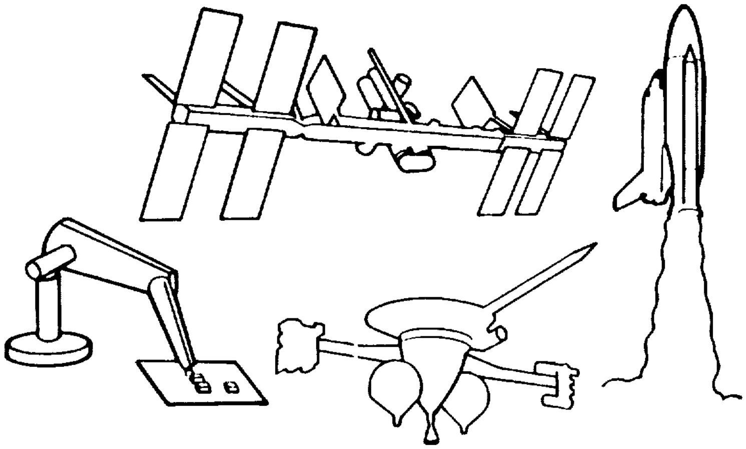
JPL Publication 89-45, Vol. 2

Proceedings of the 3rd Annual Conference on Aerospace Computational Control

Volume 2

D. E. Bernard
G. K. Man
Editors



December 15, 1989

NASA
National Aeronautics and
Space Administration

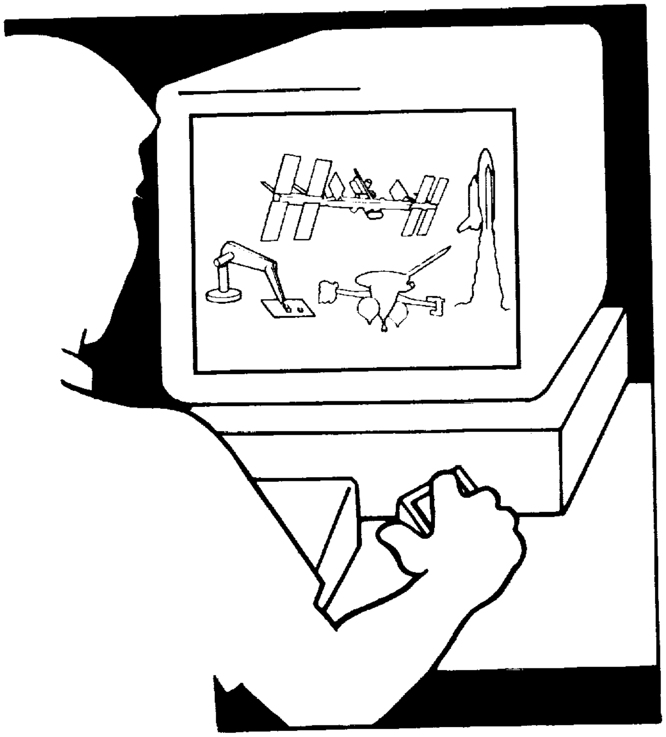
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

JPL Publication 89-45, Vol. 2

Proceedings of the 3rd Annual Conference on Aerospace Computational Control

Volume 2

D. E. Bernard
G. K. Man
Editors



December 15, 1989

NASA
National Aeronautics and
Space Administration

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

This publication was prepared by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

ABSTRACT

Volume II of two volumes of the Proceedings of the 3rd Annual Conference on Aerospace Computational Control begins with the article "Concurrent Processing Simulation of the Space Station."

The term Computational Control was coined this year to encompass that range of computer-based tools and capabilities needed by aerospace control systems engineers for design, analysis, and testing of current and future missions. This year's conference furthered the dialogue in this area begun at the 1987 Workshop on Multibody Simulation in Pasadena and continued at the 1988 Workshop on Computational Aspects in the Control of Flexible Systems in Williamsburg, Virginia.

A group of over 200 engineers and computer scientists representing government, industry, and universities convened at Oxnard for a three-day intensive conference on computational control. The conference consisted of thirteen sessions with a total of 107 technical papers in addition to opening and closing panel discussions.

Conference topics included definition of tool requirements, advanced multibody simulation, formulations, articulated multibody component representation descriptions, model reduction, parallel computation, real time simulation control design and analysis software, user interface issues, testing and verification, and applications to spacecraft, robotics, and aircraft.

CONTENTS

CONTENTS

PARALLEL PROCESSING I

MULTIBODY ADVANCED FORMULATION II

CONTENTS

CONTROL DESIGN AND ANALYSIS II

ILLUSTRATIVE DYNAMICAL SYSTEMS

CONTENTS

PARALLEL PROCESSING II

CONTENTS

EMERGING INTEGRATED CAPABILITIES

LOW ORDER CONTROLLERS

CONTENTS

REAL TIME SIMULATION

MBODY COMPONENT REPRESENTATION

CONTENTS

USER ENVIRONMENT

DISTRIBUTED PARAMETER TECHNIQUES

CLOSING SESSION

CONCURRENT PROCESSING SIMULATION OF THE SPACE STATION

by

R. Gluck, TRW Space & Technology Group, Redondo Beach, CA. 90278
A. L. Hale, Supercomputing Solutions Inc., San Diego, CA. 92121
J. W. Sunkel, NASA Johnson space Center, Houston, TX., 77058

ABSTRACT

This report describes the development of a new capability for the time-domain simulation of multibody dynamic systems and its application to the study of a large-angle rotational maneuvers of the Space Station. The effort was divided into three sequential tasks, which required significant advancements of the state-of-the art to accomplish. These were: a) the development of an explicit mathematical model via symbol manipulation of a flexible, multibody dynamic system; b) the development of a methodology for balancing the computational load of an explicit mathematical model for concurrent processing, and c) the implementation and successful simulation of the above on a prototype Custom Architectured Parallel Processing System (CAPPS) containing eight processors.

The throughput rate achieved by the CAPPS operating at only 70 percent efficiency, was 3.9 times greater than that obtained sequentially by the IBM 3090 supercomputer simulating the same problem. More significantly, analysis of the results leads to the conclusion that the relative cost-effectiveness of concurrent vs. sequential digital computation will grow substantially as the computational load is increased. This is a welcomed development in an era when very complex and cumbersome mathematical models of large space vehicles must be used as substitutes for full-scale testing which has become impractical.

1.0 INTRODUCTION

The Space Station exemplifies future NASA missions, which contemplate the use of large, flexible multibody space vehicles requiring structural dynamics control to meet their objectives. Because of their large size and limberness, full scale development and verification testing of these vehicles in the laboratory is impractical. Even if such tests could be made, results obtained in the earth gravitational environment are often misleading or inconclusive regarding the vehicle's on-orbit behavior. For these reasons, analytical modeling and simulation have become essential tools for large space structures design.

To satisfy the designer's needs, analytical modeling and simulation tools for large space structures must possess the following attributes:

- Accommodate all desired rigid-and flexible-body degrees of freedom of the system and incorporate acceptable models of its control system(s) and external forces and torques acting on it.

- Require short computation times and keep computation costs within reasonable bounds.

- Are versatile enough to accommodate radical variations in space structure configuration from one study to the next.

The most readily available analytical simulation tools in the aerospace industry are sequential digital computers. The most common among these are large mainframe computers and supercomputers which do meet high fidelity and versatility requirements, but only with a crippling penalty of simulation time and cost. Moreover, experience gathered at TRW over the past several years strongly suggests that the execution speed of conventionally coded software on commercially available sequential computers is rapidly approaching a limit; only relatively modest improvements in simulation throughput rate can be expected for these computers in the near future. Yet, the cost-per-run, at present, for even the most efficient of them is excessive and precludes comprehensive simulation studies or meaningful support of the design process.

This paper describes the results of a project undertaken to demonstrate the application of a specific concurrent processing system, the Custom Architectured Parallel Processing System (CAPPS), in determining the control/structure interaction of a representative Space Station undergoing a large angle maneuver. The project was carried out under a NASA contract (NAS 9-17778) with the Johnson Space Center. It consisted of the following three tasks:

(a) Develop an explicit control/structure interaction model of the Space Station. This task was a joint effort of TRW and NASA personnel, the latter providing the structural data and control models and the former applying these data to the development of an explicit mathematical model of the Space Station via symbol manipulation.

(b) Distribute the computational load for the CAPPS. A methodology for a balanced computational load distribution was applied to the Space Station model of Task (a) to prepare it for concurrent processing on the CAPPS.

(c) Demonstrate the CAPPS multiprocessor. In this task, the control/structure interaction of the Space Station model was simulated using a CAPPS containing 8 Computational Units (processors). The simulation speedup achieved by this concurrent processor was measured and compared to the performance of sequential digital computers simulating the same problem.

This paper is divided into 5 sections. Sections 2, 3 and 4 are devoted to the work accomplished under Tasks (a), (b) and (c), respectively. Section 5 contains the conclusions drawn from the results obtained. Further details of the Space Station simulation and CAPPS implementation are contained in Reference 1.

2.0 SPACE STATION MODEL DEVELOPMENT

2.1 Derivation of the Equations of Motion

A non-linear mathematical model describing the fully coupled rigid-and flexible-body motion of the Space Station undergoing a large angle maneuver was derived in explicit (scalar) mathematical form using Kane's dynamical equations. Explicit equations provide the analyst with considerable engineering insight into the problem being solved, permitting fine tuning of the mathematical model, including the elimination of superfluous operations, such as additions of zeros, multiplications by unity, or the computations of dot products of orthogonal vectors. Moreover, the derivation of explicit dynamical equations of motion is performed only once, in contrast with conventional implicit formulations (such as Programs DISCOS and Treetops, References 2 and 3, respectively) in which the equations of motion are essentially rederived at each time step of the numerical integration. This leads to a significant reduction in simulation time of explicit models compared to implicit ones. In one example, a 4-fold increase in simulation speed was realized at TRW by an explicit model compared to that obtained with Program DISCOS simulating the same problem. Another advantage of explicit models is the ability to determine the degree of accuracy to which important parameters must be known to achieve a desired accuracy of the solution. Finally, explicit equations lend themselves well to "coarse grain" computational load distribution in preparation for concurrent processing simulation, as described in Section 3.

Explicit equations of motion are developed by applying the Symbolic Manipulation Program (SMP, see Reference 4) to the Space Station model. This method of generating explicit equations of motion in SMP using Kane's formulation will be hereafter designated as Program SYMBOD (Symbolic Multi-Body).

Program SYMBOD generates a set of ordinary differential equations of the form: A(q,t)ud= b(q,u,t), qd = f(q,u,t), where q and qd are generalized coordinates and their first time derivatives, respectively, u and ud are, respectively, generalized speeds and their first time derivatives, and t is time. Elements of A, b, and f are generated by SYMBOD and then translated into FORTRAN via file. Symbolically deriving the model eliminates the many coding errors and debugging steps required when equations of motion are formulated implicitly.

Developing an operational symbol manipulation methodology for deriving Kane's dynamical equations requires a systematic method of reducing the number of algebraic operations in the formulation of these equations. Frequently the intermediate computations of expressions, such as velocity terms, produce expressions so large that their storage requirements exceed the computer's capacity. Therefore, a procedure for systematically introducing new intermediate symbols to replace recurring combinations of algebraic subexpressions was developed in SYMBOD. This procedure eliminates repetitious calculations and results in efficient computational algorithms requiring fewer arithmetic operations and a vastly reduced computer storage.

A series of utility procedures were developed to generate symbolic expressions for partial velocities, partial angular velocities, their associated time derivatives, and the equations of motion. One important advantage of this novel approach of formulating the equations of motion is the analyst's ability to redefine quantities such as generalized speeds and partial velocities to fit his needs. This can be done very easily with just minor modifications to Program SYMBOD. In contrast, these revisions would require such a major modification in a conventional implicit formulation code, often making it impractiacal to accomplish. This very desirable feature is not available in any other simulation code for multibody dynamic systems. Its application, however, requires intensive interaction of an experienced analyst well versed in Kane's formalism.

2.2 Model Description

The physical system of the Space Station was described by three flexible bodies interconnected at the two ALPHA gimbals (or hinges) to form the topological tree configuration of Figure 1. The main central body (Body 1), containing the pressurized modules inboard of the two ALPHA gimbals, was selected as the reference body for the Space Station model. The starboard body (Body 2) and the port body (Body 3) each consisted of all the components, including the solar arrays, on the transverse boom outboard of the ALPHA gimbals.

Finite element models were developed for each body of the Space Station. They consisted of an unconstrained (free-free) model of the central body and two constrained (fixed-free) models of the starboard and port bodies cantilevered at the ALPHA gimbals. The characteristics of the finite element models are shown in Table 1. The MSC/NASTRAN program was used to obtain the natural modes of vibration within a 10.0 Hz frequency band. The spectrum of natural frequencies for each of the three finite element models is shown in Figure 2. Note that these are characterized by a number of low frequency modes (below 1 Hz) spaced closely together. Each of the bodies in the model was described by its own assumed admissible spatial functions which were extracted from the modal data.

The three-body Space Station model contained eight (8) large-motion, rigid-body degrees-of-freedom (dof), three translational and three rotational for the central body, and one rotational for each of the extraneous bodies relative to the central body. Full coupling between the rigid-and flexible-body dof was facilitated in the model. The flexibility of Body 1 was described by 44 "free-free" natural modes used here as assumed admissible functions. The flexibilities of Bodies 2 and 3 were each described by 44 "fixed-free" natural modes serving also as assumed admissible functions. The entire model consisted of 140 coupled rigid-and flexible-body dof.

The Space Station model was used to simulate a transient maneuver involving a large-angle, rigid-body rotation of the flexible solar arrays connected to the transverse booms, while maintaining the central body in a three-axis attitude control mode. Two separate control systems were incorporated in the model to simulate this maneuver. The first one was a three-axis attitude control system using uncoupled proportional-differential feedback control laws, designed to regulate the Space Station orientation and keep a longitudinal axis of the central body aligned with the local vertical, while maintaining a plane containing this axis perpendicular to the velocity vector. The control system consisted of attitude sensing instrumentation, control moment gyros, and electronics to cause corrective control moments to be applied to the Space Station central body whenever it moved away from the commanded attitude. The attitude rate sensors and the control moment gyros were co-located at the central body's undeformed center of mass.

The second control system executes the large-angle rotations of the ALPHA gimbals. This control system was designed to maintain the solar arrays pointing in a direction perpendicular to the sun line. The second order control law uses angular position and rate feedback of the ALPHA gimbal to calculate the controller's motor torque. Options were provided in the control law to rewind the solar arrays during eclipse. This control system was activated by rotating the spacecraft-sun line a specified angle away from the solar array's normal.

3.0 COMPUTATIONAL LOAD DISTRIBUTION

The optimization of a concurrent processor performance is achieved by minimizing that part of the computational load which must be performed sequentially. The realization of this statement, often identified as Amdahl's Law, is what makes the computational load distribution for concurrent processing a formidable task.

The explicit first-order Kane's equations of motion are integrated numerically using a fourth order Adams-Bashforth algorithm. This involves evaluating new u and q vectors at each time step based on computed values of ud and qd at the current and 3 preceding time steps. Evaluating the current ud and qd vectors, the derivative evaluation phase is based on computed values of u and q at the previous time step as well as t.

The derivative evaluation and numerical integration for the Space Station model were distributed among 8 CAPPS processors based on a "coarse-grain" decomposition of the data. Guided by the problem physics, the 8 rigid-body dof were allocated to processor 1, and 22 of the 44 flexible-body dof's per body were allocated to processors 2, 3, 5, 6, 7, and 8, which were paired so that processors 2 and 3 were dedicated to body 1, processors 5 and 6 to body 2, and processors 7 and 8 to body 3. Processor 4 was allocated computation associated with the coupling of bodies 2 and 3 to body 1, but it was not allocated any dof. Both computation and communication "costs" were considered carefully before choosing this distribution.

The computations for evaluating ud and qd at each time step, which are sequential for sequential execution, were next divided into numerous subroutines appropriate for the concurrent computation. Finally, the subroutines were distributed among the processors and communication of data was added as shown in Figure 3. The arrows in the figure show communication among the processors. The distribution is heterogeneous, i.e., different processors execute quite different sequences of operations.

Note that the routines "com1", "com2", and "com3" compute intermediate data that are common between the rigid-body and flexible-body computations for bodies 1, 2, and 3, respectively. Since the amount of computation involved in these routines is relatively small compared to that in other parts of the code, it was concluded that computing them once and communicating the results would take longer than repeating the computations. Therefore, these computations were repeated in appropriate processors rather than being distributed. This is indicative of the care that must be taken to minimize the sequential part of the overall computation in concurrent processing as implied by Amdhal's Law cited above.

Also, note that a distributed block Successive Over-Relaxation (SOR) algorithm (e.g. Reference 5) was used to solve the simultaneous linear equations, A*ud=b, for ud at each time step. for the Space Station simulation on CAPPS, the SOR algorithm is more advantageous than L-U or other direct decomposition algorithms. There are 3 major advantages. First, while SOR is iterative, the solution from the previous time step is an effective starting guess to the solution at the current time step. Second, since the iterative algorithm is self-adaptive to variations in the computational load and the average number of SOR iterations decreases as the simulation progresses, the SOR algorithm is actually more efficient than L-U decomposition. And third, the communication pattern among processors is simple and allows high performance to be achieved on CAPPS.

Finally, the load distribution just discussed for the Space Station (Figure 3) was done by extensively editing the FORTRAN equations generated by SYMBOD. Editing the FORTRAN was a laborious but one-time experience. This experience taught us how the process can be imbedded in the SYMBOD code in a generalized form, a task left for future implementation.

4.0 SIMULATION PERFORMANCE AND RESULTS ON CAPPS

To demonstrate the CAPPS, a transient maneuver of the Space Station was simulated. The maneuver involved 10 degree rotations of both solar arrays about the ALPHA gimbals. The maneuver represents reorienting and then controlling the solar arrays to be perpendicular to the sun line. The control system executes the solar-array maneuver and simultaneously acts to maintain the central body of the Space Station in a fixed attitude with one axis pointing along the local vertical, and a plane containing that axis pointing along the velocity vector. Starting with quiescent initial conditions and no external disturbances, the control systems were turned on at time t=0 and the maneuver was terminated after simulating 200 seconds.

Simulation results and execution times were obtained on 1 and 8 CAPPS processors as well as on a SUN workstation and an IBM 3090 supercomputer (see Table 2). The IBM 3090 was chosen for comparison here because in prior benchmarks conducted by TRW, using a comparative simulation problem, the IBM 3090 throughput rate exceeded those of the Cray XMP, Cray 1S, Cray 2, and CYBER 205 supercomputers by 5, 17, 74, and 162 percent, respectively.

Table 2 contains both the CPU times for the 200 second simulated maneuver and the corresponding ratios of CPU time to real time. The 1-processor CAPPS, SUN workstation, and IBM-3090 all ran the same sequential code. The 8-processor CAPPS ran the parallelized version of the same simulation code. The simulations were performed with a fixed integration time step of 0.005 seconds, which was dictated by the highest frequency (10 Hz) present in the differential equations of motion. The

8-processor CAPPS simulation is a factor of 5.61 times faster than the 1-processor version, indicating an overall efficiency of 70.4 percent.

Execution times for the "coarse-grain" balanced computational load distribution among CAPPS' 8 processors are shown in Figure 4. the computational elements shown in the figure correspond to those shown in Figure 3 of Section 3. Note the idle times in the distributed load of each of the processors. The largest idle time was in CU4, which was not allocated any dof. Also note that roughly 40% of the total computation time was spent in the SOR solution and numerical integration.

It is interesting to consider in more detail the SOR linear equation solution part of the simulation. The algorithm is similar to block SOR (Reference 5), but it was specially tailored to the CAPPS and Space Station simulation. The distributed algorithm was run on the CAPPS with 1, 2, 4, and 8 processors and with different size matrices representing multibody systems of different numbers of dof. The execution times are presented in Figure 5, where the speedup factor is plotted against the number of processors with the computational load as a parameter. The speedup factor is the ratio of computational time with 1 processor to that with m processors solving the same, fixed size problem. Since memory size of the prototype CAPPS used limited the largest matrix that could be held by 1 processor to approximately n=500, the speedup factors for large problems are scaled factors as discussed in Reference 6.

A significant conclusion based on the results of Figure 5 is that the efficiency (defined as the speedup factor divided by the number of processors) of the CAPPS increases sharply as a function of the computational load. As the latter increased from 72 to 1200 dof, the 8-processor system's efficiency increased from 40 to 92 percent. This behavior of a loosely coupled concurrent processing system is explained by the observation that, to a first approximation, the parallel parts of the problem scale with the problem size, whereas the non-parallel parts (including communication) do not. As the problem size increases, the non-parallel operations constitute a smaller percentage of the total computational load.

Finally, Figure 6 contains 4 temporal plots of representative state vector entries. They are: a) the relative angular rotation of the starboard ALPHA gimbal, b) the first time derivative of the relative angular rotation of the starboard ALPHA gimbal, c) the inertial angular velocity of the central body along the 1 axis, and d) the fourth elastic displacement function of the starboard body. Comparing the ALPHA gimbal rotation and rotation rate plots, one can see evidence of flexible motion superposed on the rigid-body motion at the beginning of the maneuver. Also, one can see evidence in the elastic displacement function shown that the bending deformation of the solar arrays is fully coupled to the rigid-body motion of the system. While only 4 plots are presented here, all entries of the state vector and its first time derivative as obtained from the four simulations were compared and found indistinguishable.

5.0 CONCLUSIONS

This work represents a major advance in the state of the art for analytical simulation of large space systems. Concurrent processing now offers the capability of simulating very large and complex mathematical models of multibody dynamical systems at high speeds and at an acceptable cost.

The performance to cost ratio of loosely coupled concurrent processors (CAPPS) vis-a-vis sequential computers was demonstrated to increase with computational load.

Having an explicit mathematical model is invaluable for "coarse-grain" computational load distribution, balancing, tuning, and otherwise maximizing the simulation throughput rate. The Symbol Manipulation Program (SMP) conveniently generates the explicit model.

The simulation process is divided into model development, computational load distribution, and computational load balancing steps. For practical application, all three steps must be mechanized to render most of the explicit model generation and load balancing process transparent to the user. This is feasible, based on the experiences reported herein.

Finally, on going work endeavors to incorporate an n-order algorithm for multibody equations together with explicit modeling and concurrent processing. Preliminary results, not reported here, demonstrates that this provides the capability of simulating, in real time, multibody systems with hundreds of large motion degrees of freedom.

ACKNOWLEDGEMENTS

The authors wish to express their sincere appreciation to Dr. Sam Lee of TRW Space & Technology Group for his development of the explicit mathematical model of the Space Station, and to Dr. Paul Cooper, Tom Sutter and John Young of NASA/LaRC and Jalal Mapar of Grumman Aerospace Corp. for supplying the "raw" data necessary for this development. Special thanks are due Dr. Kenneth Cox of NASA/JSC for his encouragement and support throughout the Space Station simulation development program.

REFERENCES

1. _____, "A Custom architectured Parallel Processing System for the Space Station," TRW No. EML-003, Final report, Contract No. NAS9-17778, NASA Lyndon B. Johnson Space Center, May 1989.

2. Bodley, C.S., Devers, A.D., Park, A.C., and Frisch, H.P., "A Digital Computer Program for Dynamic Interaction Simulation of Controls and Structures (DISCOS)," Technical Paper 1219, Vols. 1 and 2, NASA, May 1978.

3. Singh, R.P., VanderVoort, R.J., and Likins, P.W., "Dynamics of Flexible Bodies in Tree Topology -- A Computer Oriented Approach," Paper No. AIAA-84-1024, AIAA/ASME/ASCE 25th Structures, Structural Dynamics, and Materials Conference, Palm Springs, CA. May 14-18, 1984.

4. Lee, S.S., "Symbolic Generation of Equations of Motion for Dynamics/Control Simulation of Large Flexible Multibody Space Systems," Ph.D. Dissertation, University of California, Los Angeles, 1988.

5. J. M. Ortega, Introduction to Parallel and Vector Solution of Linear Systems, Plenum Press, New York, 1988.

6. Gustafson, J.L., et al., "Development of Parallel Methods for a 1024-Processor hypercube," SIAM Journal of Scientific and Statistical Computations, Vol. 9, No. 4, July 1988.

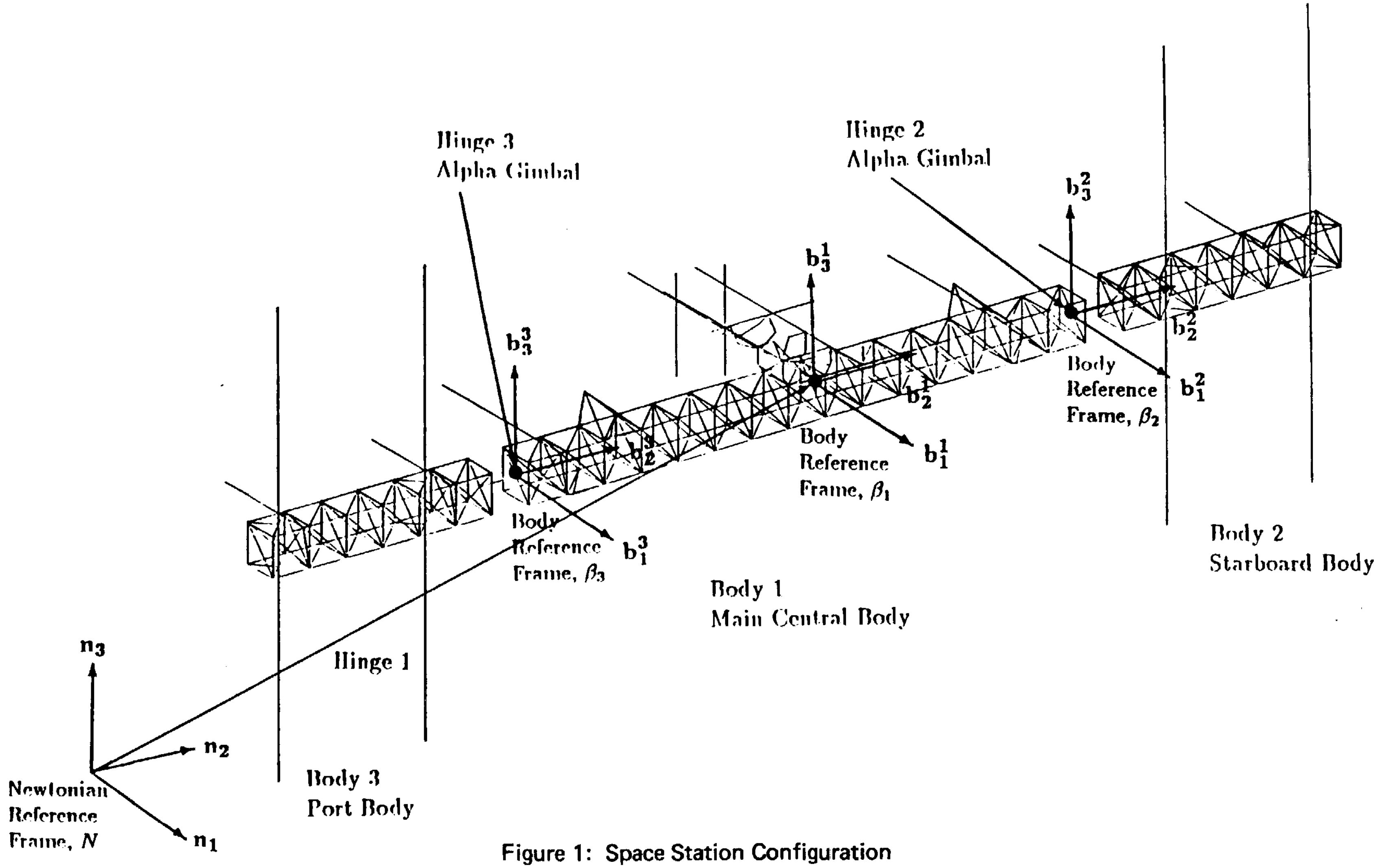


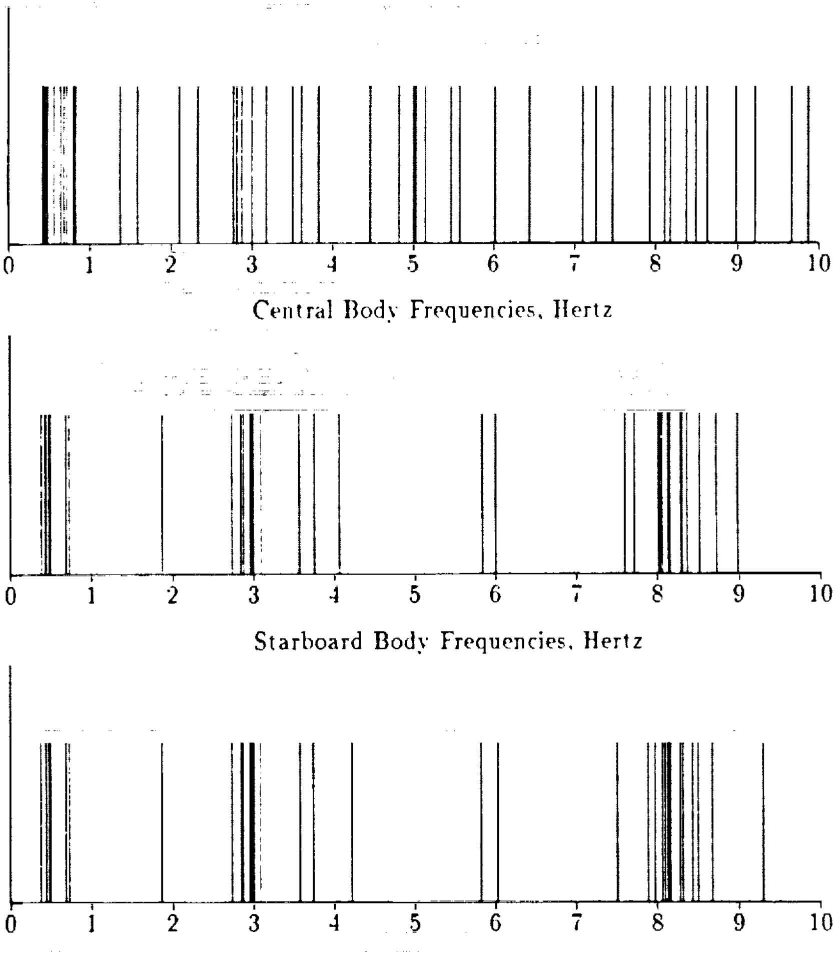
Figure 1: Space Station Configuration

Table 1: Space Station Model and Mass Properties Data

Model	**Central Body**	**Starboard Body**	**Port Body**
Finite Element Models:			
Grids	160	72	72
Elements	315	120	120
DOF	942	270	270
Mass and Inertia Data:			
Mass data (lb)			
Mass	373786	26685	26685
Center of mass (in)[a]			
X_1	0.0	-29.4	-29.4
X_2	0.0	733.3	-733.3
X_3	0.0	16.4	16.4
Centroidal Inertia data (lb · in^2)[b]			
I_{11}	8.047E10	6.973E09	6.973E09
I_{22}	6.749E10	3.162E09	3.163E09
I_{33}	1.114E11	4.836E09	4.836E09
I_{12}	9.092E08	3.408E07	-3.557E07
I_{13}	-5.099E09	1.243E07	1.243E07
I_{23}	3.296E09	2.292E07	-2.441E07

[a] measured from origin of β reference frame

[b] at CM about β reference fram axes



Central Body Frequencies, Hertz

Starboard Body Frequencies, Hertz

Port Body Frequencies, Hertz

Figure 2: Frequency Spectra of the Space Station Model

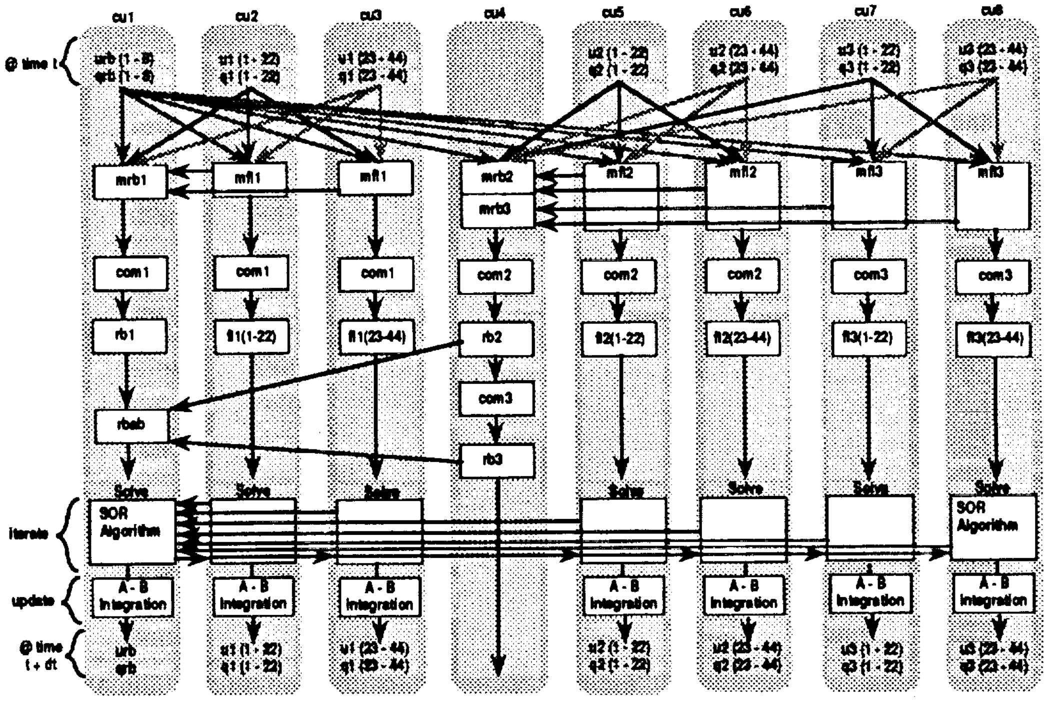


Figure 3: Computational Load Distribution for the Space Station Simulation on CAPPS

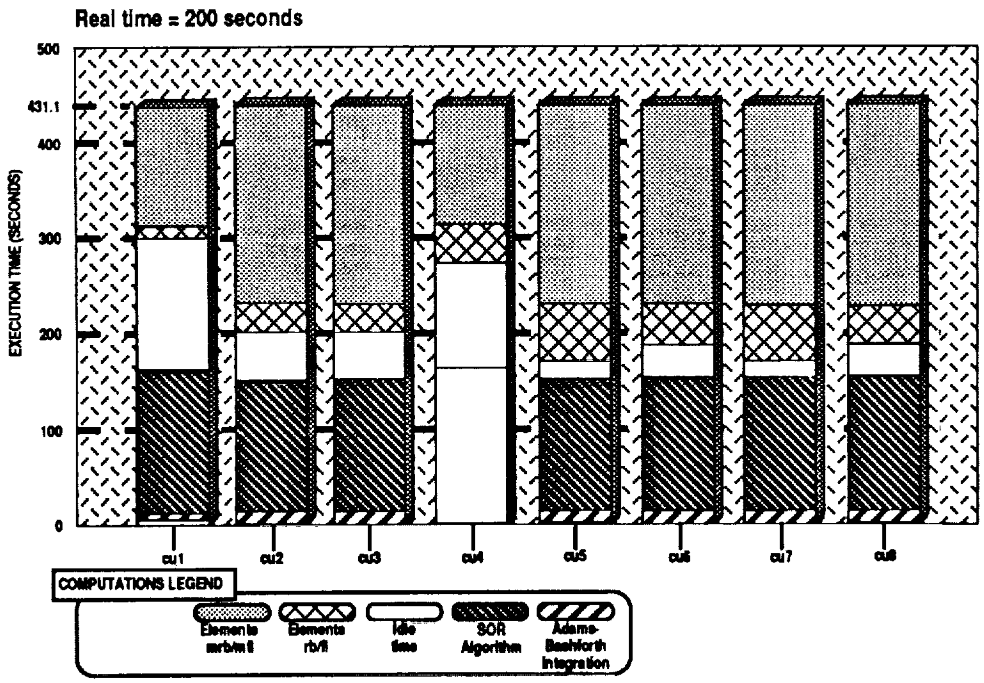


Figure 4: Execution Time for Coarse-Grain Balanced Computational Load

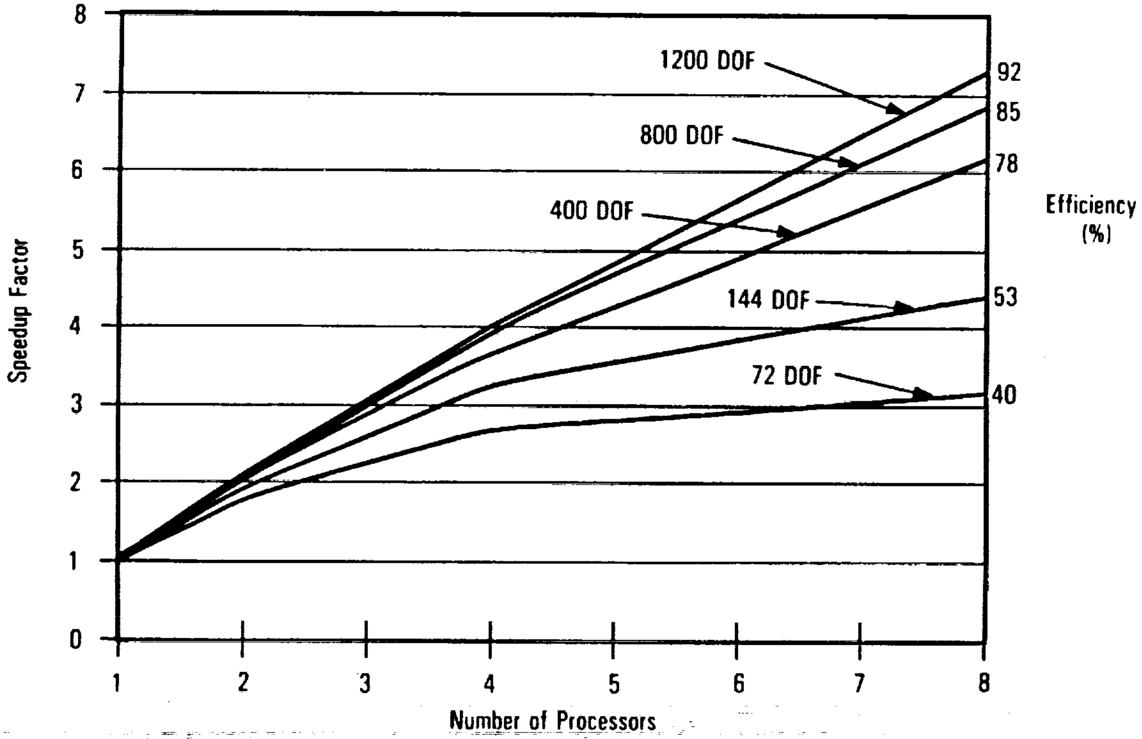


Figure 5: Speedup Factors for the Successive Over Relaxation Algorithm on CAPPS B–32

Table 2: Space Station Simulation Results

PARAMETER	CAPPS B - 32		IBM	SUN
	1 CU	8 CUS**	3090/180E	25MHz
CPU TIME (MINUTES)	40.3	7.2	28.2	1844.8
CPU TIME/REAL TIME *	12.1	2.2	8.5	553.4

* Real time simulation - 200 seconds

** 8 - CU CAPPS speedup factor: 5.6 (70 percent overall efficiency)

ORIGINAL PAGE IS
OF POOR QUALITY

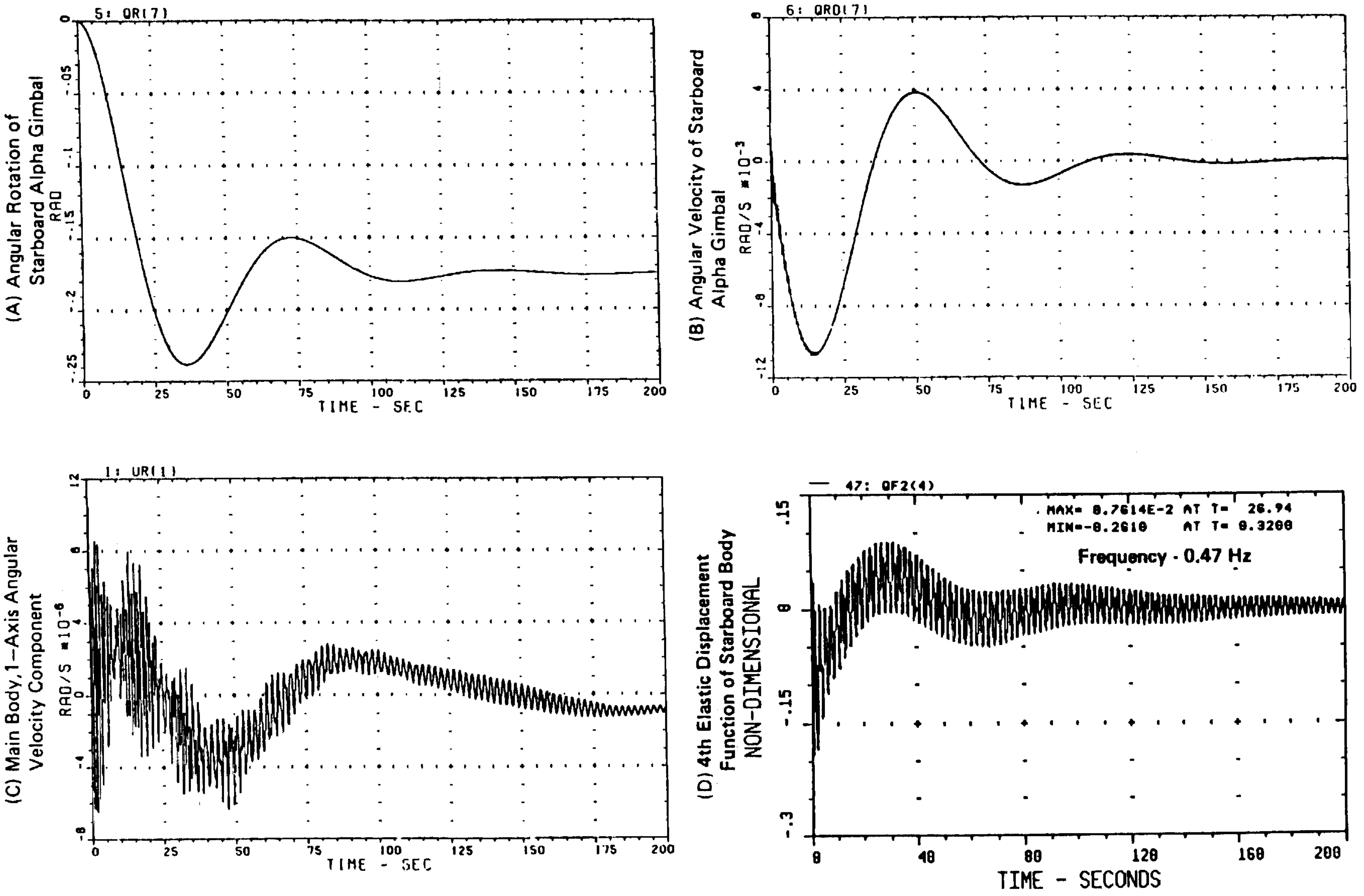


Figure 6: Representative Time Histories of the Flexible Space Station

N90-23042

A DECOUPLED RECURSIVE APPROACH FOR CONSTRAINED FLEXIBLE MULTIBODY SYSTEM DYNAMICS

Hao-Jan Lai, Sung-Soo Kim, and Edward J. Haug
The Center for Simulation and Design Optimization of Mechanical Systems
The University of Iowa
Iowa City, Iowa 52242

Dae-Sung Bae
The University of Kansas
Lawrence, Kansas 66045

Abstract

A variational-vector calculus approach is employed to derive a recursive formulation for dynamic analysis of flexible multibody systems. Kinematic relationships for adjacent flexible bodies are derived in a companion paper [7], using a state vector notation that represents translational and rotational components simultaneously. Cartesian generalized coordinates are assigned for all body and joint reference frames, to explicitly formulate deformation kinematics under small deformation assumptions. Relative coordinate kinematics for joints are decoupled from deformation kinematics and an efficient flexible dynamics recursive algorithm is developed. Dynamic analysis of a closed loop robot is performed to illustrate efficiency of the algorithm.

1. Introduction

A recursive dynamics formulation was proposed by Armstrong [1] to analyze a robot manipulator, beginning with Cartesian equations of motion in a joint reference frame. Reaction forces were introduced as unknown forces into the equations of motion. These unknown forces were then eliminated to obtain recursion formulas for calculation of reduced equations of motion. The method was reformulated by Featherstone [2] and used to analyze a robot arm that consists of revolute and/or translational joints. He used a spatial notation to relieve notational complexity and introduced a new "articulated inertia" terminology that reflects inertia effects of all outboard bodies in a kinematic chain. Neither method considered the effect of flexibility of components.

Variational approaches have dominated structural analysis for the last decade. The variational method has recently been combined with vector calculus, to permit systematical transformation of the equations of motion from Cartesian space to joint coordinate space [3]. The same variational approach was used to derive a recursive formulation for constrained rigid body mechanical system dynamics in Ref. 4.

A variational equation of motion for constrained flexible systems was derived in Ref. 5, using Cartesian coordinates. The variational approach was applied to extend the rigid body recursive formulation to flexible body systems by Kim [6]. Kinematic relationships between reference frames for a pair of bodies that are connected by a joint are expressed in terms of joint relative coordinates and modal deformation coordinates of bodies. As a result, joint and modal coordinate equations of motion are coupled and must be solved simultaneously. This requires inversion of a moderately large matrix, for coupled modal and joint coordinates.

In order to enhance graph theoretic analysis of deformation characteristics, kinematics of flexible multibody systems is represented in a companion paper [7]. Based on this kinematic analysis, a recursive formulation for dynamic analysis is represented in this paper that decouples relative joint and deformation coordinates, to improve computational efficiency. The proposed formulation can be used with a rigid body formulation by eliminating terms related to modal coordinates, due to its decoupled treatment of gross motion and deformation.

State vector representations and kinematics of flexible multibody systems, defined in Refs. 8 and 7 respectively, are summarized in Section 2. The equation of motion for a flexible body is transformed from the Cartesian space to a state space setting in Section 3. System topology is defined in Section 4 and recursive equations of motion for a single closed loop subsystem are derived in Section 5. Cut joint constraint acceleration equations that are needed in the equations of motion are derived in Section 6. The base body equation of motion is defined in Section 7. Numerical examples and results are presented in Section 8.

2. Decoupled Recursive Relationships for Flexible Bodies

To derive the variational equations of motion, state vector notation and decoupled recursive relationships for adjacent reference frames [7, 8] are briefly reviewed here. A matrix representation of the Cartesian velocity of a reference frame with origin at point P, as shown in Fig. 1, is given as $\mathbf{Y}_p = [\dot{\mathbf{r}}_p^T \ \boldsymbol{\omega}_p^T]^T$, where $\dot{\mathbf{r}}_p^T$ is the velocity of point P and $\boldsymbol{\omega}_p$ is the angular velocity of the x_p'- y_p'- z_p' body reference frame. A generalized velocity state vector $\hat{\mathbf{Y}}_p$, based on screw and motor algebra [8, 9], is defined here as

$$\hat{\mathbf{Y}}_p \equiv \begin{bmatrix} \dot{\mathbf{r}}_p + \tilde{\mathbf{r}}_p\boldsymbol{\omega}_p \\ \boldsymbol{\omega}_p \end{bmatrix} = \mathbf{T}_p\mathbf{Y}_p \equiv \mathbf{T}_p\begin{bmatrix} \dot{\mathbf{r}}_p \\ \boldsymbol{\omega}_p \end{bmatrix} \qquad (2.1)$$

where the 6x6 nonsingular matrix $\mathbf{T}_p$ is defined as

$$\mathbf{T}_p = \begin{bmatrix} \mathbf{I} & \tilde{\mathbf{r}}_p \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \qquad (2.2)$$

The tilde operator is used here to define a skew symmetric matrix as

$$\tilde{\mathbf{r}} \equiv \begin{bmatrix} 0 & -r_z & r_y \\ r_x & 0 & -r_x \\ -r_y & r_x & 0 \end{bmatrix} \qquad (2.3)$$

that is associated with a vector $\mathbf{r} = [r_x, r_y, r_z]^T$.

The Cartesian virtual displacement $\delta\mathbf{Z}_p$ is defined as

$$\delta\mathbf{Z}_p \equiv \begin{bmatrix} \delta\mathbf{r}_p \\ \delta\boldsymbol{\pi}_p \end{bmatrix} \qquad (2.4)$$

where $\delta\mathbf{r}_p$ is virtual displacement of point P and $\delta\boldsymbol{\pi}_p$ is virtual rotation of the x_p'- y_p'- z_p' frame. The state variation can be obtained by replacing $\dot{\mathbf{r}}_p$ and $\boldsymbol{\omega}_p$ by $\delta\mathbf{r}_p$ and $\delta\boldsymbol{\pi}_p$, respectively, in Eq. 2.1; i.e.,

$$\delta\hat{\mathbf{Z}}_p \equiv \begin{bmatrix} \delta\mathbf{r}_p + \tilde{\mathbf{r}}_p\delta\boldsymbol{\pi}_p \\ \delta\boldsymbol{\pi}_p \end{bmatrix} = \mathbf{T}_p\delta\mathbf{Z}_p \qquad (2.5)$$

The acceleration state vector $\dot{\hat{\mathbf{Y}}}_p$ is defined as the time derivative of velocity state $\hat{\mathbf{Y}}_p$ of Eq. 2.1; i.e.,

$$\dot{\hat{\mathbf{Y}}}_p = \begin{bmatrix} \ddot{\mathbf{r}}_p + \tilde{\mathbf{r}}_p\dot{\boldsymbol{\omega}}_p + \dot{\tilde{\mathbf{r}}}_p\boldsymbol{\omega}_p \\ \dot{\boldsymbol{\omega}}_p \end{bmatrix} = \mathbf{T}_p\dot{\mathbf{Y}}_p + \mathbf{X}_p \qquad (2.6)$$

where $\mathbf{X}_p$ is the 6x1 vector

$$\mathbf{X}_p = \begin{bmatrix} \dot{\tilde{\mathbf{r}}}_p \omega_p \\ \mathbf{0} \end{bmatrix} \tag{2.7}$$

The inverse relationships between Cartesian and state vector quantities can be derived from Eqs. 2.1, 2.5, and 2.6 as

$$\mathbf{Y}_p = \mathbf{T}_p^{-1} \hat{\mathbf{Y}}_p \tag{2.8}$$

$$\delta \mathbf{Z}_p = \mathbf{T}_p^{-1} \delta \hat{\mathbf{Z}}_p \tag{2.9}$$

$$\dot{\mathbf{Y}}_p = \mathbf{T}_p^{-1} \dot{\hat{\mathbf{Y}}}_p - \mathbf{X}_p \tag{2.10}$$

where the 6x6 inverse matrix $\mathbf{T}_p^{-1}$ of the matrix $\mathbf{T}_p$ is simply

$$\mathbf{T}_p^{-1} = \begin{bmatrix} \mathbf{I} & -\tilde{\mathbf{r}}_p \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \tag{2.11}$$

Three flexible bodies, with their body and joint reference frames, are shown in Fig. 2. The x-y-z frame is the global reference frame, denoted as F. Two joint reference frames are attached to a body i at each joint definition point P_{ij}. The x_{ij}'''-y_{ij}'''-z_{ij}''' frame, denoted as F_{ij}''', is fixed to body i and is parallel to the x_i'-y_i'-z_i' frame, denoted as F_i', in the undeformed state. The $x_{ij}^{\bullet}$- $y_{ij}^{\bullet}$- $z_{ij}^{\bullet}$ frame, denoted as $F_{ij}^{\bullet}$, is body fixed and has fixed orientation, relative to the F_{ij}''' frame, since both are fixed to the body at the same joint definition point where the body is assumed to be very stiff.

Recursive relationships between reference frames in a joint, for example, between the $x_{ij}^{\bullet}$- $y_{ij}^{\bullet}$- $z_{ij}^{\bullet}$ and $x_{ji}^{\bullet}$- $y_{ji}^{\bullet}$- $z_{ji}^{\bullet}$ frames of joint (i,j), are

$$\hat{\mathbf{Y}}_{ji} = \hat{\mathbf{Y}}_{ij} + \boldsymbol{\Pi}_{ij} \dot{\mathbf{q}}_{ij} \tag{2.12}$$

$$\delta \hat{\mathbf{Z}}_{ji} = \delta \hat{\mathbf{Z}}_{ij} + \boldsymbol{\Pi}_{ij} \delta \mathbf{q}_{ij} \tag{2.13}$$

$$\dot{\hat{\mathbf{Y}}}_{ji} = \dot{\hat{\mathbf{Y}}}_{ij} + \boldsymbol{\Pi}_{ij} \ddot{\mathbf{q}}_{ij} + \boldsymbol{\Theta}_{ij} \tag{2.14}$$

where $\hat{\mathbf{Y}}$, $\dot{\hat{\mathbf{Y}}}$, and $\delta\hat{\mathbf{Z}}$ are state representations of velocity, acceleration, and virtual displacement, respectively, and $\mathbf{q}_{ij}$ is a vector of joint relative coordinates [7].

Recursive relationships between inboard and outboard joint reference frames of a flexible body are

$$\hat{\mathbf{Y}}_{ij} = \hat{\mathbf{Y}}_{i\ell} + \boldsymbol{\Gamma}_{ij} \dot{\mathbf{a}}_i \tag{2.15}$$

$$\delta \mathbf{Z}_{ij} = \delta \mathbf{Z}_{i\ell} + \boldsymbol{\Gamma}_{ij} \delta \mathbf{a}_i \tag{2.16}$$

$$\dot{\hat{\mathbf{Y}}}_{ij} = \dot{\hat{\mathbf{Y}}}_{i\ell} + \boldsymbol{\Gamma}_{ij} \ddot{\mathbf{a}}_i + \boldsymbol{\Delta}_{ij} \tag{2.17}$$

where $\mathbf{a}$ is the deformation modal coordinate vector of the flexible body [5-7].

The recursive relationships between frames $\mathbf{F}_{ij}^{\bullet}$ and $\mathbf{F}_i'$ of a flexible body are

$$\hat{\mathbf{Y}}_{ij} = \hat{\mathbf{Y}}_i + \Lambda_{ij}\dot{\mathbf{a}}_i \tag{2.18}$$

$$\delta\hat{\mathbf{Z}}_{ij} = \delta\hat{\mathbf{Z}}_i + \Lambda_{ij}\dot{\mathbf{a}}_i \tag{2.19}$$

$$\dot{\hat{\mathbf{Y}}}_{ij} = \dot{\hat{\mathbf{Y}}}_i + \Lambda_{ij}\ddot{\mathbf{a}}_i + \Xi_{ij} \tag{2.20}$$

Detailed expressions for matrices Γ_{ij}, Δ_{ij}, Π_{ij}, Θ_{ij}, Λ_{ij}, and Ξ_{ij} may be found in Ref. 7.

3. Equation of Motion for a Flexible Body

The variational Cartesian equations of motion for a flexible multibody system are derived in Ref. 5. They can be written for a typical body i, using the notations defined in Section 2, as

$$\left[\delta\mathbf{Z}_i^T \;\; \delta\mathbf{a}_i^T\right]\left\{\mathbf{M}_i\begin{bmatrix}\dot{\mathbf{Y}}_i\\ \ddot{\mathbf{a}}_i\end{bmatrix} + \mathbf{S}_i + \mathbf{V}_i - \mathbf{Q}_i\right\} = 0 \tag{3.1}$$

which must hold for all kinematically admissible $\delta\mathbf{Z}_i$ and $\delta\mathbf{a}_i$. The mass matrix $\mathbf{M}_i$ is a function of the generalized coordinates, $\mathbf{S}_i$ is a collection of quadratic velocity terms, $\mathbf{V}_i$ is the elastic generalized force, and $\mathbf{Q}_i$ is the applied generalized force.

The equations of motion in Cartesian space are transformed to state vector form by substituting kinematic relationships between the spaces. The state variation and acceleration relationships of Eqs. 2.9 and 2.10 are substituted into Eq. 3.1, to yield

$$\left[\delta\hat{\mathbf{Z}}_i^T \;\; \delta\mathbf{a}_i^T\right]\left\{\hat{\mathbf{M}}_i\begin{bmatrix}\dot{\hat{\mathbf{Y}}}_i\\ \ddot{\mathbf{a}}_i\end{bmatrix} - \hat{\mathbf{Q}}_i\right\} = 0 \tag{3.2}$$

where the state representation $\hat{\mathbf{M}}_i$ of the mass matrix is partitioned into 4 submatrices, based on state and modal coordinates,

$$\hat{\mathbf{M}}_i = \begin{bmatrix}\bar{\mathbf{M}}_i^{mm} & \bar{\mathbf{M}}_i^{ma}\\ \bar{\mathbf{M}}_i^{ma} & \bar{\mathbf{M}}_i^{aa}\end{bmatrix} \equiv \begin{bmatrix}\mathbf{T}_i^{-1} & \mathbf{0}\\ \mathbf{0} & \mathbf{I}\end{bmatrix}^T \mathbf{M}_i \begin{bmatrix}\mathbf{T}_i^{-1} & \mathbf{0}\\ \mathbf{0} & \mathbf{I}\end{bmatrix} \tag{3.3}$$

Similarly, the state representation of $\hat{\mathbf{Q}}_i$, which accounts for generalized force and coupling terms, is divided into two subvectors as

$$\hat{\mathbf{Q}} = \begin{bmatrix}\bar{\mathbf{Q}}^z\\ \bar{\mathbf{Q}}^a\end{bmatrix} \equiv \begin{bmatrix}\mathbf{T}_i^{-1} & \mathbf{0}\\ \mathbf{0} & \mathbf{I}\end{bmatrix}^T\left(\mathbf{M}_i\begin{bmatrix}\mathbf{X}_i\\ \mathbf{0}\end{bmatrix} - \mathbf{S}_i - \mathbf{V}_i + \mathbf{Q}_i\right) \tag{3.4}$$

The equations of motion in Eq. 3.2 can be rewritten, using the notations of Eqs. 3.3 and 3.4, as

$$\delta\hat{\mathbf{Z}}_i^T(\bar{\mathbf{M}}_i^{mm}\dot{\hat{\mathbf{Y}}}_i + \bar{\mathbf{M}}_i^{ma}\ddot{\mathbf{a}}_i - \bar{\mathbf{Q}}_i^z) + \delta\mathbf{a}_i^T(\bar{\mathbf{M}}_j^{am}\dot{\hat{\mathbf{Y}}}_i + \bar{\mathbf{M}}_i^{aa}\ddot{\mathbf{a}}_i - \bar{\mathbf{Q}}_i^a) = 0 \tag{3.5}$$

where $\delta\hat{\mathbf{Z}}_i$ and $\delta\mathbf{a}_i$ must be consistent with all constraints that act on body i.

4. System Topology

An extended flexible multibody graph model, in which nodes represent reference frames and edges represent transformations between frames, is presented in Ref. 7. The corresponding graph for a single closed loop system is shown in Fig. 3. Body *l* is the junction body, at which chains 1 and 2 of the spanning tree of Fig. 4 meet. If joint J(n,n+1) between bodies n and n+1 is cut, both bodies n and n+1 are treated as tree end bodies.

5. Equations of Motion of a Single Closed Loop

The variational equation of motion for the system shown in Fig. 3 is

$$\sum_{i=l}^{m}\{\delta\hat{\mathbf{Z}}_i^T(\bar{\mathbf{M}}_i^{mm}\dot{\hat{\mathbf{Y}}}_i + \bar{\mathbf{M}}_i^{ma}\ddot{\mathbf{a}}_i - \mathbf{Q}_i^z) + \delta\mathbf{a}_i^T(\bar{\mathbf{M}}_i^{am}\dot{\hat{\mathbf{Y}}}_i + \bar{\mathbf{M}}_i^{aa}\ddot{\mathbf{a}}_i - \mathbf{Q}_i^a)\} = 0 \tag{5.1}$$

which must hold for all kinematically admissible virtual displacements that satisfy joint and deformation constraints in Fig. 3.

State variation and accelerations of each body reference frame are expressed in corresponding joint terms and modal coordinates from Eqs. 2.19 and 2.20 and substituted into Eq. 5.1. The resulting equations of motion are as follows:

$$\sum_{i=l}^{m}\{\delta\hat{\mathbf{Z}}_{i(i-1)}^T(\mathbf{M}_i^{mm}\dot{\hat{\mathbf{Y}}}_{i(i-1)} + \mathbf{M}_i^{ma}\ddot{\mathbf{a}}_i - \mathbf{Q}_i^z) + \delta\mathbf{a}_i^T(\mathbf{M}_i^{am}\dot{\hat{\mathbf{Y}}}_{i(i-1)} + \mathbf{M}_i^{aa}\ddot{\mathbf{a}}_i - \mathbf{Q}_i^a)\}$$
$$= \mathrm{EQM}(1) + \mathrm{EQM}(2) = 0 \tag{5.2}$$

which must hold for all kinematically admissible virtual displacements. Terms arising in Eq. 5.2 are as follows:

$$\begin{aligned}
\mathbf{M}_i^{mm} &= \bar{\mathbf{M}}_i^{mm} \\
\mathbf{M}_i^{ma} &= \bar{\mathbf{M}}_i^{ma} - \bar{\mathbf{M}}_i^{mm}\Lambda_{i(i-1)} \\
\mathbf{M}_i^{am} &= \mathbf{M}_i^{ma^T} \\
\mathbf{M}_i^{aa} &= \bar{\mathbf{M}}_i^{aa} - \bar{\mathbf{M}}_i^{am}\Lambda_{i(i-1)} + \Lambda_{i(i-1)}^T(\bar{\mathbf{M}}_i^{mm}\Lambda_{i(i-1)} - \bar{\mathbf{M}}_i^{ma}) \\
\mathbf{Q}_i^z &= \bar{\mathbf{Q}}_i^z + \bar{\mathbf{M}}_i^{mm}\Xi_{i(i-1)} \\
\mathbf{Q}_i^a &= \bar{\mathbf{Q}}_i^a + \bar{\mathbf{M}}_i^{am}\Xi_{i(i-1)} - \Lambda_{i(i-1)}^T(\bar{\mathbf{Q}}_i^z + \mathbf{M}_i^{mm}\Xi_{i(i-1)})
\end{aligned} \tag{5.3}$$

$$\mathrm{EQM}(1) = \sum_{i=l}^{n}\{\delta\hat{\mathbf{Z}}_{i(i-1)}^T(\mathbf{M}_i^{mm}\dot{\hat{\mathbf{Y}}}_{i(i-1)}+\mathbf{M}_i^{ma}\ddot{\mathbf{a}}_i-\mathbf{Q}_i^z)+ \delta\mathbf{a}_i^T(\mathbf{M}_i^{ma}\dot{\hat{\mathbf{Y}}}_{i(i-1)}+\mathbf{M}_i^{aa}\ddot{\mathbf{a}}_i-\mathbf{Q}_i^a)\} \tag{5.4}$$

$$\mathrm{EQM}(2) = \sum_{i=n+1}^{m}\{\delta\hat{\mathbf{Z}}_{i(i-1)}^T(\mathbf{M}_i^{mm}\dot{\hat{\mathbf{Y}}}_{i(i-1)}+\mathbf{M}_i^{ma}\ddot{\mathbf{a}}_i-\mathbf{Q}_i^z)+ \delta\mathbf{a}_i^T(\mathbf{M}_i^{ma}\dot{\hat{\mathbf{Y}}}_{i(i-1)}+\mathbf{M}_i^{aa}\ddot{\mathbf{a}}_i-\mathbf{Q}_i^a)\} \tag{5.5}$$

The Jacobian matrix of the cut joint constraint function $\Phi^{J(n,n+1)}$ is obtained by differentiation as

$$\delta\Phi^{J(n,n+1)} = \Phi_{\hat{\mathbf{Z}}_{n(n+1)}}\delta\hat{\mathbf{Z}}_{n(n+1)} + \Phi_{\hat{\mathbf{Z}}_{(n+1)n}}\delta\hat{\mathbf{Z}}_{(n+1)n} = \mathbf{0} \tag{5.6}$$

where state variations $\delta\hat{\mathbf{Z}}_{n(n+1)}$ and $\delta\hat{\mathbf{Z}}_{(n+1)n}$ are obtained by employing the state vector representation of Cartesian virtual displacements. There exists a Lagrange multiplier vector such that

$$\mathrm{EQM}(2) + \sum_{i=l}^{n} \{\delta\hat{\mathbf{Z}}^{T}_{i(i-1)}(\mathbf{M}^{mm}_{i}\dot{\hat{\mathbf{Y}}}_{i(i-1)} + \mathbf{M}^{ma}_{i}\ddot{\mathbf{a}}_{i} - \mathbf{Q}^{z}_{i})$$

$$+ \delta\mathbf{a}^{T}_{i}(\mathbf{M}^{am}_{i}\dot{\hat{\mathbf{Y}}}_{i(i-1)} + \mathbf{M}^{aa}_{i}\ddot{\mathbf{a}}_{i} - \mathbf{Q}^{a}_{i})\} + \delta\hat{\mathbf{Z}}^{T}_{n(n+1)}\Phi^{T}_{\mathbf{Z}_{n(n+1)}}\lambda^{n(n+1)} = 0 \tag{5.7}$$

where the virtual displacements need only be consistent with kinematic admissibility conditions for all tree structure joints and deformation constraints. Similarly, the equation of motion for chain 2 is

$$\mathrm{EQM}(2) \equiv \sum_{i=n+1}^{m} \{\delta\hat{\mathbf{Z}}^{T}_{i(i-1)}(\mathbf{M}^{mm}_{i}\dot{\hat{\mathbf{Y}}}_{i(i-1)} + \mathbf{M}^{ma}_{i}\ddot{\mathbf{a}}_{i} - \mathbf{Q}^{z}_{i})$$

$$+ \delta\mathbf{a}^{T}_{i}(\mathbf{M}^{am}_{i}\dot{\hat{\mathbf{Y}}}_{i(i-1)} + \mathbf{M}^{aa}_{i}\ddot{\mathbf{a}}_{i} - \mathbf{Q}^{a}_{i})\} + \delta\hat{\mathbf{Z}}^{T}_{(n+1)n}\Phi^{T}_{\mathbf{Z}_{(n+1)n}}\lambda^{n(n+1)} \tag{5.8}$$

The virtual displacement $\delta\hat{\mathbf{Z}}_{n(n+1)}$ may be expressed in terms of $\delta\hat{\mathbf{Z}}_{n(n-1)}$ and $\delta\mathbf{a}_{n}$ from Eq. 2.16. Substituting this relationship into Eq. 5.7, to obtain

$$\mathrm{EQM}(2) + \sum_{i=l}^{n-1} \{\delta\hat{\mathbf{Z}}^{T}_{i(i-1)}(\mathbf{M}^{mm}_{i}\dot{\hat{\mathbf{Y}}}_{i(i-1)} + \mathbf{M}^{ma}_{i}\ddot{\mathbf{a}}_{i} - \mathbf{Q}^{z}_{i}) + \delta\mathbf{a}^{T}_{i}(\mathbf{M}^{am}_{i}\dot{\hat{\mathbf{Y}}}_{i(i-1)} + \mathbf{M}^{aa}_{i}\ddot{\mathbf{a}}_{i} - \mathbf{Q}^{a}_{i})\}$$

$$+ \delta\hat{\mathbf{Z}}^{T}_{n(n-1)}(\mathbf{M}^{mm}_{n}\dot{\hat{\mathbf{Y}}}_{n(n-1)} + \mathbf{M}^{ma}_{n}\ddot{\mathbf{a}}_{n} - \mathbf{Q}^{z}_{n} + \Phi^{T}_{\mathbf{Z}_{n(n+1)}}\lambda^{n(n+1)})$$

$$+ \delta\mathbf{a}^{T}_{n}(\mathbf{M}^{am}_{n}\dot{\hat{\mathbf{Y}}}_{n(n-1)} + \mathbf{M}^{aa}_{n}\ddot{\mathbf{a}}_{n} + \Gamma^{T}_{n(n+1)}\Phi^{T}_{\mathbf{Z}_{n(n+1)}}\lambda^{n(n+1)}) = 0 \tag{5.9}$$

which must hold for all virtual displacements that are consistent with tree structure joints and deformation constraints in Fig. 4. Since $\delta\mathbf{a}_{n}$ is arbitrary, the coefficient of $\delta\mathbf{a}^{T}_{n}$ in Eq. 5.9 must be zero. As a result, the following expression for $\ddot{\mathbf{a}}_{n}$ is obtained:

$$\ddot{\mathbf{a}}_{n} = \mathbf{R}^{z}_{n(n-1)}\dot{\hat{\mathbf{Y}}}_{n(n-1)} + \mathbf{R}^{a}_{n(n-1)} + \mathbf{R}^{c^{T}}_{n(n-1)}\lambda \tag{5.10}$$

where

$$\mathbf{R}^{z}_{n(n-1)} = -\mathbf{M}^{aa^{-1}}_{n}\mathbf{M}^{am}_{n}$$

$$\mathbf{R}^{a}_{n(n-1)} = \mathbf{M}^{aa^{-1}}_{n}\mathbf{Q}^{a}_{n}$$

$$\mathbf{R}^{c^{T}}_{n(n-1)} = -\mathbf{M}^{aa^{-1}}_{n}\Gamma^{T}_{n(n+1)}\Phi^{T}_{\mathbf{Z}_{n(n+1)}} \tag{5.11}$$

Note that superscript n(n+1) for the Lagrange multiplier vector has been dropped, for notational convenience.

Substituting the modal acceleration of body n from Eq. 5.10 into Eq. 5.9,

$$\mathrm{EQM}(2)+\sum_{i=l}^{n-1}\{\delta\hat{\mathbf{Z}}^T_{i(i-1)}(\mathbf{M}^{mm}_i\dot{\hat{\mathbf{Y}}}_{i(i-1)}+\mathbf{M}^{ma}_i\ddot{\mathbf{a}}_i-\mathbf{Q}^z_i)+\delta\mathbf{a}^T_i(\mathbf{M}^{am}_i\dot{\hat{\mathbf{Y}}}_{i(i-1)}+\mathbf{M}^{aa}_i\ddot{\mathbf{a}}_i-\mathbf{Q}^a_i)\}$$

$$+\delta\hat{\mathbf{Z}}^T_{n(n-1)}(\mathbf{G}^z_{n(n-1)}\dot{\hat{\mathbf{Y}}}_{n(n-1)}-\mathbf{G}^q_{n(n-1)}+\mathbf{G}^{c^T}_{n(n-1)}\lambda)=0 \qquad (5.12)$$

where

$$\mathbf{G}^z_{n(n-1)}=\mathbf{M}^{mm}_n+\mathbf{M}^{ma}_n\mathbf{R}^z_{n(n-1)}$$
$$\mathbf{G}^q_{n(n-1)}=\mathbf{Q}^z_n-\mathbf{M}^{ma}_n\mathbf{R}^a_{n(n-1)}$$
$$\mathbf{G}^{c^T}_{n(n-1)}=\mathbf{\Phi}^T_{\mathbf{Z}_{n(n+1)}}+\mathbf{M}^{ma}_n\mathbf{R}^{c^T}_{n(n-1)} \qquad (5.13)$$

and Eqs. 5.9 and 5.12 have the same kinematic admissibility conditions.

The variational equations of motion can be reduced further by substituting $\delta\hat{\mathbf{Z}}_{n(n-1)}$ and $\dot{\hat{\mathbf{Y}}}_{n(n-1)}$ in terms of $\delta\hat{\mathbf{Z}}_{(n-1)n}$, $\delta\mathbf{q}_{(n-1)n}$, $\dot{\hat{\mathbf{Y}}}_{(n-1)n}$, and $\ddot{\mathbf{q}}_{(n-1)n}$, employing Eqs. 2.13 and 2.14. Equation 5.12 thus becomes

$$\mathrm{EQM}(2)+\sum_{i=l}^{n-1}\{\delta\hat{\mathbf{Z}}^T_{i(i-1)}(\mathbf{M}^{mm}_i\dot{\hat{\mathbf{Y}}}_{i(i-1)}+\mathbf{M}^{ma}_i\ddot{\mathbf{a}}_i-\mathbf{Q}^z_i)+\delta\mathbf{a}^T_i(\mathbf{M}^{am}_i\dot{\hat{\mathbf{Y}}}_{i(i-1)}+\mathbf{M}^{aa}_i\ddot{\mathbf{a}}_i-\mathbf{Q}^a_i)\}$$

$$+\delta\hat{\mathbf{Z}}^T_{(n-1)n}\{\mathbf{G}^z_{n(n-1)}\dot{\hat{\mathbf{Y}}}_{(n-1)n}+\mathbf{G}^z_{n(n-1)}\mathbf{\Pi}_{(n-1)n}\ddot{\mathbf{q}}_{(n-1)n}$$
$$+\mathbf{G}^z_{n(n-1)}\mathbf{\Theta}_{(n-1)n}-\mathbf{G}^q_{n(n-1)}+\mathbf{G}^{c^T}_{n(n-1)}\lambda\}$$
$$+\delta\mathbf{q}^T_{(n-1)n}\mathbf{\Pi}^T_{(n-1)n}\{\mathbf{G}^z_{n(n-1)}\dot{\hat{\mathbf{Y}}}_{(n-1)n}+\mathbf{G}^z_{n(n-1)}\mathbf{\Pi}_{(n-1)n}\ddot{\mathbf{q}}_{(n-1)n}$$
$$+\mathbf{G}^z_{n(n-1)}\mathbf{\Theta}_{(n-1)n}-\mathbf{G}^q_{n(n-1)}+\mathbf{G}^{c^T}_{n(n+1)}\lambda\}=0 \qquad (5.14)$$

which must hold for all virtual displacements that satisfy constraints inboard of body n-1. Since the kinematic relationship for joint (n-1,n) has been substituted into the equations of motion, $\delta\mathbf{q}_{(n-1)n}$ is arbitrary; i.e., the coefficient of $\delta\mathbf{q}_{(n-1)n}$ in Eq. 5.14 must be zero, which gives

$$\ddot{\mathbf{q}}_{(n-1)n}=\mathbf{R}^z_{(n-1)n}\dot{\hat{\mathbf{Y}}}_{(n-1)n}+\mathbf{R}^a_{(n-1)n}+\mathbf{R}^{c^T}_{(n-1)n}\lambda \qquad (5.15)$$

where

$$\mathbf{R}^z_{(n-1)n}=-(\mathbf{\Pi}^T_{(n-1)n}\mathbf{G}^z_{n(n-1)}\mathbf{\Pi}_{(n-1)n})^{-1}\mathbf{\Pi}^T_{(n-1)n}\mathbf{G}^z_{n(n-1)}$$
$$\mathbf{R}^a_{(n-1)n}=-(\mathbf{\Pi}^T_{(n-1)n}\mathbf{G}^z_{n(n-1)}\mathbf{\Pi}_{(n-1)n})^{-1}\mathbf{\Pi}^T_{(n-1)n}(\mathbf{G}^z_{n(n-1)}\mathbf{\Theta}_{n(n-1)}-\mathbf{G}^q_{n(n-1)})$$
$$\mathbf{R}^{c^T}_{(n-1)n}=-(\mathbf{\Pi}^T_{(n-1)n}\mathbf{G}^z_{n(n-1)}\mathbf{\Pi}_{(n-1)n})^{-1}\mathbf{\Pi}^T_{(n-1)n}\mathbf{G}^{c^T}_{n(n-1)} \qquad (5.16)$$

where existence of $(\mathbf{\Pi}^T_{(n-1)n}\mathbf{G}^z_{n(n-1)}\mathbf{\Pi}_{(n-1)n})^{-1}$ is proved in Ref. 11. Note that the subscripts of $\mathbf{R}^z$, $\mathbf{R}^a$, and $\mathbf{R}^c$ in Eq. 5.16 are in ascending order which are different from Eq. 5.11.

Substituting the relative joint acceleration of Eq. 5.15 into Eq. 5.14,

$$\mathrm{EQM}(2)+\sum_{i=l}^{n-1}\{\delta\hat{Z}^{T}_{i(i-1)}(M_i^{mm}\dot{\hat{Y}}_{i(i-1)}+M_i^{ma}\ddot{a}_i-Q_i^{z})+\delta a_i^{T}(M_i^{am}\dot{\hat{Y}}_{i(i-1)}+M_i^{aa}\ddot{a}_i-Q_i^{a})\}$$

$$+\,\delta\hat{Z}^{T}_{(n-1)n}\{G^{z}_{(n-1)n}\dot{\hat{Y}}_{(n-1)n}-G^{q}_{(n-1)n}+G^{c^{T}}_{(n-1)n}\lambda\}=0 \qquad (5.17)$$

where

$$G^{z}_{(n-1)n}=G^{z}_{n(n-1)}+G^{z}_{n(n-1)}\Pi_{(n-1)n}R^{z}_{(n-1)n}$$
$$G^{q}_{(n-1)n}=G^{q}_{n(n-1)}-G^{z}_{n(n-1)}\Theta_{(n-1)n}-G^{z}_{n(n-1)}\Pi_{(n-1)n}R^{a}_{(n-1)n}$$
$$G^{c^{T}}_{(n-1)n}=G^{c^{T}}_{n(n-1)}+G^{z}_{n(n-1)}\Pi_{(n-1)n}R^{c^{T}}_{(n-1)n} \qquad (5.18)$$

which must hold for all virtual displacements that satisfy the same kinematic admissibility conditions as Eq. 5.14. Note that the subscripts of G^{z}, G^{q}, and G^{c} in Eq. 5.18 are in ascending order which are different from Eq. 5. 13.

By employing the recursive relationships between inboard and outboard joint frames of Eqs. 2.16 and 2.17, the variational equations of motion can be reduced to

$$\mathrm{EQM}(2)+\sum_{i=l}^{n-2}\{\delta\hat{Z}^{T}_{i(i-1)}(M_i^{mm}\dot{\hat{Y}}_{i(i-1)}+M_i^{am}\ddot{a}_i-Q_i^{z})+\delta a_i^{T}(M_i^{am}\dot{\hat{Y}}_{i(i-1)}+M_i^{aa}\ddot{a}_i-Q_i^{a})\}$$

$$+\,\delta\hat{Z}^{T}_{(n-1)(n-2)}\{G^{z}_{(n-1)n}(\dot{\hat{Y}}_{(n-1)(n-2)}+\Gamma_{(n-1)n}\ddot{a}_{n-1}+\Delta_{(n-1)n})$$

$$-G^{q}_{(n-1)n}+G^{c^{T}}_{(n-1)n}\lambda+M^{mm}_{n-1}\dot{\hat{Y}}_{(n-1)(n-2)}+M^{ma}_{n-1}\ddot{a}_{n-1}-Q^{z}_{n-1}\}$$

$$+\,\delta a^{T}_{n-1}[\Gamma^{T}_{(n-1)n}\{G^{z}_{(n-1)n}(\dot{\hat{Y}}_{(n-1)(n-2)}+\Gamma_{(n-1)n}\ddot{a}_{n-1}+\Delta_{(n-1)n})$$

$$-G^{q}_{(n-1)n}+G^{c^{T}}_{(n-1)n}\lambda\}+M^{am}_{n-1}\dot{\hat{Y}}_{(n-1)(n-2)}+M^{aa}_{n-1}\ddot{a}_{n-1}-Q^{a}_{n-1}]=0 \qquad (5.19)$$

which must hold for all virtual displacements that are consistent with tree structure constraints inboard of joint (n-2,n-1). Since δa_{n-1} must be arbitrary, the coefficient of δa_{n-1}^{T} must be zero, which yields

$$\ddot{a}_{n-1}=R^{z}_{(n-1)(n-2)}\dot{\hat{Y}}_{(n-1)(n-2)}+R^{a}_{(n-1)(n-2)}+R^{c^{T}}_{(n-1)(n-2)}\lambda \qquad (5.20)$$

where

$$R^{z}_{(n-1)(n-2)}=-(M^{aa}_{n-1}+\Gamma^{T}_{(n-1)n}G^{z}_{(n-1)n}\Gamma_{(n-1)n})^{-1}(\Gamma^{T}_{(n-1)n}G^{z}_{(n-1)n}+M^{am}_{n-1})$$
$$R^{a}_{(n-1)(n-2)}=-(M^{aa}_{n-1}+\Gamma^{T}_{(n-1)n}G^{z}_{(n-1)n}\Gamma_{(n-1)n})^{-1}\{\Gamma^{T}_{(n-1)n}(G^{z}_{(n-1)n}\Delta_{(n-1)n}+G^{q}_{(n-1)n})-Q^{a}_{n-1}\}$$
$$R^{c^{T}}_{(n-1)(n-2)}=-(M^{aa}_{n-1}+\Gamma^{T}_{(n-1)n}G^{z}_{(n-1)n}\Gamma_{(n-1)n})^{-1}(\Gamma^{T}_{(n-1)n}G^{c^{T}}_{(n-1)n}) \qquad (5.21)$$

where existence of $(M^{aa}_{n-1}+\Gamma^{T}_{(n-1)n}G^{z}_{(n-1)n}\Gamma_{(n-1)n})^{-1}$ is proved in Ref. 11.

Substituting the modal acceleration of body n-1 from Eq. 5.20 into Eq. 5.19,

$$\mathrm{EQM}(2) + \sum_{i=l}^{n-2}\{\delta\hat{\mathbf{Z}}^{T}_{i(i-1)}(\mathbf{M}^{mm}_{i}\dot{\hat{\mathbf{Y}}}_{i(i-1)}+\mathbf{M}^{am}_{i}\ddot{\mathbf{a}}_{i}-\mathbf{Q}^{z}_{i})+\delta\mathbf{a}^{T}_{i}(\mathbf{M}^{am}_{i}\dot{\hat{\mathbf{Y}}}_{i(i-1)}+\mathbf{M}^{aa}_{i}\ddot{\mathbf{a}}_{i}-\mathbf{Q}^{a}_{i})\}$$

$$+\ \delta\hat{\mathbf{Z}}^{T}_{(n-1)(n-2)}\{\mathbf{G}^{z}_{(n-1)(n-2)}\dot{\hat{\mathbf{Y}}}_{(n-1)(n-2)}-\mathbf{G}^{q}_{(n-1)(n-2)}+\mathbf{G}^{c^{T}}_{(n-1)(n-2)}\lambda\} = 0 \qquad (5.22)$$

where

$$\mathbf{G}^{z}_{(n-1)(n-2)} = \mathbf{G}^{z}_{(n-1)n}+\mathbf{M}^{mm}_{n-1}+(\mathbf{G}^{z}_{(n-1)n}\Gamma_{(n-1)n}+\mathbf{M}^{ma}_{n-1})\mathbf{R}^{z}_{(n-1)(n-2)}$$
$$\mathbf{G}^{q}_{(n-1)(n-2)} = \mathbf{G}^{q}_{(n-1)n}+\mathbf{Q}^{z}_{n-1}-(\mathbf{G}^{z}_{(n-1)n}\Gamma_{(n-1)n}+\mathbf{M}^{ma}_{n-1})\mathbf{R}^{a}_{(n-1)(n-2)}$$
$$\mathbf{G}^{c^{T}}_{(n-1)(n-2)} = \mathbf{G}^{c^{T}}_{(n-1)n}+(\mathbf{G}^{z}_{(n-1)n}\Gamma_{(n-1)n}+\mathbf{M}^{ma}_{n-1})\mathbf{R}^{c^{T}}_{(n-1)(n-2)} \qquad (5.23)$$

If the reduction procedure is continued to the junction body l for chains 1 and 2, the following reduced variational equation of motions is obtained:

$$\delta\hat{\mathbf{Z}}^{T}_{l(l-1)}\{(\mathbf{G}^{z}_{l(l+1)}+\mathbf{G}^{z}_{lm})\dot{\hat{\mathbf{Y}}}_{l(l-1)}+(\mathbf{G}^{z}_{l(l+1)}\Gamma_{l(l+1)}+\mathbf{G}^{z}_{lm}\Gamma_{lm})\ddot{\mathbf{a}}_{l}+(\mathbf{G}^{z}_{l(l+1)}\Delta_{l(l+1)}+\mathbf{G}^{z}_{lm}\Delta_{lm})$$

$$-(\mathbf{G}^{q}_{l(l+1)}+\mathbf{G}^{q}_{lm})+(\mathbf{G}^{c^{T}}_{l(l+1)}+\mathbf{G}^{c^{T}}_{lm})+\mathbf{M}^{mm}_{l}\dot{\hat{\mathbf{Y}}}_{l(l-1)}+\mathbf{M}^{ma}_{l}\ddot{\mathbf{a}}_{l}-\mathbf{Q}^{z}_{l}\}$$

$$+\ \delta\mathbf{a}^{T}_{l}\{(\Gamma^{T}_{l(l+1)}\mathbf{G}^{z}_{l(l+1)}+\Gamma^{T}_{lm}\mathbf{G}^{z}_{lm})\dot{\hat{\mathbf{Y}}}_{l(l-1)}+(\Gamma^{T}_{l(l+1)}\mathbf{G}^{z}_{l(l+1)}\Gamma_{l(l+1)}+\Gamma^{T}_{lm}\mathbf{G}^{z}_{lm}\Gamma_{lm})\ddot{\mathbf{a}}_{l}$$

$$+(\Gamma^{T}_{l(l+1)}\mathbf{G}^{z}_{l(l+1)}\Delta_{l(l+1)}+\Gamma^{T}_{lm}\mathbf{G}^{z}_{lm}\Delta_{lm})-(\Gamma^{T}_{l(l+1)}\mathbf{G}^{q}_{l(l+1)}+\Gamma^{T}_{lm}\mathbf{G}^{q}_{lm})+(\Gamma^{T}_{l(l+1)}\mathbf{G}^{c^{T}}_{l(l-1)}$$

$$\Gamma^{T}_{lm}\mathbf{G}^{c^{T}}_{lm})\lambda+\mathbf{M}^{am}_{l}\dot{\hat{\mathbf{Y}}}_{l(l-1)}+\mathbf{M}^{aa}_{l}\ddot{\mathbf{a}}_{l}-\mathbf{Q}^{a}_{l}\} = 0 \qquad (5.24)$$

Since $\delta\mathbf{a}_{l}$ is arbitrary, the modal acceleration of junction body l can be determined as

$$\ddot{\mathbf{a}}_{l} = \mathbf{R}^{z}_{l(l+1)}\dot{\hat{\mathbf{Y}}}_{l(l-1)} + \mathbf{R}^{a}_{l(l-1)} + \mathbf{R}^{c^{T}}_{l(l-1)}\lambda \qquad (5.25)$$

where

$$\mathbf{R}^{z}_{l(l-1)} = -(\mathbf{M}^{aa}_{l}+\Gamma^{T}_{l(l+1)}\mathbf{G}^{z}_{l(l+1)}\Gamma_{l(l+1)}+\Gamma^{T}_{lm}\mathbf{G}^{z}_{lm}\Gamma_{lm})^{-1}(\mathbf{M}^{am}_{l}+\Gamma^{T}_{l(l+1)}\mathbf{G}^{z}_{l(l+1)}+\Gamma^{T}_{lm}\mathbf{G}^{z}_{lm})$$
$$\mathbf{R}^{a}_{l(l-1)} = -(\mathbf{M}^{aa}_{l}+\Gamma^{T}_{l(l+1)}\mathbf{G}^{z}_{l(l+1)}\Gamma_{l(l+1)}+\Gamma^{T}_{lm}\mathbf{G}^{z}_{lm}\Gamma_{lm})^{-1}[\Gamma^{T}_{l(l+1)}\{\mathbf{G}^{z}_{l(l+1)}\Delta_{l(l+1)}+\mathbf{G}^{q}_{l(l+1)})$$
$$+\Gamma^{T}_{l(l+1)}(\mathbf{G}^{z}_{lm}\Delta_{lm}+\mathbf{G}^{q}_{lm})-\mathbf{Q}^{a}_{l}]$$
$$\mathbf{R}^{c^{T}}_{l(l-1)} = -(\mathbf{M}^{aa}_{l}+\Gamma^{T}_{l(l+1)}\mathbf{G}^{z}_{l(l+1)}\Gamma_{l(l+1)}+\Gamma^{T}_{lm}\mathbf{G}^{z}_{lm}\Gamma_{lm})^{-1}(\Gamma^{T}_{l(l+1)}\mathbf{G}^{c^{T}}_{l(l+1)}+\Gamma^{T}_{lm}\mathbf{G}^{c^{T}}_{lm}) \qquad (5.26)$$

Substituting the modal acceleration of the junction body from Eq. 5.25 into Eq. 5.24,

$$\delta\hat{\mathbf{Z}}^{T}_{l(l-1)}(\ \mathbf{G}^{z}_{l(l-1)}\dot{\hat{\mathbf{Y}}}_{l(l-1)} - \mathbf{G}^{q}_{l(l-1)} + \mathbf{G}^{c^{T}}_{l(l-1)}\lambda\ ) = 0 \qquad (5.27)$$

which must hold for all kinematic constraints acting on junction body l. Terms arising in Eq. 5.27 are as follows:

$$\mathbf{G}^{z}_{l(l-1)} = (\mathbf{G}^{z}_{l(l+1)}+\mathbf{G}^{z}_{lm})+\mathbf{M}^{mm}_{l}+(\mathbf{G}^{z}_{l(l+1)}\Gamma_{l(l+1)}+\mathbf{G}^{z}_{lm}\Gamma_{lm}+\mathbf{M}^{ma}_{l})\mathbf{R}^{z}_{l(l-1)}$$
$$\mathbf{G}^{q}_{l(l-1)} = (\Pi_{l(l+1)}+\Pi_{lm})+\mathbf{Q}^{z}_{l}+(\mathbf{G}^{z}_{l(l+1)}\Gamma_{l(l+1)}+\mathbf{G}^{z}_{lm}\Gamma_{lm}+\mathbf{M}^{ma}_{l})\mathbf{R}^{a}_{l(l-1)}$$
$$\mathbf{G}^{c^{T}}_{l(l-1)} = (\mathbf{G}^{c^{T}}_{l(l+1)}+\mathbf{G}^{c^{T}}_{lm})+(\mathbf{G}^{z}_{l(l+1)}\Gamma_{l(l+1)}+\mathbf{G}^{z}_{lm}\Gamma_{lm}+\mathbf{M}^{ma}_{l})\mathbf{R}^{c^{T}}_{l(l-1)} \qquad (5.28)$$

6. Cut Joint Constraint Acceleration Equations

In addition to the equations of motion, cut joint constraints must be used to obtain the same number of equations as unknown accelerations and multipliers. Cut joint constraints may be differentiated twice to obtain the constraint acceleration equation,

$$\ddot{\Phi}^{(n,n+1)} = \Phi^{T}_{Z_{n(n+1)}} \dot{\hat{Y}}_{n(n+1)} + \Phi^{T}_{Z_{(n+1)n}} \dot{\hat{Y}}_{(n+1)n} - \gamma = 0 \qquad (6.1)$$

where γ is the collection of all terms that do not include $\dot{\hat{Y}}_{n(n+1)}$ and $\dot{\hat{Y}}_{(n+1)n}$. Superscript (n,n+1), which has been used to denote the cut constraint between bodies n and n+1, is omitted for notational convenience in the derivation. Accelerations $\dot{\hat{Y}}_{n(n+1)}$ and $\dot{\hat{Y}}_{(n+1)n}$ from Eq. 2.17 are substituted into Eq. 6.1, to yield

$$\begin{aligned}&\Phi^{T}_{Z_{n(n+1)}}(\dot{\hat{Y}}_{n(n-1)}+\Gamma_{n(n+1)}\ddot{a}_{n}+\Delta_{n(n+1)})\\ &+\Phi^{T}_{Z_{(n+1)n}}(\dot{\hat{Y}}_{(n+1)(n+2)}+\Gamma_{(n+1)(n+2)}\ddot{a}_{n+1}+\Gamma_{(n+1)(n+2)})-\gamma=0\end{aligned} \qquad (6.2)$$

Substituting $\ddot{a}_n$ from Eq. 5.10 and $\ddot{a}_{n+1}$, obtained by advancing subscripts in Eq. 5.10, into Eq. 6.2 yields

$$\begin{aligned}&G^{c}_{n(n-1)n}\dot{\hat{Y}}_{n(n-1)}+G^{c}_{(n+1)(n+2)}\dot{\hat{Y}}_{(n+1)(n+2)}+(L^{T}_{n(n-1)}+L^{T}_{(n+1)(n+2)})\lambda\\ &\qquad -N_{n(n-1)}-N_{(n+1)(n+2)}-\gamma=0\end{aligned} \qquad (6.3)$$

where

$$\begin{aligned}L^{T}_{n(n-1)}&=\Phi^{T}_{Z_{n(n+1)}}\Gamma_{n(n+1)}R^{c^{T}}_{n(n-1)}\\ N_{n(n-1)}&=-\Phi^{T}_{Z_{n(n+1)}}(\Delta_{n(n+1)}+\Gamma_{n(n+1)}R^{a}_{n(n-1)})\end{aligned} \qquad (6.4)$$

and $L_{(n+1)(n+2)}$ and $N_{(n+1)(n+2)}$ are obtained by substituting appropriate subscripts into Eq. 6.4.

The recursive relationship between triple primed frames of joints (n-1,n) and (n+2,n+1), obtained from Eq. 2.16, are next substituted into Eq. 6.3, to give

$$\begin{aligned}&G_{n(n-1)}(\dot{\hat{Y}}_{(n-1)n}+\Pi_{(n-1)n}\ddot{q}_{(n-1)n}+\Theta_{(n-1)n})+G^{c}_{(n+1)(n+2)}(\dot{\hat{Y}}_{(n+2)(n+1)}+\Pi_{(n+2)(n+1)}\ddot{q}_{(n+2)(n+1)}\\ &\qquad+\Theta_{(n+2)(n+1)})+(L^{T}_{n(n-1)}+L^{T}_{(n+1)(n+2)})\lambda-N_{n(n-1)}-N_{(n+1)(n+2)}-\gamma=0\end{aligned} \qquad (6.5)$$

If the relative joint accelerations $\ddot{q}_{(n-1)n}$ from Eq. 5.15 and $\ddot{q}_{(n+2)(n+1)}$, obtained by replacing subscripts of Eq. 5.15, are substituted into Eq. 6.5,

$$G_{(n-1)n}\dot{\hat{Y}}_{(n-1)n}+G_{(n+2)(n+1)}\dot{\hat{Y}}_{(n+2)(n+1)}+(L^{T}_{(n-1)n}+L^{T}_{(n+2)(n+1)})\lambda-N_{(n-1)n}-N_{(n+2)(n+1)}=0 \qquad (6.6)$$

where

$$\mathbf{N}_{(n-1)n} = \mathbf{N}_{n(n-1)} + \mathbf{G}^{c}_{n(n-1)}(\mathbf{\Theta}_{(n-1)n}+\mathbf{\Pi}_{(n-1)n}\mathbf{R}^{a}_{(n-1)n})$$

$$\mathbf{L}^{T}_{(n-1)n} = \mathbf{L}^{T}_{n(n-1)} + \mathbf{G}_{n(n-1)}\mathbf{\Pi}_{(n-1)n}\mathbf{R}^{c^{T}}_{(n-1)n} \tag{6.7}$$

and $\mathbf{L}_{(n+2)(n+1)}$ and $\mathbf{N}_{(n+2)(n+1)}$ are obtained by replacing (n-1,n) by (n+2,n+1) in Eq. 6.7.

If this sequence of elimination of modal and relative joint accelerations is repeated to junction body *l*, the following reduced constraint acceleration equations are obtained as

$$\mathbf{G}_{l(l-1)}\dot{\hat{\mathbf{Y}}}_{l(l-1)} + \mathbf{L}^{T}_{l(l-1)}\boldsymbol{\lambda} - \mathbf{N}_{l(l-1)} - \boldsymbol{\gamma} = \mathbf{0} \tag{6.8}$$

where

$$\mathbf{L}^{T}_{l(l-1)} = (\mathbf{L}^{T}_{l(l+1)}+\mathbf{L}^{T}_{lm}) + (\mathbf{G}_{l(l+1)}\mathbf{\Gamma}_{l(l+1)}+\mathbf{G}_{lm}\mathbf{\Gamma}_{lm})\mathbf{R}^{c^{T}}_{l(l-1)}$$

$$\mathbf{N}_{l(l-1)} = (\mathbf{N}_{l(l+1)}+\mathbf{N}_{lm}) - (\mathbf{G}_{l(l+1)}\mathbf{\Gamma}_{l(l+1)}+\mathbf{G}_{lm}\mathbf{\Gamma}_{lm})\mathbf{R}^{a}_{l(l-1)} - (\mathbf{G}_{l(l+1)}\mathbf{\Delta}_{l(l+1)}+\mathbf{G}_{lm}\mathbf{\Delta}_{lm}) \tag{6.9}$$

7. Base Body Equation of Motion

A single closed loop subsystem is used to derive decoupled recursive equations of motion in Sections 5 and 6. The variational equations of motion were reduced to the inboard joint reference frame of the junction body. Since the base body does not have an inboard joint, the inboard joint reference frame of the base body is assumed to coincide with the base body reference frame. If the reduction procedures that have been carried out with this subsystem are repeated along all chains of a system to the base body, the base body equation of motion is obtained as

$$\delta\hat{\mathbf{Z}}^{T}_{b}(\mathbf{G}^{z}_{bb}\dot{\hat{\mathbf{Y}}}_{bb}-\mathbf{G}^{q}_{bb}+\mathbf{G}^{c^{T}}_{bb}\boldsymbol{\lambda}) = 0 \tag{7.1}$$

where $\delta\hat{\mathbf{Z}}_{b}$ is arbitrary for a floating base body, which yields

$$\mathbf{G}^{z}_{bb}\dot{\hat{\mathbf{Y}}}_{bb}-\mathbf{G}^{q}_{bb}+\mathbf{G}^{c^{T}}_{bb}\boldsymbol{\lambda} = \mathbf{0} \tag{7.2}$$

Reduction of cut joint constraint acceleration equations to the base body yields the reduced constraint acceleration equations as

$$\mathbf{G}^{c^{T}}_{bb}\dot{\hat{\mathbf{Y}}}_{b}+\mathbf{L}^{T}_{bb}\boldsymbol{\lambda}-\mathbf{N}_{bb} = \mathbf{0} \tag{7.3}$$

Equation 7.2 may be combined with Eq. 7.3, to form the augmented base body equations of motion,

$$\begin{bmatrix} \mathbf{G}^{z}_{bb} & \mathbf{G}^{c^{T}}_{bb} \\ \mathbf{G}^{c}_{bb} & \mathbf{L}^{T}_{bb} \end{bmatrix} \begin{bmatrix} \dot{\hat{\mathbf{Y}}}_{bb} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathbf{G}^{q}_{bb} \\ \mathbf{N}_{bb} \end{bmatrix} \tag{7.4}$$

In the case of a constrained base body, a Lagrange multiplier vector $\boldsymbol{\lambda}^{b}$ and corresponding constraint accelerations are introduced into Eq. 7.4. The resulting augmented equation of motion is

$$\begin{bmatrix} G^{Z}_{bb} & G^{c^{T}}_{bb} & \Phi^{b^{T}}_{\hat{Z}_{bb}} \\ G^{c}_{bb} & L^{T}_{bb} & 0 \\ \Phi^{b}_{\hat{Z}_{bb}} & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{\hat{Y}}_{bb} \\ \lambda \\ \lambda^{b} \end{bmatrix} = \begin{bmatrix} G^{q}_{bb} \\ N_{bb} \\ \gamma^{b} \end{bmatrix} \tag{7.5}$$

where Φ^b is constraint equations acting on the base body and γ^b is the collection of all terms that do not include $\dot{\hat{Y}}_b$ in $\ddot{\Phi}^b$.

Equation 7.4 or 7.5 is solved for the base body state acceleration vector and the Lagrange multiplier vector. Detailed computational algorithm is presented in Ref. 11.

8. Example Problem

A closed loop spatial robot that consists of two flexible and three rigid bodies is shown in Fig. 5. Bodies 3 and 5 are flexible beams with rectangular cross sections. All other bodies are treated as rigid bodies. Body 1 is connected with ground, which is designated as the base body, by a revolute joint. A lumped mass is attached to body 3 at point P to represent a payload for this robot. Joints 1, 2, 3, 4, and 5 are revolute joints. The connection between bodies 3 and 5 is a spherical joint (joint 6).

One generalized coordinate is assigned for each revolute joint, and three deformation modes have been chosen for each flexible. Joint 6 is defined as the cut joint to form a tree structure.

Inertial properties and geometric data are given in Table 1. For deformation mode computation, flexible beams are discretized in to 10 equal length 3 diemensional beam elements.

Simulation is carried out for 0.5 sec., with the following actuator torques applied at joints 1, 2, and 4, respectively:

$$n_1 = 5.0E9 \cdot \sin(0.2\pi t)$$
$$n_2 = 9.0E7 - 8.0E7 \cdot t$$
$$n_4 = 8.5E9 - 3.0E9 \cdot t \tag{8.1}$$

Results of the simulation have been verified using the three dimensional dynamic analysis program DADS [18], which employes a Cartesian coordinate formulation [13]. In the Cartesian coordinate formulation, 48 generalized coordinates and 40 constraint equations are needed to represent the system. However, only 11 generalized coordinates and 3 constraint equations are required for the recursive formulation presented here. The y coordinate, velocity, and acceleration of the origin of the body reference frame for body 3 are shown in Fig. 6. Both the DADS and recursive formulations yield the same results when implemented on a VAX 11/780 serial computer, which cannot exploit parallelism in the recursive algorithm. Table 2 shows the CPU time required for both methods and the ratio of CPU time between the two methods.

References

1. Armstrong, W.W., "Recursive Solution to the Equations of Motion of an N-Link Manipulator," Proc. 5th World Congress on Theory of Machines and Mechanism, Vol. 2, Montreal, 1979, pp. 1343-1346.

2. Featherstone, R., "The Calculation of Robot Dynamics Using Articulated-Body Inertias," The International J. of Robotics Research, Vol. 2, No. 1., 1983, pp. 13-30.

3. Haug, E.J., and McCullough, M.K., "A Variational-Vector Calculus Approach to Machine Dynamics," Journal of Mechanisms, Transmissions, and Automation in Design, Vol. 108, No. 1, 1986, pp. 25-30.

4. Bae, D.S., and Haug, E.J., " A Recursive Formulation for Constrained Mechanical System Dynamics: Part I, Open Loop Systems," Mechanics of Structures and Machines, Vol. 15, No. 3, 1987.

5. Wu, S.C., Haug, E.J., and Kim, S.S., "A Variational Approach to Dynamics of Flexible Multibody Systems," Mechanics of Structures and Machines, to appear.

6. Kim, S.S,. and Haug, E.J., "A Recursive Formulation for Flexible Multibody Dynamics: Part I, Open Loop Systems," Journal of Computer Methods in Engineering, to appear.

7. Haug, E. J., Lai, H. J., and Bae, D. S., "Kinematics of Flexible Multibody Systems," in preparation.

8. Bae, D.S., Hwang, R.S., and Haug, E.J., "A Recursive Formulation for Real-Time Dynamic Simulation," to appear.

9. Bottema, O., and Roth, B., Theoretical Kinematics, Amsterdam, North Holland, 1979.

10 Ball, R.S., A Treatise on the Theory of Screws, London, Cambridge University Press, 1900.

11. Lai, H. J., A Decoupled Recursive Approach for Flexible Multibody System Dynamics and Its Application in Parallel Computation, Ph.D. dissertation, The University of Iowa, Iowa City, Iowa.

13. Haug, E.J., Computer Aided Kinematics and Dynamics of Mechanical Systems, Allyn & Bacon, New York, 1989.

14. Kim, S.S., A Recursive Formulation for Flexible Multibody Body Dynamics, Ph.D. dissertation, The University of Iowa, Iowa City, Iowa, 1988.

16. Gear, C.W., Leimkuhler, B., and Gupta, G.K., "Automatic Integration of Euler-Lagrange Equations with Constraints," J. of Computation and Applied Mathematics, 12-13, 1985, pp. 77-90.

17. Yoo, W.S., and Haug, E.J., "Dynamics of Flexible Mechanical Systems Using Vibration and Static Correction Modes," Trans. ASME Journal of Mechanisms, Transmissions, and Automation in Design, Vol. 108, Sept. 1986, pp. 315-322.

18. DADS User's Manual, CADSI, P.O. Box 203, Oakdale, Iowa, 52319.

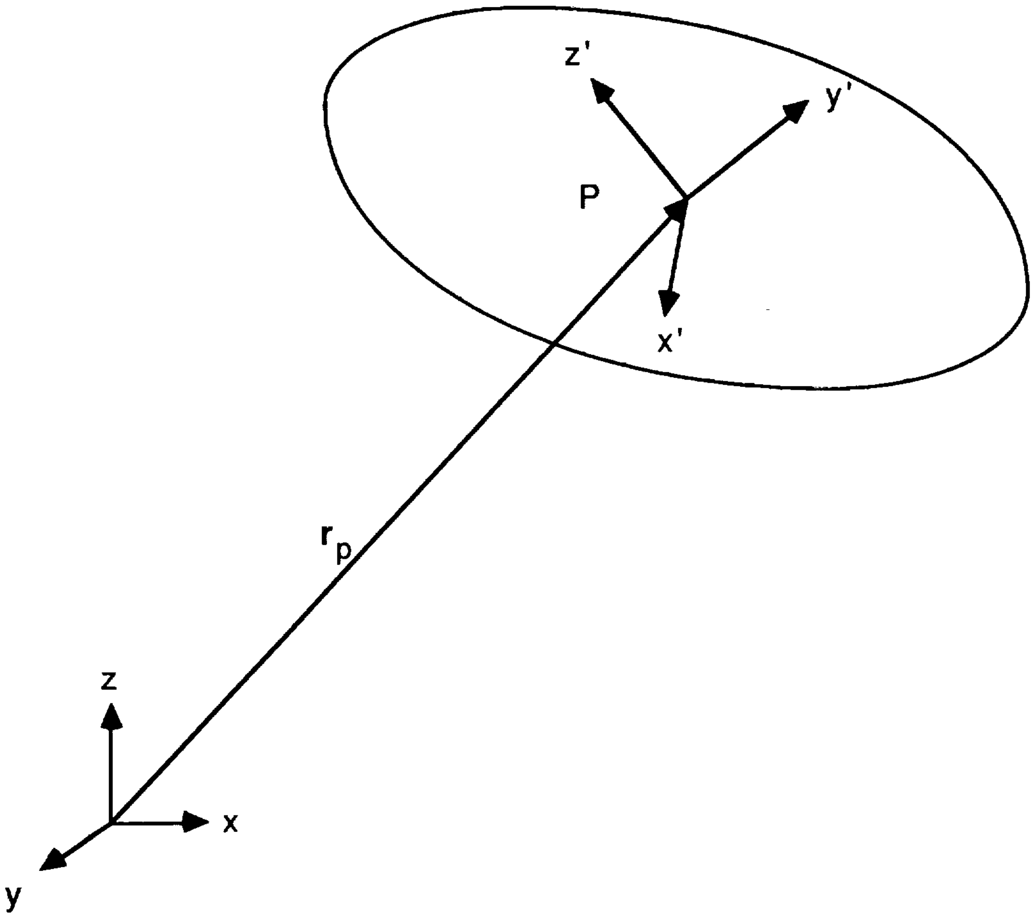


Figure 1. A Reference Frame

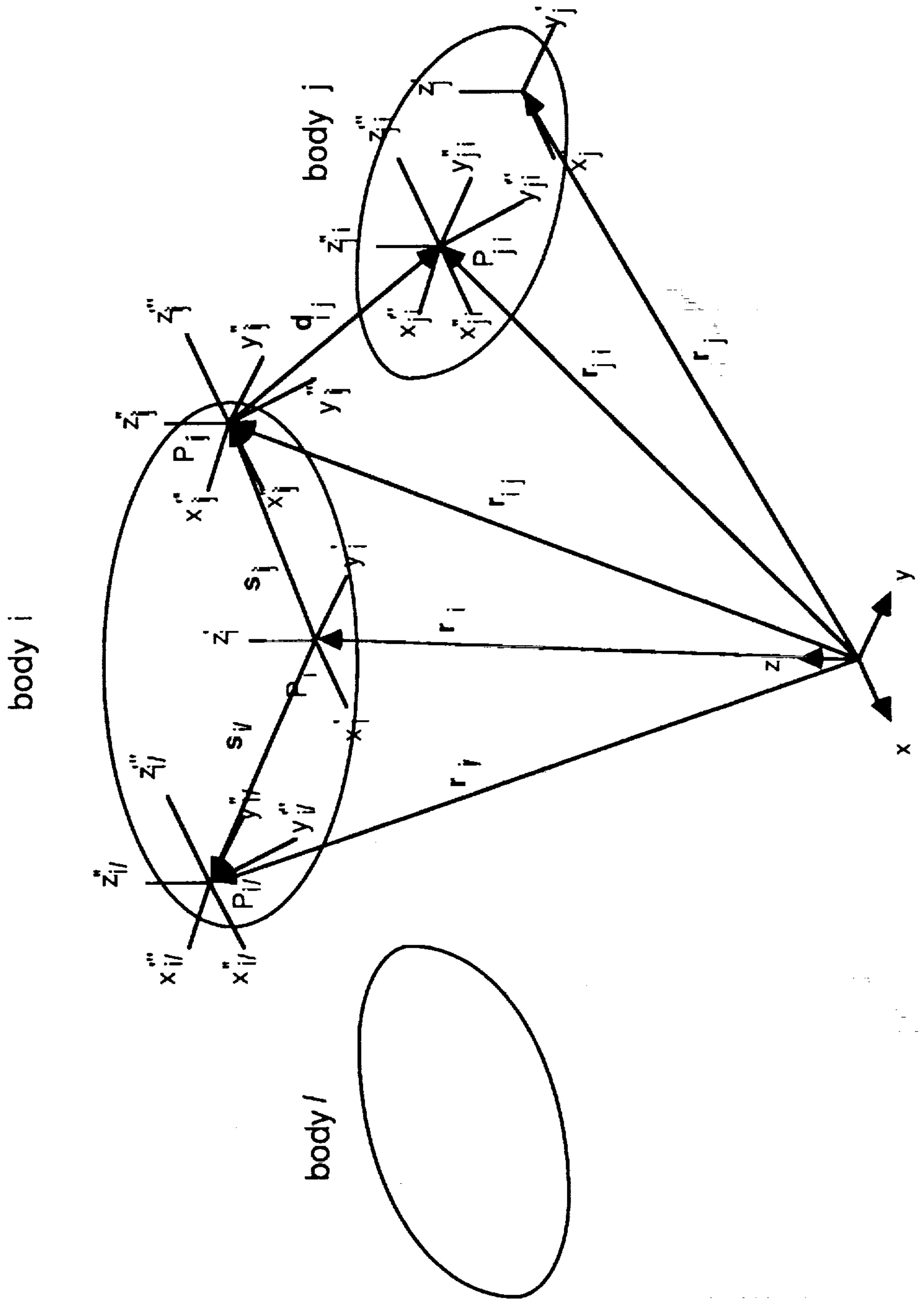


Figure. 2. Adjacent Flexible Bodies

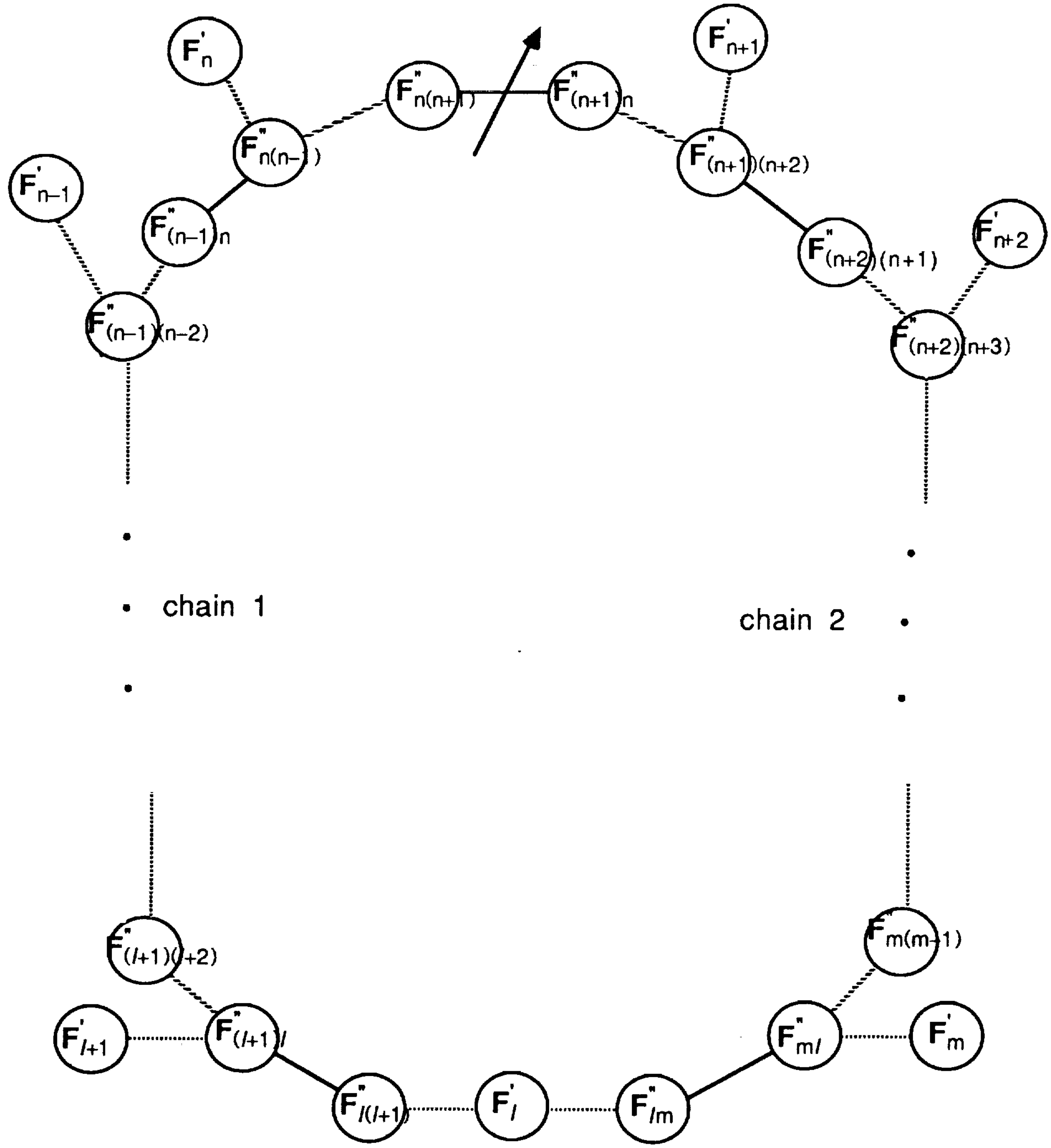


Figure 3. A Closed Loop Subsystem

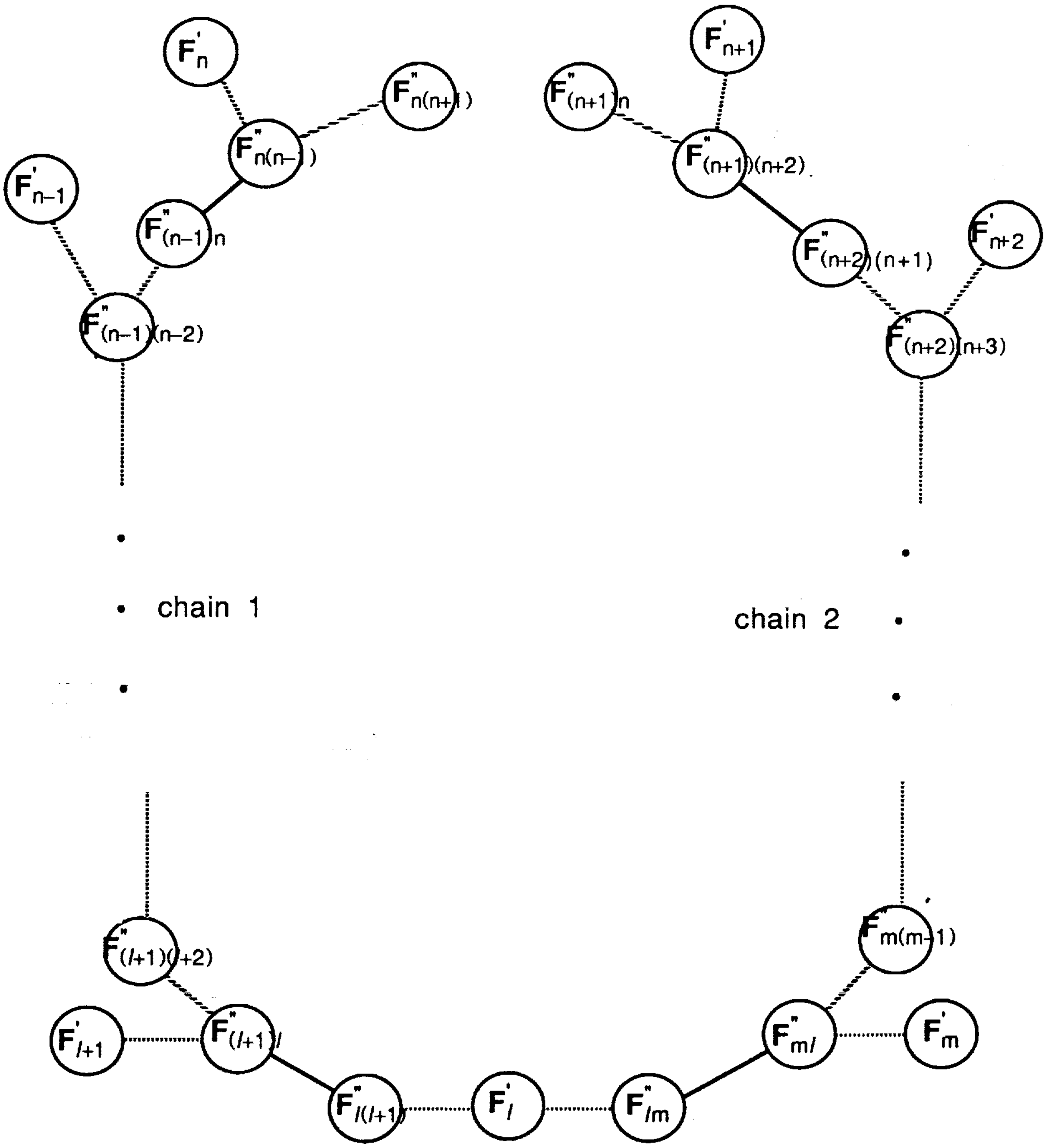


Figure 4. Spanning Tree for the Closed Loop Subsystem

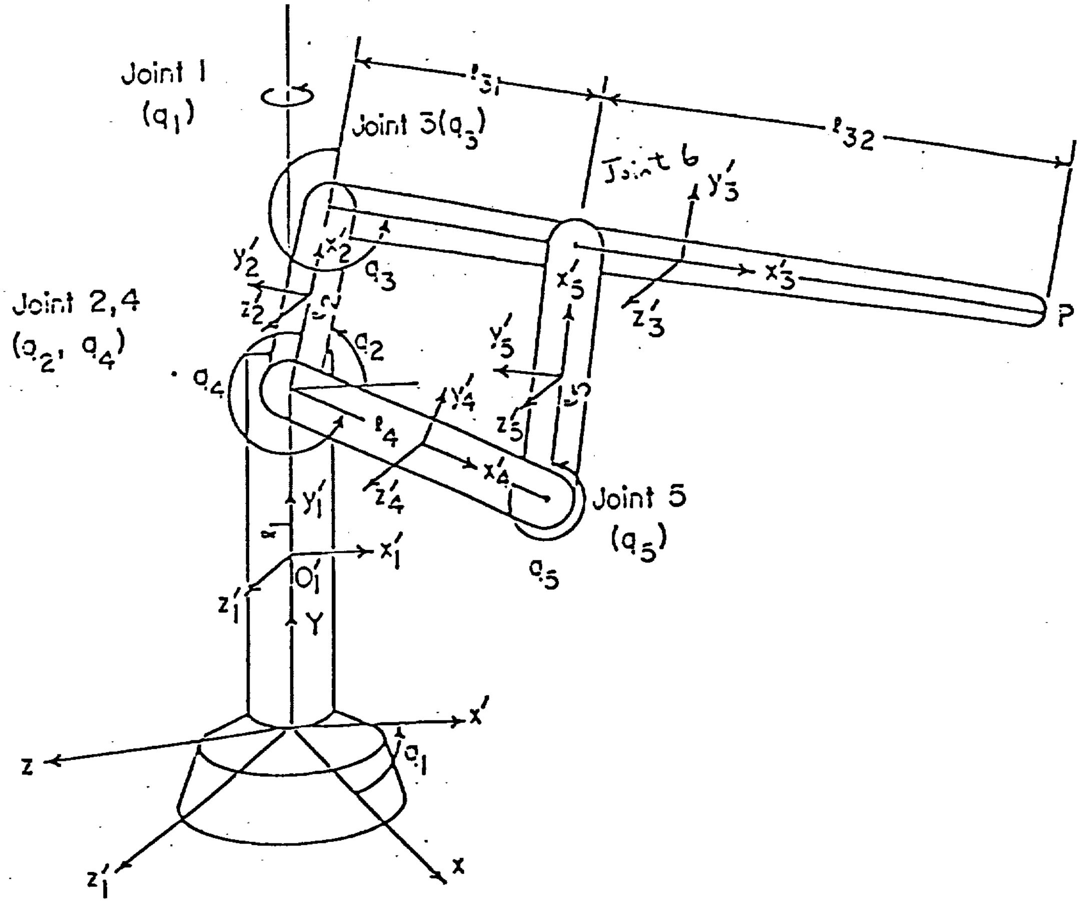


Figure 5. A Closed Loop Manipulator

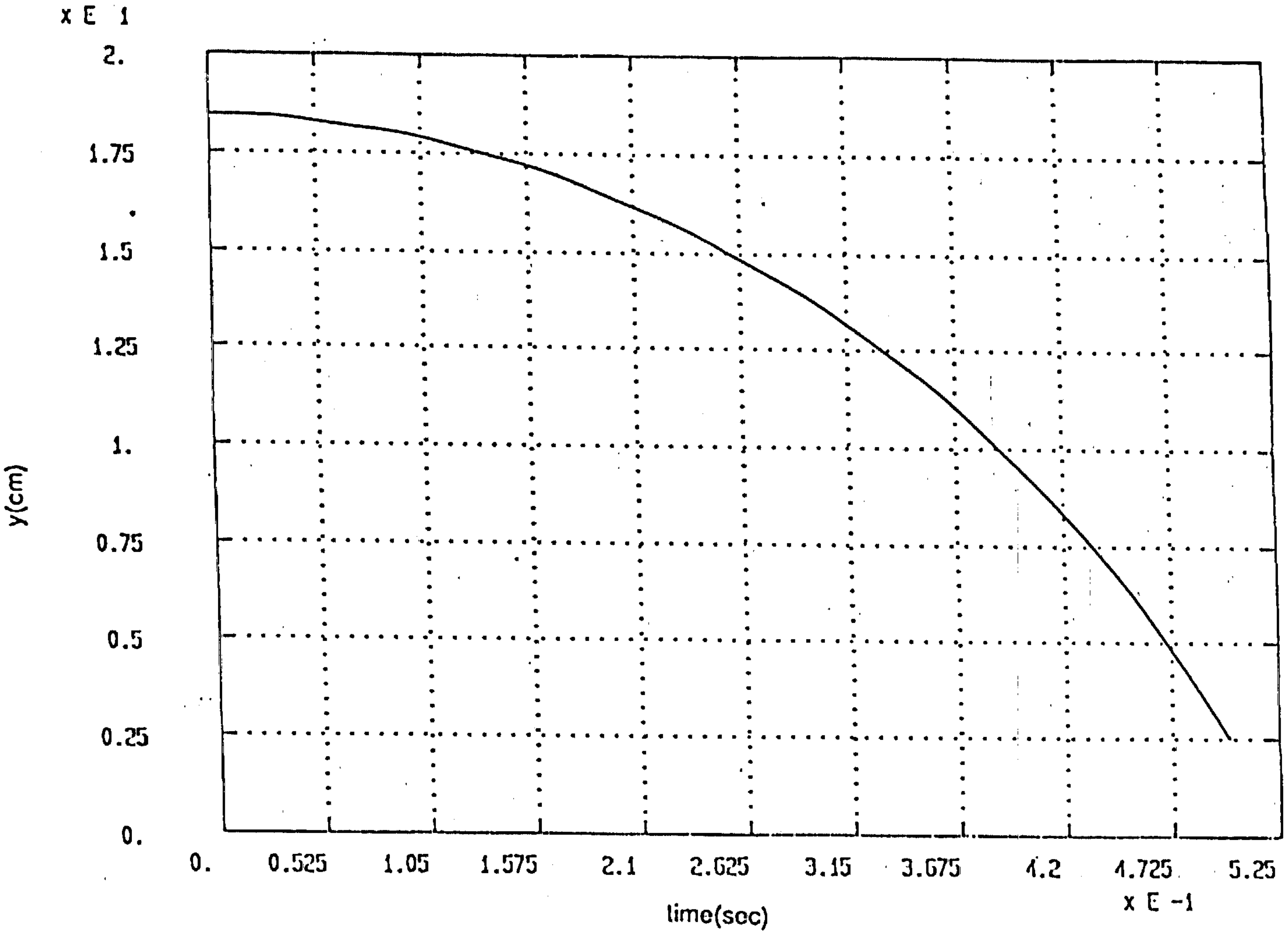


Figure 6. Y of Body 3

Table 1. Dimension and Inertia of the Closed Loop Manipulator Bodies

Body	Length (cm)	m (g)	I_{xx}	I_{yy}	I_{zz}	I_{xy}	I_{xz}	I_{yz}
1	ℓ_1 =100.0	6.1497×10^4	5.1632×10^7	7.6071×10^5	5.1632×10^7	0.0	0.0	0.0
2	ℓ_2 =37.5	1.1745×10^4	8.7108×10^4	1.4008×10^6	1.4390×10^6	0.0	0.0	0.0
3	ℓ_{31} =77.5 ℓ_{32} =57.5	4.2282×10^4	3.1359×10^5	6.4304×10^7	6.4441×10^7	0.0	0.0	0.0
4	ℓ_4 =132.5	4.1499×10^4	3.0778×10^5	6.0800×10^7	6.0935×10^7	0.0	0.0	0.0
5	ℓ_5 =100.0	3.1320×10^4	2.3229×10^5	2.6165×10^7	2.6267×10^7	0.0	0.0	0.0

Table 2. CPU Comparison

CPU		Ratio
DADS	Recursive Method	
8450 sec.	762 sec.	11.09

Table 2. CPU Comparison

N90-23043

Algorithmic Considerations of Integrated Design for CSI on a Hypercube Architecture

Ü. Özgüner and F. Özgüner

Department of Electrical Engineering, The Ohio State University
2015 Neil Ave, Columbus, OH 43210

ABSTRACT

In this paper we present an approach to the integrated design problem for actively controlled large, flexible mechanical systems for which *Control Structure Interaction* (CSI) problems are of concern.

The two coupled design problems have been identified as the optimal *Structural Design* problem and the optimal *Controller Design* problem. These two problems can be addressed within a decision making loop that would consider each seperately, and then sequentially analyze the effects of one on the other. Embedded in such a loop would be the simulation and coordination tasks as part of the decision tools required in a total (software) package.

All of the above are compute–intensive tasks. In any such task, possible decompositions and gains due to the inherent parallelism have to be exploited. We claim that the problems under consideration, as applied to large flexible mechanical structures are particularly suited to be mapped onto multi–computer systems in a hypercube topology.

1. INTRODUCTION

Issues related to accomplishing integrated design for structural and control systems are of increasing concern in the context of Large Space Structures. Indeed a number of attempts have been made to come up with unified cost criteria and optimization approaches ([1], [2]). It is evident that one of the major hurdles in all such attempts is, and is going to be, computational. For truly large scale systems all four aspects, namely finite element modeling, control algorithms, over–all optimization and finally total closed–loop simulation singly or jointly create computational problems.

The use of distributed memory multiple processors connected in a hypercube topology has proven to be very useful in many large computation intensive tasks, especially with favorable parameter structures and algorithms which are compatible with the said topology. Indeed, the above four problems have been addressed individually (possibly in different contexts) for hypercube architectures. For example in [3], as a result of finite element discretization, linear equations of banded form are obtained and solved with an approach based on the *Conjugate Gradient* method. Experimental results on a 16–node Intel 386–based iPSC/2 hypercube have shown an almost linear speedup over a single processor implementation.

Some control design related work, specifically on solutions of quadratic regulator problems have also been reported in the literature [4] on hypercube based solution approaches. Yet these approaches have not been evaluated within the context of a total design package for Large Space Structures. In this paper we shall report on such an evaluation and introduce preliminary results for the development of a a CSI design package which assumes a large system with subsystems under decentralized control. The decentralized control design approach is based on the package DOLORES [5] which is being modified for the hypercube.

Distributed memory multiprocessors interconnected in a static topology such as a mesh, a toroid or a hypercube have been proposed as architectures particularly suitable for diverse application areas of scientific computing. However, in order to use these general purpose parallel computers in a specific application, existing algorithms need to be restructured for the architecture and new algorithms developed. In fact, conventional algorithms need to be reexamined, since the best algorithm for a sequential computer may not be the best for a parallel computer. Parallelization schemes for most applications on distributed memory multiprocessors are characterized by the mapping of a physical domain or its graph representation to processors with locality of communication. However, applications exhibit different characteristics in data dependencies and interprocessor communication patterns and volume. Finite element and matrix problems have very regular structures and the volume of communication and the amount of computation can be predicted.

Therefore, the basic model we shall consider is the banded matrix structure obtained from a finite–element model. We propose the retention of the nodal form of such models as we move from modeling to controller design and back. We intend to fix the number of nodes and nodal variables through the optimal design cycle. We will argue for the possible insertion of dummy nodes to retain as broad a *design space* as necessary for doing *parametric designs* where comparative evaluations of multiple criteria will becomeneeded.

The key reason for keeping the nodal form, possibly with excessive nodes, is doing the mapping to the processors only once. Thus, the required communication structure for the various problems will remain the same for different iterations of the same problem.

The configuration of a design package which includes the FEM stage is given in Figure 1.

2. THE FINITE ELEMENT MODEL

2.1 The Problems Considered

The idea behind the finite element method has always been to provide a formulation which can exploit the resources of digital computers. The resources provided by multiprocessors, particularly hypercube topologies, are especially useful for finite element based analysis and design problems where the banded structure of relevant matrices are exploited [6]. The important stages in such an analysis are:

- Obtaining the equations of motion of a structure by deriving the element equations and then *assembling* the equations for all elements.

- *Solving* the FEM equations, i.e. essentially solving a set of linear equations corresponding to static force–displacement type relations.

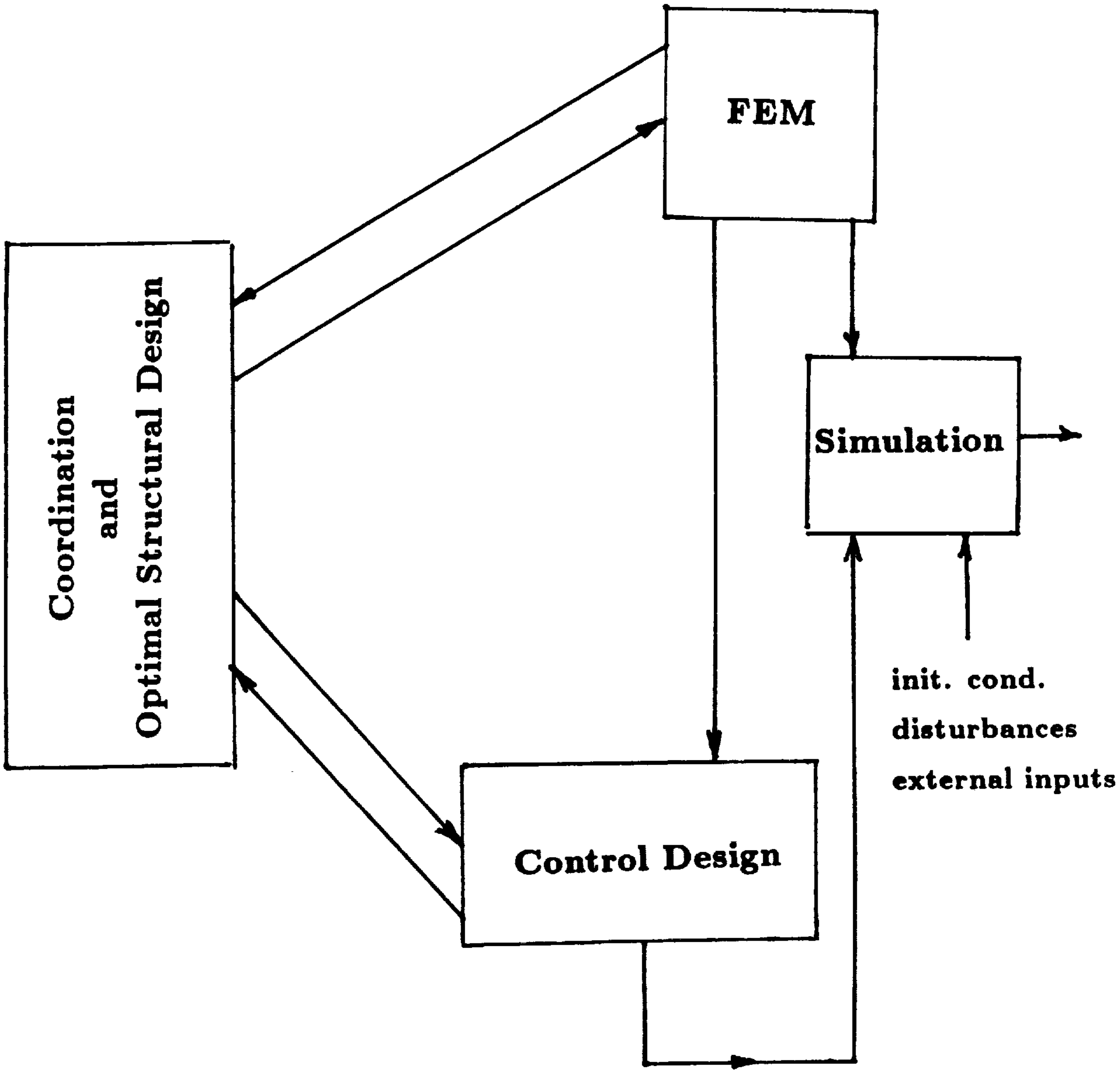


Figure 1: The configuration for a design package.

- *Modal analysis*, i.e. a dynamic analysis requiring solution for modes and mode–shapes.

Thus the problems of concern are solution of equations of the form,

$$\mathcal{K}q = \mathcal{B}F \tag{2.1}$$

for q and

$$\left(\mathcal{K} - \omega^2\mathcal{M}\right) q = 0 \tag{2.2}$$

for the set ω^2. The nxn $\mathcal{M}$ and $\mathcal{K}$ matrices are the mass and stiffness matrices respectively, and we assume that the consistent mass matrix approach has been utilized in generating $\mathcal{M}$. Thus the interdependancies implied by the elements of the two matrices (location of zeros, etc) are the same. The vector q is the nodal displacement vector, and F is the vector of applied forces to the structure and the matrix $\mathcal{B}$ denotes the influences on individual nodes.

2.2 Mapping to Processors and $\mathcal{K}$ Generation

The key to having a good total design package is having properties in parameter sets that different portions of the package, addressing different problems, can jointly exploit. The basic property we want to exploit here is the banded form arising from the FEM. Thus the important first stage is obtaining that form and mapping it onto the processors [7]. This is coupled to the so–called $\mathcal{K}$ *Generation* operation. The elemental stiffness equations give relationships between each of the nodes associated with the element. The global $\mathcal{K}$ matrix is the superposition of the individual elemental K matrices. Therefore, the individual elemental K matrices may be computed independently and this portion of the generation of $\mathcal{K}$ can be done completely in parallel. It is important to note that the adjacency matrix associated with the finite element graph is identical in its zero–nonzero structure to the $\mathcal{K}$ matrix. This knowledge is utilized in a number of subsequent iterative solution schemes.

A row partitioning of the global $\mathcal{K}$ matrix corresponds to mapping a set of nodes onto each processor of a hypercube. The mapping determines which rows of $\mathcal{K}$ need be resident on a processor in order to perform the necessary matrix–vector product operations in any algorithm under consideration. The operations to be performed in parallel are then obtained from these rows.

The set of all elements which contain nodes in the mapped node set for a processor is *the associated element set* for that processor. If the nodes are contained entirely in the mapped set, the elements are *interior elements*. The difference between the associated and interior sets is called *the boundary element* set. This set provides information for the construction of rows mapped to at least two different processors and thus represents data that must be either communicated or calculated redundantly. [1]

Since the volume of data to be transmitted is proportional to the size of the boundary element set, it is important to perform the selection of this set carefully. An example of mapping is shown in Figure 2.

[1] The so–called Component Mode Synthesis techniques promise to provide a rich avenue of further research coupling both computation and control issues into basic FEM. The clear identification of the boundary element set provides insight into these relationships.

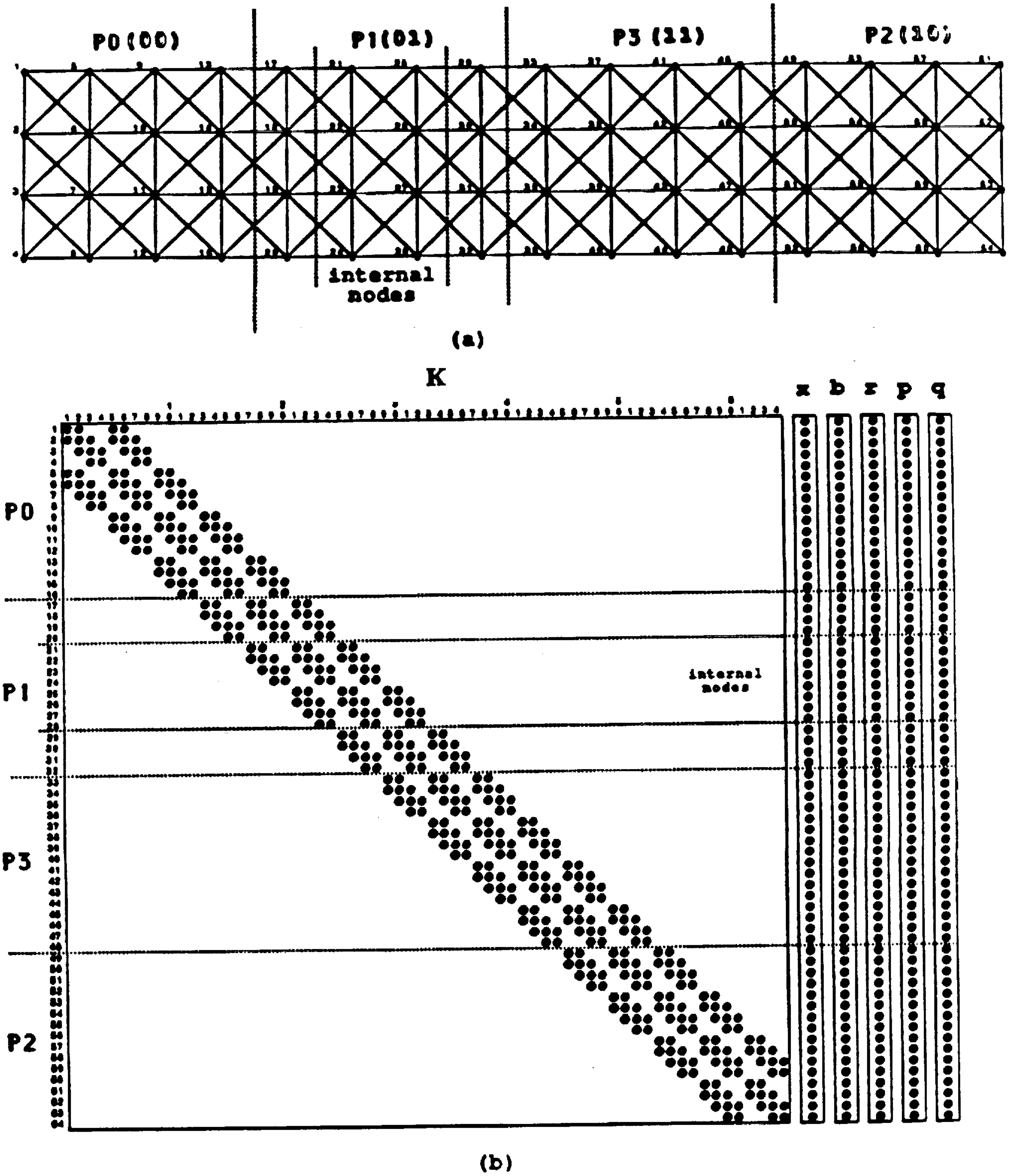


Figure 2. Strip mapping of (a) a finite element domain and (b) the corresponding K matrix onto a 2-d hypercube

2.3 Solving the FEM Equations

Methods for solving such equations on sequential computers can be grouped as: direct methods such as Gaussian elimination, LU decomposition and Cholesky factorization and iterative methods such as Gauss-Jacobi, Gauss-Seidel and Conjugate Gradient methods. Although extensive research has been done on parallelization of the solution of large sparse systems of linear equations, new architectural features such as the crossbar connection capability and massively parallel distributed memory architectures with fast communication offer the potential for new approaches to algorithm design.

Among the iterative methods, the CG algorithm [3], is increasingly being used in sparse matrix solvers, since it converges in at most n steps. The computational steps of the standard *CG* algorithm are given below:

Let $\mathbf{b} = \mathcal{B}F$. *Initially, choose* $\mathbf{x}_0$ *and compute* $\mathbf{r}_0 = \mathbf{p}_0 = \mathbf{b} - \mathcal{K}\mathbf{x}_0$. *Then, for* $k = 0, 1, 2, \ldots$

```
1.    form $\mathbf{q}_k = \mathcal{K}\mathbf{p}_k$
2.  a. form $< \mathbf{p}_k, \mathbf{q}_k >$
    b. $\alpha_k = \frac{<\mathbf{r}_k,\mathbf{r}_k>}{<\mathbf{p}_k,\mathbf{q}_k>}$
3.  $\mathbf{r}_{k+1} = \mathbf{r}_k - \alpha_k \mathbf{q}_k$
4.  $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$                                  (2.3)
5.  a. form $< \mathbf{r}_{k+1}, \mathbf{r}_{k+1} >$
    b. $\beta_k = \frac{<\mathbf{r}_{k+1},\mathbf{r}_{k+1}>}{<\mathbf{r}_k,\mathbf{r}_k>}$
6.  $\mathbf{p}_{k+1} = \mathbf{r}_{k+1} + \beta_k \mathbf{p}_k$
```

Here, $\mathbf{r}_k$ is the residual error associated with the trial vector $\mathbf{x}_k$, *i.e.* $\mathbf{r}_k = \mathbf{b} - \mathcal{K}\mathbf{x}_k$ which must be null when $\mathbf{x}_k$ is coincident with $\mathbf{x}$ which is the solution vector and $\mathbf{p}_k$ is the direction vector at the k-th iteration. As seen from (1), the *CG* algorithm has three types of operations: matrix vector product $\mathcal{K}\mathbf{p}_k$, inner products $< \mathbf{r}_{k+1}, \mathbf{r}_{k+1} >$ and $< \mathbf{p}_k, \mathbf{q}_k >$ and the vector additions required in steps 3,4, and 6. To perform these operations concurrently, the rows of $\mathcal{K}$, and the corresponding elements of the vectors $\mathbf{b}$, $\mathbf{x}$, $\mathbf{r}$, $\mathbf{q}$ and $\mathbf{p}$ must be distributed among the processors by considering the communication features of the machine. Details of the implementation can be found in [3] where a speedup of over 15 is reported for a 16–node hypercube implementation.

2.4 Modal Analysis

Although much research on linear equation solutions for banded matrices has been done, results on eigenvalue calculations on hypercubes for banded matrices are very recent. We shall not dwell further on this issue except to point out that such an approach will have to be pursued to retain the advantages of the mapping described above.

3. INCLUSION OF ACTUATORS

3.1 The Proof–Mass Actuator Model

It is clear that the addition of an actuator with no bi–directional coupling with structure dynamics is simple and will only affect the mass matrix in the FEM. Thus the number of nodes in the FEM will not change. Some of them will have to be tagged in order to do a parametrized study of actuator location variation. On the other hand, actuators with dynamics coupled to the structure dynamics are somewhat more complex. In what follows, we consider such a case, specifically the model of a generic **proof–mass actuator**. (This follows the same lines as the analysis of a specific proof–mass actuator on a simple beam, presented in [8].)

Consider a proof–mass actuator attached to a flexible structure at a certain point. We shall simplify the problem somewhat by assuming that the point of attachment has been represented by a single node in the FEM. Let the nodal position variable (in the same direction as proof–mass movement) at this point be denoted by q_i. Let the proof–mass have mass m, friction constant d_m and spring constant k_m. [1] The equations of motion of the mass m are given by,

$$m\ddot{z} + d_m(\dot{z} - \dot{q}_i) + k_m(z - q_i) = f_e \tag{3.1}$$

where z denotes the position of the mass with respect to a fixed reference frame and f_e is the force generated by the windings and can be modeled as a linear DC motor as follows,

$$\begin{aligned} f_e &= k_e i & (3.2)\\ e &= Ri + k_b(\dot{z} - \dot{q}_i) \quad , & (3.3) \end{aligned}$$

where k_e, k_b are the motor constant and back e.m.f. constant respectively, R is the electrical resistence in the the motor circuit, and i and e denote the current and input voltage to the motor. Let us define the following constants:

$$\begin{aligned} d_e &= \frac{k_e k_b}{R} & (3.4)\\ b &= \frac{k_e}{R} & (3.5) \end{aligned}$$

Thus the total dynamics of the proof–mass actuator can be given as,

$$m\ddot{z} + (d_m + d_e)\dot{z} + k_m z - (d_m + d_e)\dot{q}_i - k_m q_i = be \tag{3.6}$$

The reaction force applied to the flexible structure is,

$$F = m\ddot{z} = f_e - d_m(\dot{z} - \dot{q}_i) - k_m(z - q_i) \tag{3.7}$$

[1] The last two could also have been added by electronic means, that is by wrapping a local feedback loop around the actuator. See [9] for such an example.

3.2 Embedding into the FEM

The actuator that has been analyzed in the present paper is particularly suitable for spacially decentralized control applications since it can be distributed over several locations of a given flexible structure for active vibration damping. Some attempts along these lines (using different devices) have already been reported in the literature [10,11]. To keep the notation simple, we shall continue treating the case with a single actuator. Consider now the dynamical equations representing the structure obtained from a FEM, with no actuator dynamics:

$$\mathcal{M}\ddot{q} + \mathcal{K}q = \mathcal{B}F \tag{3.8}$$

For our case $\mathcal{B}$ is a vector with all entries zero except the i'th which is 1. We define the expanded nodal position vector $\bar{q}$ as

$$\bar{q} = \begin{bmatrix} q_1 \\ \vdots \\ q_i \\ \vdots \\ q_n \\ z \end{bmatrix} \tag{3.9}$$

This is essentially equivalent to assigning nodal variables to the proof-mass actuator and considering it as a mass-spring system. The back e.m.f. and other effects have to be treated with some care. The final result will be an expanded FEM that can be written as,

$$\bar{\mathcal{M}}\ddot{\bar{q}} + \mathcal{D}\dot{\bar{q}} + \bar{\mathcal{K}}\bar{q} = \bar{\mathcal{B}}e \tag{3.10}$$

where the various matrices can be generated from the original (non-actuated) FEM and the actuator model. (An interesting and important observation here is that the structure will have damping with the input shorted, i.e. $e = 0$).

The point of all the above discussion is:

1. Actuator dynamics could have been introduced in the FEM stage.

2. If there are choices in actuator location, all "possible" actuators can be inserted by the introduction of dummy nodal variables.

4. THE OPTIMAL CONTROLLER DESIGN

4.1 A General Decentralized Structure

A number of results have appeared in the literature on using a quadratic regulator framework for controller design for flexible structures. We shall assume here that a decentralized information structure has been imposed. This means that there exists a correspondence between the inputs (actuators) and the output measurements (sensors). Let the system be modelled as,

$$\dot{x} = Ax + \sum_{i=1}^{\nu} B_i u_i \tag{4.1}$$

$$y_i = C_i x + \ D_i u_i \qquad ; \ i = 1, \cdots, \nu \tag{4.2}$$

where $x \in \Re^n$, $u_i \in \Re^{l_i}$, $y_i \in \Re^{m_i}$ and the matrices are real and of compatible dimension. Due to the decentralization constraint, only static feedback is allowed, the control is

$$u_i \ = \ K_i \, y_i \qquad ; \ i = 1, \cdots, \nu \tag{4.3}$$

or if dynamics are allowed in each feedback loop, the control is

$$\begin{aligned} \dot{z}_i &= F_i z_i + G_i y_i & & (4.4) \\ u_i &= H_i z_i + N_i y_i \qquad ; \qquad i = 1, \ldots, \nu. & & (4.5) \end{aligned}$$

where $z_i \in \Re^{q_i}$.

4.2 The Decentralized Quadratic Regulator

Consider a large scale system (4.1)-(4.2) with decentralized control (4.3) . The basic problem is to find an optimal static feedback gain so that the following cost function is minimized:

$$J = \int_0^\infty (x^T Q x + \ + \sum_{i=1}^{\nu} u_i^T R_i u_i) \, dt \tag{4.6}$$

and the following feedback structure constraint:

$$u_i = K_i \, y_i \qquad ; i = 1, \ldots, N. \tag{4.7}$$

It can be shown [12,13,5,14] that the necessary conditions for minimizing J given by (4.6) with the controller structure (4.7) imply the solution of the following system of nonlinear algebraic equations:

$$\begin{cases} A_c^T P + P A_c + \bar{Q} = 0 \\ A_c L + L A_c^T + X_0 = 0 \end{cases}$$

and

$$\nabla_{K_i} J = B_i^T P L C_i^T + R_i K_i C_i L C_i^T = 0$$

where

$$\begin{aligned} A_c &= A + \sum_{i=1}^{N} B_i K_i C_i \\ \bar{Q} &= Q + \sum_{i=1}^{N} C_i^T K_i^T R_i K_i C_i \\ X_0 &= x_0 \, x_0^T . \end{aligned}$$

4.3 The Lyapunov Equation Solution

It is obvious from the above that the solution of a single or coupled multiple Lyapunov equations,

$$AX + XA^T + Q = 0 \tag{4.8}$$

is key to many optimal controller design problems. Indeed the Lyapunov equation has been analyzed numerous times and many solution algorithms proposed. In general Schur decomposition based algorithms have been the accepted, reliable methods of solving small size problems, where A is dense and has no particular structure. Gardiner and Laub have addressed [4] the issue of large size and hypercube implementation, however the equations they have considered are still dense. Consideration of the sparse case is very recent, and general *heuristic* algorithms have been proposed in [15] with no convergence proof.

In considering linear equation solutions and modal analysis, we have discussed the case of banded matrices arising from the FEM. We have to, however, analyze clearly what the structure of the A matrix in the Lyapunov equation will be if it arises from a FEM.

Consider the *state equations* that one can obtain from the original model,

$$\mathcal{M}\ddot{q} + \mathcal{D}\dot{q} + \mathcal{K}q = \mathcal{B}F \tag{4.9}$$

If we go into *modal* form from the *nodal* form, with the unitary transformation,

$$z = \Phi^T q \tag{4.10}$$

to obtain,

$$\ddot{z} + \bar{D}\dot{z} + \tilde{K}z = \bar{B}F \tag{4.11}$$

where the relevant matrices are,

$$\begin{aligned} \bar{D} &= \Phi^T \mathcal{D} \Phi && (4.12) \\ \tilde{H} &= \Phi^T \mathcal{K} \Phi && (4.13) \\ \bar{B} &= \Phi^T \mathcal{B} && (4.14) \end{aligned}$$

we will obtain, in state space, the A matrix as,

$$A = \begin{bmatrix} 0 & I \\ -\bar{K} & -\bar{D} \end{bmatrix} \tag{4.15}$$

The required Lyapunov equations can then be solved in terms of the A matrix above. Note, however, that the problem is in modal space and the distribution to processors, accomplished during $\mathcal{K}$ generation is no longer valid. Besides, calculation of the nodal–modal transformations have to be accomplished.

Another approach which holds some promise is, remaining in nodal space and transforming the relevant control design equations. Thus, the equation of interest will no longer be the standard Lyapunov equation, but a transformed Lyapunov equation appropriate for systems modeled by second order differential equations. This provides the benefit of retaining the banded structure in the coefficient matrices without going through a modal transformation stage.

5. CONCLUSION

In this paper we considered a possible design package for large flexible mechanical structures that would address CSI problems. We advocate the exploitation of the banded structure that can be obtained from proper mappings in the FEM stage. We furthermore advocate the utilisation of distributed memory multiprocessors in a hypercube topology as particularly suitable for addressing the algorithmic problems.

Research in this area is somewhat new, but draws on aspects of previous work, in FEM, in matrix algorithms and in control design. The key issues are related to

- Being able to cast multiple problems into a single framework
- Retaining the same framework without doing unnecessary transformations
- Introducing dummy variables and parameters which can be used in parametric studies

Samples of subproblems and cases have been given to indicate the necessity of further work along these lines.

6. REFERENCES

[1] D. Miller, V. Venkayya, and V. Tischler, "Integration of structures and controls – some computational issues," in *Proc. of 24th CDC*, Ft. Lauderdale, FL, Dec. 1985.

[2] S.S.Rao, V.B. Venkayya and N.S. Khot, "Game theory approach for the integrated design of structures and controls," *AIAA Journal*, vol. 26, pp. 463–469, 1988.

[3] C. Aykanat, F. Özgüner, F. Ercal and P. Sadayappan, "Iterative algorithms for solution of large sparse systems of linear equations on hypercubes," *IEEE Trans. on Computers*, vol. 37, pp. 1554–68, 1988.

[4] J. Gardiner and A. Laub, "Hypercube implementation of some parallel algorithms in control," in *The Application of Advanced Computing Concepts and Techniques in Control Engineering*, NATO ASI, Denham and Laub eds. 1988.

[5] F. Khorrami, S. Tien, and Ümit Özgüner, "DOLORES: A software package for analysis and design of optimal decentralized control," in *Procedings of the 40th National Aerospace and Electronics Conference*, Dayton, OH, May 1988.

[6] L.M. Adams and R.G. Voigt, "Design, development and use of the finite element machine," Technical Report, NASA ICASE Report 172250, 1983.

[7] C. Aykanat, F. Özgüner, S. Martin and S.M. Doraivelu, "Parallelization of a finite element application program on a hypercube multiprocessor," in *Proc. 2nd Conf. on Hypercube Multiprocessors*, pp. 662–673, Knoxville, TN, Oct. 1986.

[8] J. Martin, Ü. Özgüner, S. Yurkovich, "An active vibration damper for flexible structures," in *Proc. of 17th Pitt. Conf. on Modeling and Sim.*, pp. 687–692, Pitt., PA, April 1986.

[9] E. Breitfeller and Ümit Özgüner, "Development of a control oriented model of a fixed-free beam with end mass," in *Proceedings of the American Control Conference*, Atlanta, GA, June 1988.

[10] U. Özgüner, S. Yurkovich, J. Martin, and F. Al–Abbass, "Decentralized control experiments on NASA's flexible grid," in *Proceedings of the 1986 American Control Conference*, pp. 1045–1051, Seattle WA, June 1986.

[11] N. H. McClamroch, "Vibration control of flexible structures using member dampers," in *1985 Decision and Control Conference*, pp. 936–939, Fort Lauderdale, Florida, December 11–13 1985.

[12] W. S. Levine, T. L. Johnson, and M. Athans, "Optimal limited state variable feedback controllers for linear systems," *IEEE Transactions on Automatic Control*, vol. AC–16, no. 6, pp. 785–792, December 1971.

[13] J. C. Geromel and J. Bernussou, "Optimal decentralized control of dynamic systems," *Automatica*, vol. 18, no. 5, pp. 545–557, 1982.

[14] F. Khorrami and U. Özgüner, "Frequency–shaped cost functionals for decentralized systems," in *Proc. 25th CDC*, Athens, Greece, Dec. 1988.

[15] A.S. Hodel and K.R. Poolla, "Heuristic approaches to the solution of very large sparse Lyapunov and algebraic Riccati equations," in *Proc. 27th Conf. on Decision and Control*, pp. 2217–2222, Austin, Texas, Dec. 1988.

A Parallel Structure Transient Response Algorithm Using Independent Substructure Response Computation

Jeffrey K. Bennighof and Jiann-Yuarn Wu

Department of Aerospace Engineering and Engineering Mechanics
The University of Texas at Austin
Austin, Texas 78712

Abstract

An algorithm for parallel computation of transient response for structures is presented in which responses of substructures are computed independently for dozens of time steps at a time, and these substructure responses are then corrected to obtain the response of the overall coupled structure. The correction of the uncoupled substructure responses only requires the responses computed for interfaces at occasional points in time, and is done independently for different substructures in a very efficient procedure. A numerical example is presented to demonstrate the method and show the accuracy of the method.

Introduction

A significant amount of effort has been directed recently toward the development of methods for subdividing the computational effort associated with the solution of large transient response problems. The general approach of subdividing the computation associated with a given problem on the basis of a subdivision of the problem domain into subdomains has come to be known as domain decomposition in the last few years.[1,2] For transient response problems in structural dynamics, some efforts in this direction have been motivated by the need to solve problems for systems consisting of two or more well-defined subsystems, such as the Shuttle orbiter and its payloads, using modal data that have already been obtained for each of the subsystems rather than computing new modal data for the combined system.[3–5] Other work has been done in the context of the element-by-element approach to finite element analysis.[6,7] More recently, Ortiz *et al.* have proposed methods specifically intended for concurrent computation of transient response based on a subdivision of the problem domain into subdomains.[8,9] In their approach, an implicit integration scheme is used to obtain response for each subdomain for a given time step, and the results of these computations are averaged at interfaces to yield an approximation of the response of the overall system. Hajjar and Abel have investigated the accuracy of these methods for certain structural dynamics transient response problems, and have concluded that their accuracy is inadequate for these problems when practical time step sizes are used.[10]

In all of the transient response methods mentioned above, computation of response on the substructure level can only be done independently for one time step at a time. In contrast to this, an algorithm was presented recently by these authors which allows independent computation of substructure response for an arbitrary number of time steps at a time.[11] After independent substructure responses have been computed, they are corrected based on the interface motion computed for substructures at each time step, to obtain the response of the combined structure. Allowing the response to be computed independently for a number of time steps at a time reduces the interdependence between processors assigned to different substructures significantly, which can be important when the amount of computation required for different substructures is unequal. Also, if there are more substructures than processors, the cost of swapping different substructures in and out of processors will be reduced if it can be done less frequently.

In the present paper, an extension of the algorithm presented in Ref. 11 is presented in which independent substructure response computation can proceed for much longer periods of time. Independent substructure responses are corrected on the basis of computed interface motion sampled at occasional points in time. The correction procedure for obtaining the response of the structure from the computed substructure responses is extremely efficient once the transient response computation is under way, although there is some computational overhead required to set up the correction capability. A numerical example is presented which illustrates the method and shows the accuracy that is obtained.

A Method Using Substructure-Level Response Computation

The algorithm presented in this paper is for computing the transient response of structures whose motion is governed by the equation

$$M\ddot{\boldsymbol{u}} + C\dot{\boldsymbol{u}} + K\boldsymbol{u} = \boldsymbol{F}(t) \tag{1}$$

where M, C, and K are taken to be constant mass, damping, and stiffness matrices, $\ddot{\boldsymbol{u}}$, $\dot{\boldsymbol{u}}$, and $\boldsymbol{u}$ are acceleration, velocity, and displacement vectors, and $\boldsymbol{F}(t)$ is a vector of forces exciting the system. As mentioned in the Introduction, the transient response of a given structure is computed in this algorithm by solving transient response problems for the substructures defined by decomposing the structure. To introduce the notation that will be used in this paper, a mass matrix for a structure composed of two substructures is shown below, after a possible reordering of rows and columns:

$$M = \begin{bmatrix} M_{LL}^{(1)} & M_{LS}^{(1)} & 0 \\ M_{SL}^{(1)} & M_{SS}^{(1)} + M_{SS}^{(2)} & M_{SL}^{(2)} \\ 0 & M_{LS}^{(2)} & M_{LL}^{(2)} \end{bmatrix}. \tag{2}$$

The superscripts in parentheses tell which substructure a given matrix partition is associated with, and the subscripts S and L refer to matrix partitions associated with *shared*, or interface, and *local*, or internal degrees of freedom. For some of the development in this paper, a structure composed of only two substructures is considered in an effort to simplify the presentation. However, the methods presented will be applicable for an arbitrary number of substructures.

Because responses will be obtained for each of the substructures a structure is composed of, some convention must be adopted for representing the structure response in terms of the substructure responses, particularly at the interfaces. In this paper, the approach taken is similar to the standard approach for the assembly of element matrices in the finite element method. The response of the structure in interface degrees of freedom is represented as the sum of the interface responses for the substructures sharing the interface, e.g.,

$$\boldsymbol{u} = \left\{ \begin{array}{c} \boldsymbol{u}_L^{(1)} \\ \boldsymbol{u}_S^{(1)} + \boldsymbol{u}_S^{(2)} \\ \boldsymbol{u}_L^{(2)} \end{array} \right\}, \tag{3}$$

so that each substructure's interface response is only one component of the total interface response of the structure. Of course, if this convention is adopted, substructure transient response problems must be defined and solved in such a way that the response of the structure obtained by assembling together the substructure responses is accurate.

Substructure response problems can be defined for independent computation by extracting equations from the structure equations of motion, and they will be of the form

$$\begin{bmatrix} \hat{M}_{SS}^{(k)} & M_{SL}^{(k)} \\ M_{LS}^{(k)} & M_{LL}^{(k)} \end{bmatrix} \begin{Bmatrix} \ddot{u}_S^{(k)} \\ \ddot{u}_L^{(k)} \end{Bmatrix} + \begin{bmatrix} \hat{C}_{SS}^{(k)} & C_{SL}^{(k)} \\ C_{LS}^{(k)} & C_{LL}^{(k)} \end{bmatrix} \begin{Bmatrix} \dot{u}_S^{(k)} \\ \dot{u}_L^{(k)} \end{Bmatrix} + \begin{bmatrix} \hat{K}_{SS}^{(k)} & K_{SL}^{(k)} \\ K_{LS}^{(k)} & K_{LL}^{(k)} \end{bmatrix} \begin{Bmatrix} u_S^{(k)} \\ u_L^{(k)} \end{Bmatrix} = \begin{Bmatrix} \hat{F}_S^{(k)} \\ F_L^{(k)} \end{Bmatrix}, \tag{4}$$

where "hat" symbols identify matrix or vector partitions for which a policy for assigning the corresponding partitions in the structure equations of motion to the different substructures must be determined. Again, reordering of rows and columns may be necessary to collect all "shared" degrees of freedom together for a given substructure. Simply computing substructure responses that satisfy these equations and assembling them together will not result in an accurate representation of the response of the overall structure, because the interaction between substructures is neglected in such an approach. It must be noted that in the response of the structure, each substructure has two sources of excitation. One is the external applied force, which appears on the right hand side of the equation above, and the other is due to interaction with adjacent substructures at the interfaces. This suggests a two-step approach for computing the responses of substructures in the response of the coupled structure. The first step consists of obtaining independent substructure responses that satisfy the substructure equations of motion above. These responses neglect any interaction between substructures. Then the second step consists of correcting these substructure responses to obtain responses of substructures in the motion of the coupled structure. It will be shown that this second step can be accomplished with a surprisingly small amount of effort, and with very little information from the independent substructure responses.

If independent responses satisfying the substructure equations of motion are computed, and assembled together and inserted into the structure equations of motion, a residual $r(t)$ will be obtained. For a two-substructure structure the residual will be given by

$$\begin{aligned} r(t) &\equiv M\ddot{u} + C\dot{u} + Ku - F \\ &= \begin{bmatrix} M_{LL}^{(1)} & M_{LS}^{(1)} & 0 \\ M_{SL}^{(1)} & M_{SS}^{(1)} + M_{SS}^{(2)} & M_{SL}^{(2)} \\ 0 & M_{LS}^{(2)} & M_{LL}^{(2)} \end{bmatrix} \begin{Bmatrix} \ddot{u}_L^{(1)} \\ \ddot{u}_S^{(1)} + \ddot{u}_S^{(2)} \\ \ddot{u}_L^{(2)} \end{Bmatrix} \\ &+ \begin{bmatrix} C_{LL}^{(1)} & C_{LS}^{(1)} & 0 \\ C_{SL}^{(1)} & C_{SS}^{(1)} + C_{SS}^{(2)} & C_{SL}^{(2)} \\ 0 & C_{LS}^{(2)} & C_{LL}^{(2)} \end{bmatrix} \begin{Bmatrix} \dot{u}_L^{(1)} \\ \dot{u}_S^{(1)} + \dot{u}_S^{(2)} \\ \dot{u}_L^{(2)} \end{Bmatrix} \\ &+ \begin{bmatrix} K_{LL}^{(1)} & K_{LS}^{(1)} & 0 \\ K_{SL}^{(1)} & K_{SS}^{(1)} + K_{SS}^{(2)} & K_{SL}^{(2)} \\ 0 & K_{LS}^{(2)} & K_{LL}^{(2)} \end{bmatrix} \begin{Bmatrix} u_L^{(1)} \\ u_S^{(1)} + u_S^{(2)} \\ u_L^{(2)} \end{Bmatrix} - \begin{Bmatrix} F_L^{(1)} \\ F_S \\ F_L^{(2)} \end{Bmatrix}. \end{aligned} \tag{5}$$

By making use of Eq. (4), the residual can be obtained as

$$\begin{Bmatrix} r_L^{(1)}(t) \\ r_S(t) \\ r_L^{(2)}(t) \end{Bmatrix} = \begin{Bmatrix} M_{LS}^{(1)}\ddot{u}_S^{(2)} + C_{LS}^{(1)}\dot{u}_S^{(2)} + K_{LS}^{(1)}u_S^{(2)} \\ r_S(t) \\ M_{LS}^{(2)}\ddot{u}_S^{(1)} + C_{LS}^{(2)}\dot{u}_S^{(1)} + K_{LS}^{(2)}u_S^{(1)} \end{Bmatrix}, \tag{6}$$

where

$$r_S(t) = (M_{SS}^{(1)} + M_{SS}^{(2)})(\ddot{u}_S^{(1)} + \ddot{u}_S^{(2)}) - \hat{M}_{SS}^{(1)}\ddot{u}_S^{(1)} - \hat{M}_{SS}^{(2)}\ddot{u}_S^{(2)}$$

$$\begin{aligned}
&+ (C_{SS}^{(1)} + C_{SS}^{(2)})(\dot{\boldsymbol{u}}_S^{(1)} + \dot{\boldsymbol{u}}_S^{(2)}) - \hat{C}_{SS}^{(1)}\dot{\boldsymbol{u}}_S^{(1)} - \hat{C}_{SS}^{(2)}\dot{\boldsymbol{u}}_S^{(2)} \\
&+ (K_{SS}^{(1)} + K_{SS}^{(2)})(\boldsymbol{u}_S^{(1)} + \boldsymbol{u}_S^{(2)}) - \hat{K}_{SS}^{(1)}\boldsymbol{u}_S^{(1)} - \hat{K}_{SS}^{(2)}\boldsymbol{u}_S^{(2)} \\
&- \boldsymbol{F}_S + \hat{\boldsymbol{F}}_S^{(1)} + \hat{\boldsymbol{F}}_S^{(2)}.
\end{aligned} \tag{7}$$

Note that the residual associated with one substructure is given entirely in terms of the interface motion computed for adjacent substructures. Note also that $\boldsymbol{r}_S(t)$ is defined in terms of the "hat" partitions of Eq. (4), and can be obtained as a null vector, if these "hat" partitions are chosen to satisfy the following:

$$\begin{gathered}
\hat{M}_{SS}^{(1)} = \hat{M}_{SS}^{(2)} = M_{SS}^{(1)} + M_{SS}^{(2)}, \\
\hat{C}_{SS}^{(1)} = \hat{C}_{SS}^{(2)} = C_{SS}^{(1)} + C_{SS}^{(2)}, \\
\hat{K}_{SS}^{(1)} = \hat{K}_{SS}^{(2)} = K_{SS}^{(1)} + K_{SS}^{(2)}, \\
\hat{\boldsymbol{F}}_S^{(1)} + \hat{\boldsymbol{F}}_S^{(2)} = \boldsymbol{F}_S.
\end{gathered} \tag{8}$$

With this as motivation, the "hat" partitions are taken to be defined this way in this paper. A physical interpretation of this choice is that for each of the independent substructure response problems, the structure is modeled as if it were clamped one node beyond the interfaces, and the excitation acting on the structure at the interfaces is divided between the substructures that share the interfaces.

The residual in the equations associated with a given substructure can be seen to be a result of including the interface motion of adjacent substructures in the given substructure's equations of motion. This interface motion for adjacent substructures was neglected in the solution of the independent substructure response problems. In order to obtain the true response of the structure, the substructure responses must be corrected to account for adjacent substructures' interface motion, so that when the substructure responses are assembled into the structure equations of motion, the residual is zero.

For the correction to the first substructure's response, note that if the interface motion for the second substructure were given, the residual in the structure equations of motion associated with the first substructure would be defined. The first substructure's response would have to be corrected by adding a response of the first substructure to the negative of the residual resulting from the interface motion of the second substructure. The second substructure's response would have to be corrected in a similar manner, if the interface motion for the first substructure were given. However, the interface motion for both substructures is not known *a priori*, because all of the interface motion will be changed as a result of the corrections to the substructure responses. The responses of both substructures will have to be corrected simultaneously, so that the response of each substructure to the negative residual due to the other's *corrected* interface motion will be added to the independently computed substructure response. The following paragraphs present a method for accomplishing this.

Because the residual is defined in terms of interface motion, it is convenient to introduce a vector $\boldsymbol{v}^{(k)}(t)$ containing the interface accelerations, velocities, and displacements for the kth substructure as

$$\boldsymbol{v}^{(k)}(t) = \begin{Bmatrix} \ddot{\boldsymbol{u}}_S^{(k)}(t) \\ \dot{\boldsymbol{u}}_S^{(k)}(t) \\ \boldsymbol{u}_S^{(k)}(t) \end{Bmatrix}. \tag{9}$$

With this definition, the correction of the first substructure's response to account for the second substructure's interface motion will be the response to an excitation of the form

$$\boldsymbol{f}^{(1)}(t) = \begin{bmatrix} -M_{LS}^{(1)} & -C_{LS}^{(1)} & -K_{LS}^{(1)} \\ 0 & 0 & 0 \end{bmatrix} \boldsymbol{v}^{(2)}(t), \tag{10}$$

where the degrees of freedom are ordered as in the structure equations of motion. If the second substructure's interface motion $\boldsymbol{v}^{(2)}(t)$ is given only at the beginning and the end of a time interval consisting of p time steps of length Δt, the interface displacement $\boldsymbol{u}_S^{(2)}(t)$ can be approximated over the time interval $0 \leq t \leq p\Delta t$ by interpolation. Hence, $\boldsymbol{u}_S^{(2)}(t)$ is assumed to take the form

$$\begin{aligned} \boldsymbol{u}_S^{(2)}(t) &= [\psi_1(t)I \quad \psi_2(t)I \quad \psi_3(t)I]\boldsymbol{v}^{(2)}(0) \\ &+ [\psi_4(t)I \quad \psi_5(t)I \quad \psi_6(t)I]\boldsymbol{v}^{(2)}(p\Delta t) \end{aligned} \tag{11}$$

where I represents a unit matrix and $\psi_i(t)$, $i = 1, \ldots, 6$ are interpolation functions that must satisfy the following end conditions:

$$\begin{array}{ll} \ddot{\psi}_1(0) = 1, & \dot{\psi}_1(0) = \psi_1(0) = \ddot{\psi}_1(p\Delta t) = \dot{\psi}_1(p\Delta t) = \psi_1(p\Delta t) = 0, \\ \dot{\psi}_2(0) = 1, & \ddot{\psi}_2(0) = \psi_2(0) = \ddot{\psi}_2(p\Delta t) = \dot{\psi}_2(p\Delta t) = \psi_2(p\Delta t) = 0, \\ \psi_3(0) = 1, & \ddot{\psi}_3(0) = \dot{\psi}_3(0) = \ddot{\psi}_3(p\Delta t) = \dot{\psi}_3(p\Delta t) = \psi_3(p\Delta t) = 0, \\ \ddot{\psi}_4(p\Delta t) = 1, & \ddot{\psi}_4(0) = \dot{\psi}_4(0) = \psi_4(0) = \dot{\psi}_4(p\Delta t) = \psi_4(p\Delta t) = 0, \\ \dot{\psi}_5(p\Delta t) = 1, & \ddot{\psi}_5(0) = \dot{\psi}_5(0) = \psi_5(0) = \ddot{\psi}_5(p\Delta t) = \psi_5(p\Delta t) = 0, \\ \psi_6(p\Delta t) = 1, & \ddot{\psi}_6(0) = \dot{\psi}_6(0) = \psi_6(0) = \ddot{\psi}_6(p\Delta t) = \dot{\psi}_6(p\Delta t) = 0. \end{array} \tag{12}$$

Quintic polynomials were used for the results obtained in this paper. Expressions for $\dot{\boldsymbol{u}}_S^{(2)}(t)$ and $\ddot{\boldsymbol{u}}_S^{(2)}(t)$ for defining the excitation for correcting the first substructure's response are easily obtained by differentiating the interpolation functions.

With $\boldsymbol{u}_S^{(2)}(t)$ defined in terms of $\boldsymbol{v}^{(2)}(0)$ and $\boldsymbol{v}^{(2)}(p\Delta t)$, the corrected interface motion for the first substructure at the end of the time interval will be the sum of the response to the independent response problem and the response based on $\boldsymbol{v}^{(2)}(t)$, $0 \leq t \leq p\Delta t$. Hence, it will have the form

$$\boldsymbol{v}^{(1)}(p\Delta t) = \boldsymbol{v}_{ind}^{(1)}(p\Delta t) + S_{12}\boldsymbol{v}^{(2)}(0) + T_{12}\boldsymbol{v}^{(2)}(p\Delta t), \tag{13}$$

where each column of the matrices S_{12} and T_{12} contains the first substructure's interface response at $t = p\Delta t$ to a negative residual specified by a column of the first or second matrix, respectively, on the right-hand side of Eq. (11). Using a similar approach, the corrected interface response of the second substructure at the time $t = p\Delta t$ can be expressed in terms of the first substructure's interface motion as

$$\boldsymbol{v}^{(2)}(p\Delta t) = \boldsymbol{v}_{ind}^{(2)}(p\Delta t) + S_{21}\boldsymbol{v}^{(1)}(0) + T_{21}\boldsymbol{v}^{(1)}(p\Delta t). \tag{14}$$

As mentioned above, corrected interface motion for an adjacent substructure is not known before the reconciliation is accomplished. All that is known in the two equations above is the interface motion of both substructures at $t = 0$, from initial conditions, and the interface motion obtained from the solution of the independent substructure transient response problems. However, given the set of linear equations in Eqs. (13) and (14), it is straightforward to solve for the unknowns, with the result that

$$\left\{ \begin{matrix} \boldsymbol{v}^{(1)}(p\Delta t) \\ \boldsymbol{v}^{(2)}(p\Delta t) \end{matrix} \right\} = \left[I - \begin{bmatrix} 0 & T_{12} \\ T_{21} & 0 \end{bmatrix} \right]^{-1} \left\{ \begin{matrix} \boldsymbol{v}_{ind}^{(1)}(p\Delta t) + S_{12}\boldsymbol{v}^{(2)}(0) \\ \boldsymbol{v}_{ind}^{(2)}(p\Delta t) + S_{21}\boldsymbol{v}^{(1)}(0) \end{matrix} \right\}. \tag{15}$$

More compactly, the reconciled interface motion is given by

$$\left\{ \begin{matrix} \boldsymbol{v}^{(1)}(p\Delta t) \\ \boldsymbol{v}^{(2)}(p\Delta t) \end{matrix} \right\} = [I - T]^{-1} [S \quad I] \left\{ \begin{matrix} \boldsymbol{v}^{(1)}(0) \\ \boldsymbol{v}^{(2)}(0) \\ \boldsymbol{v}_{ind}^{(1)}(p\Delta t) \\ \boldsymbol{v}_{ind}^{(2)}(p\Delta t) \end{matrix} \right\}, \tag{16}$$

where the matrices S and T are readily identified. The corrected motion for the first substructure's local degrees of freedom at $t = p\Delta t$ is given by

$$\begin{Bmatrix} u_L^{(1)}(p\Delta t) \\ \dot{u}_L^{(1)}(p\Delta t) \end{Bmatrix} = \begin{Bmatrix} u_{Lind}^{(1)}(p\Delta t) \\ \dot{u}_{Lind}^{(1)}(p\Delta t) \end{Bmatrix} + [\, S_L^{(1)} \quad T_L^{(1)} \,] \begin{Bmatrix} v^{(2)}(0) \\ v^{(2)}(p\Delta t) \end{Bmatrix}, \tag{17}$$

where columns of the matrices $S_L^{(1)}$ and $T_L^{(1)}$ contain responses in local degrees of freedom to interpolation functions for representing interface motion. These two matrices are naturally obtained at the same time that the matrices S_{12} and T_{12} are obtained, from the solution of the same substructure response problems. The corrected motion in local degrees of freedom for the second substructure is obtained in the same manner. Once the motion in both local and shared degrees of freedom has been corrected for $t = p\Delta t$, the initial conditions have been obtained for ongoing computation of response for the next p time steps.

The developments presented here are easily applied to structures composed of more than two substructures. For example, if there are three substructures, the matrices S and T in Eq. (16) take the form

$$S = \begin{bmatrix} 0 & S_{12} & S_{13} \\ S_{21} & 0 & S_{23} \\ S_{31} & S_{32} & 0 \end{bmatrix}, \quad T = \begin{bmatrix} 0 & T_{12} & T_{13} \\ T_{21} & 0 & T_{23} \\ T_{31} & T_{32} & 0 \end{bmatrix}, \tag{18}$$

and modification of the rest of the procedure presented for two substructures is straightforward.

Infrequent Reconciliation of Substructure Responses

In the method of the preceding section, responses are computed independently for different substructures for p time steps at a time, and then the independent substructure responses are corrected to obtain substructure responses in the response of the overall coupled structure. In this section, a procedure for carrying out the reconciliation of independent substructure responses after a number of p-step time intervals is developed. This procedure will allow substructure responses to be computed independently for long periods of time without correcting for interaction between substructures.

The interface motion for the second substructure over the time interval $p\Delta t \leq t \leq 2p\Delta t$ can be approximated in terms of the interpolation functions introduced in the preceding section and the interface motion at the beginning and end of the time interval as

$$u_S^{(2)}(t) = [\psi_1(t^*)I \quad \psi_2(t^*)I \quad \psi_3(t^*)I]v^{(2)}(p\Delta t) + [\psi_4(t^*)I \quad \psi_5(t^*)I \quad \psi_6(t^*)I]v^{(2)}(2p\Delta t), \tag{19}$$

where $t^* = t - p\Delta t$. Recalling that substructure responses have two components including the response to external excitation, which is represented in the independent substructure responses, and the response due to interaction with adjacent substructures, which is represented in the correction to the independent substructure responses, the interface response of the first substructure at the time $t = 2p\Delta t$ will have the form

$$v^{(1)}(2p\Delta t) = v_{ind}^{(1)}(2p\Delta t) + S_{12}(2p\Delta t)v^{(2)}(0) + T_{12}(2p\Delta t)v^{(2)}(p\Delta t) + T_{12}(p\Delta t)v^{(2)}(2p\Delta t). \tag{20}$$

Here, the columns of $S_{12}(2p\Delta t)$ contain responses of the first substructure at $t = 2p\Delta t$ based on the second substructure's interface motion, which is given in terms of the interpolation functions ψ_1, ψ_2, and ψ_3 for $0 \leq t \leq p\Delta t$, and is extended as zero for $p\Delta t \leq t \leq 2p\Delta t$. Similarly, the columns of $T_{12}(2p\Delta t)$ contain responses of the first substructure at $t = 2p\Delta t$ based on interface motion of the second substructure which is given in terms of the interpolation functions ψ_4, ψ_5, and ψ_6 for $0 \leq t \leq p\Delta t$, and is extended in terms of ψ_1, ψ_2, and ψ_3 for $p\Delta t \leq t \leq 2p\Delta t$. The matrix $T_{12}(p\Delta t)$ is simply the matrix T_{12} of the preceding section.

The interface motion for both substructures at $t = 2p\Delta t$ can be written as

$$\left\{ \begin{matrix} v^{(1)}(2p\Delta t) \\ v^{(2)}(2p\Delta t) \end{matrix} \right\} = \left(\left\{ \begin{matrix} v^{(1)}_{ind}(2p\Delta t) \\ v^{(2)}_{ind}(2p\Delta t) \end{matrix} \right\} + S(2p\Delta t) \left\{ \begin{matrix} v^{(1)}(0) \\ v^{(2)}(0) \end{matrix} \right\} \right.$$
$$\left. + T(2p\Delta t) \left\{ \begin{matrix} v^{(1)}(p\Delta t) \\ v^{(2)}(p\Delta t) \end{matrix} \right\} \right) + T(p\Delta t) \left\{ \begin{matrix} v^{(1)}(2p\Delta t) \\ v^{(2)}(2p\Delta t) \end{matrix} \right\} \qquad (21)$$

with S and T matrices defined in terms of 0, S_{12}, S_{21}, 0, etc., as in the last section. Solving for $v^{(1)}(2p\Delta t)$ and $v^{(2)}(2p\Delta t)$ gives the result

$$v(2p\Delta t) = [I - T(p\Delta t)]^{-1}[\,S(2p\Delta t) \quad T(2p\Delta t) \quad I\,] \left\{ \begin{matrix} v(0) \\ v(p\Delta t) \\ v_{ind}(2p\Delta t) \end{matrix} \right\} \qquad (22)$$

where

$$v(ip\Delta t) \equiv \left\{ \begin{matrix} v^{(1)}(ip\Delta t) \\ v^{(2)}(ip\Delta t) \end{matrix} \right\}. \qquad (23)$$

Recalling that

$$v(p\Delta t) = [I - T(p\Delta t)]^{-1}[\,S(p\Delta t) \quad I\,] \left\{ \begin{matrix} v(0) \\ v_{ind}(p\Delta t) \end{matrix} \right\}, \qquad (24)$$

and letting $A \equiv [I - T(p\Delta t)]^{-1}$, $S_i \equiv S(ip\Delta t)$, and $T_i \equiv T(ip\Delta t)$, $v(2p\Delta t)$ can be obtained in terms of initial conditions and independent substructure responses as

$$v(2p\Delta t) = A[\,(S_2 + T_2AS_1) \quad T_2A \quad I\,] \left\{ \begin{matrix} v(0) \\ v_{ind}(p\Delta t) \\ v_{ind}(2p\Delta t) \end{matrix} \right\}. \qquad (25)$$

The corrected interface motion at $t = 3p\Delta t$ can be found using the same approach. When the interface motion for the different substructures is assumed in terms of interpolation functions as in Eq. (19), linear equations involving $v(3p\Delta t)$ can be written as in Eq. (21). These equations can be solved for $v(3p\Delta t)$, yielding the result

$$v(3p\Delta t) = A[\,S_3 \quad T_3 \quad T_2 \quad I\,] \left\{ \begin{matrix} v(0) \\ v(p\Delta t) \\ v(2p\Delta t) \\ v_{ind}(3p\Delta t) \end{matrix} \right\}. \qquad (26)$$

Interpolation functions are simply extended as zero into the time interval $2p\Delta t \leq t \leq 3p\Delta t$ in the generation of responses for matrices S_3 and T_3. Inserting the expressions for $v(p\Delta t)$ and $v(2p\Delta t)$ from Eqs. (24) and (25) gives $v(3p\Delta t)$ in terms of initial conditions and independent substructure responses as

$$v(3p\Delta t) = A[\,(S_3 + T_2AS_2 + (T_3A + (T_2A)^2)S_1) \quad (T_3A + (T_2A)^2)$$
$$(T_2A) \quad I\,] \left\{ \begin{matrix} v(0) \\ v_{ind}(p\Delta t) \\ v_{ind}(2p\Delta t) \\ v_{ind}(3p\Delta t) \end{matrix} \right\}. \qquad (27)$$

This result can be generalized for finding the corrected interface motion at a time $t = mp\Delta t$, with the result that

$$v(mp\Delta t) = A\left[\left(\sum_{i=0}^{m-1} B_i S_{m-i} \right) \quad B_{m-1} \quad B_{m-2} \quad \cdots \quad B_0 \right] \left\{ \begin{matrix} v(0) \\ v_{ind}(p\Delta t) \\ \vdots \\ v_{ind}(mp\Delta t) \end{matrix} \right\}, \qquad (28)$$

where $B_0 \equiv I$, and the other B_i matrices are defined by the recursive formula

$$B_i \equiv \sum_{l=0}^{i-1} (T_{i-l+1}A)B_l, \tag{29}$$

so that $B_1 = T_2A$, $B_2 = T_3A + (T_2A)^2$, $B_3 = T_4A + T_3AT_2A + T_2AT_3A + (T_2A)^3$, etc. Defining a matrix C_m as

$$C_m \equiv A\left[\left(\sum_{i=0}^{m-1} B_iS_{m-i}\right) \quad B_{m-1} \quad B_{m-2} \quad \cdots \quad B_0\right], \tag{30}$$

the corrected interface motion can be obtained separately for each substructure by partitioning C_m into upper and lower halves $\bar{C}_m^{(1)}$ and $\bar{C}_m^{(2)}$, and multiplying each by the vector on the right hand side of Eq. (28). For parallel computation, if different processors are assigned to different substructures, the processor for the kth substructure only needs to have access to $\bar{C}_m^{(k)}$ and the interface motion computed independently for all substructures for every pth time step.

After interface motion has been corrected for $t = mp\Delta t$, the motion for local degrees of freedom for each substructure can be corrected. As an example, the corrected local motion for the first substructure will be given by

$$\begin{Bmatrix} u_L^{(1)}(mp\Delta t) \\ \dot{u}_L^{(1)}(mp\Delta t) \end{Bmatrix} = \begin{Bmatrix} u_{Lind}^{(1)}(mp\Delta t) \\ \dot{u}_{Lind}^{(1)}(mp\Delta t) \end{Bmatrix} + \left[\, S_{Lm}^{(1)} \quad T_{Lm}^{(1)} \quad \cdots \quad T_{L1}^{(1)} \,\right] \begin{Bmatrix} v^{(2)}(0) \\ v^{(2)}(p\Delta t) \\ \vdots \\ v^{(2)}(mp\Delta t) \end{Bmatrix}, \tag{31}$$

where the matrices $S_{Li}^{(k)}$ and $T_{Li}^{(k)}$ contain responses in local degrees of freedom to interface motion given in terms of interpolation functions, and are analogous to the S_i and T_i matrices used above in terms of subscript numbering. The vector of the second substructure's corrected interface motion at every pth time step is given in terms of the independently computed interface responses as

$$\begin{Bmatrix} v^{(2)}(0) \\ v^{(2)}(p\Delta t) \\ \vdots \\ v^{(2)}(mp\Delta t) \end{Bmatrix} = \begin{bmatrix} 0 & I & 0 & 0 & 0 & \cdots & 0 \\ [& \bar{C}_1^{(2)} &] & 0 & & \cdots & 0 \\ & & \vdots & & & & \\ [& & & \bar{C}_m^{(2)} & & &] \end{bmatrix} \begin{Bmatrix} v(0) \\ v_{ind}(p\Delta t) \\ \vdots \\ v_{ind}(mp\Delta t) \end{Bmatrix}. \tag{32}$$

Therefore, the product of the matrix on the right hand side of Eq. (31) and the matrix on the right hand side of Eq. (32) is the matrix by which the vector of independently computed interface responses must be multiplied to obtain the correction for the motion in local degrees of freedom for the first substructure. The same approach is taken to find the correction for the motion in local degrees of freedom for the second substructure.

To summarize, the developments presented in this section permit the independent computation of response for different substructures for a total time interval of length $mp\Delta t$. The interface motion for all of the substructures at the end of this time interval can be corrected using Eq. (28), and then the motion for local degrees of freedom for each of the substructures can be corrected as shown above. Once these corrections are made, initial conditions are obtained so that independent computation of substructure responses can proceed again for another $mp\Delta t$. The amount of computation required for the corrections is very small compared to the amount of computation required for obtaining the independent substructure responses. The computational "overhead" that is required for this method consists of obtaining substructure responses to interface motion specified in terms of interpolation

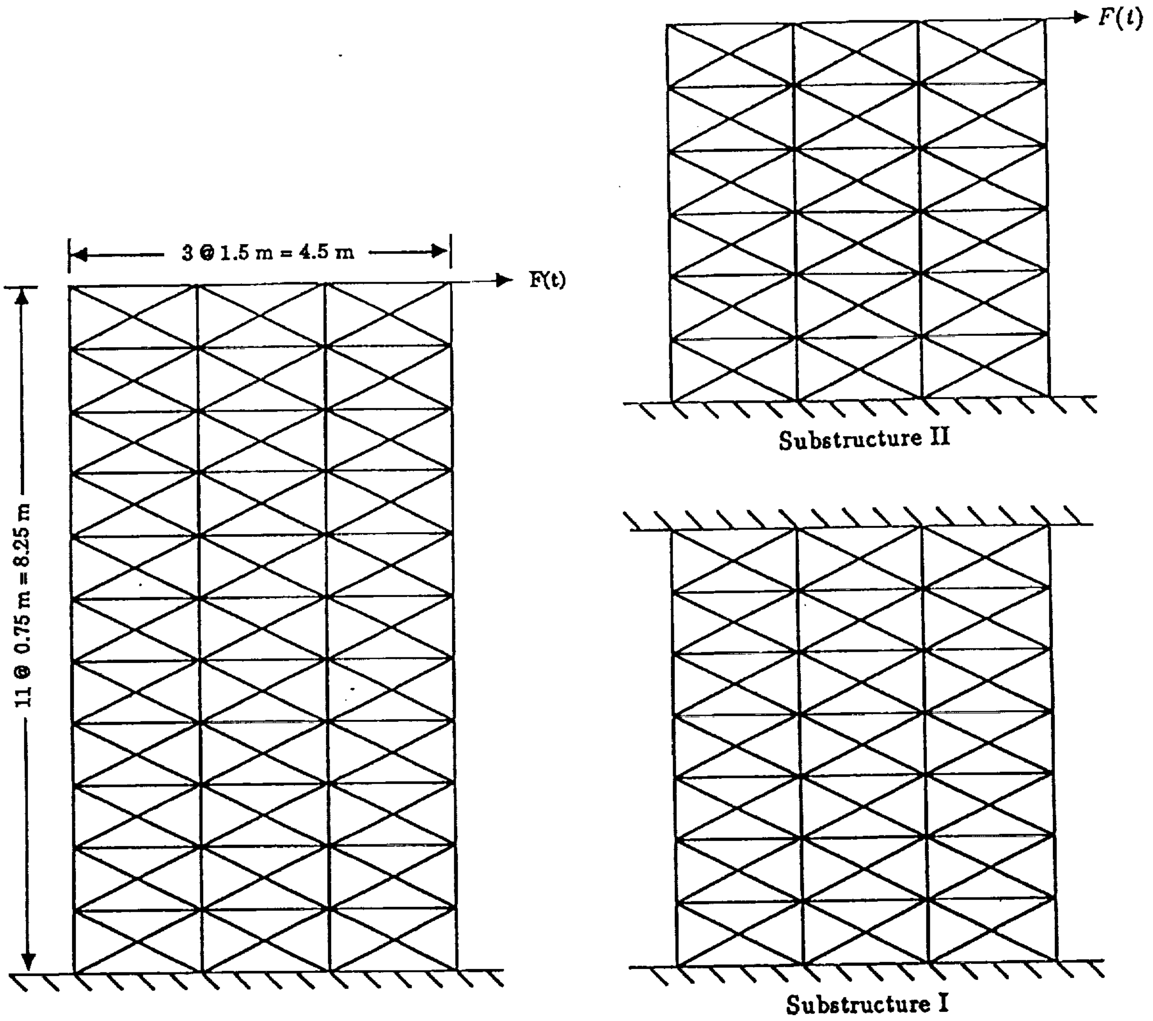


Figure 1: Plane truss used in the numerical example, and its division into substructures.

functions, and carrying out the matrix operations outlined above to obtain the matrices required for making corrections. This overhead is justified if the transient response of the structure must be computed for a long time. The amount of computation required both for the "overhead" operations and for the corrections is determined by the dimensions of the matrices involved, which is determined in turn by how many shared and local degrees of freedom are associated with each of the substructures.

Numerical Example

The algorithm presented in this paper is demonstrated on an example structure which is shown in Fig. 1. The structure is a plane truss composed of 143 aluminum members, each of which has an elastic modulus of $E = 70 \times 10^9$ N/m^2, a cross-sectional area of $A = 4 \times 10^{-4}$ m^2, and a density of $\rho = 2710$ kg/m^3. The dimensions are as shown. A force is applied to the top right corner of the truss starting at $t = 0$, and it is given by

$$F(t) = 5(1 - \cos \Omega t) \text{ (Newtons)}, \tag{33}$$

where $\Omega = 590.3$ radians per second, which is between the second and third natural frequencies of the structure. The truss has eighty-eight degrees of freedom, and is assumed to have proportional

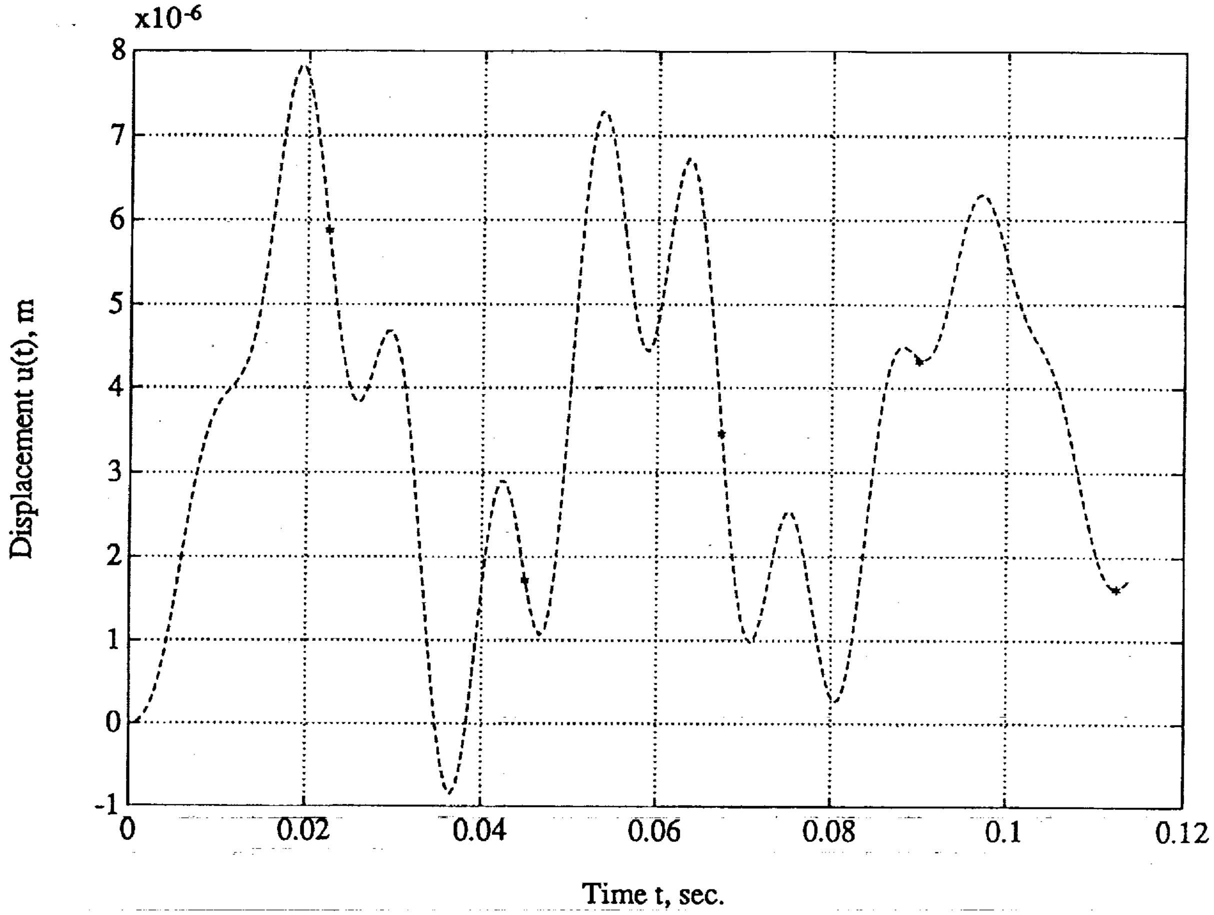


Figure 2: Plots of exact response (dashed line) and computed response (asterisks).

damping of the form $C = \alpha M + \beta K$, where α and β are chosen to give modal damping factors between one and five percent. For application of the algorithm presented in this paper, the structure was partitioned at the top of the sixth bay into two substructures, which are also shown in Fig. 1. Note that each substructure is modeled in the algorithm as being effectively clamped one truss bay beyond the interface, as shown in the figure.

In Fig. 2, the horizontal displacement of the structure at the point where the excitation is applied is plotted. The dashed line is a plot of the exact response, obtained from a mode-by-mode exact solution, and the asterisks represent values that were obtained using the algorithm of this paper. The responses of the two substructures were obtained using an algorithm that finds the exact response to a piecewise linear approximation of the excitation.[12] A time step of $\Delta t = 3.74 \times 10^{-4}$ seconds was used, which is equal to about one twenty-eighth of the period of the excitation, and is also approximately equal to the period of the highest mode of the structure. For larger time steps, the error becomes visible on a plot scaled as in Fig. 2, when the piecewise linear algorithm is used on the structure as a whole. In this example, substructure responses were computed independently for sixty time steps at a time, and then corrections to the independent substructure responses were made based on the interface motion computed for every tenth time step. Therefore, the quintic interpolation polynomials for interface motion were defined over time intervals of length $p\Delta t$ with p equal to ten, and there were six of these time intervals in each time period over which independent substructure responses were computed.

Because the response of the structure was only corrected for every sixtieth time step, the asterisks on the plot in Fig. 2 are sixty time steps apart. It should be noted, however, that the response for any degree of freedom at any time can be obtained in a straightforward manner with a small amount of additional computation. From the plot of Fig. 2, it is evident that the accuracy obtained in this example is quite adequate for most purposes, even though the corrections to independent substructure responses were made based on a very limited amount of information. The only approximations made in obtaining these results were in the piecewise linear approximation of the excitation and the piecewise quintic approximations of the interface motion.

Summary

In this paper, an algorithm is presented for computing the transient response of structures by computing the transient responses of substructures. The algorithm is well suited for parallel implementation, where a different processor would be assigned to each substructure. The fact that computation can proceed independently for different substructures for dozens of time steps at a time reduces the interdependence between processors, which can be of considerable importance when different substructures require different amounts of computational effort per time step. The correction of independently computed substructure responses to obtain the response of the structure acting as a whole requires only the interface motion computed for substructures at occasional points in time. This correction of substructure responses can be done independently for different substructures once the interface motion for all of the substructures has been computed, and this correction requires very little effort. Because of this, the total amount of computation required using this approach will be only slightly greater than the amount required to solve the transient response problem for the structure as a whole for many problems. A surprisingly high level of accuracy is obtained using this algorithm, in view of how little information is required for making corrections to the independent substructure responses.

Acknowledgement

This work was supported in part by National Science Foundation grant number EET-8709155, and monitored by Dr. George Lea, whose support is gratefully acknowledged. Additional support was provided by Office of Naval Research/Defense Advanced Research Projects Agency grant number N00014-89-J-1451, monitored by Dr. Albert J. Tucker, whose support is also gratefully acknowledged.

References

1. Glowinski, R., Golub, G. H., Meurant, G. A., and Periaux, J., eds., *Proceedings of the First International Symposium on Domain Decomposition Methods for Partial Differential Equations*, SIAM, 1988.

2. Chan, T. F., *et al.*, eds., *Proceedings of the Second International Symposium on Domain Decomposition Methods*, SIAM, 1989.

3. Engels, R. C., Harcrow, H. W., and Shanahan, T. G., "An Integration Scheme to Determine the Dynamic Response of a Launch Vehicle with Several Payloads," *Proceedings of the 24th AIAA Structures, Structural Dynamics, and Materials Conference*, Paper No. 82-0632, 1982.

4. Spanos, P. D., Cao, T. T., Jacobson, C. A., Jr., Nelson, D. A. R., Jr., and Hamilton, D. A., "Decoupled Dynamic Analysis of Combined Systems by Iterative Determination of Interface Accelerations," *Earthquake Engineering and Structural Dynamics*, Vol. 16, 1988, pp. 491-500.

5. Spanos, P. D., Cao, T. T., Nelson, D. A. R., Jr., and Hamilton, D. A., "Efficient Loads Analyses of Shuttle-Payloads Using Dynamic Models With Linear Or Nonlinear Interfaces," *Proceedings of the 30th AIAA Structures, Structural Dynamics and Materials Conference*, Mobile, Alabama, April 1989, Paper No. AIAA-89-1203, pp. 414-424.

6. Ortiz, M., Pinsky, P., and Taylor, R., "Unconditionally Stable Element-by-Element Algorithms for Dynamic Problems," *Computer Methods in Applied Mechanics and Engineering*, Vol. 36, No. 2, 1983, pp. 223-239.

7. Hughes, T. J. R., Levit, I., and Winget, J. "An Element-by-Element Solution Algorithm for Problems of Structural and Solid Mechanics," *Computer Methods in Applied Mechanics and Engineering*, Vol. 36, No. 2, 1983, pp. 241-254.

8. Ortiz, M. and Nour-Omid, B., "Unconditionally Stable Concurrent Procedures for Transient Finite Element Analysis," *Computer Methods in Applied Mechanics and Engineering*, Vol. 58, 1986, pp. 151-174.

9. Ortiz, M., Nour-Omid, B., and Sotelino, E., "Accuracy of a Class of Concurrent Algorithms for Transient Finite Element Analysis," *International Journal for Numerical Methods in Engineering*, Vol. 26, 1988, pp. 379-391.

10. Hajjar, J. and Abel, J., "On the Accuracy of Some Domain-by-Domain Algorithms for Parallel Processing of Transient Structural Dynamics," *International Journal for Numerical Methods in Engineering*, Vol. 28, 1989, pp. 1855-1874.

11. Bennighof, J. K. and Wu, J.-Y., "A Substructure-Based Parallel Algorithm for Structure Transient Response Calculations," Paper No. AIAA-89-1338, 30th AIAA Structures, Structural Dynamics and Materials Conference, Mobile, Alabama, April 1989 (in review, *AIAA Journal*).

12. Craig, R. R., Jr. *Structural Dynamics—An Introduction to Computer Methods*, John Wiley & Sons, New York, 1981, pp. 139-142.

N90-23045

Parallel Conjugate Gradient Algorithms for Manipulator Dynamic Simulation

Amir Fijany and Robert E. Scheid
Jet Propulsion Laboratory/California Institute of Technology

Abstract

In this paper parallel conjugate gradient algorithms for the computation of multibody dynamics are developed for the specialized case of a robot manipulator. For an n-dimensional positive-definite linear system, the Classical Conjugate Gradient (CCG) algorithms are guaranteed to converge in n iterations, each with a computation cost of $O(n)$; this leads to a total computational cost of $O(n^2)$ on a serial processor. We present conjugate gradient algorithms that provide greater efficiency using a preconditioner, which reduces the number of iterations required, and by exploiting parallelism, which reduces the cost of each iteration. Two Preconditioned Conjugate Gradient (PCG) algorithms are proposed which respectively use a diagonal and a tridiagonal matrix, composed of the diagonal and tridiagonal elements of the mass matrix, as preconditioners. Parallel algorithms are developed to compute the preconditioners and their inversions in $O(\log_2 n)$ steps using n processors. A parallel algorithm is also presented which, on the same architecture, achieves the computational time of $O(\log_2 n)$ for each iteration. Simulation results for a seven degree-of-freedom manipulator are presented. Variants of the proposed algorithms are also developed which can be efficiently implemented on the Robot Mathematics Processor (RMP).

Methodology for Analysis and Simulation of Large Multidisciplinary Problems

by

William C. Russell, Paul J. Ikeda, and Robert G. Vos

I. Introduction

The Integrated Structural Modeling (ISM) program is being developed for the Air Force Weapons Laboratory and will be available for Air Force work. Its goal is to provide a design, analysis, and simulation tool intended primarily for directed energy weapons (DEW), kinetic energy weapons (KEW), and surveillance applications. The code is designed to run on DEC (VMS and UNIX), IRIS, Alliant, and Cray hosts. Several technical disciplines are included in ISM, namely structures, controls, optics, thermal, and dynamics. Four topics from the broad ISM goal will be discussed in this paper. The first is project configuration management and includes two major areas: the software and database arrangement and the system model control. The second is interdisciplinary data transfer and refers to exchange of data between various disciplines such as structures and thermal. Third is a discussion of the integration of component models into one system model, i.e. multiple discipline model synthesis. Last is a presentation of work on a distributed processing computing environment.

II. Overview of ISM

An overview of the ISM architecture is described to provide a framework for the various subjects that will be covered. Figure 1 shows the general organization of the system. The ISM user accesses the system capabilities through the ACE (Analysis Capability Executive) executive program. The executive provides a graphical menu interface and a command language interface which can operate in either an interactive or batch mode. It allows running of other modules such as NASTRAN and SINDA (Systems Improved Numerical Differencing Analyzer), detailed manipulation and query of data, and interfacing with the database and host operating system. Operational features include command procedures, macros, journaling, and variable computations with looping and branching.

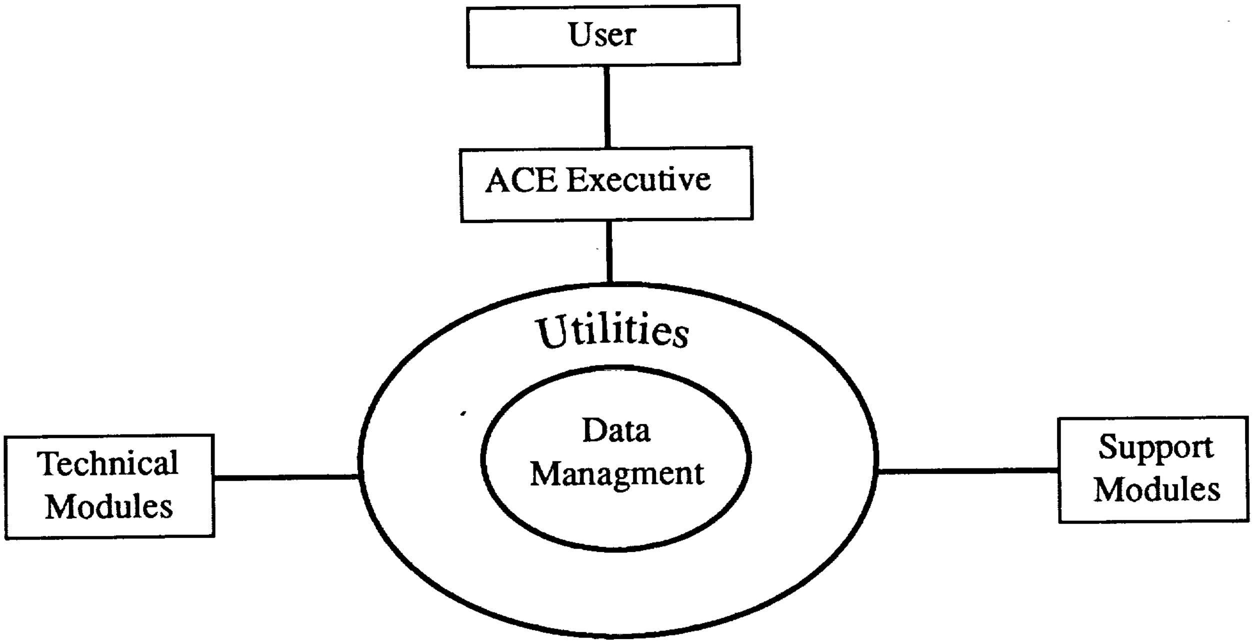


Figure 1. ISM Architecture

The ISM utilities support application programs as well as the executive. The utilities provide a toolkit for managing the database, performing file transfers, setting up and accessing a dynamic memory workspace, and handling system parameters and messages.

The system data storage includes three parts: a multi-user database, with formal cataloging and retrieval capabilities; a temporary workspace, specific to each user or application program; and the host file system. Data may be stored in a general ISM defined table form, as arrays (matrices with labels), or in more arbitrary data structures defined by users or application programs.

Technical modules address the structures, controls, optics, thermal, and system dynamics disciplines. Support modules provide graphics and pre/post processing, interfaces between technical modules, and data flow with CAD and other external databases.

III. Project Configuration Management

On large multidiscipline problems, management of software, database, and models becomes critical to the flow of a project. Multidiscipline software packages often begin as a set of stand alone modules with ad hoc data organization/ management and user interfaces. As the number of modules and supported solution paths increases, common executive program and database features become increasingly important to both the software developers and the users.

Because the number of module interfaces can approach an n^2relationship as shown in figure 2, costs for expansion of a poorly managed system can become prohibitive. The solution to this problem is to develop a common executive and database. With this organization, only one interface between the executive and each module needs to be created. Both multidiscipline users and single discipline specialists on multidiscipline teams, benefit from standard data structures, advanced data handling tools and a consistent top-level user interface. The benefits include less costly learning/relearning curves, faster data flow, and fewer user/software errors. Both program developers and users benefit from more efficient maintenance and the easier incorporation of new technology.

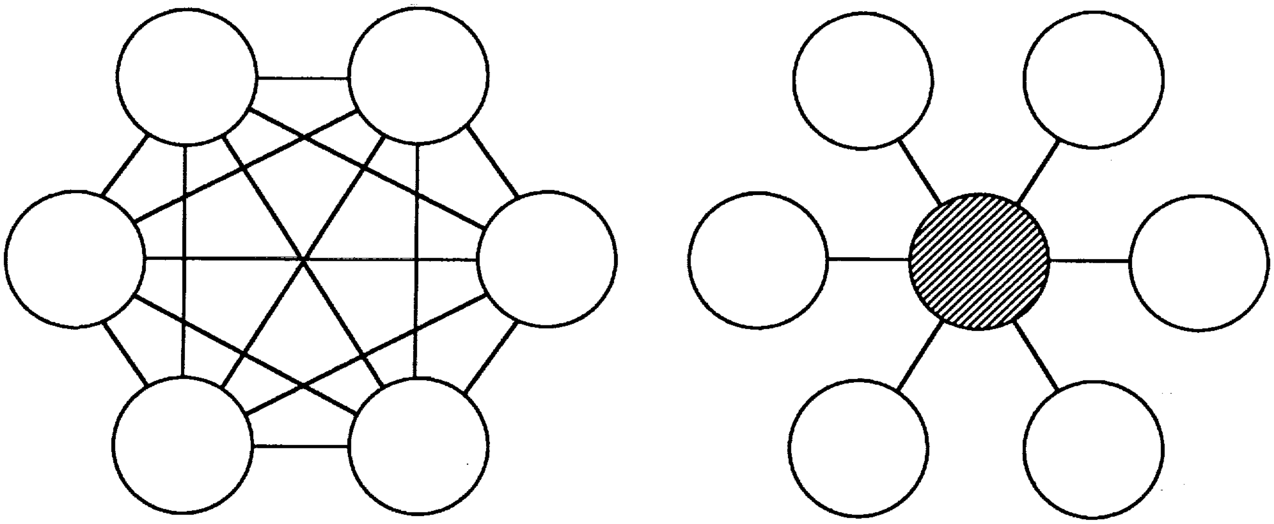


Figure 2. Software /Data Configuration Management

Large aerospace projects are costly and require the close coordination of many groups and disciplines. Participants include program management, design and manufacturing personnel, analysts, and the customer. Much of the configuration management problem arises because several project configurations must be managed.

The current baseline system configuration must be stored in the common database, for access by all. Typically one or two older versions of this baseline also exist. These versions must be accessible for backup and to maintain traceability. The system configuration must be formally controlled by the project chief engineer, and only authorized updates allowed.

Partial configurations and component models are also required for the designers and analysts, and versions of these may either lead or lag the baseline. Each analysis technology (dynamics, controls, thermal, etc.) and design area will require one or more models. For example, the thermal models almost always require different mesh refinement than the structural models, thermal insulation material may be non-structural in nature, etc. The analysis database must support these diverse needs, but still allow required data flow (e.g. thermal loads on the structural model to compute thermal deformations).

Configuration management requirements include data cataloging and query/retrieval, version control, database interfaces to project subgroups and to the outside world, and compatibility with the major project hardware, and software components.

IV. Interdisciplinary Data Transfer

Data is often passed between various disciplines. For example, results from a normal modes analysis may be passed to a controls or multibody code. In order to pass the data, ISM transfers the information to a common database where it can be accessed by other modules. The database is a logical environment for configuration management. In addition any number of modules can gain simultaneous access to the current system baseline or developmental versions. There are generic data manipulation tools that provide an extra dimension of flexibility (in addition to the discipline specific modules).

To illustrate some conventional paths, consider an optical beam analysis where a finite element structural model of the mirrors and associated structure is constructed in NASTRAN (or ANSYS). The system mass, stiffness, damping, node locations, generalized matrices, etc. can be stored in the ISM database. Optical sensitivity coefficients that relate structural displacements to beam direction are calculated and stored in the database. Sensor and actuator models are defined and stored. All or some of this data is combined for control design. Once the controller is designed a high fidelity simulation can be conducted using all the component models. Finally, detailed performance evaluation is done via simulation data post processing and wavefront propagation analysis. Each part of the system can be modeled several times with varying degrees of sophistication. See figure 3.

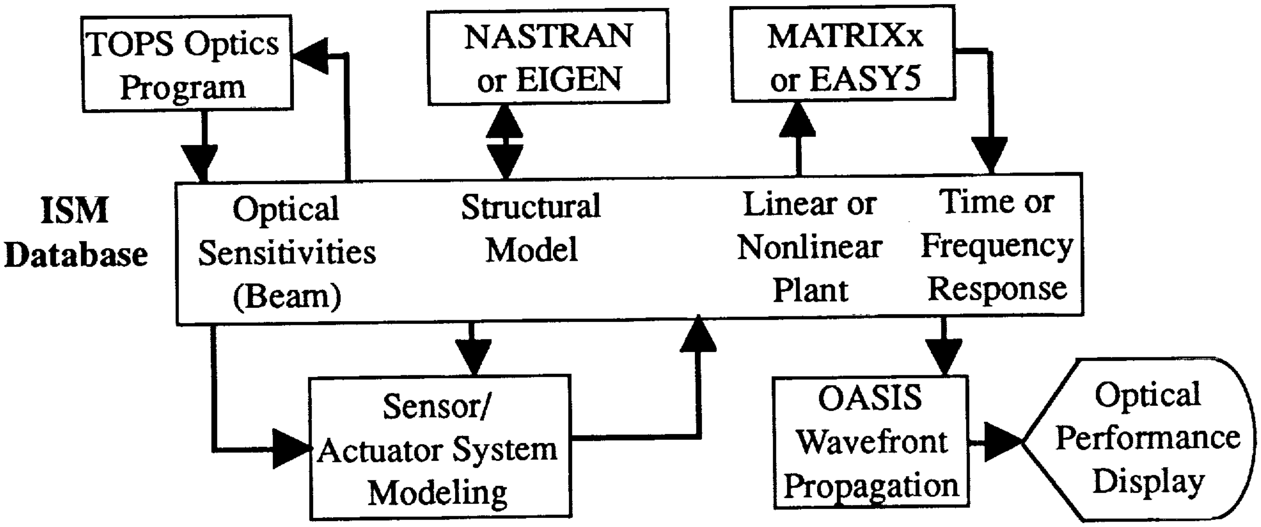

TOPS Optics Program
NASTRAN or EIGEN
MATRIXx or EASY5
ISM Database
Optical Sensitivities (Beam)
Structural Model
Linear or Nonlinear Plant
Time or Frequency Response
Sensor/ Actuator System Modeling
OASIS Wavefront Propagation
Optical Performance Display

Figure 3. Simple Integrated Modeling Blocls

Several examples of interdisciplinary data transfer are given. The first is the LEAP (Lightweight Exoatmospheric Projectile) ISM demonstration exercise. Results from dynamics and optics analysis are combined with disturbance data to predict line of sight jitter. Two structural models were converted from ANSYS and SAP into a combined NASTRAN dynamics model (11,000 DOF). The ISM EIGEN module was used to find the modes and frequencies using the NASTRAN generated nodal mass and stiffness matrices.

The ISM TOPS module was used to derive sensitivity coefficients. Once these tasks were completed ORACLS used the optical sensitivity matrix and the modes and frequencies of the structure to create a state space model which formed the basis for the system simulation in MATRIX$_x$. See figure 4.

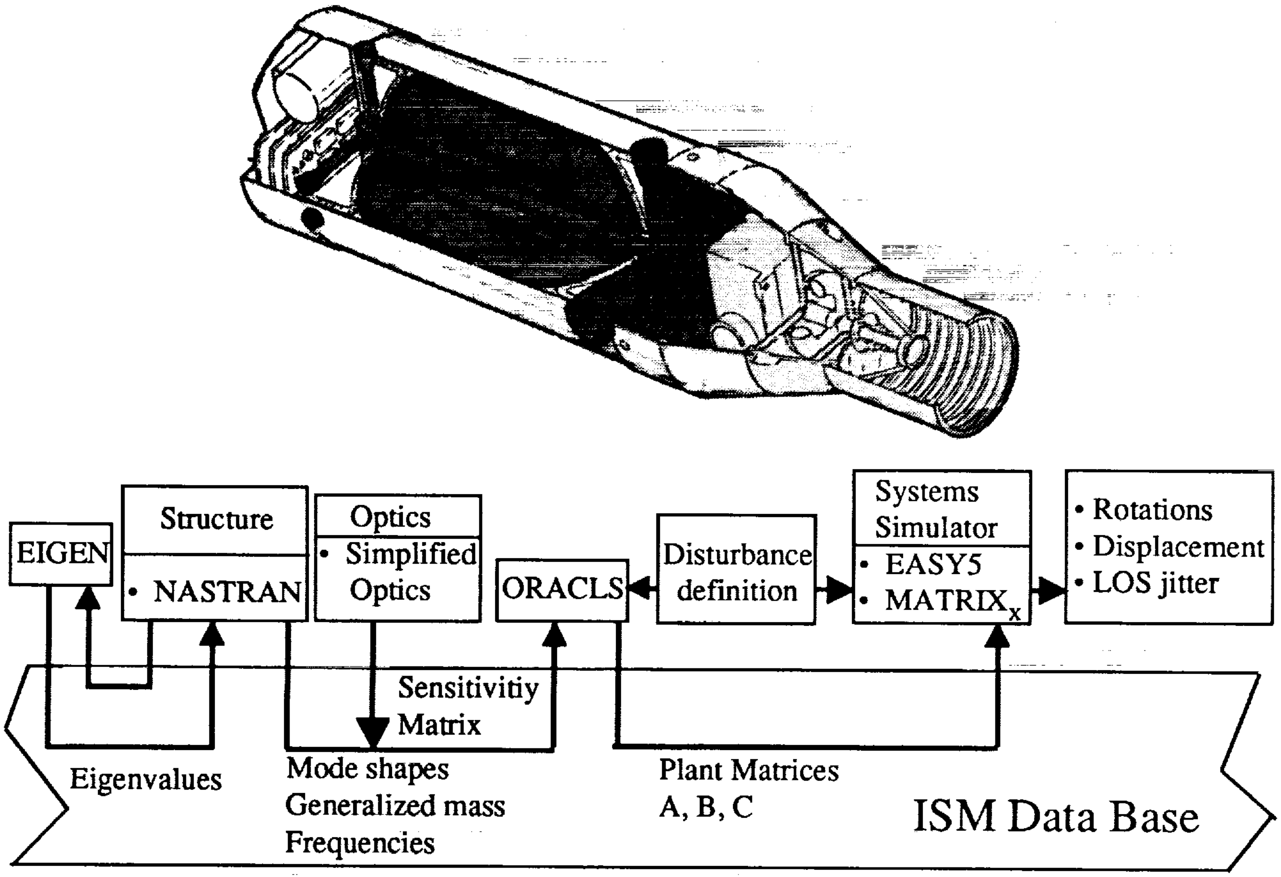
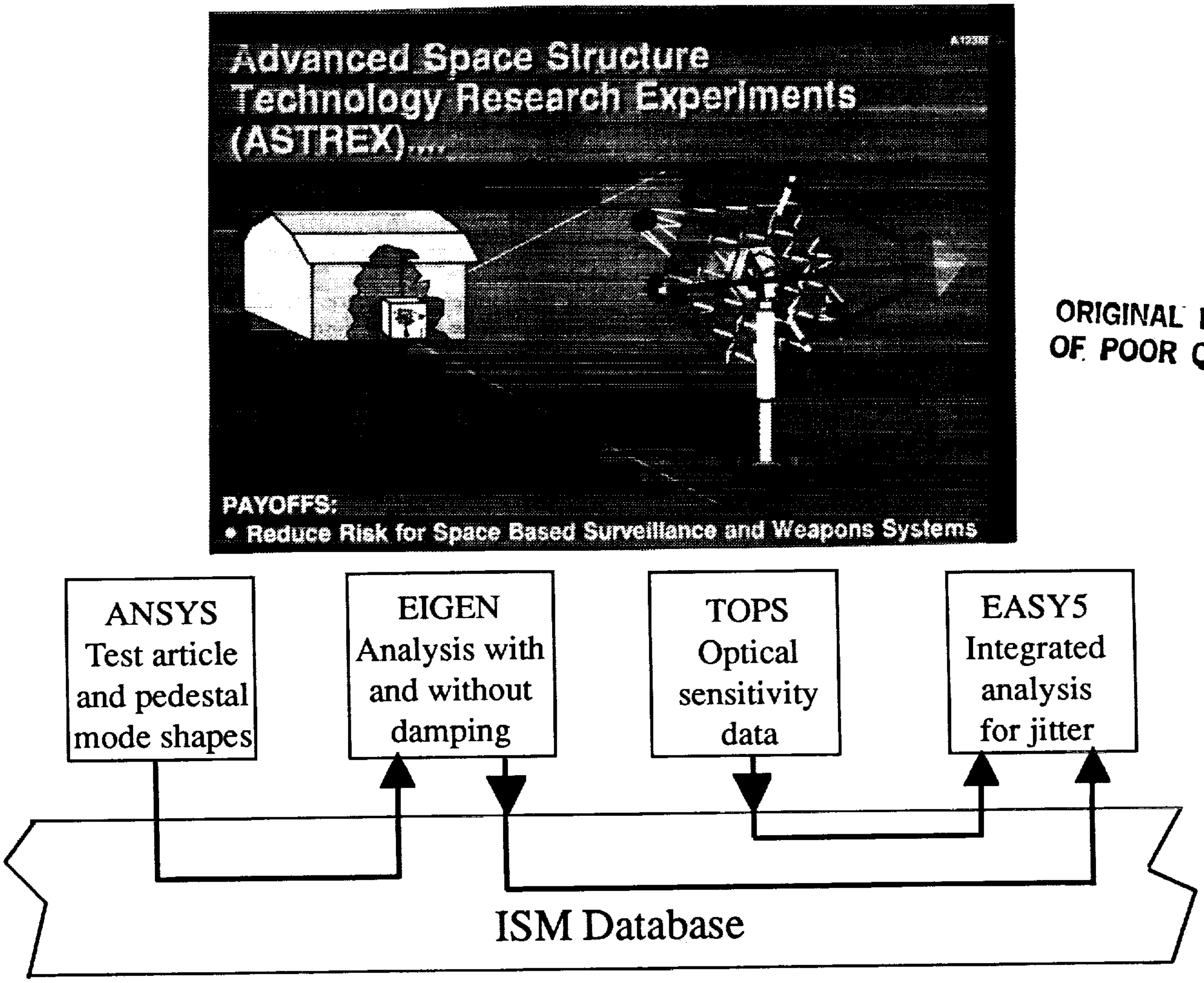


Figure 4. LEAP Interdisciplinary Data Transfer

Another example where interdisciplinary data transfer was used was in support of the design of the ASTREX (Advanced Space Strucutres Technology Research Experiments) test facility for the Air Force Astrodynamics Laboratory. The application involved a substructure merge of an air bearing pedestal, a test article, and an air cushion. The structural model was rooted and the modes and frequencies were used with the optical sensitivity coefficients obtained from TOPS, to conduct closed loop simulations in EASY5, a controls code. As in the LEAP analysis, data from structures, optics, and controls was combined in the final simulation. The ISM database minimized the burden on the analyst during the transfer of data. See figure 5.



Advanced Space Structure Technology Research Experiments (ASTREX)....

PAYOFFS:
- Reduce Risk for Space Based Surveillance and Weapons Systems

ANSYS
Test article and pedestal mode shapes
EIGEN
Analysis with and without damping
TOPS
Optical sensitivity data
EASY5
Integrated analysis for jitter
ISM Database

ORIGINAL PAGE IS OF POOR QUALITY

Figure 5. ASTREX Demonstation Analysis

The final example involves the Carbon-Carbon Wingbox analysis that combined data from a thermal analysis with a structural model. SINDA was using to perform the thermal analysis. The radiation exchange factors within the wingbox were calculated using the program TRASYS (Thermal Radiation Analysis System). The stresses were determined using ANSYS. Two separate meshes were created for the wingbox. One mesh was the finite element mesh for ANSYS, and the other was a finite element mesh that was then converted into a finite difference model for SINDA. The results from the thermal analysis were stored in the database. Using the module MIMIC (Model Integration via Mesh Interpolation Coefficients), the temperatures in the thermal model were interpolated to the nodes of the stress model. ISM commands were then used to write the data out to a file that is a suitable format for ANSYS. See figure 6.

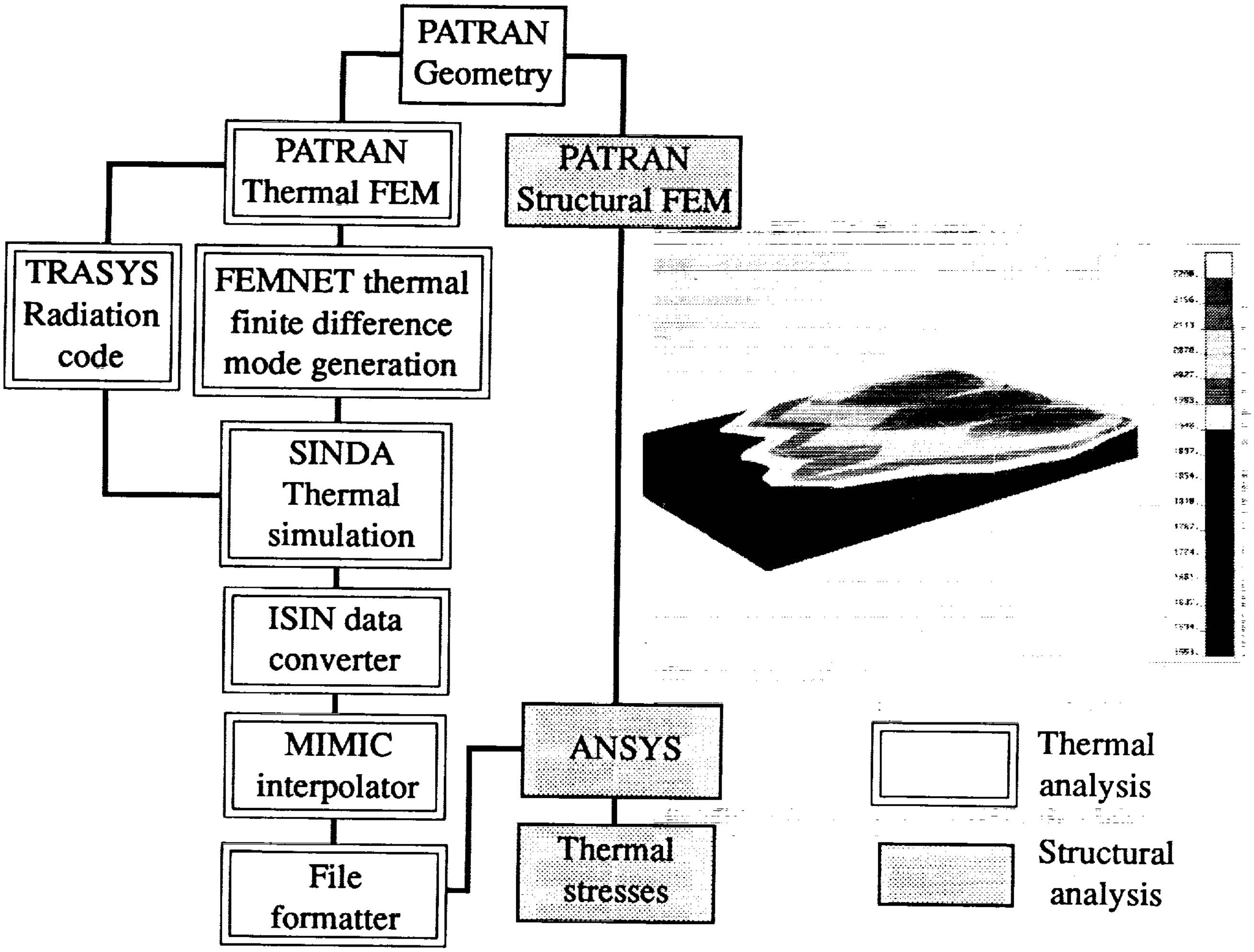
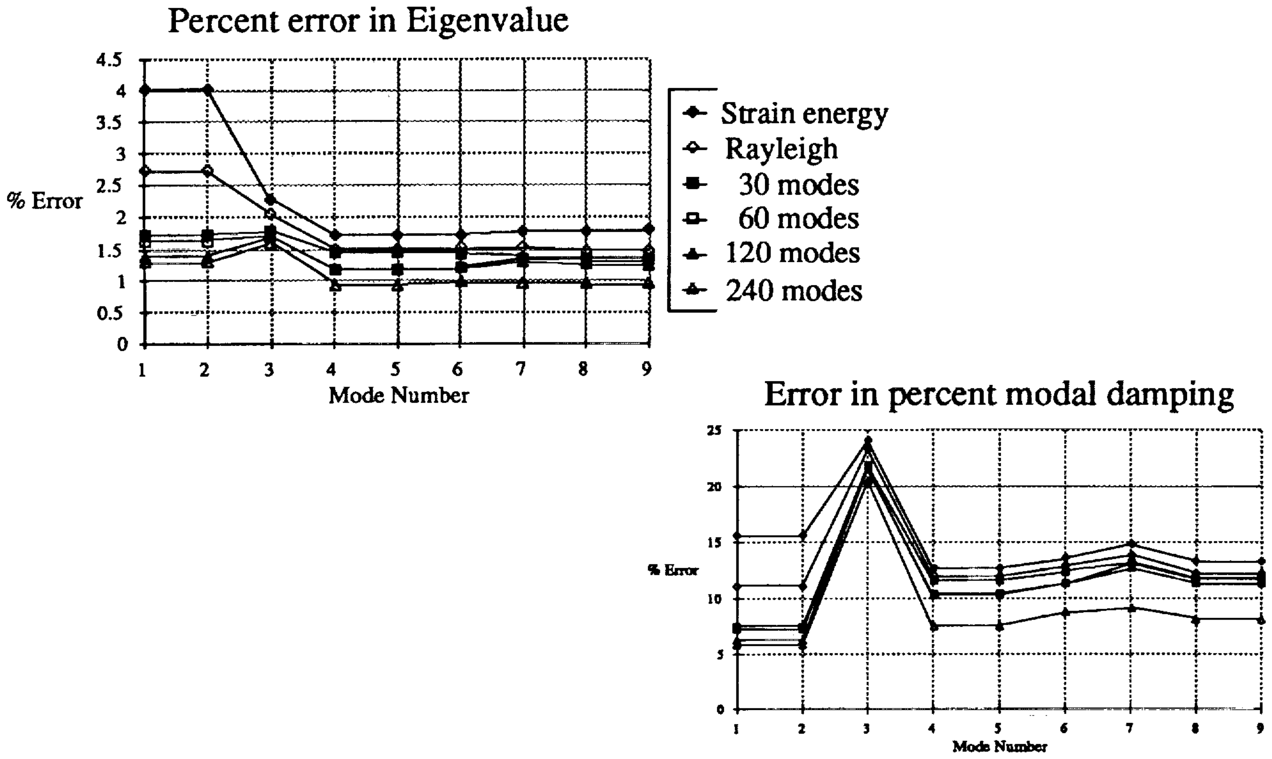


Figure 6. Carbon-Carbon Wingbox Analysis

V. Multiple Discipline Model Synthesis

The ability to transfer data between disciplines allows the formulation of multiple discipline models. Before considering multiple disciplines, note that the need for synthesizing models from incompatible software packages is common in multiple company contracts. The LEAP project is an example of combining models from two different organizations and two different software packages (an ANSYS model with a SAP model). The larger the model the greater the difficulty associated with the translation due to unique features found in each package. Controls, a discipline in itself, already performs simulations to predict system response. Many times these simulations are reasonable in size and are easily handled by one person. Typically a structural model within a simulation consists of a reduced model (a set of mode shapes). The assumption is that these mode shapes characterize the solution space of the system. However, when the control loop is closed, the system mode shapes change. In many cases the reduced set of mode shapes from the open loop system is not sufficient to characterize the mode shapes for the closed loop system. In this case it would be appropriate to combine the controls and structures models before reducing the model. The resulting closed loop basis can improve the accuracy in ensuing analysis.

An example of the errors that can be associated with an open loop basis is shown. The size of the structural model mandates the use of a reduced basis in most problems. The use of an inadequate basis will result in significant error. This is demonstrated with a passive damping problem and is analogous to the errors that can occur in a rate feedback problem. The graphs show the percent difference from the exact complex Lanczos solution, for the several approximate solution methods shown. The points that vary the most from the Lanczos solution are calculated using the Strain Energy Method. A slightly better solution is found using a Rayleigh proportional approximation. The next four solutions are performed using a reduced basis. The lowest modes for an undamped system are used as an approximate solution (a basis) to solve the damped problem. This is analogous to using open loop structural modes as a basis for a closed loop active system. In solutions using a reduced basis, 30, 60, 120, and 240 modes are used as the reduced basis. Increasing the number of modes improves the accuracy of the solution, but predictions of the performance (modal damping) based on a reduced set of modes result in large error. See figure 7.


Percent error in Eigenvalue

Error in percent modal damping

Figure 7. Error In The Use Of An Open Loop Basis

V. Distributed Processing

A distributed processor computing environment is key to productive large multidicipline simulations. The primary use of distributed processing will be to allow the ISM user to function within a workstation environment while processing data on the "optimum" computers. The use of the term optimum may simply refer to the computer on which a particular piece of software is hosted or may refer to the one with the most unused time.

A secondary yet powerful use of distributed processing is to spread the computational burden of a multiple discipline ISM simulation over several available processors. The simulation must be distributed in a rational manner. There are two key factors involved. It is important to minimize the communication between the processors. It is also important to balance the computational efforts required by each component with the capabilities of the processors at hand. See figure 8.

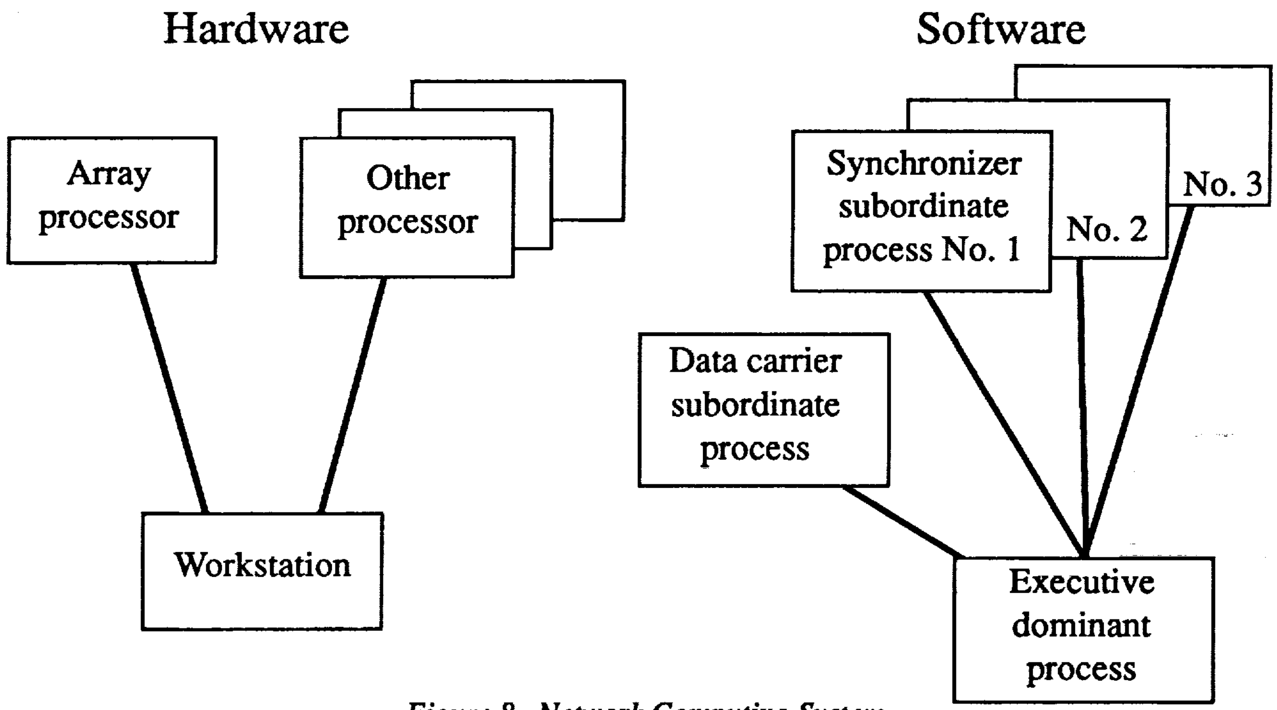

Hardware
Software
Array processor
Other processor
Workstation
Synchronizer subordinate process No. 1
No. 2
No. 3
Data carrier subordinate process
Executive dominant process

Figure 8. Network Computing System

A third capability that distributing processing allows is real time processing. ISM has not been designed with this in mind; however the tools to do this kind of work will be in place.

Distributed processing software enables the user to distribute the processing of a single application to several computers and maintain copies of data on several interconnected networks.

A demonstration of distributed processing has been completed.

VI. Summary

ISM is under development and is actively being used to support current multiple discipline simulations. Among the tools that are developed are those for controlling the project models and analysis. These include a database where key models and information can be stored and a protection scheme to limit the access and ability to change the database. In addition, utilities and interfaces exist to transfer data between various analysis modules (for example, between thermal analysis and structural analysis, or between optics and controls). These tools will facilitate the multidiscipline analysis that can be used to analyze large COSI (control, optics, structure interaction) problems. Distributed processing is utilized to make better use of computer facilities.

Enhanced Modeling Features Within TREETOPS

R. J. VanderVoort
and
Manoj N. Kumar
Dynacs Engineering Co., Inc.
Clearwater, FL 34623

August 16, 1989

MOTIVATION

The original motivation for TREETOPS was to build a generic multi-body simulation and remove the burden of writing multi-body equations from the engineers. The motivation of the enhancement was twofold: 1)to extend the menu of built in features (sensors, actuators, constraints, etc) that did not require user code and 2)to extend the control system design capabilities by linking with other government funded software (NASTRAN and MATLAB).

These enhancements also serve to bridge the gap between structures and controls groups. It is common on large space programs for the structures groups to build hi-fidelity models of the structure using NASTRAN and for the controls group to build lower order models because they lack the tools to incorporate the former into their analysis. Now the controls engineers can accept the hi-fidelity NASTRAN models into TREETOPS, add sensors and actuators, perform model reduction and couple the result directly into MATLAB to perform their design. The controller can then be imported directly into TREETOPS for non-linear, time-history simulation.

SUMMARY OF ENHANCEMENTS

TREETOPS is more than a time-history simulation. It is a suite of programs oriented toward design and analysis of embedded control systems for flexible spacecraft. The enhancements to TREETOPS can be divided into two categories. *Internal* enhancements add to the menu of time history simulation features and *External* enhancements provide links to other software and provide stand alone utility features.

Internal Enhancements

- Orbital Environment–Standard NASA models of the earth's magnetic field and atmosphere have been added to TREETOPS. You can specify the "shape" of each body and appendage, the orbit and certain atmospheric paramenters and TREETOPS will in turn compute the aerodynamic drag forces. You can also include gravity gradient moments on each body in your model through a simple menu option. The magnetic field operates through magnetic actuators to produce controllable moments at any point on the structure.

- Actuators–The new actuators include reaction wheels, single gimbal control moment gyros (SGCMG), double gimbal CMG's (DGCMG) and seven variations of motor drive actuators.

  The previous actuators were simply input devices that applied a force or moment. The reaction wheels, SGCMG's and DGCMG's have their own dynamics that are closely coupled to the structure dynamics. Furthermore, you can select friction models for the gimbals, such as the Dahl friction model, that are described by non-linear differential equations. When you select one of the actuators from the menu TREETOPS will form the associated differential equations augment them to the structure equations and simultaneously solve the complete set.

  The motor drive actuators simulate a wide variety of electric motors driving through gear trains. These models include gear train compliance, backlash, coulomb friction and no-back-drive options. Seven varieties of motor drive actuators have been added to TREETOPS representing different degrees of complexity.

- Devices–the new devices include cables, brakes and locks.

  A cable is connected between any two nodes on the structure and constrains the distance between the two nodes to be less than or equal to the cable length. An alternate way of looking at a cable is that node 2 is constrainted to be within or on a sphere centered at node 1. The sphere radius is equal to the cable length.

  A brake is a device that mounts on a hinge degree of freedom and produces a force that can inhibit motion but not cause motion. The braking force is an input command, just like other actuators, and when the velocity goes to zero a constraint is instituted just like a coulomb damper.

  Locks are like brakes except that they mount between two nodes rather than on a hinge axis.

- Manipulators–Two enchancements have been made to facilitate the simulation of manipulators. The first is a grapple option where the manipulator tip can attach to another point on the structure and create a closed tree topology. The second option is a contact option where the manipulator tip makes contact with a surface producing equal and opposite forces but does not form a closed loop constraint.

- Cut joints–A cut joint facilitates the modeling of closed tree topologies. It lets you constrain any combination of the relative degrees of freedom between two nodes and is specified just like a TREETOPS hinge. A "closed loop constraint" can be thought of as a subset of cut joints because it constrains all 3 relative translations between two nodes but allows all 3 relative rotations.

- Flexibility–Previous versions of TREETOPS had a restriction that the deformation of any flexible body must be zero at the attach point node. This restriction has been removed.

External Enhancements

As stated previously, TREETOPS is a suite of programs for performing analysis, design and simulation of spacecraft control systems. By linking TREETOPS with NASTRAN and MATLAB, the job is now much easier. All three programs were developed under government funding and are available free or at low cost. Furthermore, all 3 programs run on UNIX workstations which brings the cost of performing complex design tasks down to an acceptable level.

- TREEFLEX–A new program that links NASTRAN and TREETOPS. Each flexible body in a TREETOPS simulation has certain volume integrals that are required and others that are optional. TREEFLEX will read data from a NASTRAN output file, compute all required volume integrals and write them in the proper format. All data files are in an ASCII format to facilitate transfer of data from one computer to another or from one subcontractor to another. Options are available in TREEFLEX to specify the NASTRAN modes that will be retained, the number of optional terms that will be retained and the damping for each mode. In addition, the computation of augmented modal data has been further automated making it easier to "augment" the modes of inboard bodies to account for the mass and inertia of outboard bodies.

- TREESEL– A model reduction program. A common problem in multibody systems is that the number of system modes which is equal to the sum of individual component modes becomes very large. TREESEL uses a modified Component Cost Analysis (CCA) method to rank individual component modes based on their contribution to overall system costs. The modes having the least impact on system response are then deleted using TREEFLX. In other words, the substructure modes are selected based on system level cost criteria. All model reduction is done in physical coordinates.

- Link to MATLAB–MATLAB is a public domain program, developed under government funding. It provided the basis for several well-known commercial controls analysis programs. We have incorporated the public domain version into the TREETOPS suite because the user interface is well-known and widely accepted. Portability and low cost were emphasized in this link. An ASCII file format was adopted to store linear system data (ABCD matrix quadruples). TREETOPS writes linear plant models in this format and reads controller models in the same format from the same file. The link to MATLAB was completed by simply adding two functions, one for reading quadruples and the other for writing them.

- TREEFREQ–Frequency domain analysis. This is a post-processor that reads linear system data and performs frequency domain analysis. Poles and zeros are computed, frequency response in computed and plotted in Bode, Nichols or Nyquist format.

- Utility Function–Each of these utilities operates on linear system data.

  - EIGEN computes and prints system Eigen values.
  - LINC will combine several linear systems into one.
  - BL2M converts a block diagram controller into matrix form.
  - SPRT produces a pretty printout of ABCD System quadruples.

EXAMPLE PROBLEM NO. 1 - AXAF

The Advanced X-ray Astrophysics Facility (AXAF) is a proposed NASA spacecraft for the 1990's. It *is* a multi-body spacecraft.

Two different models of the spacecraft were used for two different controller design Tasks. The first task was basic attitude or pointing control in the presence of external disturbances. A three body model was chosen with a rigid core body and two flexible solar panel appendages as sketched in figure 1. A simple proportional-derivative controller, defined in block diagram form, used reaction wheels for actuators and attitude and rate sensors for feedback. Disturbances included aerodynamic drag, gravity gradient and rotating machinery vibrations. Note that no user supplied code is required and the problem is completely defined by the .INT and .FLN files. A typical line-of-sight disturbance response is shown in figure 2.

The second task was to design a closed loop controller for calibrating the optical system. A two body model was used with the optical bench and solar panels combined to form the first rigid body and the High Resolution Mirror Assembly (HRMA) formed the second body as shown in figure 3. Six motor drive actuators in a tripod configuration gave control of all 6 relative degrees of freedom. Relative attitude and rate sensors simulated the optical sensors which are not yet defined. A user controller was incorporated to illustrate the technique of incorporating user supplied code. The pointing system was operating in parallel with the optical calibration system. The .INT file and user controller listing completely defines this problem. The same system model was used to design the actuator servo loop and then to design the controller loops. Figure 4 shows the relative velocity frequency response between two motor drive actuator end points in response to a force input. As you would expect the transfer function is approximately 1/ks and the effect of the other 5 actuators constraining the 6 relative degrees of freedom is negligible for this particular example.

EXAMPLE PROBLEM NO. 2 - SPACE STATION

This example demonstrates the NASTRAN/TREETOPS interface and model reduction procedures as they apply to large dimension, real-world problems.

The NASTRAN model known as the MB-1 configuration was developed by the structures group at NASA Langley Research Center and is shown in figure 5. The NASTRAN model has 1062 degrees of freedom but only the first 89 modes for each body were computed within NASTRAN giving us the first rough cut model reduction. In the second stage of model reduction TREEFLEX was used to arbitrarily select the first 15 modes for body 1 and the first 50 modes for body 2. This produced a baseline TREETOPS model with 72 DOF's. (6 Rigid body for hinge 1, 15 flex for body 1, 1 rigid for hinge 2 and 50 flex for body 2). The third step was to perform component cost analysis using TREESEL and select a reduced subset of modes based on relative importance. This step reduced the model from 72 to 41 degrees of freedom and further reduction is quite likely for the control design problem. A comparison of the transient response for the 72 and 41 DOF models is presented in figure 6.

IMPORTANCE OF WORK TO COMMUNITY

For many years government funded software has been available to model, analyze and design complex spacecraft systems. Now these software codes have been integrated. With NASTRAN you can create and analyze complex finite element models. TREETOPS can combine these NASTRAN single body models into a multi-body model, add sensors and actuators and linearize the total for use in MATLAB. The controller can be designed in MATLAB and passed back to TREETOPS for verification via non-linear simulation.

Strict adherence to FORTRAN 77 standards enhances TREETOPS portability. All software has been installed on engineering workstations.

No user supplied code is required.

FUTURE DIRECTION

The major complaint against generic, multi-body simulations is run time. We are currently using three approaches to reduce run time: Order-N algorithms, symbolic programming and parallel processing. Order-N and symbolics can be done without compromising portability but parallel processing requires the proper hardware. Two orders of magnitude reduction in run time is feasible without resorting to approximate solutions. This puts a large class of problems in the realm of real time simulation.

The second area of advancement is graphical interfaces. A menu/mouse method of building models is very desirable but hardware dependent. NASA funded software is already available for 3-D solid modeling and building NASTRAN input files. The same 3-D model built with the graphical preprocessor can be used in a graphical post processor to produce movies from your simulation. The simulation merely supplies the coordinate transformation for each of the graphic objects created with the preprocessor.

Last, but not least, a collection of MATLAB macros is being collected to solve the common controls problems.

C-2

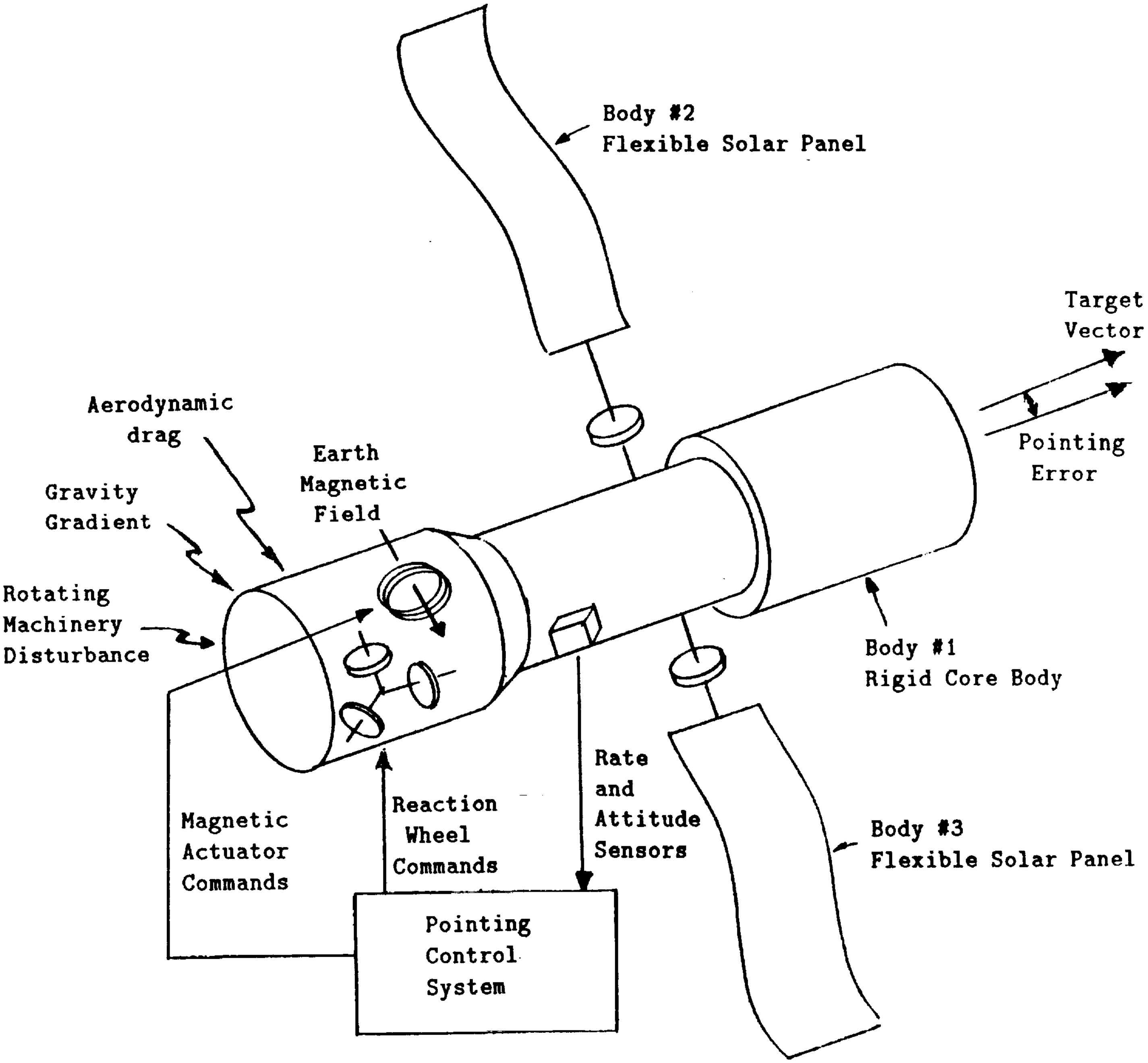


Figure 1 A 3-body representation of the AXAF spacecraft.

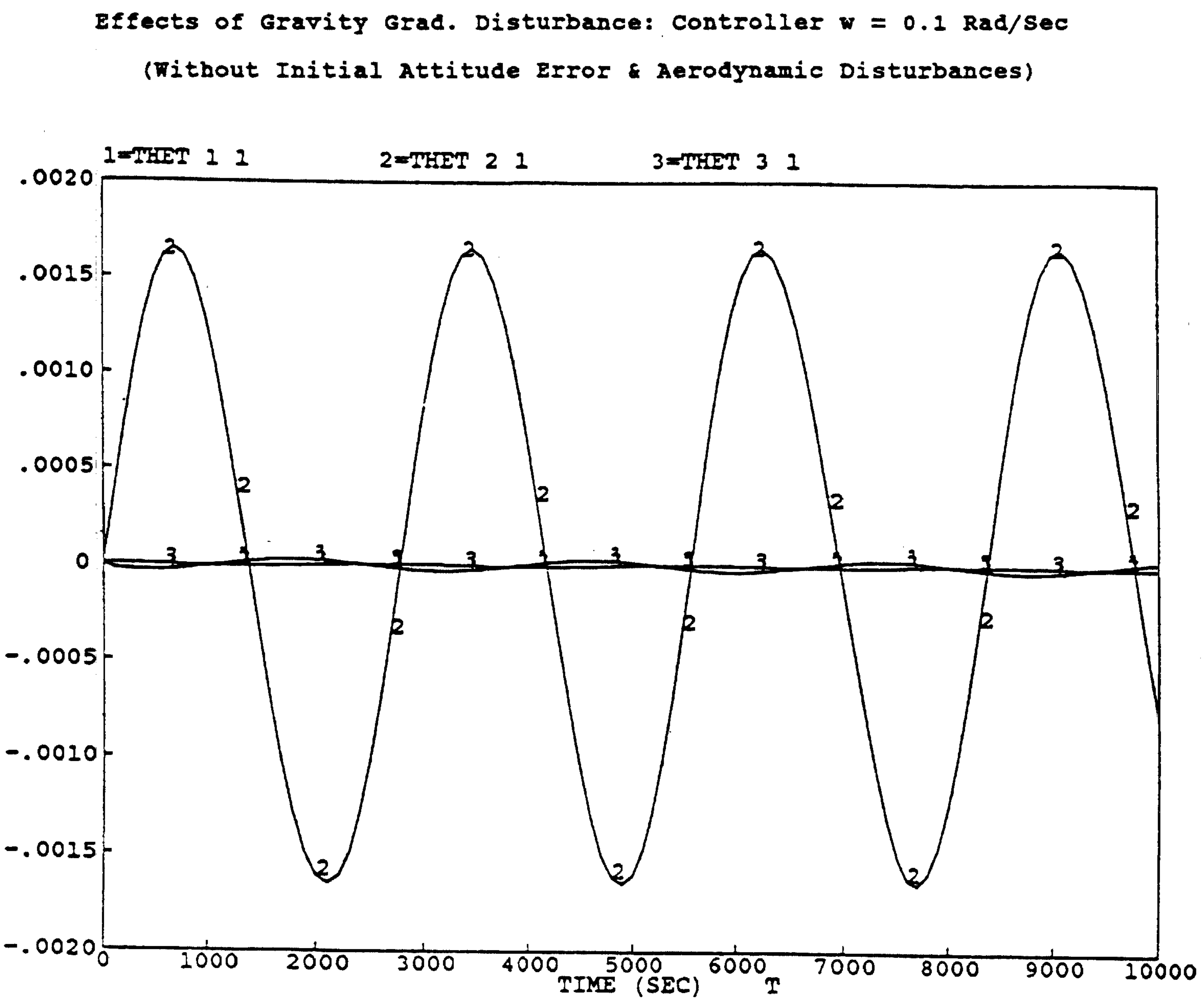


Figure 2 AXAF pointing error response to disturbance inputs.

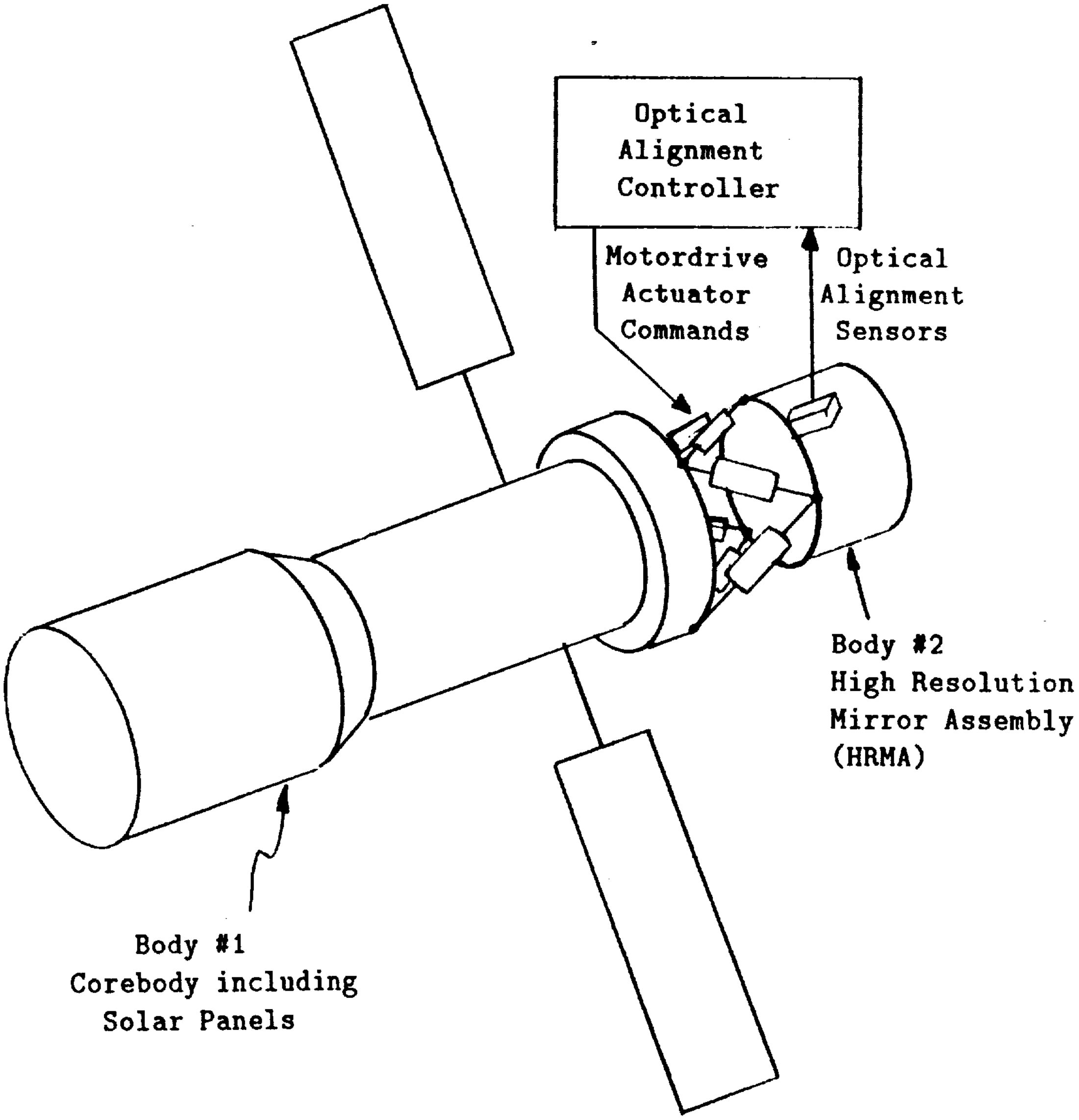
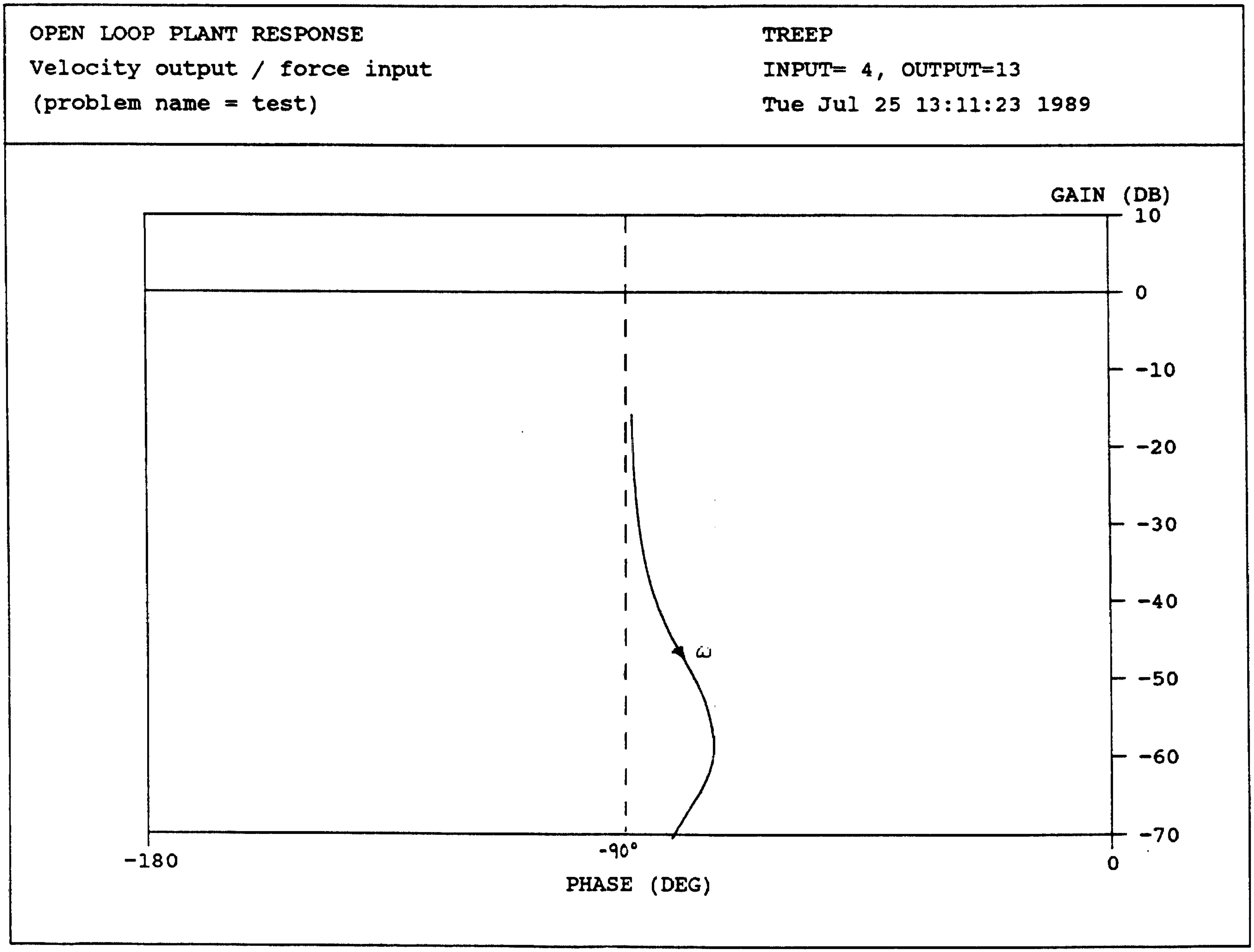


Figure 3 A 2-body representation of the AXAF spacccraft.



OPEN LOOP PLANT RESPONSE
Velocity output / force input
(problem name = test)

TREEP
INPUT= 4, OUTPUT=13
Tue Jul 25 13:11:23 1989

Figure 4 AXAF typical open loop frequency response.

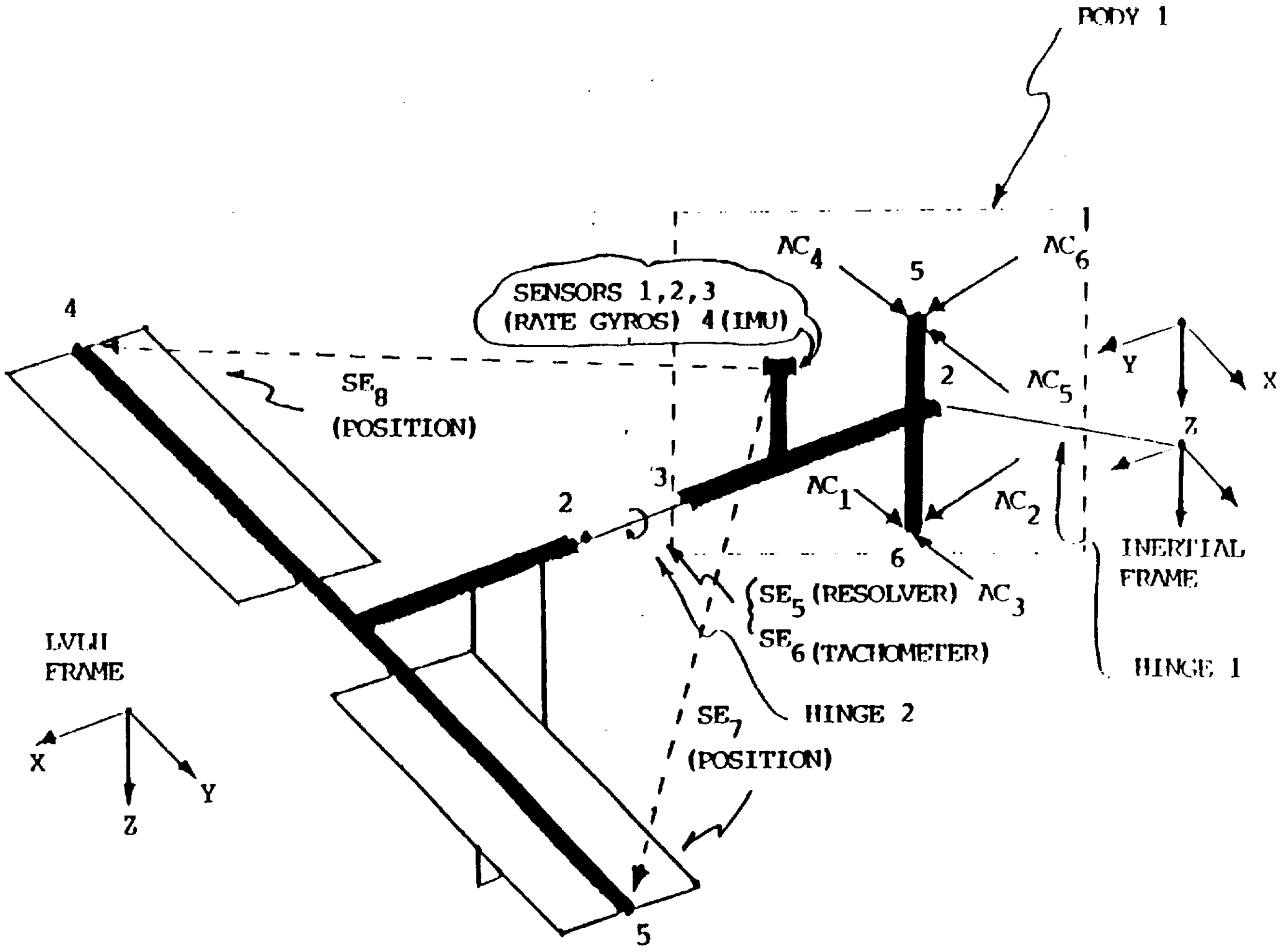


Figure 5 Space Station MB1 configuration.

ORIGINAL PAGE IS
OF POOR QUALITY

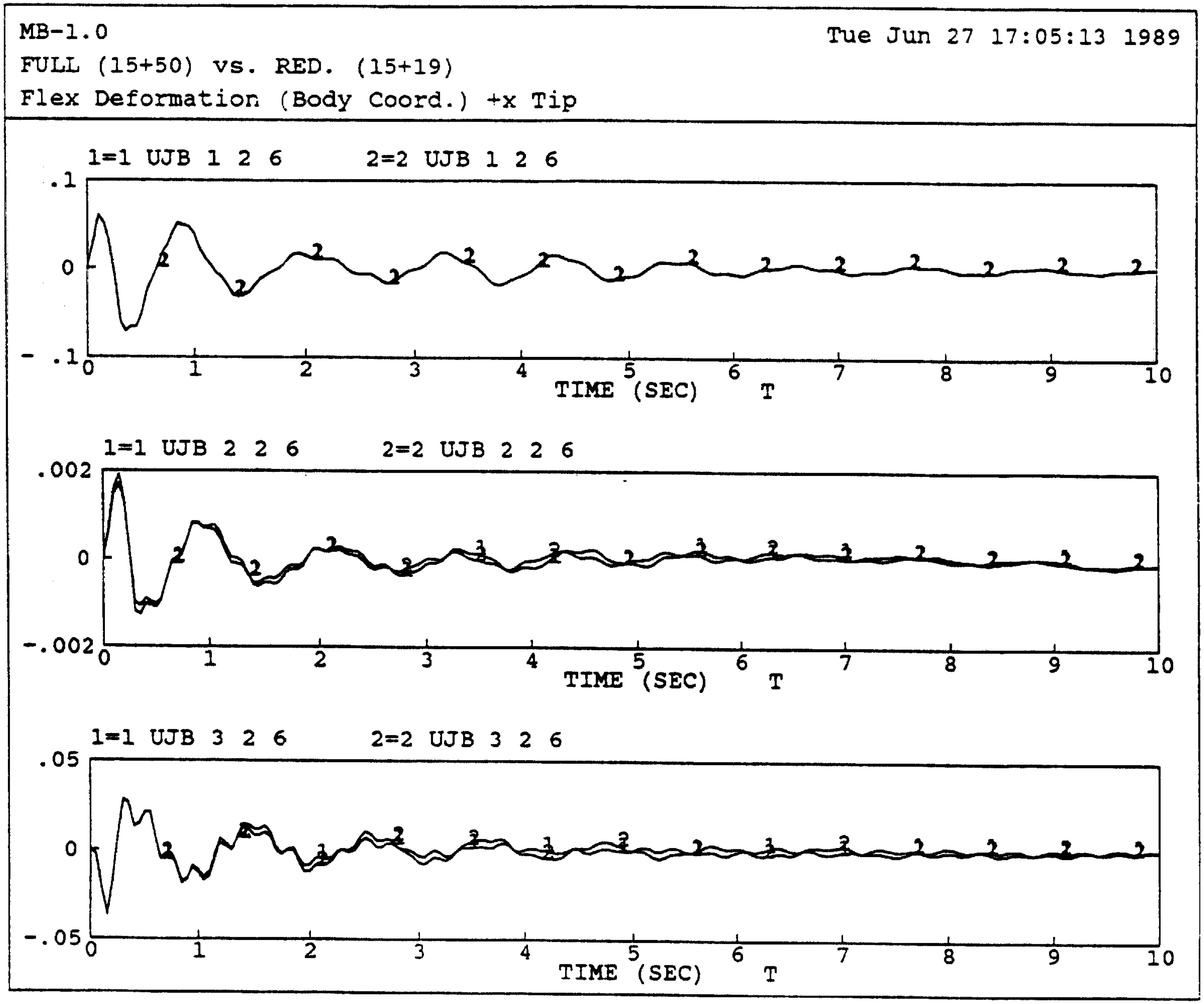


Figure 6 A comparison of Space Station reduced order models.

N90-23048

Implementation of Generalized Optimality Criteria in a Multidisciplinary Environment

R.A. Canfield and V.B. Venkayya
WRDC/FIBRA
WPAFB, OH 45433-6553

INTRODUCTION

The aerospace industry has begun to incorporate optimization methods into their design procedures in recent years. The Automated Structural Optimization System (ASTROS),[1] is an example of an automated multidisciplinary tool to assist in the preliminary design or modification of aircraft and spacecraft structures. The Air Force has distributed ASTROS to more than 90 organizations in the aerospace community in the last 18 months. The philosophy behind this system is to integrate proven and reliable analysis methods with numerical optimization using modern executive system and database management concepts (Fig 1). The engineering disciplines include structural analysis, aerodynamic loads. aeroelasticity, control response, and structural optimization.

The structural analysis, based on and highly compatible with NASTRAN,[2] uses the finite element method to calculate: deflections and stresses from static, thermal, or gravity loads; normal modes; and transient or frequency response due to time dependent loads including gust loads. The air loads module, an advanced paneling method based on USSAERO-C,[3] calculates flexible loads and determines a trimmed configuration. The aeroelastic module employs the Doublet Lattice method[4] for subsonic unsteady aerodynamics and the Constant Pressure method[5] for the supersonic regime. Flutter solutions are found using the PK method. When response quantities in any of these disciplines are constrained, the sensitivity analysis calculates analytic derivatives for each active constraint. These derivatives are fed to the optimization module which employs the Automated Design Synthesis program[6] to minimize structural weight. The final design's dynamic response in the presence of a given control system can be simulated by the control response module. The ability to simultaneously consider multiple boundary conditions, flight conditions, store loadings, and disciplines uniquely qualify ASTROS for structural design in a production environment.

The structural optimization methodology in ASTROS utilizes design variable linking and approximation concepts[7] to efficiently handle large problems. Even so, the maximum number of design variables that can be handled effectively is no more than a few hundred. In the past the aircraft industry has handled thousands of variables by using optimality criterion methods, *e.g.*, Stress Ratio Method: however. their application in industry has been limited to considering a single discipline at a time. *i.e.*, stresses, or displacements, or flutter alone. Fleury and Schmit demonstrated the equivalence of optimality criterion and mathematical programming methods,[8] and more recently Venkayya formulated a generalized optimality criteria approach for general mathematical functions.[9] Present efforts meld

Fleury and Schmit's dual solution method and Venkayya's compound scaling algorithm in ASTROS to handle multidisciplinary structural design with thousands of variables.

MATHEMATICAL STATEMENT OF THE DESIGN PROBLEM

The objective is to minimize the weight (or equivalently, mass)

$$\min W(\mathbf{v}) \tag{1}$$

subject to m normalized constraints

$$g_j(\mathbf{v}) \leq 0; \qquad j = 1, \ldots, m \tag{2}$$

and side constraints on the n design variables, $\mathbf{v}$

$$v_i^L \leq v_i \leq v_i^u; \qquad i = 1, \ldots, n. \tag{3}$$

The constraint functions are formed by normalizing the response quantities, $z_j(\mathbf{v})$ by their allowable values, z_{bj}.

$$g_j = \pm \left(\frac{z(\mathbf{v}) - z_{bj}}{|z_{bj}|} \right) \tag{4}$$

The finite element cross-sectional properties (areas of rods and thicknesses of membranes), **d**, are controlled by the design variables, **v** through a linking matrix, **T**.

$$\mathbf{d} = \mathbf{T}\mathbf{v} \tag{5}$$

The approximation concepts develop first order Taylor series for the constraint functions in the reciprocal design variable space.

$$x_i = \frac{1}{v_i} \tag{6}$$

When a row of the linking matrix, **T**, in eq (5), has only one non-zero element, a single design variable controls one or more finite elements. This *physical linking* can accommodate the use of the reciprocal variables defined in eq (6). When a row of **T** has more than one non-zero element, the design variables can be interpreted as coefficients that scale some shape function defined by a column of the linking matrix. Because this *shape function* design variable may be zero, reciprocal variables cannot be used, in which case $x_i = v_i$.

After each complete analysis of the structure, an approximate sub-problem is formed using a first order Taylor series to represent the constraint functions. For the finite elements used in ASTROS the objective function is a non-linear but explicit function of the reciprocal variables.

$$\min W(\mathbf{x}) = \mathbf{x}_o + \sum_{i=1}^{n} \frac{w_i}{x_i} \tag{7}$$

The approximate constraints are linear in the reciprocal variables.

$$\bar{\mathbf{g}} = \bar{\mathbf{g}}_o + \mathbf{N}^t\mathbf{x} \tag{8}$$

where $N_{ij} = \frac{\partial g_j}{\partial x_i}$ is the gradient matrix and $\bar{g}_{oj} = \mathbf{N}_j^t\mathbf{x}_o$. To stay within the region of validity for the Taylor series, move limits are applied to the design variables

$$\frac{x_i}{f} \leq x_i \leq f x_i \tag{9}$$

where the move limit factor, f, is set to two as a default in ASTROS.

THE DUAL PROBLEM

Optimality criterion methods are derived using the Lagrangian function which augments the objective function with a summation of terms that weight each constraint by a *Lagrangian multiplier* (later referred to as a *dual variable*).

$$L(\mathbf{x}, \lambda) = W(\mathbf{x}) + \lambda^t \mathbf{g}(\mathbf{x}) \tag{10}$$

Application of the well-known *Kuhn-Tucker* conditions to this convex and separable approximate sub-problem results in a min-max optimization problem. The solution of this *dual problem* also defines the global optimum of the original *primal problem*.[8]

$$\max_{\lambda} L(\lambda) = \sum_{i=1}^{n} \min_{\mathbf{x}} \left[w_i / x_i + \lambda^t \bar{\mathbf{g}}(\mathbf{x}) \right] \tag{11}$$

where $\mathbf{x}$ is found explicitly from the condition that $\frac{\partial L}{\partial x_i} = 0$ for any given λ as

$$x_i = \sqrt{\frac{x_i}{\mathbf{N}_i \lambda}} \tag{12}$$

for all free x_i, *i.e.*, those not at their lower or upper bounds.

This dual problem is an unconstrained maximization problem. The approximate constraints are derivatives of the objective (Lagrangian) function.

$$\frac{\partial L}{\partial \lambda_j} = \bar{g}_j \tag{13}$$

The advantage of solving eq (11) in place of eq (7) is that the dimensionality of the problem is reduced from n design variables to m_a dual variables corresponding to only the strictly active constraints. By definition the Lagrange multipliers are zero for inactive constraints and positive for active constraints ($g = 0$). Whenever the number of positive dual variables (active constraints) is fewer than the number of primal (design) variables, the dual problem is more efficient to solve. One of the numeric difficulties, however, is the problem of terminating the optimization when eq (13) is zero. Other termination criteria (*e.g.*. relative change in the objective) in are often satisfied before the approximate constraints are within a tolerance acceptable for the primal problem. Fleury and Schmit accounted for this potential pitfall in two ways. First, their dual solver "does not seek the maximum of the dual function along the [search] direction S, rather it is designed to assure that either: (a) a regular Newton unit step is taken without any change in the set of free primal variables: or (b) the move distance is selected so that the value of the dual function increases. Second, they offer the option of reducing the size of the dual space by using zero order approximations—side constraints on the primal variables based on the element's stress ratio—for some stress constraints." Numeric difficulties occur less often when there are fewer dual variables.

Another approach would be to solve the dual problem as a constrained optimization problem in order to explicitly require the objective's derivatives not be greater than zero. The required derivatives of the approximate constraints with respect to the dual variables were derived in Ref 8.

COMPOUND SCALING ALGORITHM

The approach used here for preventing constraint violoations is different: a compound scaling algorithm[10] to guarantee the approximate constraints are satisfied. In formulating a generalized optimality criteria Venkayya defines the target response ratio as

$$\beta_j \equiv \frac{z_{bj}}{z_j} \tag{14}$$

and a sensitivity parameter,

$$\mu_{ij} \equiv \frac{N_{ij} x_i}{z_j}. \tag{15}$$

For each constraint the design vector is partitioned into groups based on the sign of the constraint derivative.

$$\mu_j^N = -\sum_{i=1}^{n} \min\left[0, \mu_{ij}\right], \quad \mu_j^P = \sum_{i=1}^{n} \max\left[0, \mu_{ij}\right] \tag{16}$$

Each partition of the design vector can be scaled by a factor, Λ, to be determined.

$$\mathbf{x}^N = \Lambda_j^N \mathbf{x}_o^N, \quad \mathbf{x}^P = \Lambda_j^P \mathbf{x}_o^P \tag{17}$$

Substituting these definitions into eq (8) yields an approximation of the target response ratio as a function of the two scale factors.

$$\beta_j(\Lambda_j^N, \Lambda_j^P) \approx 1 - \mu_j^N(\Lambda_j^N - 1) + \mu_j^P(\Lambda_j^P - 1) \tag{18}$$

Contours of the approximate target response ratio can be plotted as a function of the two scale factors. The desired target response lies on the contour line for $\beta = 1$ shown in Fig 2. Selecting a unique pair of scale factors requires a second equation in addition to eq (18). For reference, point S in Fig 2 represents simple scaling where the entire design vector is scaled by a single factor. The original derivation of the scale factors in Ref 10, represented by points A and B, assumed that scaling either partition alone would achieve the target response. The current approach is to select the point M on the scaling line that minimizes the distance to the current design at point O. This is the point closest to the small region about point O where the Taylor series approximation is accurate. The solution for the scale factors is

$$\Lambda_j^N = \left(\beta_j^N\right)^{\frac{-1}{\mu_j^N}}, \qquad \Lambda_j^P = \left(\beta_j^P\right)^{\frac{1}{\mu_j^P}} \tag{19}$$

where the partial target response ratios are defined as

$$\beta_j^N = 1 + \frac{\mu_j^N}{\mu_j^N + \mu_j^P}\left(\beta_j - 1\right), \qquad \beta_j^P = 1 + \frac{\mu_j^P}{\mu_j^N + \mu_j^P}\left(\beta_j - 1\right). \tag{20}$$

Two tables are used to select scale factors for multiple constraints. The Scale Factor Table is simply formed using the scale factor for the corresponding partition of the design vector.

$$\Lambda_{ij} = \begin{cases} \Lambda_j^N & \text{if } \mu_{ij} < 0 \\ \Lambda_j^P & \text{if } \mu_{ij} > 0 \\ 1.0 & \text{if } \mu_{ij} = 0 \end{cases} \tag{21}$$

The Scale Factor Assignment Table is formed from the sensitivity factors, except that the total differential term is normalized by the allowable value instead of the response value.

$$t_{ij} = \left| \frac{N_{ij} x_i}{z_{bj}} \right| = \left| \mu_{ij} \beta_j \right| \tag{22}$$

A scale factor is selected for each design variable according to the following three cases.

$$\begin{array}{lll}
\text{Case 1} & \overset{\text{all}}{\underset{j}{}} N_{ij} \leq 0, & \lambda_i = \overset{\text{max}}{\underset{j, N_{ij} \neq 0}{}} (\Lambda_{ij}) \\
\text{Case 2} & \overset{\text{all}}{\underset{j}{}} N_{ij} \geq 0, & \lambda_i = \overset{\text{min}}{\underset{j, N_{ij} \neq 0}{}} (\Lambda_{ij}) \\
\text{Case 3} & \overset{\text{max}}{\underset{j}{}} t_{ij}, & \Lambda_i = \Lambda_{ij} \begin{cases} \text{if } w_i(\Lambda_{ij} - 1) < 0 \\ \text{or } z_j > z_{bj} \end{cases}
\end{array}$$

The above three cases replace the rules presented in Ref 10 and simplify the scale factor selection procedure by avoiding the special cases of simple scaling and compound scaling with a single constraint. Also, starting from a uniform design is unnecessary.

Each design cycle in ASTROS consists of a complete multidisciplinary analysis followed by redesign based on the approximate problem. The new generalized optimality criterion algorithm begins with a dual solution scheme to find the Lagrange Multipliers and corresponding primal variables, followed by iterative compound scaling until approximate constraint violations are tolerable.

RESULTS

Three design problems with 200 to 1527 design variables were solved to compare primal and dual solution methods for large optimization problems. Although only the second problem is multidisciplinary, the approach is the same for all problems regardless of the disciplines considered. Iteration history plots are shown for each example. The label "Primal" refers to the current algorithm in ASTROS that solves the approximate subproblem directly. The label "Dual" refers to the current generalized optimality criteria being tested. Normalized CPU times for the entire execution (analyses, sensitivity, and redesign for all iterations) are shown as a factor next to each label in the legend of each plot.

200 Member Plane Truss

A 72 node plane truss made of two hundred steel elements subject to five loading conditions[11] (Fig 3 and Table 1) was used to demonstrate the efficiency of a generalized optimality criterion approach for statics. Stress and displacement limits together accounted for 2500 applied constraints. The dual method required one fourth the computational effort compared the primal method (Fig 4).

Intermediate Complexity Wing

The next example considered was an intermediate complexity wing.[12] The structural model has 158 elements and 234 degrees of freedom (Fig 5). The composite cover skins are made of graphite epoxy with the properties given in Table 2. Stress constraints were imposed on all membrane elements and displacement constraints were imposed at the tip of the wing in the transverse direction for two independent static loading conditions. A flutter speed limit of 925 knots corresponding to a flight condition of 0.8 Mach number at sea level was also applied, resulting in 722 constraints and 350 design variables. For comparison, the results from Ref 12 using 22 shape function variables and two physical design variables are shown as well (Fig 6). The slightly higher weight for this case demonstrates the penalty for constraining the skin thicknesses to vary quadratically with the span and the spar web thicknesses to vary linearly with the span. The shape function solution was more efficient since 60 constraints were active at the optimum of the dual problem. Nevertheless, even with an additional iteration, the dual method is twice as efficient as the primal method when all 350 variables are considered. The improvement is less dramatic than for the previous example because the multidisciplinary analysis (statics, modes, and flutter) is more costly relative to the optimization.

High Altitude Long Endurance (HALE) Aircraft

The finite element model for the right wing of a HALE aircraft (Fig 7) is comprised of a truss substructure and metallic cover skins. The mission of this 270 foot span airplane is to patrol for several days at 150 to 250 knots at an altitude of 65,000 feet. Since the ASTROS steady and unsteady aerodynamics models were not yet complete, three static loads were applied to an aluminum version of this wing. Stresses and wing-tip deflections were constrained, producing a total of 6124 constraints (Table 3). All 1527 elements were designed independently. The primal method could not be solved within the memory available to ASTROS, so a Fully Stressed Design (FSD) method was used as the basis for comparison (Fig 8). A design with deflection constraints alone was one order of magnitude more costly than the FSD due to the sensitivity analysis (optimization was negligible). A weight penalty was incurred, of course, when designing for the stress constraints as well.

CONCLUSIONS

A generalized optimality criterion method consisting of a dual problem solver combined with a compound scaling algorithm has been implemented in the multidisciplinary design tool, ASTROS. This method enables, for the first time in a production design tool, the determination of a minimum weight design using thousands of independent structural design variables while simultaneously considering constraints on response quantities in several disciplines. Even for moderately large examples, the computational efficiency is improved significantly relative to the conventional approach.

REFERENCES

1. Neill, D.J., Johnson, E.H., and Canfield, R.A., "ASTROS—A Multidisciplinary Automated Structural Design Tool," AIAA/ASME/ASCE/AHS28th Structures, Structural Dynamics, and Materials Conference, AIAA-87-0713, Monterey, CA, 1987, pp. 44-53.
2. MacNeal, R.H., "The NASTRAN User's Manual," NASA SP 222(08), June 1986.
3. Woodward, F.A., "An Improved Method for the Aerodynamic Analysis of Wing-Body-Tail Configurations in Subsonic and Supersonic Flow, Part I—Theory and Applications," NASA CR-2228, May 1973.

4. Albano, E. and Rodden, W.P., "A Doublet-Lattice Method for Calculating Lift Distributions on Oscillating Surfaces in Subsonic Flows," *AIAA Journal,* Vol. 7,November 1969, p. 2192.

5. Appa, K., "Constant Pressure Panel Method for Supersonic Unsteady Airloads Analysis," *Journal of Aircraft,* Vol. 24,October, 1987, pp. 696–702.

6. Vanderplaats, G.N., "An Efficient Feasible Directions Algorithm for Design Synthesis," *AIAA Journal,* Vol. 22,no. 11,November 1984, pp. 1633–1640.

7. Schmit, L. A., and Miura, H., "Approximation Concepts for Efficient Structural Synthesis," NASA CR–2552, 1976.

8. Fleury, C. and Schmit, L.A., Jr., "Dual Methods and Approximation Concepts in Structural Analysis," NASA CR 3226, December, 1980.

9. Venkayya, V.B., "Optimality Criteria: A Basis for Multidisciplinary Design Optimization," *Computational Mechanics,* Vol. 5,no. 1,1989, pp. 1–21.

10. Venkayya, V.B. and Tischler, V.A., "A Compound Scaling Algorithm for Mathematical Optimization," WRDC–TR–89–3040, February 1989.

11. Venkayya, V.B., Khot, N.S., and Reddy, S., "Optimization of Structures Based on the Study of Energy Distribution," *Proc. Second Conference on Matrix Methods in Structural Mechanics,* AFFDL–TR–68–150, December 1969, pp. 111–153.

12. Johnson, E.H. and Neill, D.J., "Automated Structural Optimization System (ASTROS), Vol. III—Applications Manual," AFWAL–TR–88–3028, December 1988.

Material	Steel
Modulus of Elasticity	E = 30 x 10^6 psi
Weight Density	0.283 lb / cu in
Stress Limits	30,000 psi
Lower Limit on rod areas	0.1 sq in
Displacement Limits on all nodes (horizontal and vertical directions)	0.5 in
Number of Loading Conditions	5
Loading Condition 1	1000 lb acting in +X direction at nodes 1, 6, 15, 20, 29, 34, 43, 48, 57, 62, 71
Loading Condition 2	1000 lb acting in -X direction at nodes 5, 14, 19, 20, 28, 33, 42, 47, 56, 61, 70, 75
Loading Condition 3	10,000 lb acting in -Y direction at nodes 1, 2, 3, 4, 5, 6, 8, 10, 12, 14, 15, 16, 17, 18, 19, 20, 22, 24, . . . , 71, 72, 73, 74, 75
Loading Condition 4	Loading Conditions 1 and 2 together
Loading Condition 5	Loading Conditions 2 and 3 together

Table 1: 200 Member Plane Truss Design Conditions

Isotropic Material	Aluminum
Modulus of Elasticity	$E = 30 \times 10^6$ psi
Poisson's Ratio	0.30
Weight Density	0.10 lb / cu in
Tensile Stress Limit Compressive Stress Limit Shear Stress Limit	55,000 psi 55,000 psi 45,000 psi
Lower Limit on Thickness	0.04 in
Orthotropic Material	Graphite Epoxy
Modulus of Elasticity	$E_1 = 18.5 \times 10^6$, $G_{12} = 0.65 \times 10^6$ $E_2 = 1.6 \times 10^6$ psi
Poisson's Ratio	0.25
Weight Density	0.055 lb / cu in
Stress Limits	115,000 psi
Lower Limit on Plies	0.00525 in
Limit on Transverse Tip Displacements	10.0 in
Flutter Speed Limit	925 knots

Table 2: Intermediate Complexity Wing Design Conditions

Material	Aluminum
Modulus of Elasticity	$E = 10.5 \times 10^6$ psi
Weight Density	0.10 lb / cu in
Poisson's Ratio	0.30
Stress Limits	60,000 psi
Lower Limit on Thicknesses	0.021 in
Lower Limit on rod areas	0.10 sq in
Limits on Transverse Tip Displacements	200.0 in
Number of Loading Conditions	4

Table 3: HALE Design Conditions

AUTOMATED STRUCTURAL OPTIMIZATION SYSTEM

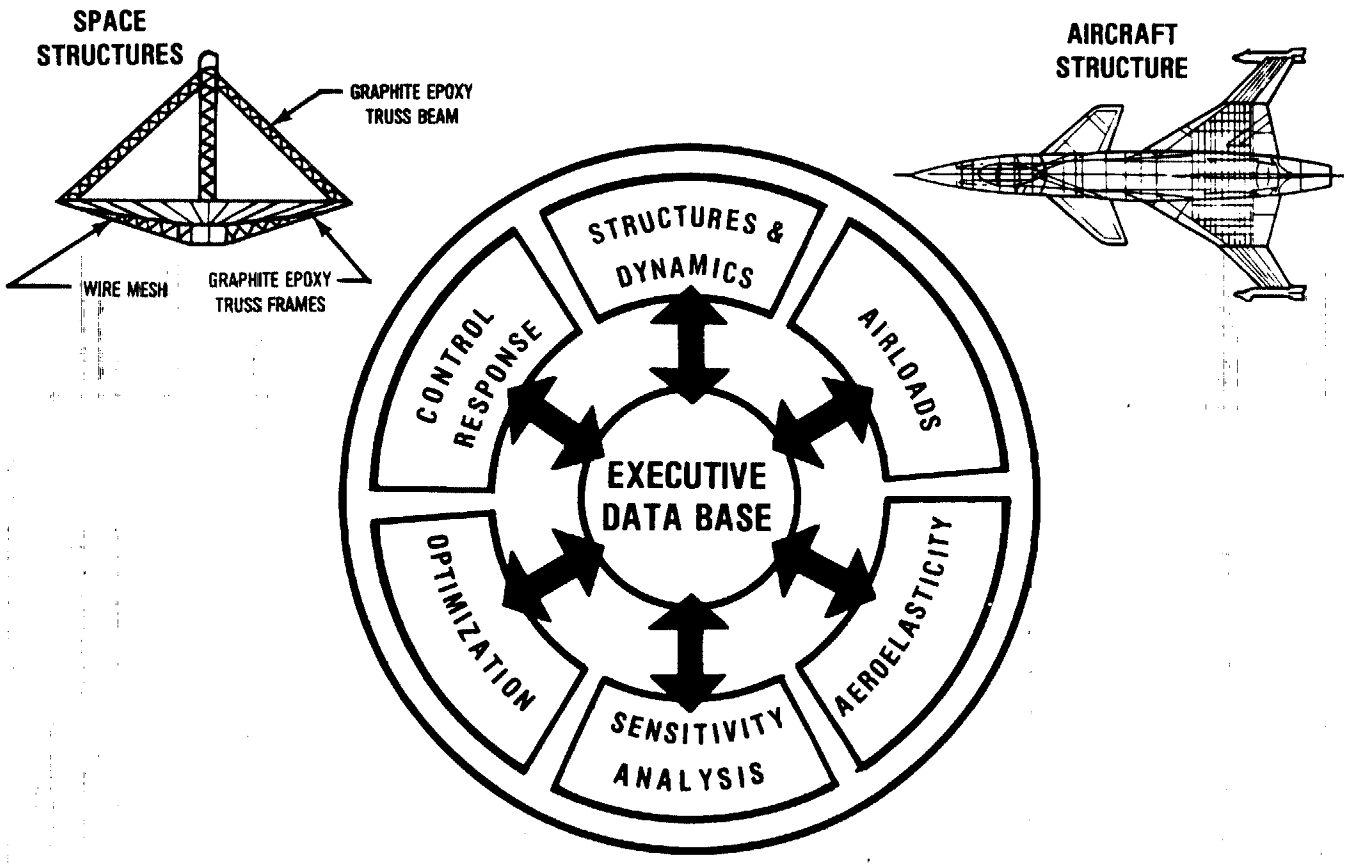
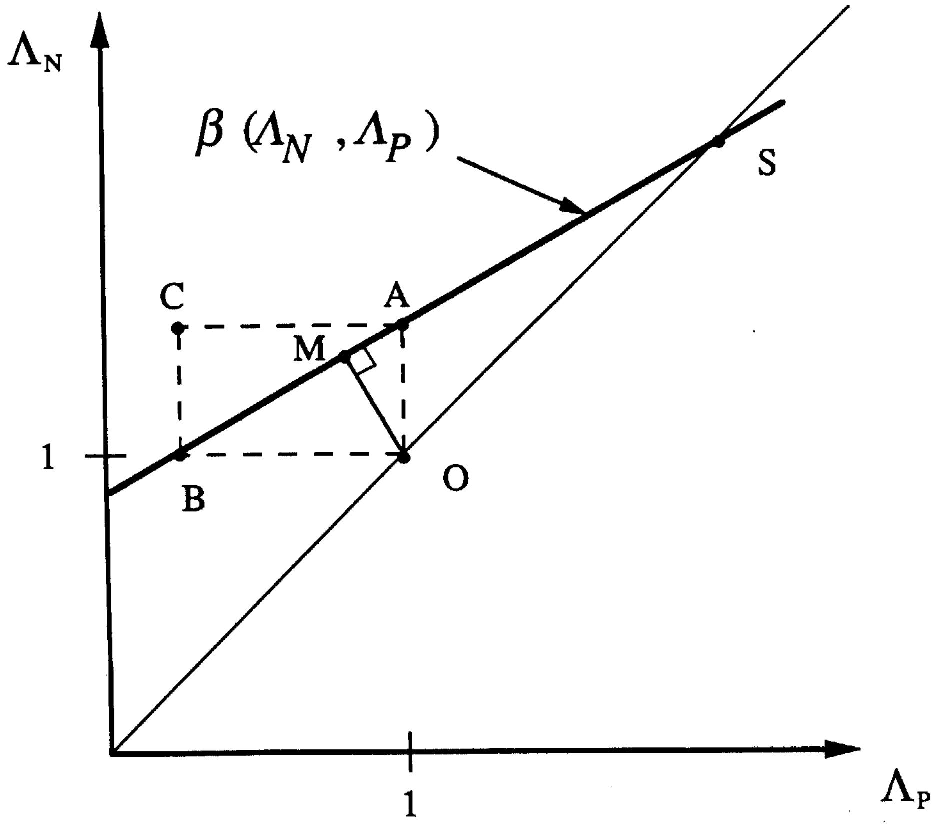


Figure 1: ASTROS Diagram



Point A is $\left(\Lambda_N = \beta^{-1/\mu_N}, \Lambda_P = 1\right)$

Point B is $\left(\Lambda_N = 1, \Lambda_P = \beta^{1/\mu_P}\right)$

Point C is $\left(\Lambda_N = \beta^{-1/\mu_N}, \Lambda_P = \beta^{1/\mu_P}\right)$

Point M is $\left(\Lambda_N = \beta_N^{-1/\mu_N}, \Lambda_P = \beta_P^{1/\mu_P}\right)$

Point S is $\left(\Lambda_N = \beta^{1/\mu} = \Lambda_P\right)$

Figure 2: Approximate Target Response Scaling Line

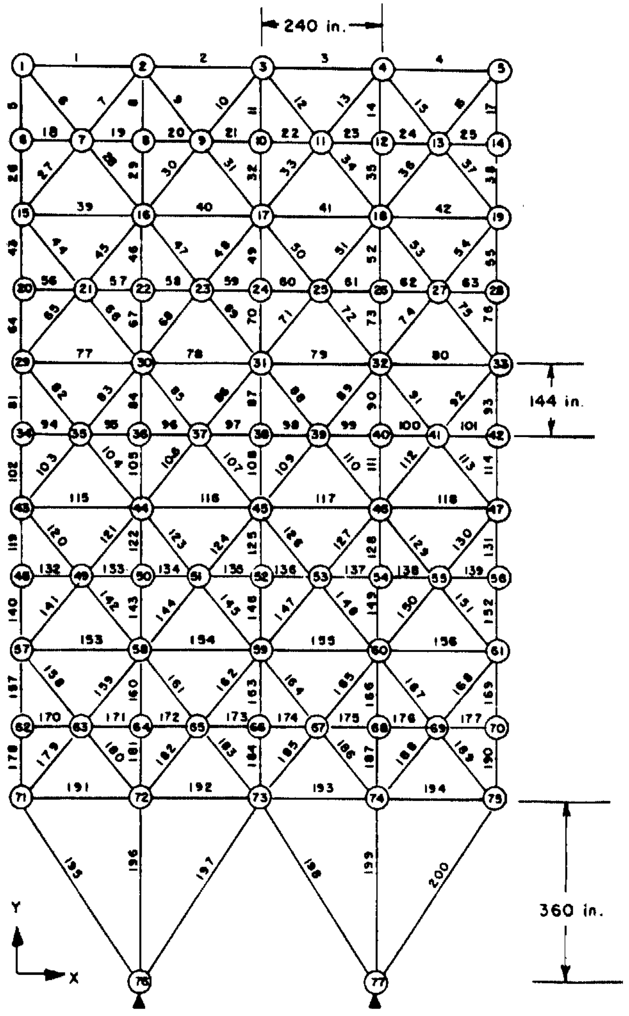


Figure 3: 200 Member Plane Frame Model

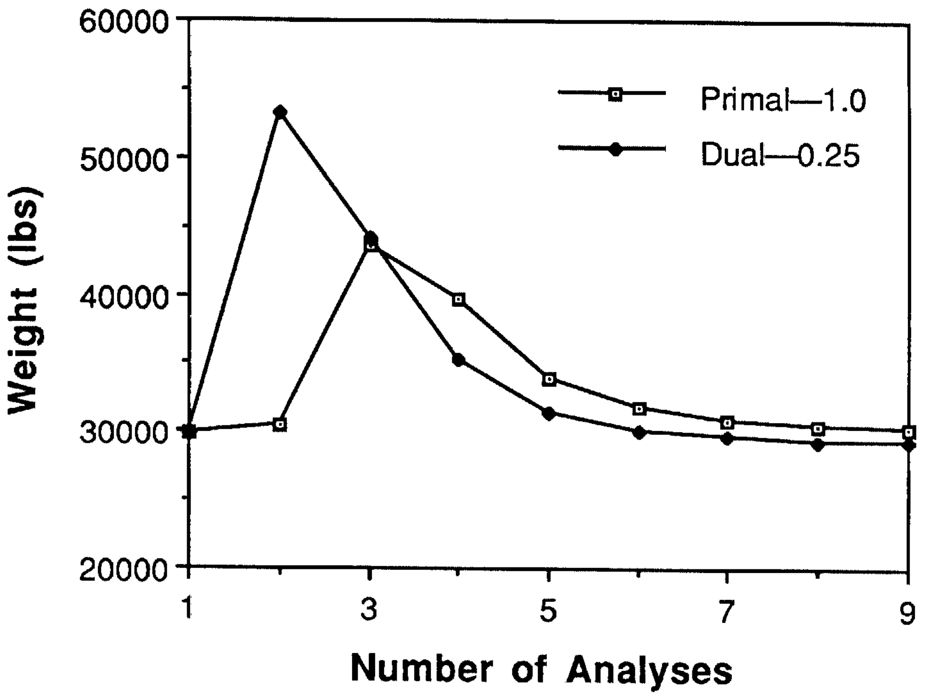


Figure 4: 200 Member Frame Iteration History

ORIGINAL PAGE IS
OF POOR QUALITY

No. of Nodes	No. of Elements		No. of DOF's	
88	39	Rods	294	Constrained
	55	Shear Panels	234	Unconstrained
	62	Quadrilateral Membrane	528	Total
	2	Triangular Membrane		
	158	Total		

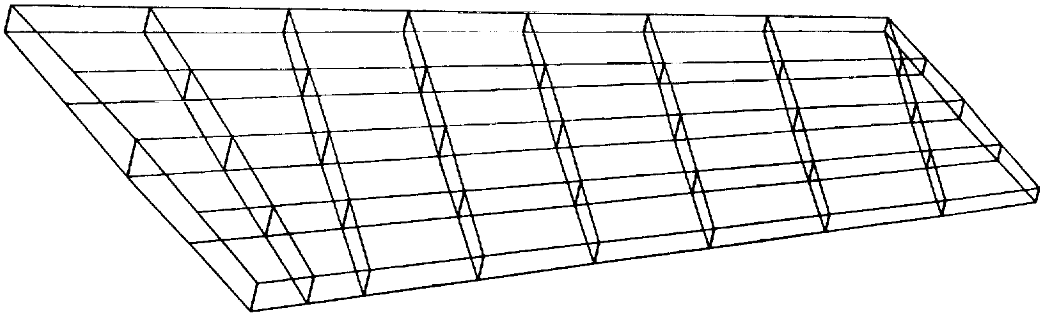


Figure 5: Intermediate Complexity Wing Model

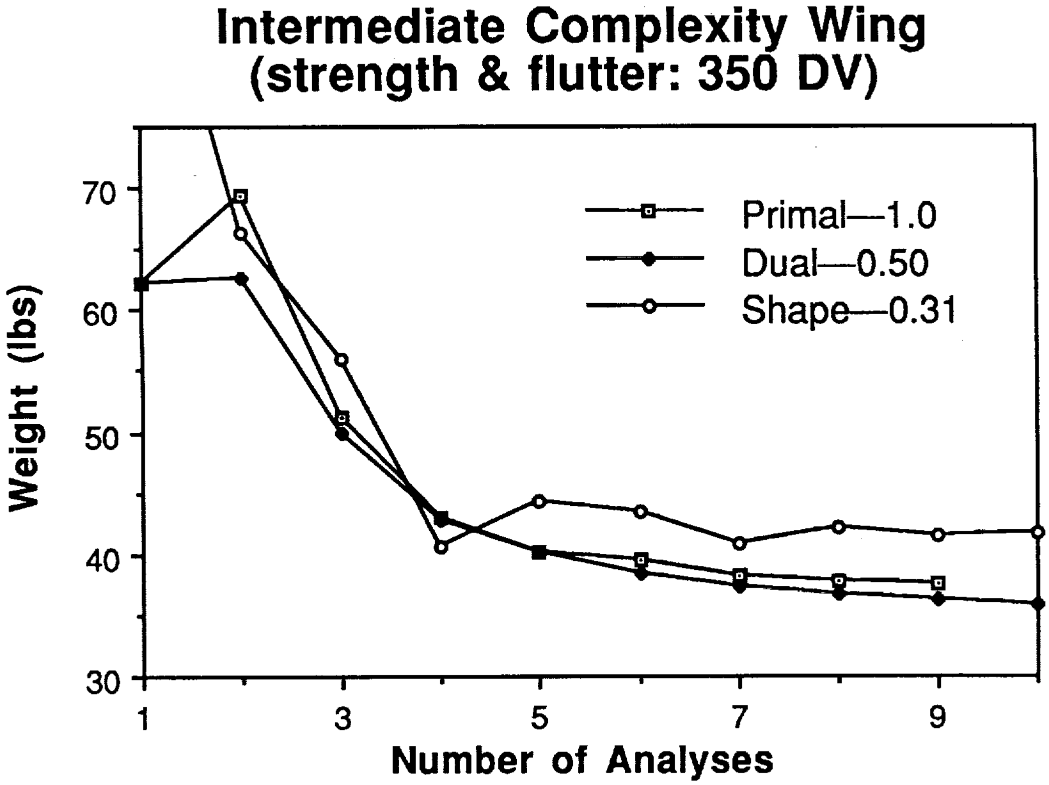


Figure 6: Intermediate Complexity Wing Iteration History

High Altitude Long Endurance Aircraft Wing

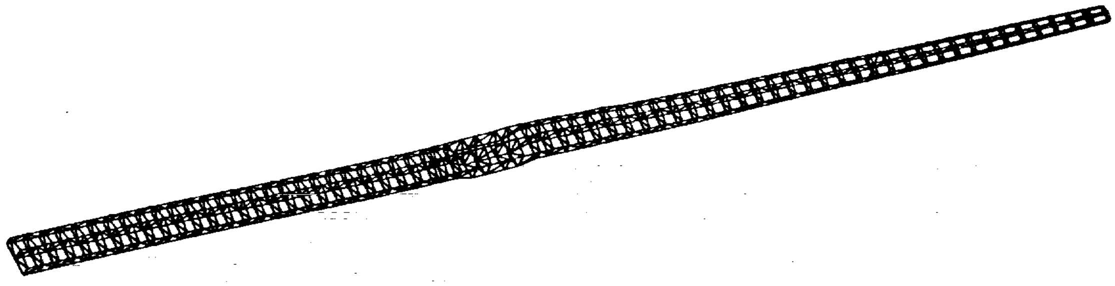


Figure 7: HALE Aircraft Wing Model

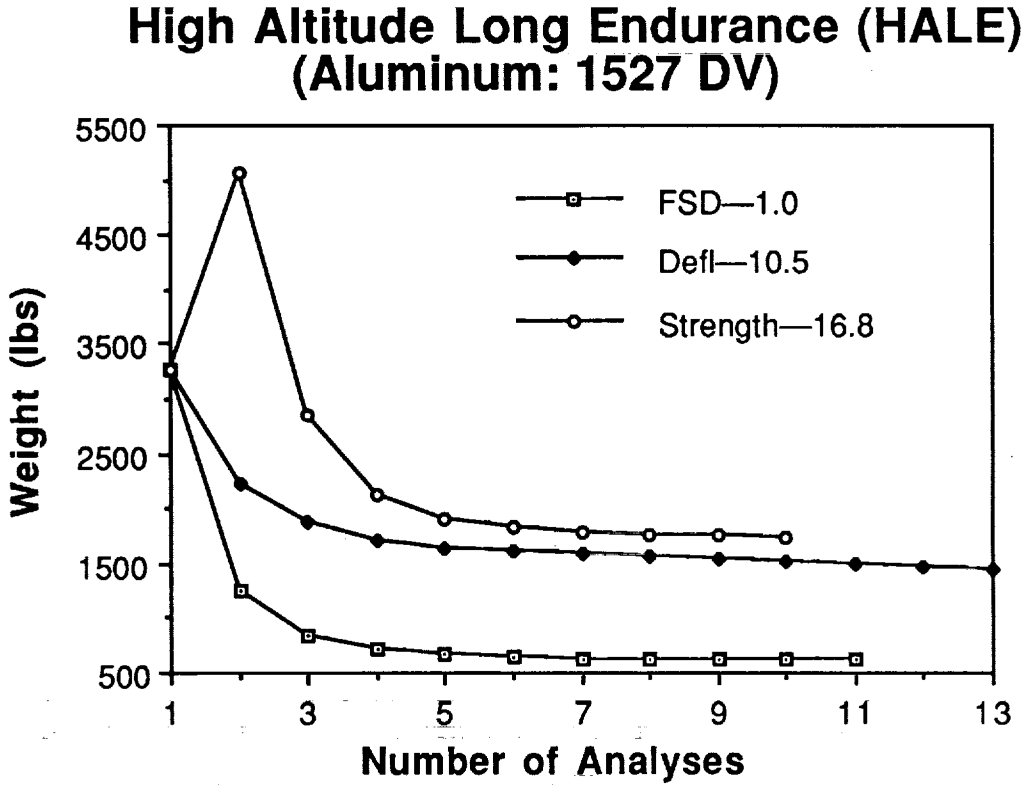


Figure 8: HALE Iteration History

N90-23049

New Multivariable Capabilities of the INCA Program

Frank H. Bauer, NASA/Goddard Space Flight Center
John P. Downing, NASA/Goddard Space Flight Center
Christopher J. Thorpe, Fairchild Space Company

Abstract

The INteractive Controls Analysis (INCA) program was developed at NASA's Goddard Space Flight Center to provide a user friendly, efficient environment for the design and analysis of control systems, specifically spacecraft control systems. Since its inception, INCA has found extensive use in the design, development, and analysis of control systems for spacecraft, instruments, robotics, and pointing systems. The INCA program was initially developed as a comprehensive classical design analysis tool for small and large order control systems. The latest version of INCA, expected to be released in February of 1990, has been expanded to include the capability to perform multivariable controls analysis and design.

1 Introduction

The INteractive Controls Analysis (INCA) program was developed at NASA's Goddard Space Flight Center (GSFC) as a computer aided control system design tool primarily for use by engineers in the Guidance and Control Branch. The program couples a user friendly interface with excellent, well conditioned computational algorithms to provide control system design analysis engineers tools which are simple to use, quick, and provide accurate results.

Initial development of the INCA program began in 1982 with the first release of program to members of the Guidance and Control Branch in 1983. Since then, INCA has been used extensively to design or analyze control systems for all of Goddard's spacecraft programs. This includes spacecraft attitude control systems, instrument servomechanisms, pointing control systems and robotic control systems. Numerous flight proven designs have been developed or validated using INCA's analytic capabilities. These include the Earth Radiation Budget Satellite (ERBS) spacecraft and the Advanced TIROS-N (ATN) [1] weather satellite series.

In 1985 INCA was delivered to COSMIC [2], NASA's computer program dissemination source, for general distribution. Since then it has been used in industry for spacecraft, instrument and robotic controller design development, university research, classical control system instruction and for use on Department of Defense payloads. Because INCA is a public domain program with source code provided, numerous enhancements have been incorporated in INCA by outside users which are now in the COSMIC version. The most notable enhancement in this category is the describing function analysis capability which was developed by Dr. Bong Wie and Mr. Tobin Anthony of the University of Texas at Austin [3]. This capability was later installed in INCA Version 3.13.

2 *Program Overview*

INCA was developed for use on VAX/VMS computers. The program was written primarily in Pascal; however many of the analysis algorithms are written in FORTRAN, particularly the matrix and multivariable control algorithms which were obtained from the SAMSAN [4] subroutine package. Since INCA was written to perform control system design analysis on very expensive spacecraft and instruments, emphasis was placed on doing the analysis **right the first time**. Thus, the program utilizes algorithms which strive for numerical accuracy and attempt to prevent ill conditioned situations. In some instances these algorithms may be less efficient than others available; however, when expensive spacecraft are at stake, accurate results are of supreme importance.

Figure 1 illustrates the INCA user interface. The INCA executive incorporates a default menu driven command structure with a VAX/VMS-type command option. These executive commands can be used to access the graphic and editor modes, query the help library, retrieve data from input files, and create and manipulate transfer functions and matrices. All commands invoked in INCA are parsed, interpreted, and checked for errors. The parsing capability allows the user to either type the full command, or simply use one or two letters in the command. Also, VAX DCL commands can be invoked without leaving the INCA environment.

Input data can include user defined transfer functions, gains, matrices, or specially generated INCA command sequences.

Transfer functions can be developed using the function editor, as alphanumeric input data from other programs, or by using FORTRAN-like arithmetic expressions. For example, a closed loop system can be computed as follows:

```
INCA> G = 1/I/s^2
INCA> H = KR * s +KP
INCA> CLTF = G / (1 + G*H)
```

The results of each INCA session are documented in an output file with the project name. Transfer functions and matrices are stored in either a binary format (the default) or alphanumeric format. INCA generated plots are displayed on local graphics terminals and can be copied using standard hard copy capabilities. Graphics can be viewed on a wide variety of color and monochrome terminals including Tektronix color and monochrome graphics terminals, DEC VT terminals and some Macintosh and IBM PC terminal emulators. Graphic output devices include, but are not limited to, the Tektronix color graphics printers, DEC Laser printers and local terminal hardcopy devices.

3 *Classical Design Analysis Tools*

Using INCA's command menu, the controls engineer can quickly generate system models. These models are then stored in transfer function form for Single-Input-Single Output (SISO) systems or in matrix form for Multi-Input-Multi-Output (MIMO) systems. INCA provides the capability to analyze continuous systems, sample data systems and hybrid (continuous/sample data) systems. In addition, systems with computational delays or transport lags can also be analyzed via the modified z-transform. The resultant linear models can then be manipulated to modify the design or can be used to determine system stability, parameter robustness or control system performance. Standard filter and control law templates have also been included to expedite system design.

INCA provides a comprehensive host of standard classical controls analysis techniques including Root Locus, Frequency Response (Bode, Nichols, Nyquist) and Linear Time Response capabilities. In addition, frequency response analyses of nonlinear systems can be performed using the describing function method [3]. Analysis results are presented as multi-colored plots with a full range of simple plot manipulation commands. Plot zooming and multiple plot windows capabilities are available as well as simple plot documentation capabilities. Analysis results can also be presented in tabular form. Figures 2-4 represent some of the design analysis plots which can be generated by INCA users.

4 Multivariable Capabilities

The multivariable capabilities incorporated in latest version of INCA, version 4.0, will serve as the foundation for all future multivariable enhancements to the program. The primary goal of version 4.0 was to incorporate all those capabilities necessary to perform rudimentary MIMO analyses. Additional MIMO enhancements will follow later.

First and foremost, a user-friendly matrix input/output environment was developed. This includes:

- Line & full screen editing
- Matrix parsing, row & column concatenation, and submatrices
- Matrix development capabilities such as identity matrices and diagonal matrices from vectors, and
- Input/output from various ASCII and binary files.

INCA will accept matrices with real, complex, and string or so-called "dynamic" elements. The dynamic matrices, similar to the dynamic transfer functions in INCA, work like a spreadsheet. As a variable such as an inertia changes, the matrix is recomputed to incorporate this change. INCA's MIMO extension is unique in that it accepts matrices of 0th order (with just an information header), 1st order (vectors), 2nd order (standard mxn matrices), 3rd order (mxnxo matrices) and beyond.

Matrix arithmetic and linear algebra utilities have been installed in the program. The matrix math operations include addition, subtraction, multiplication, matrix inverse, matrix transpose and exponentiation. As with transfer functions, these operations can be performed in symbolic form, e.g.:

```
INCA> C= ((A*B)~ + D))^3
INCA> E= D'
```

where ˜ and ' represent matrix inversion and matrix transpose operations respectively and the other operands should be obvious to BASIC or FORTRAN computer program users. Standard linear algebra routines are also available to the user including Eigenvalue/Eigenvector computations, solving the determinant of a matrix, trace computations and singular value decomposition calculations.

Controls specific MIMO capabilities include transfer function to state variable form transformations, state variable to transfer function matrix computations, linear quadratic regulator and linear quadratic estimator design and a structural finite element model reduction capability.

5 Structural Mode Significance

The structural mode significance module embedded in INCA was developed to provide a user-friendly model order reduction capability. This module was designed to expedite the model order reduction process from the initial step of loading the model information into the INCA data base through to the final step of automatically generating reduced order flexible body plant transfer matrices.

Structural dynamic models, such as that provided by the NASA Structural Analysis program, NASTRAN, [5] can be described in the following form:

$$M\ddot{x} + D\dot{x} + Kx = F \qquad (1)$$

where M, D, and K represent the system mass, damping and stiffness matrices respectively, x represents the flexible system's translation and rotation degrees of freedom, and the external forces and torques which act on the body are defined as F. Using the modal coordinate transformation:

$$x = \phi q \qquad (2)$$

where ϕ represents the mass normalized eigenvector matrix computed via programs such as NASTRAN, then the system finite element model of equation (1) can be expressed as:

$$\ddot{q} + C\dot{q} + \lambda q = \phi^{T} F \qquad (3)$$

where λ is a diagonal matrix of system eigenvalues (the square of the modal frequencies), C is a diagonal matrix representing the modal damping and ϕ^T is the transpose of the mass normalized eigenvector matrix defined in (2). The mode significance analysis in INCA relies on the system modeling and modal coordinate definitions described in equations 1-3. Data input to INCA includes the mass normalized eigenvector matrix, ϕ, and the system eigenvalues, λ, obtained through NASTRAN, the user defined modal damping matrix, C, and the locations of the force/torque and sensor locations defined in x.

Once the system model parameters are defined in INCA, a number of model reduction techniques are available to the user. These include simple gain techniques such as modal gain criteria, [6], and frequency weighted significance criteria such as the Peak

Amplitude [7] and Gregory's model reduction criteria [7] [8]. Moreover, user-defined frequency weighting criteria, such as sensor dynamics frequency shaping or control bandwidth shaping, can be used to augment the criteria described above.

Mode significance analysis results can be displayed through the use of two and three dimensional color plots (figure 5) or by viewing the results in tabular form. These results can then be used to automatically or manually select the modes to keep in the model. Special commands have also been included to automatically generate the plant transfer matrices from the reduced order model.

6 Future Plans

The multivariable controls capability described in this paper will be incorporated in Version 4.0 of INCA which is expected to be released to COSMIC in February of 1990. INCA enhancements planned beyond Version 4.0 include extending the multivariable controls and mode significance capabilities, transfer function identification from frequency response plots, and an interactive compensator synthesis capability. In addition, a PC and MacIntosh derivative of INCA is currently under development at NASA Goddard called ASTEC (Analysis and Simulation Tools for Engineering Controls). ASTEC, described in detail in reference 9, is planned for release to COSMIC in late 1990.

7 Conclusions

The INteractive Controls Analysis (INCA) program couples a user friendly interface with excellent, well conditioned computational algorithms to provide control system design analysis engineers tools which are simple to use, quick, and provide accurate results. The latest version of the program, expected to be released in early 1990, will extend INCA from it present status as comprehensive classical controls tool to one which includes multivariable controls analysis and structural model order reduction capabilities. These enhancements promise to make INCA and it's PC derivative called ASTEC the premier controls analysis tools into the 1990's and beyond.

References

1. Bauer, F. H. and Downing, J. P., "Control System Design and Analysis using the INteractive Controls Analysis (INCA) Program", AIAA Guidance, Navigation and Control Conference Paper 87-2517, 1987.

2. Bauer, F. H. and Downing, J. P., "INteractive Controls Analysis (INCA) Version 2.0", Program Number GSC-12998, COSMIC, University of Georgia, Athens GA., 1985, updated to Version 3.13, 1989.

3. Anthony, T., Wie, B., and Carroll, S., "Pulse-Modulated Controller Synthesis for a Flexible Spacecraft", AIAA Paper No. 89-3433, 1989.

4. Frisch, H. P., and Bauer, F. H., "Modern Numerical Methods for Classical Sampled Systems Analysis (SAMSAN Version 2) User's Guide", Program Number GSC-12827, COSMIC, University of Georgia, Athens, GA., 1984.

5. "The NASTRAN Theoretical Manual", NASA SP-221-(06), COSMIC, University of Georgia, Athens, GA., January 1981.

6. McGlew, D. E., "MODESIG, a Computer Program to Determine the Significant Flexible Modes", GSFC Guidance and Control Branch Report No. 324, November 1982.

7. Class, B. F., Bauer, F. H., Strohbehn, K., and Welch, R. V., "Space Infrared Telescope Facility/Multimission Modular Spacecraft Attitude Control System Conceptual Design", AAS Paper No. 86-031, AAS Guidance and Control Conference, 1986

8. C. Z. Gregory, "Reduction of Large Flexible Spacecraft Models Using Internal Balancing Theory," J. Guidance and Control, Vol. 7, No. 6, Nov-Dec 1984, pp. 725-732.

9. Downing, J. P., Bauer, F. H., and Thorpe, C. J., "ASTEC--Controls Analysis for Personal Computers", 3rd Annual Conference on Aerospace Computational Control, Oxnard CA, August 28-30, 1989.

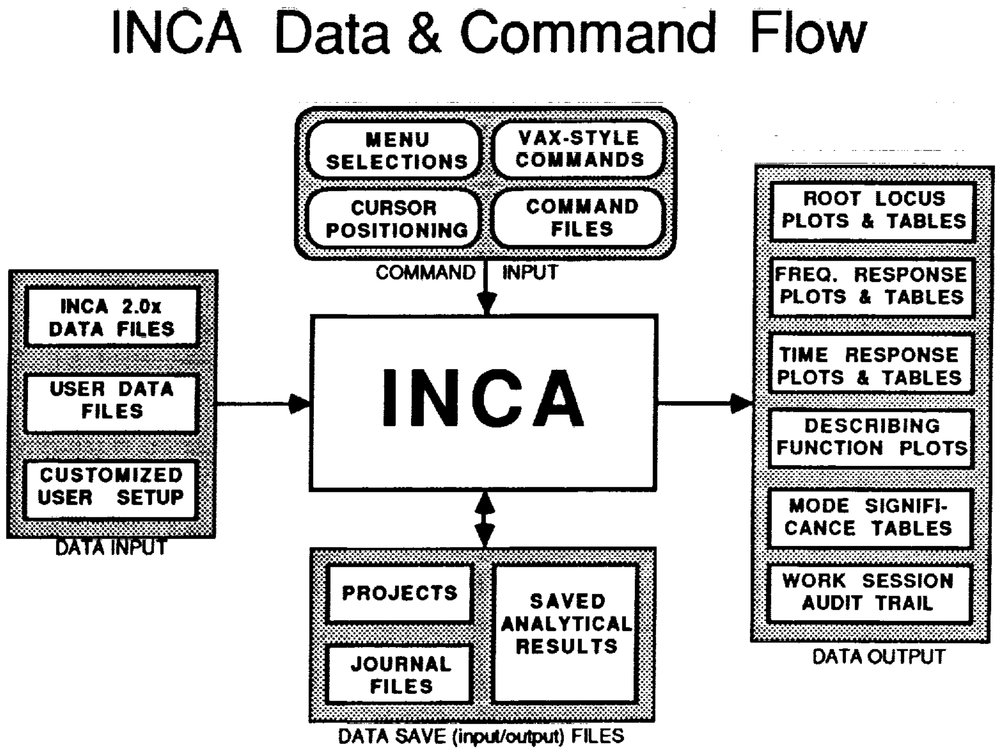


Figure 1

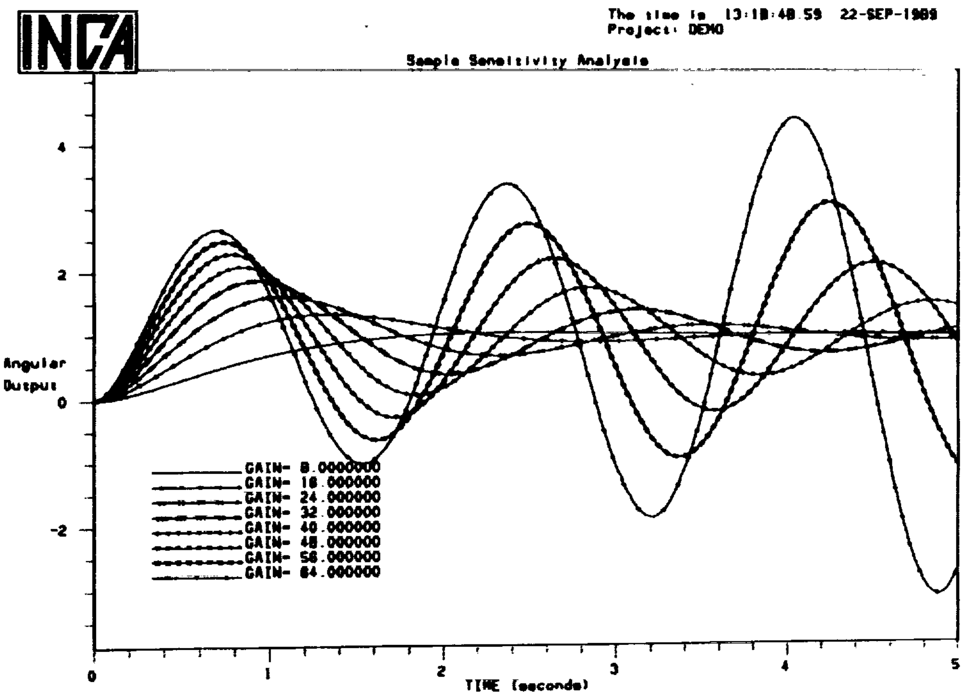
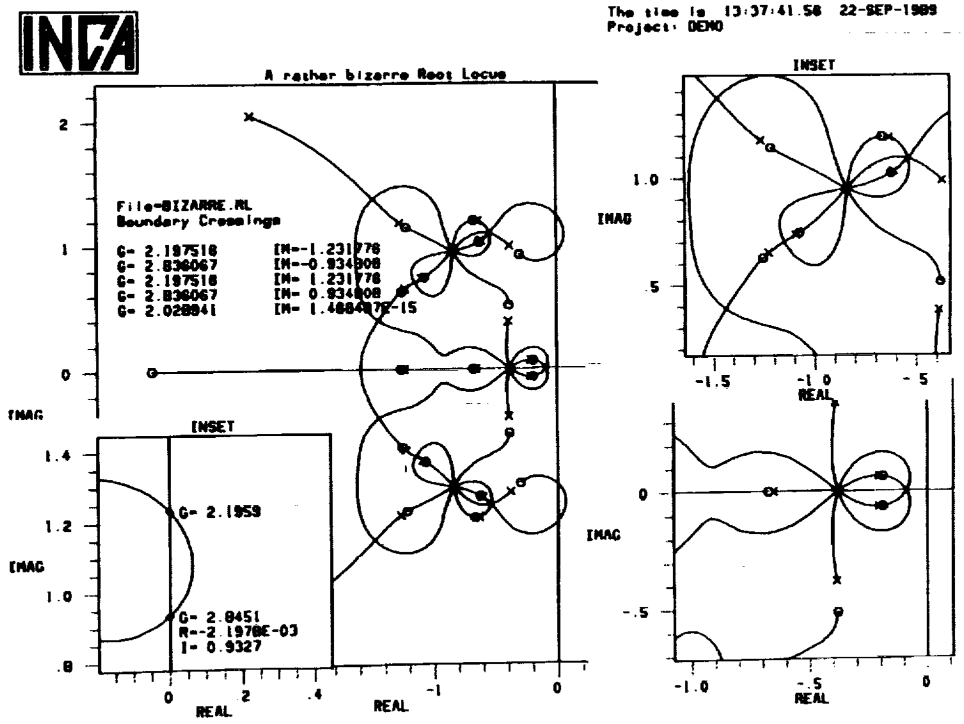


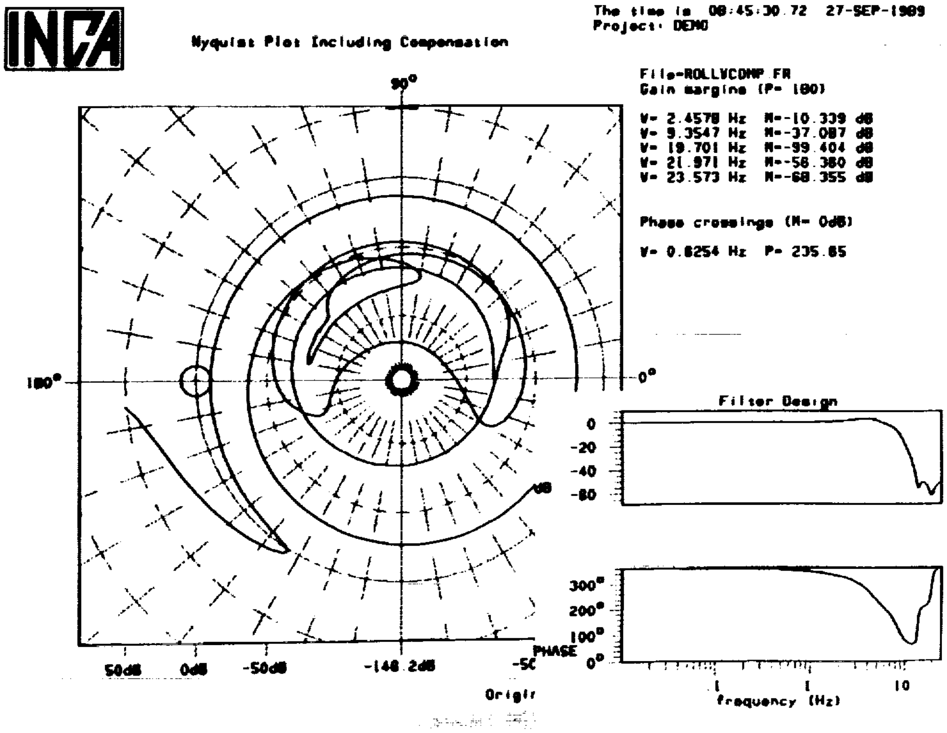
Figure 2



```
File-BIZARRE.RL
Boundary Crossings

G- 2.197516        IM--1.231776
G- 2.836067        IM--0.934808
G- 2.197516        IM- 1.231776
G- 2.836067        IM- 0.934808
G- 2.028941        IM- 1.4884072E-15
```

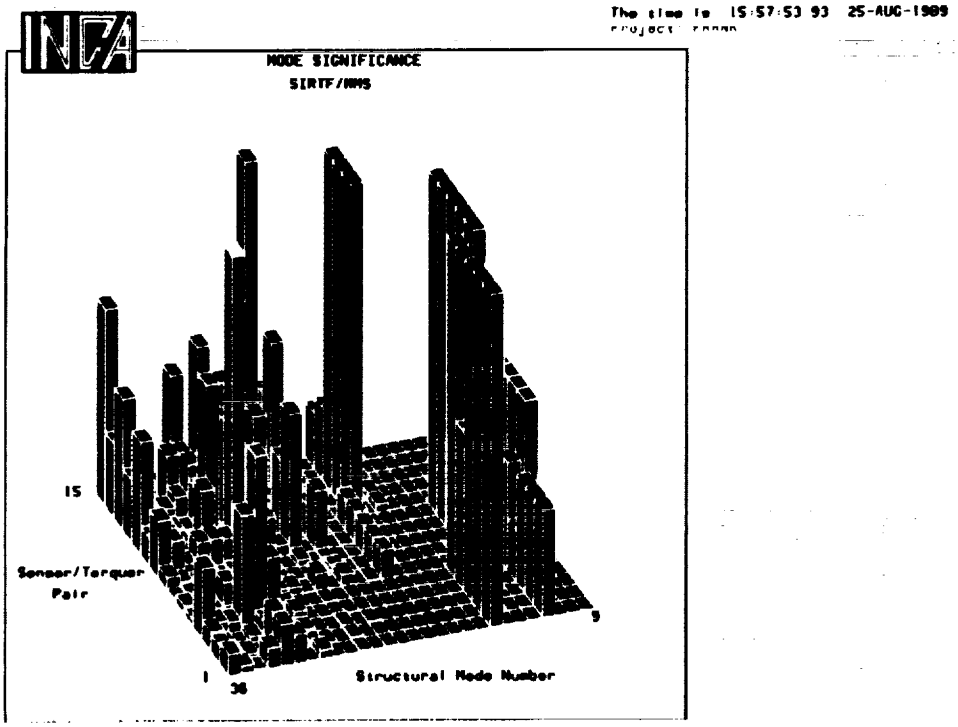
Figure 3

ORIGINAL PAGE IS
OF POOR QUALITY



```
The time is  08:45:30.72  27-SEP-1989
Project: DEMO

File-ROLLVCOMP.FR
Gain margins (P= 180)

V= 2.4578 Hz  M=-10.339 dB
V= 9.3547 Hz  M=-37.087 dB
V= 19.701 Hz  M=-99.404 dB
V= 21.971 Hz  M=-58.380 dB
V= 23.573 Hz  M=-68.355 dB

Phase crossings (M= 0dB)

V= 0.8254 Hz  P= 235.85
```

Figure 4



```
The time is  15:57:53 93  25-AUG-1989
Project: rnnnn
```

Figure 5

N90-23050

MULTIDISCIPLINARY EXPERT-AIDED ANALYSIS AND DESIGN (MEAD)

Thomas C. Hummel
Flight Dynamics Lab / FIGCA
Wright R&D Center
Wright-Patterson AFB, OH 45433

James H. Taylor
Control Systems Lab / KWD-209A
GE Corporate R&D
PO Box 8, Schenectady, NY 12301

ABSTRACT

The MEAD Computer Program (MCP) is being developed under the Multidisciplinary Expert-Aided Analysis and Design (MEAD) Project as a CAD environment in which integrated flight, propulsion, and structural control systems can be designed and analyzed. The MCP has several embedded computer-aided control engineering (CACE) packages, a user interface (UI), a supervisor, a data-base manager (DBM), and an expert system (ES). These modules have widely different interfaces and are written in several programming languages, so integrating them into a single comprehensive environment represents a significant achievement.

The supervisor monitors and coordinates the operation of the CACE packages, the DBM, the ES, and the UI. The DBM tracks the control design process. Models created or installed by the MCP are tracked by date and version, and results are associated with the specific model version with which they were generated. In addition, every model and result may have notes stored in the data base for user-supplied on-line documentation. The ES is used to relieve the control engineer from tedious and cumbersome tasks in the iterative design process. The UI provides the capability for a novice as well as an expert to utilize the MCP easily and effectively. Using the menu-driven access mode, a first-time MCP user may readily use the CACE packages, the ES, and the DBM. The expert user, on the other hand, may use MCP macros and two command entry modes to take advantage of the flexibility and extensibility of the MEAD environment.

The MCP version 2 (MCP-2.0) is fully developed for flight control system design and analysis. Propulsion system modeling, analysis, and simulation is also supported; the same is true for structural models represented in state-space form. The ultimate goal is to cover the integration of flight, propulsion, and structural control engineering, including all discipline-specific functionality and interfaces. This paper will discuss the current MCP-2.0 components and functionality.

1. INTRODUCTION

1.1 Motivation/Goal

Future aircraft designs will place more emphasis on the integration of aircraft control subsystems such as flight, propulsion, and structures. To a great extent, the design and analysis of these subsystems require similar analytical methods and software tools, yet the exchange of data and information among such disciplines is inefficient and time consuming during the conceptual and preliminary design phases because of varying notations, reference systems, and conventions. To effect this exchange, a computerized development environment is needed that contains a set of tools capable of accomplishing control system design and analysis tasks required by each discipline. This environment should allow automatic transfer of data between disciplines, thus enabling systems design and analysis to be accomplished efficiently in the preliminary design cycle. Such a system would allow each discipline to develop subsystems in the same time frame, thereby making integration feasible and producing designs that reduce aircraft complexity and improve overall performance. In addition, this environment should support users with CAD experience ranging from novice to expert, and provide rigorous tracking of the data generated throughout the design cycle. To meet these needs, the US Air Force initiated the Multidisciplinary Expert-Aided Analysis and Design Program to define and create a computer-aided engineering environment to facilitate the design and analysis of modern flight control systems.

1.2 Approach

The MCP represents the culmination of three major tasks. Task 1 researched and documented the design methodologies for individual disciplines and for integrated flight, propulsion, and structural control (IFPSC). The second task built on Task 1 to develop the MEAD software requirements, specifications, and architecture for a computer program that would support the design methodologies identified in Task 1. Task 3 involved the implemention and testing of the first MEAD computer program (MCP-1.0) based on a subset of the definition developed in Task 2. MCP-1.0 was completed in March 1989. The test and evaluation period is in progress and is expected to be completed in late 1989. A general-purpose nonlinear simulation package (SIMNON) is being incorporated in the MCP as part of the second phase of the program which also includes substantial refinements and extensions; these will comprise MCP-2.0 which will enter the alpha test stage in September 1989.

1.3 Definition of the MCP

The MCP is a computer-aided control engineering environment for modeling, simulation, design, and analysis of linear and nonlinear airframes, engines, and structural models in state-space form. The CACE packages currently integrated into the MCP include the MATRIX_X® package for linear analysis and design, GENESIS†, ALLFIT†, and AUTOSPEC†; the SIMNON® program for nonlinear simulation, equilibrium determination, and linearization is being added at the present time. The MCP utilizes a supervisor that acts as the package integrator: All communications between the CACE packages, expert system (ES), data-base manager (DBM), and the user interface (UI) are coordinated by the supervisor.

The MCP tracks, documents, and dates models created and revised by the user. Notes may also be tagged to models if the user wishes. Results are documented and associated with a specific class of a model; these too may be annotated using the MCP Note Facility. Conditions specified for each simulation, analysis, or design result are also associated with the corresponding result file (e.g., duration of simulation, type of input, etc.). All of these capabilities are handled by the DBM automatically.

In terms of CACE activity, the MCP supports many general nonlinear and linear systems operations (see Section 2.2). Some functionality specific to flight control is also implemented in

® MATRIX_X is a registered trademark of Integrated Systems, Inc, Santa Clara CA 95054; SIMNON is a registered trademark of Lund University, Lund, Sweden.

† GENESIS, ALLFIT, and AUTOSPEC are flight-control-specific packages developed by Northrop Corp. Aircraft Division (NCAD), Hawthorne CA 90250; see Section 2.1 for more details.

the MCP. For example, once a nonlinear aircraft system model is developed, the MCP has the ability to linearize it and run tolerance checks against the military flying qualities standard MIL-F-8785C. The MCP utilizes ALLFIT, AUTOSPEC (flying qualities assessment) along with the ES to do this. In the event that the system is out of tolerance, the MCP will iterate the flight control gains to bring the flight control system into compliance with specifications.

The user may operate the MCP using one or several different modes. For the more inexperienced CACE package user, the 'menu-driven' mode makes it possible to quickly and easily execute the basic CACE package functionity, as illustrated in Section 2.3. The more experienced user may access the CACE packages via menu-driven mode, either of two 'command-line' modes (MEAD commands and 'package' commands, i.e., MATRIX_X commands at present), or by using the 'MEAD Macro Facility', which allows an arbitrary mixture of MEAD commands, CACE package commands, and DCL commands to be executed within a single macro. MEAD commands are commands recognized by the MCP supervisor and converted internally into package commands; in some cases the translation is quite direct (e.g., the MEAD command for finding system model eigenvalues), in other cases MEAD commands are converted into a large number of package commands (e.g., the MEAD command for generating an input signal for a linear system in MATRIX_X). The command-line modes would be used by the more experienced user whenever a MEAD macro would not make sense (i.e., when one or two commands are to be issued).

1.4 Outline

The remaining sections of this paper will present and discuss the following subject areas:

2.0 MEAD Computer Program Version 2.0 (MCP-2.0)
2.1 The MCP Architecture
2.2 MCP Functionality
2.3 The MCP User Interface
2.4 The MCP Data-Base Manager
2.5 The MCP Supervisor
2.6 MCP Expert-Aiding
2.7 MCP Hardware Requirements
3.0 Future Directions
4.0 Summary and Conclusion

2. MEAD COMPUTER PROGRAM VERSION 2.0 (MCP-2.0)

2.1 Architecture

The MEAD project approach to creating the MCP was to take maximum advantage of existing software modules. The resulting architecture of the MCP-2.0 is portrayed in Fig. 1. MEAD integrates and serves as a front end to the CACE packages, ES, and DBM. The CACE packages available in the MCP-1.0 software include: MATRIX_X, GENESIS, ALLFIT, and AUTOSPEC. MATRIX_X provides MEAD with the linear design and analysis capability. GENESIS was developed by NCAD and provides the simulation, trimming and linearization of nonlinear airframes. ALLFIT and AUTOSPEC are also products of NCAD; these packages provide equivalent low-order fits to high-order flight control systems and military specification verification (MIL-F-8785C) respectively. All of these CACE packages were written in Fortran.

The UI, supervisor, DBM, the Delphi© ES shell, and the MCP rule bases were developed by General Electric Corporate R & D. The UI is implemented in the Computer/Human Interface Development Environment (CHIDE), supplied by GE Corporate R & D; this software in turn uses the Relational Object System for Engineering (ROSE) software, an object-oriented DBM developed by the Rensselaer Polytechnic Institute Center for Interactive Computer Graphics. ROSE and CHIDE are written in C. The supervisor coordinates all communications between the other software modules and translates MEAD commands into the equivalent core package commands in its package interface routines. The supervisor is written primarily in Ada, although the package interface routines involve some FORTRAN programming. The MCP's DBM consists of the ROSE DBM and the DEC® Code Management System (DEC/CMS®) software for version control and efficient model storage. The ES shell Delphi and portions of the rule bases are coded in LISP.

2.2 Functionality

The MEAD Computer Program provides an environment that supports modeling, simulation, analysis, and design of linear and nonlinear airframes, engines, structures (in state-space form), and control systems. The MCP is fully developed for flight control engineering. This means that the MCP has specialized software (i.e., ALLFIT, AUTOSPEC, GENESIS, and several rule bases in the ES) for flight control engineering. However, the MCP does not have the equivalent specialized software for propulsion and structural systems analysis and design.

The computer-aided control engineering packages in the MCP provide the analysis, design, simulation, trimming, and linearization capabilities. These capabilities and the functionality of the Expert System and DBM can be accessed via the Aircraft Integrated Design Environment (AIDE) menu tree (see Fig. 2). The corresponding top-level functionality that supports all three disciplines is outlined in the following sections.

2.2.1 Actively or Passively Manipulating the DBM - The data base can be examined by selecting the menu items 'Browse Projects', 'Browse Models', and 'Browse Results'. Browsing the data base reveals much information, e.g., classes and versions of models and components respectively, model and component type, creation and last modification dates, existence of associated notes, etc.; the Browse Models Screen in Fig. 3 illustrates the presentation of such material. Relations among components used in multiple models in the data base are indicated (such components are stored in one model and used elsewhere via 'linking'); also, the associations between results and components that have been created by 'modelizing' the results (an airframe model linearization result may be installed in the data base as a model component; we call this process modelizing) are tracked and displayed in the Description and Browse Results forms respectively. Active operations on the data base include deleting, updating, purging, and configuring models. ('Configuring' in MEAD terminology means loading component definition files into a core package and connecting them according to the user's specification so the model is ready to use for simulation, analysis, or design.) The MCP has a Note Facility for storing note files for projects, models, components, and results that are installed or generated in the system. Notes can consist of any information relevent to the corresponding data element; they are automatically time-stamped. The note files created and managed by this facility are thus an important vehicle for rigorous on-line documentation of the user's analysis and design effort.

© Delphi is a copyright of the General Electric Company, 1985.

® DEC and DEC/CMS are registered trademarks of Digital Equipment Corporation, Maynard, MA.

2.2.2 Defining Conditions for Simulations - A user can set parameters in nonlinear models, initialize the states of linear models, or define inputs to either linear or nonlinear system models. These capabilities are available in forms under the 'Def Condition' menu item, as shown in Fig. 2, by clicking 'Set Parameters', 'Init Variables', and 'Define Inputs', respectively.

2.2.3 Running Simulations - Simulations for linear and nonlinear models are handled in the same form, portrayed in Fig. 4. In that form there is an option to choose the type of model to simulate (i.e., linear or nonlinear). Parameters such as simulation time step and duration can also be set in the Simulation Form.

2.2.4 Trimming and Linearizing Nonlinear Models - A nonlinear model can be trimmed about the longitudinal, lateral-directional, or both axes in the full 6-degrees-of-freedom (6-DOF) case. When an axis set is chosen, a corresponding Trim Set-up form is generated, as shown in Fig. 5 for the longitudinal case. Within this form the user has the options to 'Set States' (defines the altitude, Mach number, and load along the lift vector for the desired flight condition), 'Set Limits' (defines the minimum, initial guess, and maximum values for the trim controls), and 'Set Constraints' (defines the trim states to be nulled and the tolerances). In the Linearize menu the options available are 'User Defined' (this option gives the user the ability to choose from a predefined list of possible inputs and outputs), and 'MIL Spec HOS' (this option automatically creates a linear model suitable for using ALLFIT to match the requirements for high-order flight control system model fitting - see Section 2.2.5).

2.2.5 Performing Linear Model Transforms - The Linear Model Transforms options consist of state-space to transfer function form, state-space to discrete-time state-space form, controllable part, observable part, minimal form, balanced form, reduced order, combine (which creates a single component from a multiple-component model), MIL Spec (military specification) Fit to obtain low-order equivalent linear systems for flying qualities assessment, and expert MIL spec fit. In many of these transformations, linear models are created and can be configured and studied. For a full description of these functionalities refer to the MEAD User's Guide [6].

2.2.6 Linear Analysis and Design - Many of the classical linear analysis and design techniques available in MATRIX$_X$ have been incorporated in the MCP; these are indicated in the menu items under 'Lin Analysis' and 'Linear Design' in Fig. 2. The Lin Analysis menu also includes 'Flying Q Check', providing direct access to AUTOSPEC to assess the flying qualities of an aircraft. Rule bases are also used in conjunction with AUTOSPEC [3] and ALLFIT [4] (see Section 2.6); these are accessed via 'Expert FQ Chk' under Lin Analysis and 'Expert MIL Fit' under Lin Mdl Xform. Finally, in the Linear Design menu the item 'Exp Lead-Lag' invokes a rule base to generate lead and lag compensation to achieve specifications for band-width, gain margin and steady-state error (see section 2.6).

2.3 User Interface

The MEAD system integrates an ES, a DBM, and a variety of CACE packages in a single environment. As a stand-alone system, each of these pieces of software contains its own user interface. One objective of the MCP UI is to provide direct access to the various package functionalities, while relieving the user from needing to be intimately knowledgeable about the software packages as stand-alone systems and adapting to their various styles and syntaxes. This means, for example, that a person wishing to obtain a standard time history of a linear system should not need to know MATRIX$_X$ commands for simulation and plotting; this is true using the MCP 'point-and-click' UI mode. However, the MCP UI should not unnecessarily confine the experienced user; to meet this requirement, the MCP 'Package Mode' provides the capability of executing MATRIX$_X$ commands within the MCP environment to perform any simulation and plotting activity allowed by that package, thus fully supporting the expert CACE package user. These examples illustrate how the UI objectives have been met by providing a multi-modal interface [7] which is overviewed below.

The MCP functionality can be accessed via point-and-click mouse operations on menu- and form-driven screens, as shown in Figs. 4 and 5. The top-level MEAD menu consists of Aircraft Integrated Design Environment (AIDE), Active, Command, Package, DCL, Macro, Help, and Exit (see also Fig. 2). The AIDE option was designed to provide the most conveniently accessible and functionally robust CACE capability. The 'Active' option displays the identifiers of the model(s) currently configured in the MCP Core Packages. (The user may have one linear model configured in MATRIX$_X$ and one nonlinear model in either SIMNON or GENESIS at any time in the session.) The 'Command' mode permits MEAD commands to be entered directly and dispatched to the supervisor, thus bypassing the UI. The 'Package' mode (currently operational for MATRIX$_X$ only) permits CACE package commands to be executed while operating under the MCP. VAX/VMS® DCL commands can be entered and passed directly to the operating system under the 'DCL' mode. The 'Macro' button accesses the MCP Macro Facility, which includes macro-execute mode and macro-edit mode and can support MEAD commands, package commands, DCL commands, or any combinations of these. The 'Help' facility is also menu-driven and has an extensive data base which can be accessed by subject menus. AIDE contains the linear design and analysis functions, simulation capability, nonlinear trimming and linearization routines, and DBM access, as outlined above.

As an example of the use of the MCP to obtain a standard result, the action flow for determining the controllability of a model is illustrated in Fig. 6. The 'AIDE' option is first chosen, which causes its first-level sub-menu to be displayed. The user then clicks 'Lin Mdl Xform', which brings up the second-level menu that includes the item 'Cntrl Part'. At this level, the form is created for defining and executing this operation. The entire menu tree down to the desired functionality becomes visible upon the selection of submenus. The user is given the option to execute the function (which transforms the configured continuous- or discrete-time model to obtain its controllable part), or the user can set the tolerance and/or perform the Grammian test. Once the controllable part has been determined the user has the option to 'Modelize' the result. This means the result will be installed in the data base as a model which may later be used for analysis and design. The user also can display and save the results at this point.

The UI provides an open, customizable, and flexible environment. The adept user of package commands, MEAD commands, and DCL can accomplish any task that is supported by the CACE packages. The user may also combine any of the these languages in a single MEAD macro to tailor the MCP to their preference. For example, a user can write macros to select projects in the DBM, configure models, and set up simulations using MEAD commands. It is also possible to take advantage of the MATRIX$_X$ command environment to achieve any result that can be calculated using that package alone; such results can still be stored and documented in the MEAD data base. This is just a small subset of the possibilities when using the MCP Macro Facility functionality.

® VAX and VMS are registered trademarks of Digital Equipment Corporation, Maynard, MA.

2.4 Data-Base Manager (DBM)

Data-base management requirements for CACE were determined as part of Task 1. Data-base elements were catalogued and categorized, and the relations among them were established. The MEAD data base is organized hierarchically with several levels of objects. The top-level, most general category of objects is Projects; below each project are found Models; and finally below each model are its Attributes (Descriptions and Results). Descriptions are comprised of representations of the components and their connection, type, etc.; the set of Results encompasses all data elements produced with the model (time-histories, eigenvalues, frequency response data, LQR designs, etc.). The user accesses a data base by selecting a project and operating on the corresponding unique data elements (e.g., configuring a model, displaying a result). The only "sharing" that can occur in the data base is at the component level: Models can share components with other models (via a mechanism called 'linking'); this allows a component to be maintained in one place in the data base, thus eliminating the trouble and risks involved in keeping and updating several copies of the same element. This scheme is illustrated in Fig. 5. Further details regarding the MEAD DBM are provided in [1, 2].

A user can think of this paradigm much like the DEC/VAX directory system for file organization. For example, a user might have set up subdirectories *[user.project1], [user.project2]* to manage the projects in Fig. 5, then *[user.project1.engine], [user.project1.airframe],* etc. for the models, and perhaps even *[user.project1.engine.descript]* and *[user.project1.engine.results]* for the lowest level of the data base. In the case of MEAD, there is no need for the user to create subdirectories or track the random collection of files that accumulate therein.

While the CACE data-base categories outlined above are few in number and simple, there are several factors that complicate the DBM problem: models tend to change over the lifetime of the project, some results are also models (e.g., linearizations of nonlinear models or transformed linear models), and components tend to be used in several models yet they should be stored in one location to simplify their maintenance. The MCP DBM includes mechanisms to handle all of these situations with little or no burden to the user. This was in accordance with the specific design goal of providing DBM support with minimal changes in the way the IFPSC engineer works and with minimal added overhead. More specifically,

- *Versions and Classes* - The primary need for 'version control' in the conventional software engineering sense exists in the model level of the hierarchy. The DBM must be able to keep track of system models that evolve over time (e.g., as better modeling information becomes available or as preliminary modeling errors are corrected) so that each analysis or design result can be associated with the correct model instance. This observation motivated the use of a tool that tracks each *version* of a model component (e.g., airframe model) so that version = 1, 2, 3, ... refers to the orginal and subsequent refinements of this component model, and each *class* of a model (e.g., flight-control system) that incorporated the component.
- *Linking* - The CACE DBM requirements for tracking models also give rise to the need for non-redundant model management, since maintaining the integrity of the model level of the data base is nearly impossible if several copies of various components are separately stored and maintained. The MCP DBM supports this via *links,* which allows the engineer to maintain each component in one model (the 'home' model) and use it elsewhere by bringing it out of the home data base (DB) and incorporating it in other models.

- *Modelizing*- One remaining relation that complicates the hierarchical DB organization is that which associates a linearization as a result obtained using a nonlinear model with a linearization used as a model component. The same situation exists with regard to linear model transformation - for example, one may find the controllable part of a linear model, and desire to save this as both a result and model for further study by clicking 'Save' and 'Model' in Fig. 6. These associations are tracked in the MCP DBM using a mechanism called the *reference*. The engineer may inspect a linearization result and check the reference to see if it exists as a component in any model. From the other perspective, a linear model component may be checked to determine if it was obtained as a result generated with a particular nonlinear model and trace that result back to determine how it was obtained (e.g., at what flight condition). The value of a linear model is greatly reduced if component traceability in this sense cannot be assured.

An important secondary data element not depicted in Fig. 5 is the *condition specification*. This element contains information regarding operations performed on a model before a result is obtained. These include actions such as changing a parameter from its nominal value, specifying an initial condition and/or input signal before performing a simulation, defining a frequency list before obtaining Bode plot data, etc. The condition specification also records numerical conditions, such as setting a tolerance for determining controllability or observability. Selecting this data is critical, since it is the combination of model instance and condition specification that determines the result and thereby allows the engineer to document or repeat the result. Condition specifications are stored in the MEAD data base and may be recovered for any result that has been saved.

2.5 Supervisor

The supervisor monitors and coordinates all message handling from the user to the CACE packages, the ES, and the DBM. The supervisor interprets the user commands and translates them into package commands. Upon completion of a task/function, a package response is returned to the supervisor from a CACE package, the DBM, or the ES. The supervisor then translates this package response into standard form and conveys this information to the UI for display to the user. The DBM calls are all handled automatically based on the user's activity (e.g., creating models and results, editing models, annotating the data base, deleting data elements).

2.6 Expert-Aiding

The expert-aiding capability operates under the "control engineer's assistant" paradigm [8]. This means that the expert system is activated only when the user requests that it be used to perform a specific task. Once a user invokes the ES, results will then be generated and presented. The user has the opportunity to accept or reject the result from the ES; if the user chooses to reject the result, it is possible to respecify constraints/specifications and let the ES execute its rule base again.

The main purpose of the ES is to provide aid in clear-cut but substantial iterative control design procedures. Task 1 of the MEAD program identified several areas where expert-aiding could be used. Each area was evaluated and compared with the others in respect to time savings and feasibility. Four areas were selected for implementation; the resulting expert system consists of four main rule bases for Expert Military-Specification Fitting ('Expert MIL-Fit' in Fig. 2), Expert

Eigenvalue ('Expert Eigen'), Expert Flying Quality Assessment ('Expert FQ Chk'), and Expert Lead-Lag Compensator Design ('Exp Lead-Lag'). 'Expert MIL-Fit' takes advantage of the advanced frequency-dependent weighting functionality of ALLFIT by iterating these dependent weights to improve the low-order system fit for the axis selected (i.e., longitudinal) of the flight control system design. 'Expert Eigen' invokes Eigen Analysis, scrutinizes the results, and comments on them (e.g., 'The minimum damping ratio is zeta = 0.5148' for a stable system with complex poles). (This functionality was programmed only for software integration, demonstration purposes, and proof of concept.) 'Expert FQ Chk' provides flying qualities assessment using AUTOSPEC combined with control system design iteration to bring the flight control system into compliance with specifications.

2.7 Hardware Requirements

The MEAD Computer Program may currently be hosted only on the DEC VAX 11-xxxx family of computer systems and certain DEC workstations under the VMS operating system. The MCP UI requires use of a Tektronix® 4107 terminal (or a higher version), or of a personal computer with a suitable terminal emulator. This hardware platform was selected to support the Air Force on its existing computer facility.

3. FUTURE DIRECTIONS

Many extensions and refinements are being considered for future versions of the MCP. Selection will be based on the comments and recommendations of beta-test MCP users and practical issues of cost and implementability. Areas of high priority include execution speed, portability, flexibility (e.g., UI based on high-resolution graphics and windowing), user-friendly handling of linear models, and adding capabilities to cover specialized computer-aided control engineering for propulsion and structural control. Conversion of the operating system to UNIX® and porting the MEAD Computer Program to a workstation environment is anticipated in 1990. Other applications for expert-aiding are also under investigation.

4. CONCLUSION/SUMMARY

MEAD represents a new and innovative approach to CAE support for the control design process. The MCP is a supportive and flexible environment, designed to meet the user's needs in an appropriate and effective fashion regardless of the user's level of expertise. The important novel features of the MCP are the integrated engineering data-base manager, expert-aiding, and a multi-modal user-friendly interface. MEAD eliminates the need for the user to mentally track the data elements and design process during system development or to do so via manual means.

The MEAD Computer Program version 1.0 (MCP-1.0) represented the culmination of the MEAD Project's Task 3 effort. MCP-2.0 has been defined and implemented under an extension of the original effort. The MCP software is currently being tested and evaluated on several large program applications, e.g., the Wright R & D Center/TXAD High Altitude Long Endurance (HALE) program. Users' comments and evaluations are being recorded, and extensions and modifications are being planned on the basis of this feedback. The most important areas of future

® Tektronix is a registered trademark of Tektronix, Inc., Beaverton, OR.

® UNIX is a registered trademark of AT&T Bell Laboratories, Holmdel, NJ.

enhancement have been outlined in Section 3.

The MCP-2.0 is most fully developed for flight control systems engineering. Certain generic aspects of propulsion and structural systems analysis and design are also supported, including modeling, analysis, and simulation based on system models represented in state-space form. As a long-term goal, the MEAD project is striving for the total integration of flight, propulsion, and structural control engineering.

ACKNOWLEDGEMENTS

A major part of the work described above was performed for the Wright Research and Development Center, United States Air Force, Wright-Patterson AFB, Ohio 45433-6523, under contract F33615-85-C3611 by GE Corporate Research and Development (GE CR&D) as prime contractor, with primary subcontractor support from Northrop Corporation Aircraft Division (NCAD), GE Aircraft Engine (GE AE), and Aule-Tek Inc. The contributions of the following individuals are most gratefully recognized: NCAD - Dinesh Joshi, Pierre Wong, Peter Shaw, Juri Kalviste; GE AE - John Polley, Shrider Adibhatla; Aule-Tek - David Kassover, James Trojan, Michael Charbonneau, Albert Antoniotti; and GE CR&D - Tim Johnson, Magnus Rimvall, Hunt Sutherland, and Walt Dixon. In addition, Prof. Dean K. Frederick of Rensselaer Polytechnic Institute provided invaluable consulting services over the course of this project. Finally, numerous personnel at the USAF WRDC Flight Dynamics Laboratory have greatly influenced the course of this effort: 1st Lt Mark Schiller and Capt Mike Dunbar, and Mssrs. David Bowser, Stan Lash, John Bowlus, Tom Gentry, and Philip McKeehen.

REFERENCES

[1] Taylor, J.H., Nieh, K.H., and Mroz, P.A., "A Data-Base Management Scheme for Computer-Aided Control Engineering", *Proc. American Control Conference,* Atlanta, GA, June 1988.

[2] Mroz, P.A., McKeehen, P.D., and Taylor, J.H., "An Interface for Computer-Aided Control Engineering Based on an Engineering Data-Base Manager", *Proc. American Control Conference,* Atlanta, GA, June 1988.

[3] Wong, J.P., "AUTOSPEC Flying Qualities of Piloted Airplanes", Northrop Corporation Aircraft Division Control Systems Research, Hawthorne, CA, January 1989.

[4] Wong, J.P., "ALLFIT Equivalent Low-Order Systems Program", Northrop Corporation Aircraft Division Control Systems Research, Hawthorne, CA, January 1989.

[5] Taylor, J.H., and McKeehen, P.D., "A Computer-Aided Control Engineering Environment for Multi-Disciplinary Expert-Aided Analysis and Design (MEAD)", *Proc. National Aerospace and Electronics Conference (NAECON),* Dayton, Ohio, May 1989.

[6] Taylor, J.H., "User's Guide for MEAD Computer Program Version 1.0 (MCP-1.0)", General Electric CR&D, Schenectady, NY, March 1989.

[7] Rimvall, C.M., Sutherland, H.A., and Taylor, J.H., "GE's MEAD User Interface - a Flexible Menu- and Forms-Driven Interface for Engineering Applications", accepted for *Proc. CACSD '89,* Tampa FL, December 1989.

[8] Taylor, J.H., "Expert-Aided Environments for CAE of Control Systems" (plenary lecture), *Proc. 4th IFAC Symposium on CAD in Control Systems '88,* Beijing, PR China, August 1988.

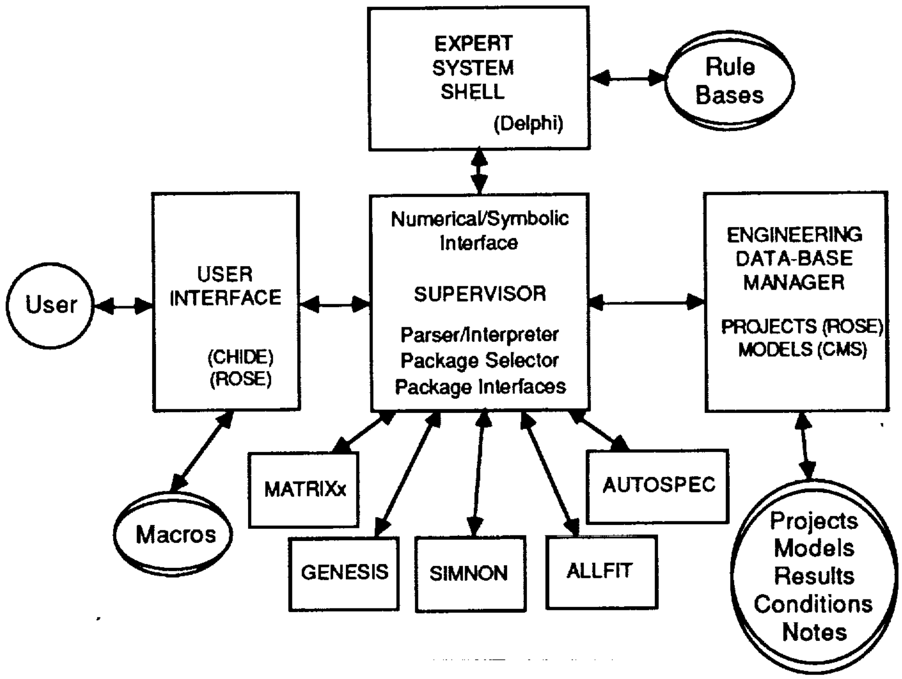


Figure 1. MCP Architecture

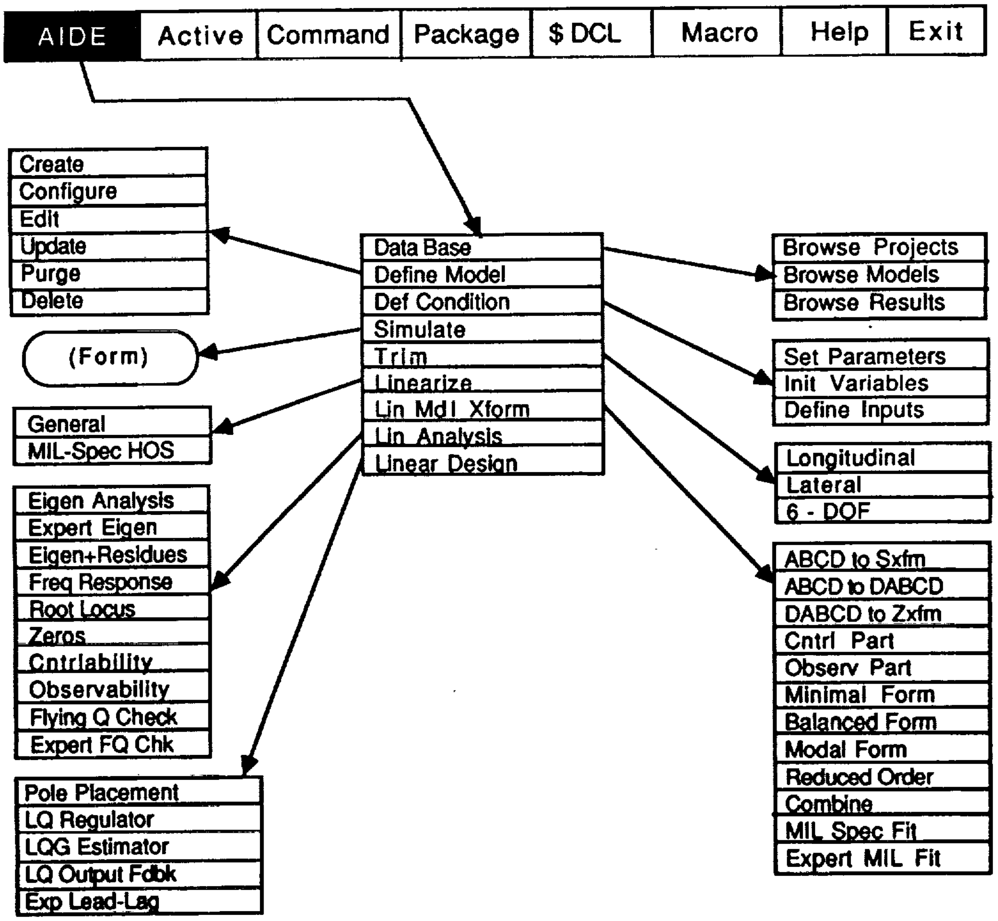
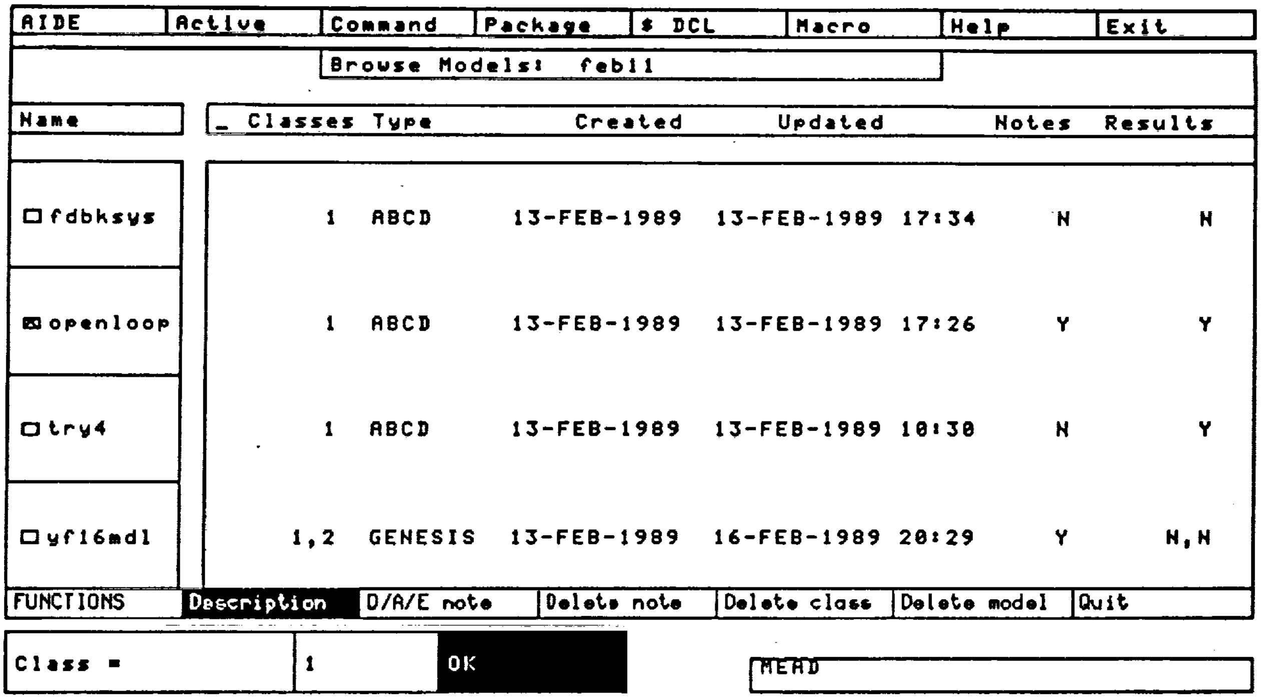


Figure 2. MCP Menu Tree

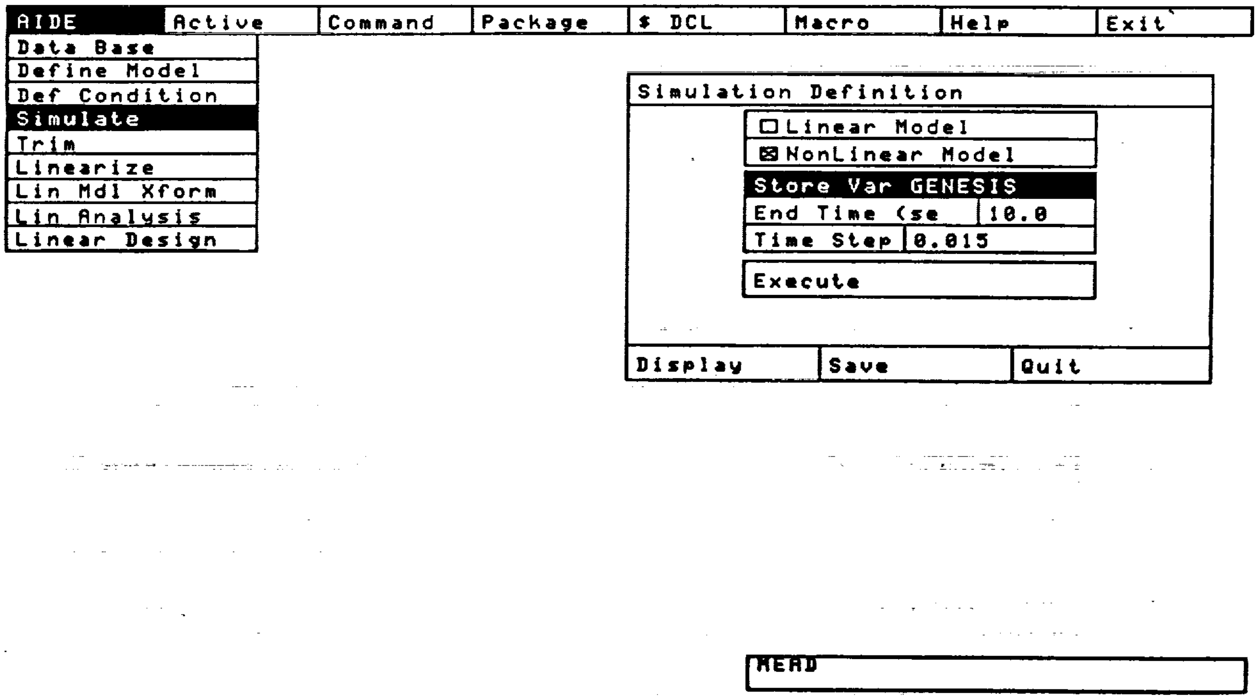


Browse Models: febll

Name	Classes	Type	Created	Updated		Notes	Results
□ fdbksys	1	ABCD	13-FEB-1989	13-FEB-1989	17:34	N	N
☒ openloop	1	ABCD	13-FEB-1989	13-FEB-1989	17:26	Y	Y
□ try4	1	ABCD	13-FEB-1989	13-FEB-1989	10:30	N	Y
□ yf16mdl	1,2	GENESIS	13-FEB-1989	16-FEB-1989	20:29	Y	N,N

Class = 1

Figure 3. MCP Browse Models Screen

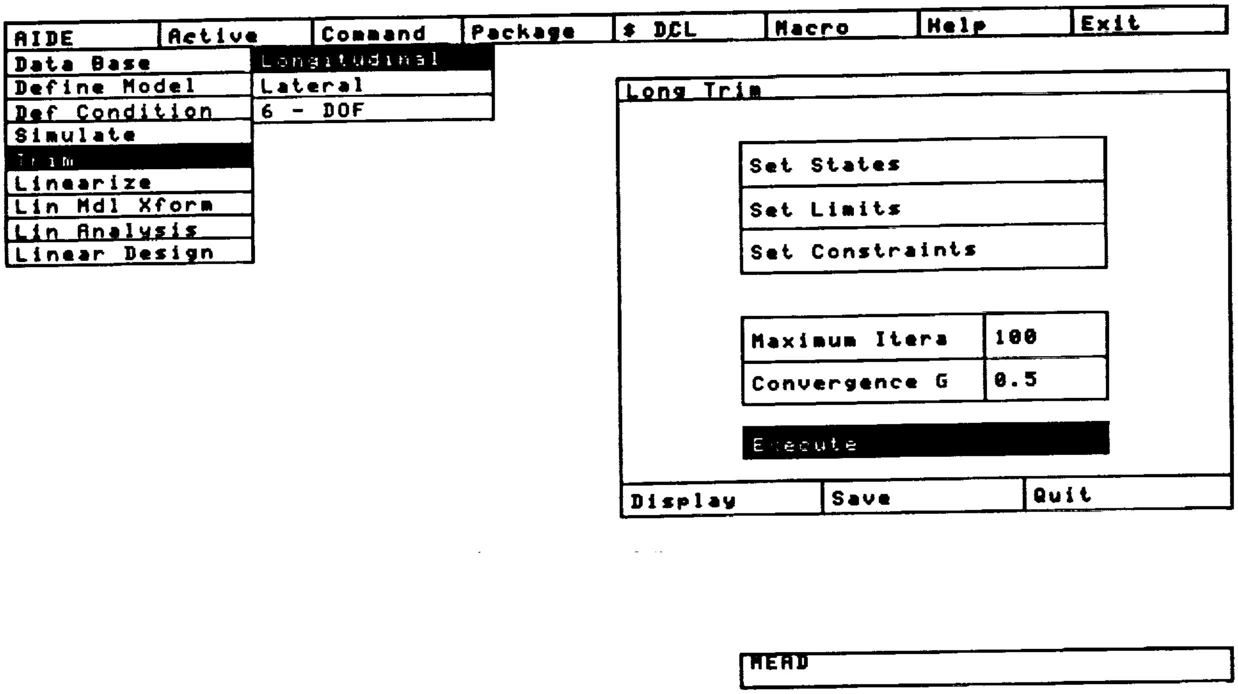


Simulation Definition

□ Linear Model
☒ NonLinear Model
Store Var GENESIS
End Time (se 10.0
Time Step 0.015

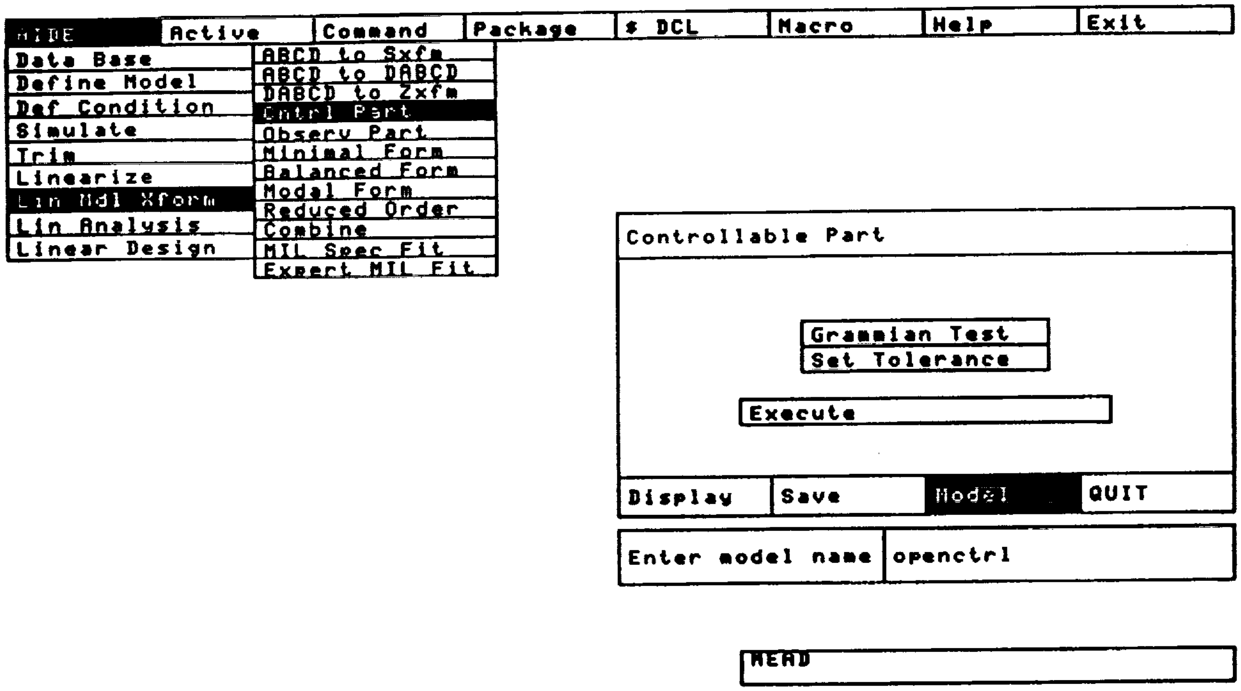
Figure 4. MCP Simulation Screen

ORIGINAL PAGE IS
OF POOR QUALITY



Maximum Itera	100
Convergence G	0.5

Figure 5. MCP Trim Set-up Screen



Enter model name	openctrl

Figure 6. MCP Controllability Analysis Screen

ORIGINAL PAGE IS
OF POOR QUALITY

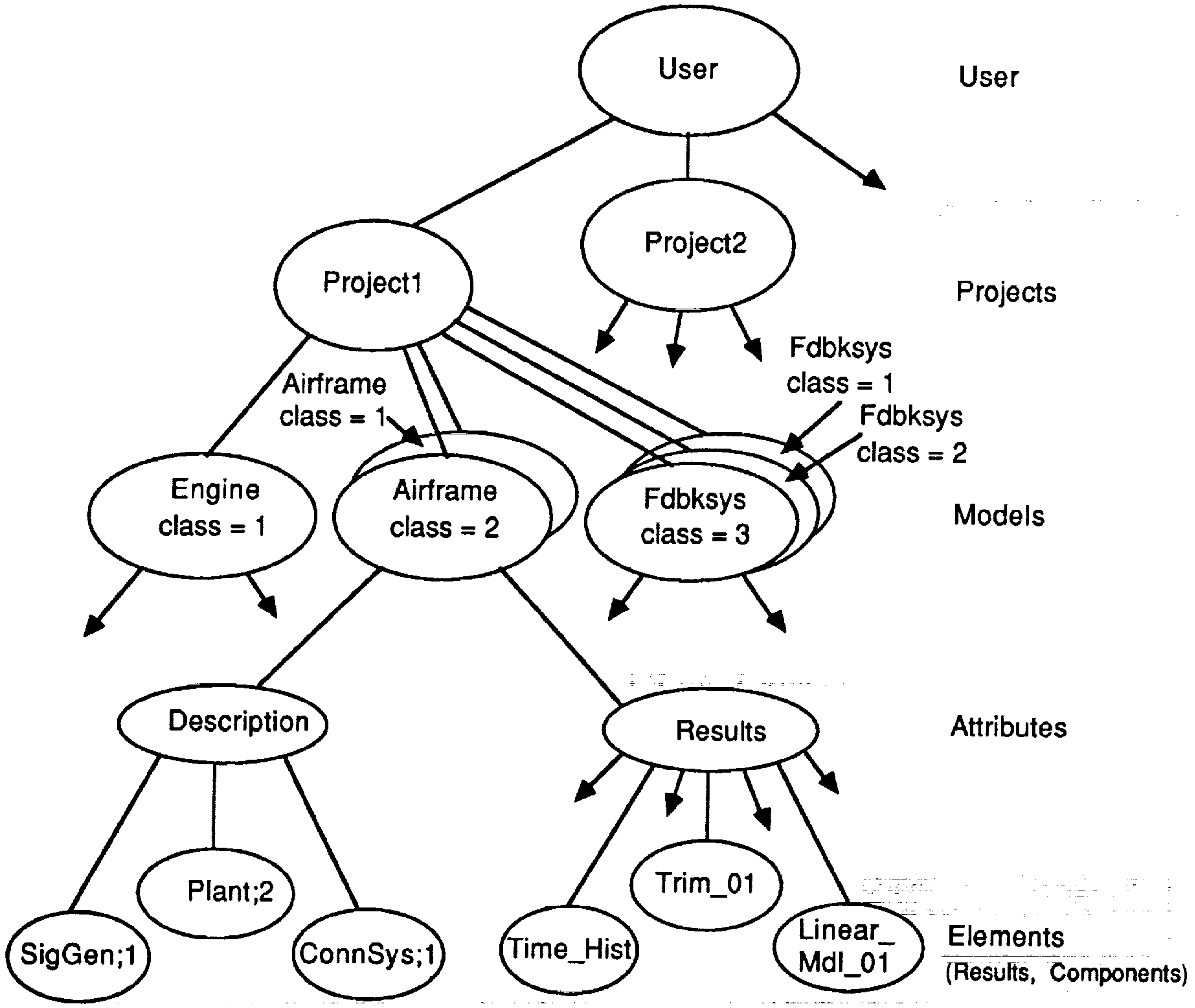


Figure 7. MCP Data-Base Version Control Schematic

Overview of Computational Control Research at UT Austin

Bong Wie
Dept. of Aerospace Engineering
The University of Texas at Austin
Austin, Texas 78712

Abstract

An overview of current research activities at UT Austin is presented to discuss certain technical issues in the following areas:

- **Computer-Aided Nonlinear Control Design:** In this project, the describing function method is employed for the nonlinear control analysis and design of a flexible spacecraft equipped with pulse modulated reaction jets. INCA program has been enhanced to allow the numerical calculation of describing functions as well as the nonlinear limit cycle analysis capability in the frequency domain.

- **Robust LQG Compensator Synthesis:** Robust control design techniques and software tools are developed for flexible space structures with parameter uncertainty. In particular, an interactive, robust multivariable control design capability is being developed for INCA program.

- **LQR-Based Autonomous Control System for the Space Station:** In this project, real-time implementation of LQR-based autonomous control system is investigated for the space Station with time-varying inertias and with significant multibody dynamic interactions.

N90-23052

ASTEC -- CONTROLS ANALYSIS FOR PERSONAL COMPUTERS

John P. Downing, NASA/Goddard Space Flight Center

Frank H. Bauer, NASA/Goddard Space Flight Center

Christopher J. Thorpe, Fairchild Space Company

Abstract

The ASTEC (Analysis and Simulation Tools for Engineering Controls) software is under development at Goddard Space Flight Center (GSFC). The design goal is to provide a wide selection of controls analysis tools at the personal computer level, as well as the capability to upload compute-intensive jobs to a mainframe or supercomputer. The project is a follow-on to the INCA (INteractive Controls Analysis) [1] program that has been developed at GSFC over the past five years. While ASTEC makes use of the algorithms and expertise developed for the INCA program, the user interface has been redesigned to take advantage of the capabilities of the personal computer. This paper describes the design philosophy and the current capabilities of the ASTEC software.

1 Introduction

The ASTEC software is being developed as a comprehensive controls tool for the 1990s. An attempt is being made to provide a broad framework upon which many different techniques can be added. ASTEC will eventually consist of over a dozen separate programs, most of which can have more than one copy running at a time. This paper will first summarize the recent history of controls software development at GSFC, and then describe the design philosophy and current capabilities of ASTEC.

2 LSAP, INCA, and ASTEC

The LSAP (Linear Systems Analysis Program) was developed by D. J. Duven at Iowa State University, and enhanced by T. P. Weis, C. J. Herget et al. at Lawrence Livermore Laboratory [2]. It was received at Goddard in 1981 and was perceived as a useful engineering tool. It was soon ported to the VAX/VMS computer and a series of enhancements was begun. Eventually the program was entirely rewritten and expanded. It was rechristened INCA, and published in 1985 through COSMIC. A greatly enhanced version was released in early 1988. The program has been reasonably popular being used at approximately 40 government and industrial sites. Several operational spacecraft have been developed or analyzed using the INCA program, the best example the Advanced Tiros-N (NOAA F-G-H) series spacecraft [3]. Most recently, state methods have been added to INCA. The routines used have been drawn from the SAMSAN library [4].

However, a number of limitations in the software have become apparent, and various improvements have been proposed both at Goddard and from outside sources. These include:

- Expansion of the multi-variable and state space capability.
- Simulation capablility, both linear and non-linear.
- Speeding the production of plots by improving terminal drivers, and adding more drivers for the ever-increasing variety of terminals.
- Conversion from VAX/VMS to other computers, including Unix, IBM-PCs, Macintosh, and Sun/Apollo workstations.
- A block diagram user interface.

While certain improvements (first three) are feasible, others would necessitate a complete reworking of the program. Any extensive changes would require modification to the user interface and command structure, and thus cause relearning and compatibility problems for many existing users. For these reasons it was decided to consider INCA a "mature" program. Only additions would be made and the basic structure would not be changed.

A new software system was desired for the needs of the future. An attempt has been be made to remove as many of the existing restrictions as possible. The following goals were established:

- Maintain most of INCA's capabilities.
- The main user interface would be based on block diagram construction and manipulation. A high level of "user-friendliness" is desired throughout.
- For enhanced graphics capability and speed, the main program should be at personal computer / workstation level. Use of the program would not then be restricted to the often unavailable or overloaded VAX computers.
- Develop a capability to upload compute-intensive jobs to other (faster) computers.
- Develop a capability to generate compilable simulations, similar to the ACSL [5] or MODEL [6] programs.
- User expandability.
- Capability to generate viewgraphs, both of results and system block diagrams.

3 Overall Design

We shall now describe the environment chosen for the initial development of ASTEC. The main program is to run on a IBM-PC or compatible under Microsoft Windows. Uploadable programs will be written in Ada for maximum portability. Versions of these Ada programs will also exist on the PC. ASTEC will consist of several independent program modules instead of a large monolithic design as in INCA. This design will keep the complexity at a more moderate level. The main development tool will be the block diagram manipulator, which will create "jobs" to be executed by other program modules in the background. The results of these jobs may be examined by display and analysis plotting programs.

3.1 Hardware Architecture and Development Configuration

ASTEC was conceived as a platform for many types of analyses. Microsoft Windows was chosen as the baseline architecture. This operating environment has many advantages as listed below.

- Near universal availability of the IBM-PC and compatible systems. While there are additional requirements for Windows (Hard Disk, Hercules or EGA display, and mouse) these are of moderate cost.
- Supports Pascal, the language in which INCA was written.
- A friendly graphical user interface.
- Device independence, in that display, printer and mouse drivers are already written. A large portion of the INCA development effort was spent on device drivers.
- Wide popularity, ensuring device driver support for future display and printer enhancements.
- Easy port to OS/2, possibly one that can be automated.
- Possible port to Macintosh, though not so easy.
- Possible port to PM/X for Unix systems.
- Easy interchange of data with commercial Windows programs for improved generation of data and reports.

Problems with the Microsoft Windows environment include:

- Often is slow, especially on and 8088 based computer.
- Complicated programming environment.
- Possible successful legal action by Apple Computer.

The Ada language was selected for use in the compute-intensive routines. Since these routines must run on a wide variety of computers, it was felt that Ada gave the best combination of portability and modern software capability.

3.2 ASTEC Architecture

ASTEC will consist of several modules. Many of the currently implemented or planned modules are described below.

• Manager	(ASTEC)	Manage files and launch other jobs.
• Modeler	(MODEL)	Build systems and launch analyses.
• Control Panel	(PANEL)	Control system parameters.
• Template Librarian	(LIBRN)	Allow user design of building blocks.
• Job Scheduler	(SCHED)	Schedule executable jobs (JOBxx).
• State Space	(STATE)	Execute operations using state space methods.
• Plotter	(GRAPH)	Plot results of JOBxx modules.
• Root Locus	(JOBRL)	Execute root locus analysis.
• Contour Locus	(JOBRC)	Execute root contour locus analysis.
• Dynamic Locus	(DYLOC)	New capability to display root locus as poles and zeroes are adjusted in real time.

• Frequency Response	(JOBFR)	Execute frequency response analysis.
• Freq. Resp. GH*	(JOBFS)	Execute GH-star analysis.
• Negative Real Axis	(JOBNR)	Execute a negative real axis analysis.
• Bode-Siggy	(JOBSG)	Execute a Bode-Siggy analysis.
• Describing Function	(JOBDF)	Execute describing function analysis.
• Time Response	(JOBTR)	Execute time response analysis.
• Linear Simulation	(JOBLS)	Compile, Link, and Execute a linear simulation.
• Nonlinear Simulation	(JOBNS)	Compile, Link, and Execute a non-linear simulation.
• Mode Significance	(JOBMS)	Execute a mode significance analysis.

The user will interact primarily with Block Diagram Manipulation (MODEL) Module. In many ways this module is the heart of ASTEC, and it was one of the first developed. The capabilities of ASTEC include classical control methods, simulation both linear and non-linear, multivariable controls and matrix methods, and new experimental capabilities -- including dynamic locus and three dimensional frequency response.

3.3 Uploading of problems to Mainframes and Super-Computers.

While personal computers are quite good in the fields of graphics and interactiveness, they often fall short in the field of number crunching, especially if hardware floating point (a coprocessor) is not installed. For this reason a capability to transfer compute intensive jobs away from the PC was deemed essential. The number crunching routines such as simulation and root locus evaluation were to be written as simple batch-oriented file processing programs, without any links to the Windows environment. These routines would read a text (ASCII) file containing the problem, solve the problem, and write out the results to another text file. The only other output would be periodic progress reports on the state of the computation. Such reports would be useful if the job was executing on the PC, or if the user wished to check on the status of the job on the remote computer.

As much as possible, it was desirable to have only one version of the source code for these job programs. Thus the code is written in or will be converted to the Ada language. Preliminary versions were written in Pascal for testing purposes.

A disadvantage of this technique is the necessity to transfer data between the PC and the remote computer. Since numeric formats vary between machines, it was decided that all transferable data should be in ASCII format only. A second disadvantage was the requirement for a job scheduling program that could interact with the remote computer in a generalizable way.

3.4 Downloading of results for convenient analyses.

Once the remote job has been completed, the results must be retrieved and displayed. The result files must be transferred to the PC, and a plotting program started. These operations are ideally under the control of the scheduler module, which would poll the remote computer to see if a job were completed. Upon completion it transfers the file and notifies the user, and possibly even starts a plotting-analysis program.

4 Current status

ASTEC is currently being developed on two systems, MS-DOS/Windows and the Macintosh. More progress has been made in the Windows version. A progress report of the latest developments is summarized here. The status is reported circa June 1989.

4.1 Microsoft Windows environment.

Development using Microsoft Windows is simultaneously most frustrating and most rewarding. While the Pascal language gives the compile-time checking needed in a large project of this sort, the Windows development tools are designed for the C language, in which much of the checking is the responsibility of the programmer. Thus a lot of time was spent chasing runtime problems that should have been caught in the compile and link steps. There are also a number of outright bugs and omissions in the development tools. Better development tools are certainly needed. A "prelink" program was developed at GSFC to automate the generation of the various linker definition files.

The advantages of Windows programming make the frustrations worthwhile. It is almost trivial to create menus, and only slightly less so to create dialog boxes and "read" the mouse buttons. Graphics generation is intricate, but no more so than many other graphics toolboxes. And the fact that only one print routine is sufficient for a vast variety of printers is absolutely wonderful.

4.2 Macintosh environment.

The Macintosh development of ASTEC, as was previously mentioned, is somewhat behind that of the Microsoft Windows development. The Macintosh Programmer's Workshop (MPW) was chosen as the programming environment. The Programmer's Workshop is versatile, and it is primarily intended for the creation of stand alone applications. The Pascal used with the workshop is run as a tool, that is part of the MPW's shell and thus frees the programmer from having to initialize Mac Toolbox managers, and menus or events are preformed by the MPW shell.

5 Future plans

In the next year the remaining capabilities of the INCA program will be added to ASTEC. At that point the software will be submitted to COSMIC for publication and distribution. In the following year work will proceed on the simulation module. This is anticipated to be the most involved portion, involving new routines (i.e., not from INCA) and a new language (Ada).

6 Conclusion

The ASTEC system shows that personal computers can be used to perform sophisticated controls analyses. The use of user interface techniques pioneered in the business word (word processors and spreadsheets) can be used to make the life of the controls engineer easier as well. Also, there is no need to give up the power available in the mainframe environment. The algorithms used in ASTEC are equivalent to those used on mainframes, and are only slightly inconvenienced by the somewhat limited capacities of today's personal computers. When completed, ASTEC is a tool that will serve engineers in the 90's.

References

1. Bauer, F. H. and Downing, J. P., INteractive Controls Analysis (INCA) User's Manuals (4 Volumes), Program Number GSC-12998, COSMIC, University of Georgia, Athens, Georgia, 1985. Updated

2. Herget, C. J. and Weis, T. P., Linear Systems Analysis Program User's Manual, Lawrence Livermore Laboratory, UCID-30184, Developed under contract W-7405-Eng-48.

3. Bauer, F. H. and Downing, J. P., Control System Design and Analysis using the INteractive Controls Analysis (INCA) program.

4. Frisch, H. P. and Bauer, F. H., Modern Numerical Methods for Classical Sampled Systems Analysis (SAMSAN Version 2) User's Guide, Program Number GSC-12827, COSMIC, University of Georgia, Athens, Georgia, 1984.

5. Advanced Continuous Simulation Language (ACSL) Reference Manual, Mitchell and Gauthier Associates, Concord, Mass, 1986

6. Zimmerman, B. G., Multi-Optimal Differential Equation Language (MODEL) User's Manual, Program Number GSC-12830, COSMIC, University of Georgia, Athens, Georgia, 1980.

Control/Structure Interaction

Design Methodology

Presented at the

3rd Annual Conference on

Aerospace Computational Control

Oxnard, CA

28-30 August, 1989

By

Dr. Hugh C. Briggs, Deputy Manager

William E. Layman, Technical Manager

JPL CSI Program

Jet Propulsion Laboratory

California Institute of Technology

Pasadena, CA

Control/Structure Interaction Design Methodology

Hugh C. Briggs, Deputy Technical Manager
William E. Layman, Technical Manager

Jet Propulsion Laboratory
4800 Oak Grove Drive
Pasadena, CA 91109

Abstract

The Control/Structure Interaction Program is a technology development program for spacecraft that exhibit interactions between the control system and structural dynamics. The program objectives include development and verification of new design concepts - such as active structure - and new tools - such as a combined structure and control optimization algorithm - and their verification in ground and possibly flight test. The new CSI design methodology is centered around interdisciplinary engineers using new tools that closely integrate structures and controls. Verification is an important CSI theme and analysts will be closely integrated to the CSI Test Bed laboratory. Components, concepts, tools and algorithms will be developed and tested in the lab and in future Shuttle-based flight experiments.

The design methodology is summarized in block diagrams depicting the evolution of a spacecraft design and descriptions of analytical capabilities used in the process. The multiyear JPL CSI implementation plan is described along with the essentials of several new tools. A distributed network of computation servers and workstations has been designed that will provide a state-of-the-art development base for the CSI technologies.

The NASA Control/Structure Interaction Program

The NASA CSI Program is an element of the Control of Flexible Structures Task in the NASA Civilian Space Technology Initiative. Three NASA Centers participate in the CSI Program: Langley Research Center, Marshall Space Flight Center and the Jet Propulsion Laboratory. This multiyear program to develop and validate new design technologies is organized around five elements: Systems and Concepts, Analysis and Design, Ground Test Methods, Flight Experiments and Guest Investigation Program. The CSI program goal is to develop validated technology that will be needed to design, verify and operate interactive control/structure systems to meet the ultraquiet structure requirements of 21st century NASA missions.

The CSI Program will integrate the advances made in other discipline technology programs to make the new spacecraft design methodology (see Figure 1). Controls programs such as Computational Control will develop a new generation of tools for multibody simulation, multibody component representation, and control analysis and synthesis. Structures technology programs such as Computational Mechanics will develop advanced finite element analysis codes. CSI will integrate these tools into a multidisciplinary environment and develop additional tools such as simultaneous structure and control optimization methods, and conceptual design tools for flexible spacecraft structure/control architectures. New CSI systems and concepts, such as active structure, will be developed and integrated into focus mission design examples.

Other developments that will enable high performance, flexible spacecraft design include an investigation of microdynamics and development of ground test methods for controlled flexible spacecraft structures. Microdynamic characterizations of spacecraft components such as joints and struts will identify the linearity of typical elements when dynamic motions are restricted to the submicron regimes required for future spacecraft. In addition, disturbance sources will be characterized at the microdynamic level to support analysis of ultraquiet spacecraft systems.

CSI Philosophy

A new philosophy is behind the CSI Design Methodology that supports improved integration of the traditional engineering disciplines utilized by the design team. These concepts emphasize integration, information sharing, and an environment that facilitates the development of new ideas and analytical capabilities. Flexible spacecraft design is a multidisciplinary process that involves several traditional engineering disciplines. For example, most organizations are structured to provide the design team with engineers from configuration design, controls, structures, mechanical design and electronics design. A major CSI objective is to demonstrate better integration of these disciplines in a working environment.

Optimal spacecraft design requires engineers who are interdisciplinary, who understand the operation and analysis of various spacecraft subsystems and who can capitalize on that understanding. The benefit of developing and utilizing the new CSI engineer is the extra margin of performance that can be gained by simultaneous optimization and the increased effectiveness of the design team that results. Beyond this, systems are sufficiently complex and must meet such intricate constraints that an interdisciplinary approach is required to generate feasible designs. Fortunately, in most cases spacecraft system design does not require great, in-depth knowledge in any one engineering discipline. CSI system engineering, if supported by a good analytical environment, needs only a working-level understanding of the central disciplines.

Spacecraft design is typically executed in a team environment because of the complexity, size and engineering breadth required. The design team is staffed with several engineers, each contributing one or more of the traditional engineering capabilities, but all working the systems issues on multiple fronts. The team is led by a system engineer who coordinates the team efforts, maintains the team focus and the uniformity of analysis. The team reports to one or more decision makers when analyses alone cannot form a basis for a choice and judgements are required. Organizations differ in their approach to decision making, in some instances giving a single manager sweeping decision-making authority, and in other situations constructing a tiered or layered decision-making system. In all cases, the design team and decision makers are acting on behalf of one or more groups of stakeholders and/or sponsors. The design methodology must be compatible with such organizational environments and surroundings, providing support and drawing resources as necessary.

To foster the development of interdisciplinary engineers and to facilitate the execution of the design process, the team members need to be collocated. Information exchange is critical to the design process and, although electronic media can help, geographical dispersion is a significant impediment. Syngerism occurs quite readily when structures engineers and controls engineers work side by side with the opportunity to share techniques, brainstorm ideas and teach each other tricks of the trade. Collocation is essential to building and maintaining an atmosphere of enthusiasm and excitement.

The design team must be supported by a modern computer environment to realize the potential of the new methodology. State-of-the-art tools are required and the boundaries of practical computation are always being stretched by new mission requirements. The computer system must provide rapid iteration and convergence of the spacecraft design if insight and ingenuity are to provide further system performance gains. Support for traceability, documentation, and reporting must be inherent in the computer environment and not simply a task that is levied after the completion of the design process. It is the computer system underlying the CSI methodology that will enable the verification of the spacecraft design in ground and flight test, and verification is an essential step in the methodology.

CSI Methodology

Systems built by humans have a readily observed life cycle that consists of progressive stages of activity from design to production to retirement (see Table 1). Various systems progress through the life cycle at different rates and organizations provide different tools for segments of the cycle. The CSI technology development activities primarily support the early system design activities. Certain analysis tools such as simulations can also be used to support mission operations. Other developments in computer aided engineering could provide access mechanisms to fabrication steps through design transfers. The design process is conveniently partitioned into three segments, conceptual design, preliminary design and detail design, although the boundary between the last two is expected to soften as computer-based analytical capabilities improve. This partitioning allows exploitation of the best features of existing, large-scale modeling and analysis tools, as well as the smaller model optimization abilities of the new tools. See Figure 2.

Conceptual Design

Experience indicates that most of the really significant trades and design decisions are made by the system design team in arriving at a system concept that, based upon simple analysis, should meet most objectives and constraints. This was borne out during the early design steps of a Focus Mission Interferometer. The system conceptual design is typically depicted in a mechanical layout incorporating all major subsystems.

Several significant choices may be imbedded in the conceptual design that may be difficult to change or revisit. For example, the location, arrangement and connectivity of essential mission critical elements is defined and used as the basis for subsequent analysis. Without efficient design tools, most certainly computer-based, this step can not be repeated without significant elapsed time and labor. Aspects of the statement of the design problem might include maneuver sequence and operational scenarios. Since these form the initial conditions for the design team, any significant change would certainly invalidate the conceptual design.

Conceptual design trades are typically based upon engineering judgement and backed by simple analyses. Little documentation is usually prepared to send forward with the completed design. The design team at this stage is quite small, perhaps consisting of the systems engineer and one or two discipline engineers. The justification, assumptions and trades are carried mentally and the design is advanced until too many ideas get lost in the process. Often, the end user is consulted frequently as the design progresses and this raises questions about the users' true intentions. The design progresses until a meaningful problem can be stated and answered with minimal number of uncertain aspects.

When the design process is viewed as the ultimate selection of a single point design from a large, multidimensional design parameter space, it can be seen that the decisions leading to the conceptual design substantially constrict the spaces to be considered in the following design steps. Indeed, the fundamental operating characteristics of the system are set by the end of the conceptual design phase.

The CSI methodology emphasizes the early application of analytical methods to the conceptual design phase. To demonstrate this, a conceptual design tool will be developed which will (1) support definition and tracing of requirements, (2) provide 3-dimensional modeling for concept depiction, and (3) provide integrated analytical methods to facilitate system trades.

Preliminary Design

With one or two system concepts in hand as a result of the conceptual design phase, the space of design parameters can be explored with new numerical optimization and performance analysis tools. The design variables might include structural parameters such as truss element areas and control parameters such as feedback gains. This simultaneous optimization of structure and control parameters will lead to a better system optimum than sequential optimization of the individual subspaces. Multiple objective optimization techniques, better known as vector optimization, allow the performance functions to include system mass, system power, closed loop performance, robustness and system cost. Notice that these are competing and incommensurate objectives and that application of vector optimization will lead to a family of (Pareto-optimal) solutions.

In general, the design variables fall into the two categories of either continuous or discrete variables. Member cross-sectional area is an example of a continuous design variable and actuator locations are examples of discrete variables. The optimization with respect to the continuous variable can be based upon homotopy or multiple objective techniques while model changes or dynamic programming is required for the discrete variables. Furthermore, certain performance functions, for example those that are not expressible as analytic functions of the design variables, might be utilized in a final manual analysis step using traditional analysis tools. System settling time and certain frequency domain transfer function properties are typical examples of such performance measures. For these metrics, numerical gradients might be computed a priori for representative locations in the design space and utilized with interpolation in subsequent optimizations.

The limitations of current hardware and optimization algorithms will place restrictions on the size of the design problem at this stage. Models with less than a few hundred degrees of freedom will be required initially to keep the design session interactive. This is sufficient to allow the designer to explore the intracies of the system design space and perform design trades with analytical support. The results of these analyses and optimizations are presented to the project decision maker to select from the design space one or two concepts with tightly bounded decision parameter ranges, to take forward into detailed design.

Detailed Design

Within certain restrictions, a detailed evaluation and tuning of the surviving candidate design(s) can be adequately executed based upon current state-of-the-art tools. The traditional large model analysis faithfully represents the physical behavior of the system and can be validated with component, subsystem and system level testing of most constituent technologies. However, if significant non-linear behavior is present in the problem or system models must be developed from many large component models, significant limitations remain.

In this phase, the system design parameters must be tuned to meet detailed performance specifications and all phases of the mission must be analyzed. Realistic operating scenarios must be developed to provide maneuver profiles, environmental effects and disturbance characterizations. The modest optimization models must be expanded or extrapolated into detailed models and analyzed in the realistic mission contexts.

Several analysis systems currently exist that support this analysis phase. Representative systems include Boeing's IAC/ISM, SDRC's I-DEAS and NASA's IMAT. Further development in this technology will be to improve data manipulation and retrieval mechanisms, to improve the human-machine interface and presentation manager, and to include new analytical methods, for example, optics and thermal analysis.

Implementation of the CSI Methodology - The Design Environment

The design environment represents the instantiation of the methodology and consists of several elements. The following section will address the computer systems and the laboratory testing facilities. The software and analytical tools were described in the preceding methodology overview section.

The CSI computer system is a distributed network-based system consisting of workstations and servers (Figure 3). Laboratory testing computers are attached to the network to support the close integration of verification in test to the development of systems concepts and tools. Sufficient commercial technology exists to support a heterogeneous equipment set based upon standard network interfaces. For example, systems from Apple, DEC, Sun, Apollo, HP, Silicon Graphics and others can all participate in an Ethernet network using TCP/IP. This capability supports various user preferences and capabilities as well as providing the mechanism to protect existing corporate investments in computer systems.

The distributed system utilizes servers for those functions not allocated to the per-engineer workstations. Large computers, such as a CRAY or departmental VAX, function as compute servers to provide an execution site for large, compute intensive jobs. Other servers might provide specialized capabilities for animation, data base management or communications. Most workstation companies make it financially attractive to collect most of the system disk resources in one or more file servers that support some form of a network disk system (e.g. Sun's NFS). These file servers are repositories for large data sets, system executables and application libraries.

The workstations must support the interactive design environment with excellent speed and graphics. The CSI methodology requires computation of intermediate sized (ie. 100+ states) problems and presentation of solid models on the workstations. Representative derived requirements for workstations are: 3-10 MIP 32 bit CPU, 12-16 Mb memory, Unix operating system, 200 Mb disk, Ethernet interface, 3-D vector graphic accelerator and windowed presentation manager with a mouse.

The network environment also extends into the laboratory where verification and validation experiments are executed on the CSI Test Bed. The computing environment internal to the lab is shown in Figure 4. The four functions are: real-time control, experiment supervision, modal analysis and software development. Individual systems can be readily purchased to perform each function although it is possible to configure certain commercial systems to perform multiple duties. In any case, the software development system will most probably not be instantiated in the laboratory, using individual analyst workstations and the experiment supervisory computer instead.

The real-time control computer system will be a distributed, multiprocessor computer based upon commercial VMEbus products. The operating system supports remote consoles, software loading and unloading, a prioritized scheduler and shared memory message passing. An excellent example is VxWorks from Wind Rivers although the underlying kernel requires additional multiprocessor extensions. Analysts will prepare simple control subroutines on their workstation and produce a load module just as they would any program for execution. Remote login facilities are provided for access to any real-time CPU and a C-like shell provides the operator interface. Products such as Dbx-Works provide source level symbolic debugging.

The experiment supervisory computer provides the laboratory operator console and overall control of the Test Bed. This system monitors and logs environmental variables such as temperature and air velocity, monitors a panic button during experiment execution and collects measurements from the external truth sensor. Remote access from any network workstation allows remote execution of experiments.

The modal analysis and data acquisition system is a standard commercial product and supplies a necessary function found in all dynamics laboratories. To characterize the structural dynamics of the test article, a modal survey can be performed utilizing a large number of accelerometers distributed over the structure. This is typically done to verify open loop system models but should also be an integral part of closed loop system performance measurement. Results are available to any analyst via the network.

For precision controlled structures, the laboratory environmental requirements are quite severe. Noise and seismic disturbance constraints will require all personnel and actively cooled electronics to be seqestered in an adjacent control room. During tests, the test chamber must be unoccupied, closed, and carefully maintained at constant temperature. This will require development of control procedures for remote experiments and forms the basis for emulation of on-orbit flight experiments. Shuttle command, communication and control features can be readily emulated with the network-based computer system and the computational capabilities of space-qualified computers can be replicated in the ground test hardware. Figure 5 illustrates scale and complexity of a test bed that models a space-based interferometer.

The CSI Design Handbook

To provide the essential technology transfer mechanism, a CSI Design Handbook will be developed over the life of the CSI program. This Handbook will contain verified design standard practices, definitions, examples and an implementation guide. It will be published by NASA with contributions from all participating centers in intermediate and final forms. Table 2 shows the Table of Contents of the Handbook.

CSI Testing Requirements

CSI will validate the system concepts, components and tools in realistic ground tests. Where the ground environment precludes acceptable verification due to such effects as the gravity field, seismic, acoustic disturbances and size limitations, flight tests will be proposed to complete the development and validation of the technology.

Testing is recognized as an essential component of the design process. The design methodology will include close coupling of the analysis with the testing and evaluation of results. This will foster verification of new system concepts and designs as well as provide analytical support for new ground test techniques. In addition, the CSI flight experiments will be designed to develop techniques for extending ground testing methods to on-orbit flight tests.

As a result of integrating testing into the design process, several capabilities must be built into the ground test facility. Interactive evaluation of control system performance must be provided to explore system phenomena and to enable reconciliation of measured behavior with predicted behavior. To validate the new optimization methods and to evaluate system robustness properties, substitution of any structural element will be provided without dismantling large subsections of the test article. Support for remote investigation of system performance via the electronic network, already mentioned as a requirement for CSI analysts, will also include support for off-site Guest Investigators. This access includes all test measurement data as well as the control programs of the real-time control computer. Finally, emulation of all essential Shuttle command, communications and control features that impact proposed flight experiments will be provided.

Summary

Control/Structure Interactions is a NASA technology development program to develop new methods for designing integrated control/structure systems and to develop new methods to test control of large flexible CSI systems. Missions of the near future such as advanced Earth observation platforms and large, flexible antennas will be significantly enhanced, and new classes of missions such as large optical interferometers and large optical telescopes will be enabled.

Acknowledgement

The research described in this paper was performed as part of the Control Structure Interaction (CSI) program at the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

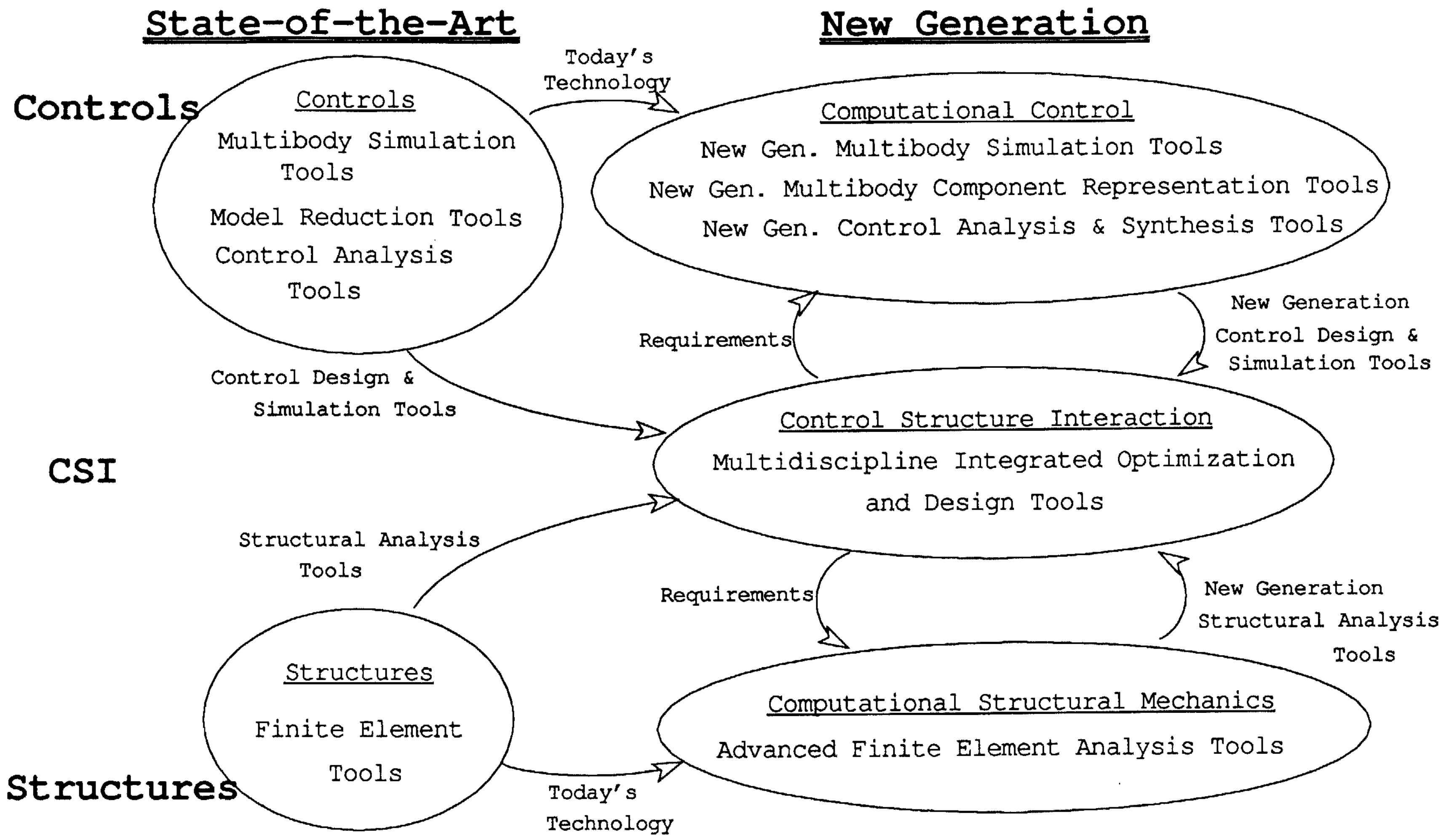
Table 1. System Life Cycle

- Pre-Project Planning
- Conceptual Design
- Detail Design
- Fabrication and Production
- Functional and Environmental Testing
- Mission Operations
- Retirement

Table 2. CSI Design Handbook Table of Contents

Philosophy
Procedure
Worked Examples
Lessons Learned
Appendices
    Tool Descriptions
    Implementation Guides

Figure 1. Relationship Between Controls, Structures, and CSI Tools



State-of-the-Art

New Generation

Controls

CSI

Structures

Controls

Multibody Simulation Tools

Model Reduction Tools

Control Analysis Tools

Today's Technology

Computational Control

New Gen. Multibody Simulation Tools

New Gen. Multibody Component Representation Tools

New Gen. Control Analysis & Synthesis Tools

Requirements

New Generation Control Design & Simulation Tools

Control Design & Simulation Tools

Control Structure Interaction

Multidiscipline Integrated Optimization and Design Tools

Structural Analysis Tools

Requirements

New Generation Structural Analysis Tools

Structures

Finite Element Tools

Today's Technology

Computational Structural Mechanics

Advanced Finite Element Analysis Tools

Figure 2. Analysis Phases of the CSI Design Methodology

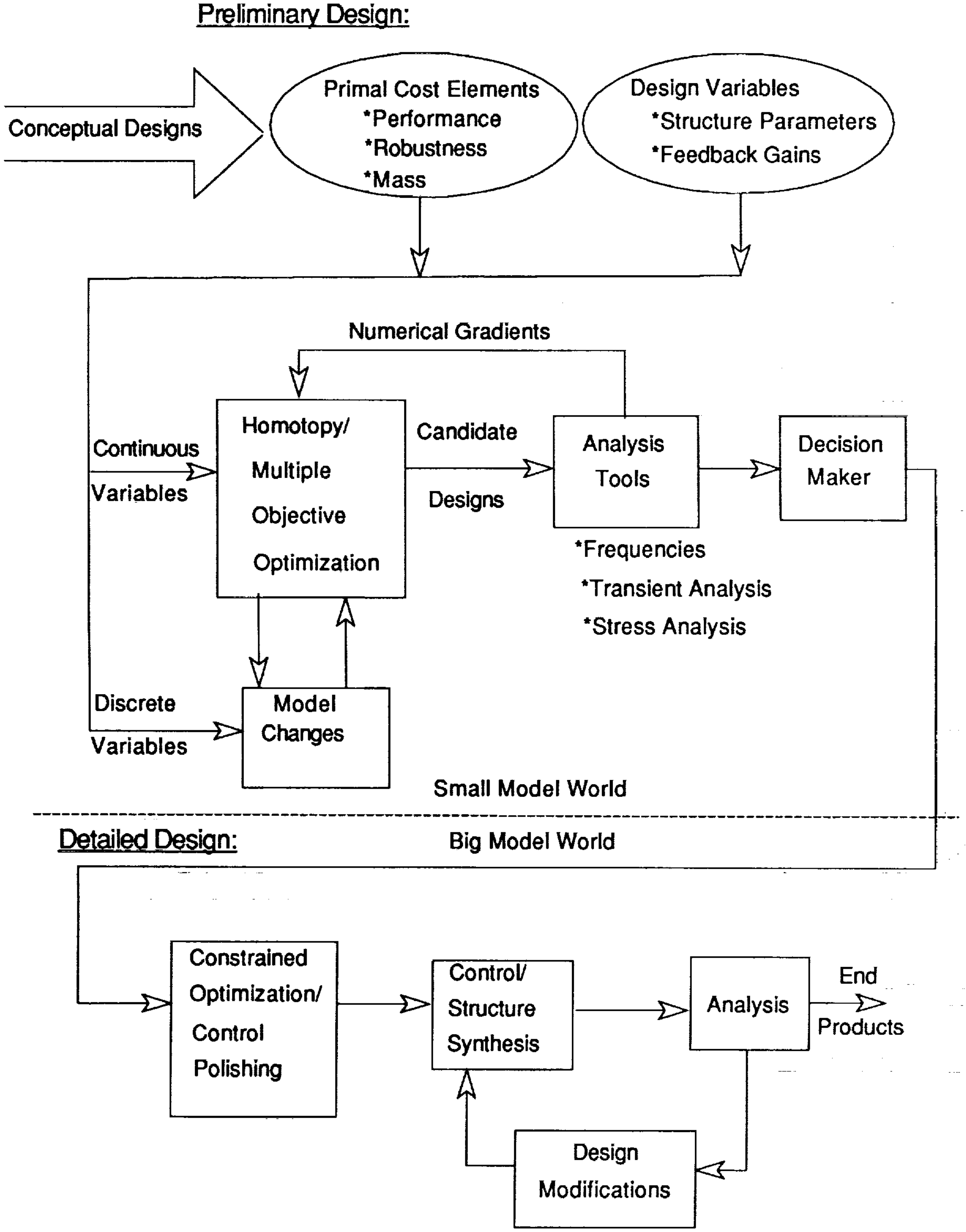
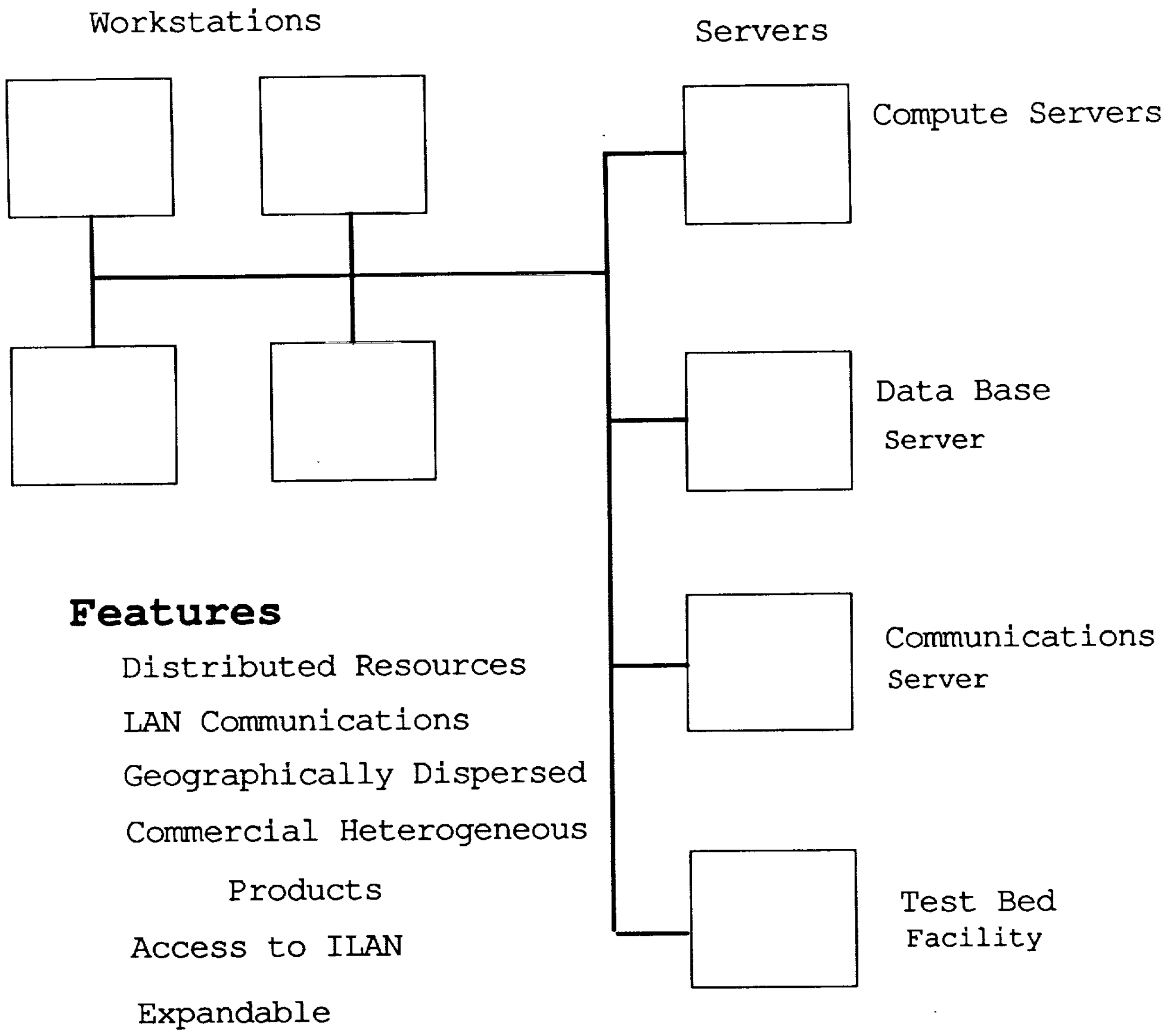


Figure 3. CSI Computing Network



Features

Distributed Resources

LAN Communications

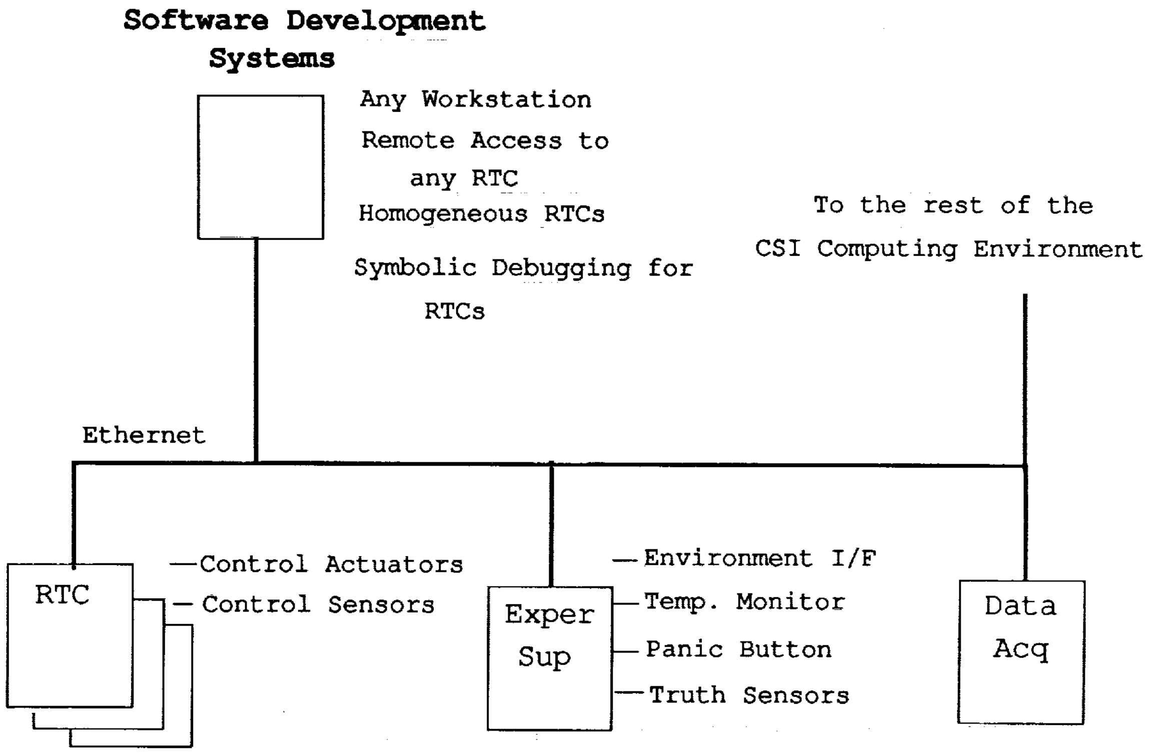
Geographically Dispersed

Commercial Heterogeneous Products

Access to ILAN

Expandable

Figure 4. Test Bed Computing Environment



Real-Time Controllers

Shell I/F & Real-Time Kernel

Multi-Processor Functions

Hardware Control

S/W Development Communications Via Enet

RT Messaging via Shared Memory

Heterogeneous RTC CPUs

Experiment Supervisor

Real-Time Unix System

Experiment Control

Environment Monitor

Facility Operator Station

Remote Access I/F

Record Keeping

Modal Analysis & Data Acquisition

Modal Test

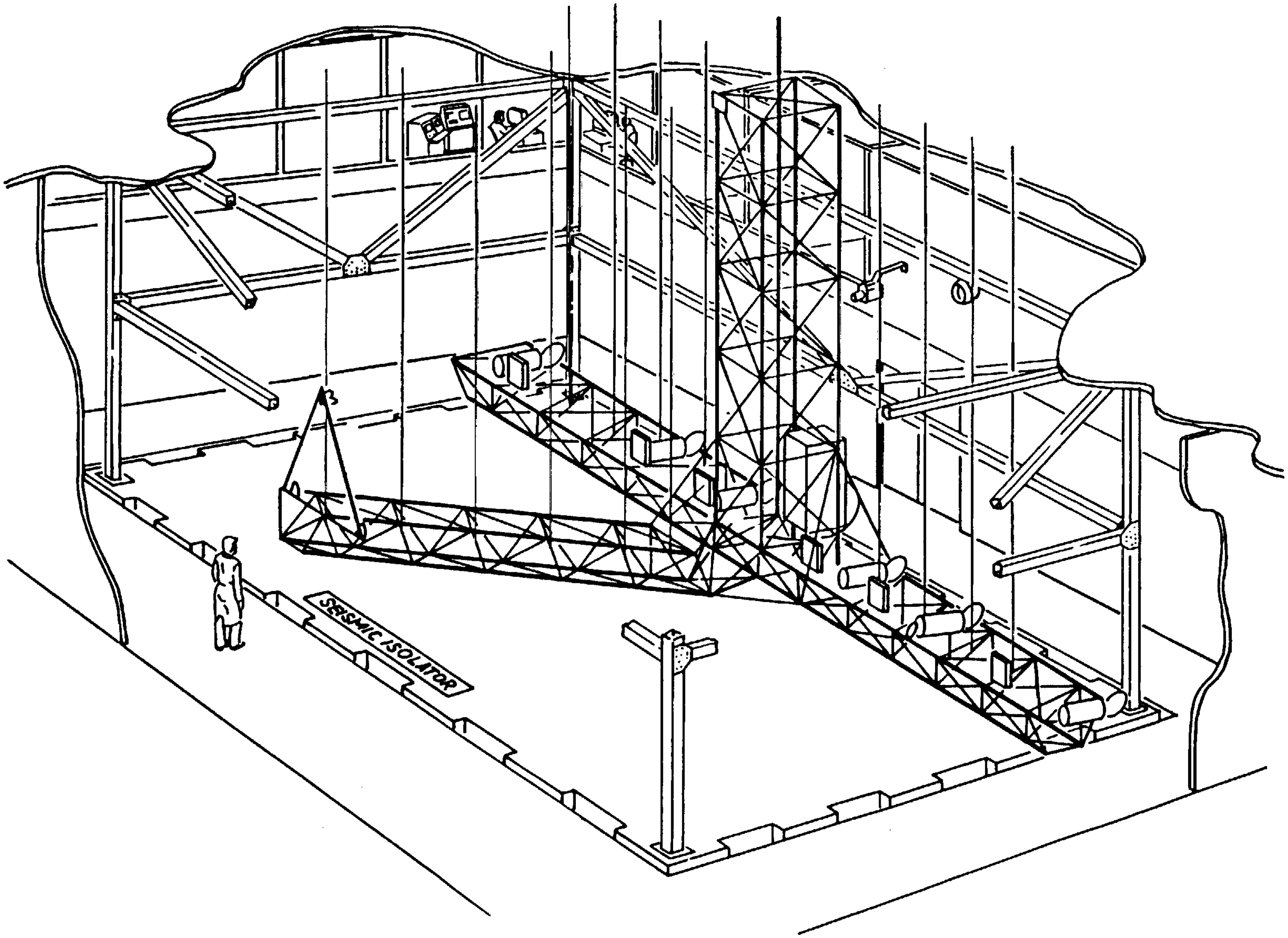
System Identification

Commercial Product

JPL

Figure 5. Integrated Controls and Structures Laboratory

CARL



N90-23054

Model Reduction for Flexible Spacecraft with Clustered Natural Frequencies

T.W.C. Williams
NASA Langley Research Center
Hampton, VA 23665
W.K. Gawronski
Jet Propulsion Laboratory
Pasadena, CA 91109

Two approaches to the problem of modal reduction for flexible spacecraft that have proved very useful are balancing and modal truncation. Furthermore, it is well-known that a modal representation of a lightly damped flexible structure with widely spaced natural frequencies is approximately balanced. Consequently, reduction in either coordinate system gives similar results for this case. It is important to note, however, that flexible space structures typically have clusters of closely spaced frequencies. In such cases, reduction in modal coordinates can give large errors, while the error obtained using balancing is generally much smaller. A new reduction procedure which combines the best features of modal and balanced reduction is therefore developed in this paper. It is more efficient than balanced reduction of the full system, as it only involves balancing those subsystems of close modes that are highly correlated, yet is shown to yield results which are essentially as good.

N90-23055

Substructural Controller Synthesis

Tzu-Jeng Su Roy R. Craig, Jr.

Department of Aerospace Engineering and Engineering Mechanics
The University of Texas at Austin
Austin, Texas 78712

A decentralized design procedure which combines substructural synthesis, model reduction, decentralized controller design, subcontroller synthesis, and controller reduction is proposed for the control design of flexible structures. The structure to be controlled is decomposed into several substructures, which are modelled by component mode synthesis methods. For each substructure, a subcontroller is designed by using the linear quadratic optimal control theory. Then, a controller synthesis scheme called Substructural Controller Synthesis (SCS) is used to assemble the subcontrollers into a system controller, which is to be used to control the whole structure.

1. Introduction

Component mode synthesis (CMS) methods [1,2] have proved to be indispensible for analyzing the dynamic response of large structural systems. Finite element models of order 10^4 are reduced, by the use of CMS methods, to order 10^2 to make possible the accurate numerical simulation of dynamic events. The most widely used CMS methods are those described in Refs. [3]-[6].

For the past decade there has been a growing interest in the topic of control of flexible structures, or control-structure interaction (CSI), but so far little has been done to employ CMS concepts in the design of controllers for flexible structures. Although many decentralized control methods have been developed for general linear systems, there have been few applications of decentralized control to flexible structures. In Ref. [7], Young applies the overlapping decomposition method, which was formulated by Ikeda and Šiljak for large scale systems [8,9], to develop a control design approach called Controlled Component Synthesis (CCS). The component finite element models employed by Young include boundary stiffness and inertia loading terms in the manner introduced in the CMS literature in Ref. [6]. The controller design is carried out at the component level. Then, the large controlled structure is synthesized from the controlled components. The idea behind the CCS approach is the same as that behind the CMS method. However, the way the structure is decomposed is not the same. Recently, in an attempt to simplify the decentralized control design for structures, Yousuff extended the concept of inclusion principle, which was developed along with the overlapping decomposition method by Ikeda et al. [9], to systems described in matrix second-order form [10]. The substructural model in Yousuff's work is an expanded component, i.e., the original boundary of the component is expanded into the adjacent component, which is similar to the substructure used in Young's CCS method. The expanded component is a result of overlapping decomposition.

The terms *component synthesis* and *substructure coupling* both refer to procedures whereby structures are considered to be composed of interconnected components, or substructures.

The need to "load" the boundary of one component with stiffness and inertia terms from the adjacent components is considered to be a drawback of the CMS method of Ref. [6] in comparison with the methods of Ref. [3]-[5]. Likewise, a decentralized control design procedure that is not based on overlapping components should have an advantage over the methods described in Refs. [7] and [10]. In this paper a decentralized control design process called *Substructural Controller Synthesis* (SCS), which was developed in Ref. [11], is described. Figure 1 shows the various steps involved in the SCS method described in this paper. First, a natural decomposition, called *substructuring decomposition,* of a structural dynamics system is defined. It is well known that for structural dynamics equations described in matrix second-order form, the system matrices of the whole structure can be assembled from the system matrices of substructures. For each substructure, a subcontroller is designed by an optimal control design method. Then, the system controller, which is to be used to control the whole structure, is synthesized from the subcontrollers by using the same assembling scheme as that employed for structure matrices. The last step is to reduce the order of the system controller to a reasonable size for implementation. This can be done by employing any existing efficient controller reduction method, for instance, the *Equivalent Impulse Response Energy Controller Reduction Algorithm* developed in Ref. [12]. The final control implementation in Figure 1 is a centralized control, which means the final controller for implementation is a system controller. However, the control design is decentralized, because each subcontroller is designed independently.

The substructure used in the Substructural Controller Synthesis method is a natural component, i.e., not an expanded component like that in Young's method. One advantage of using natural components is that SCS can be effectively incorporated with the Component Mode Synthesis method to design controllers for large scale structures. The substructures can be modelled by a CMS method and then assembled together to form an approximate model for the whole structure. The subcontrollers can be designed based on the CMS substructures and can then be assembled together to form a system controller for the whole structure. Another attractive feature of the SCS controller is that it can be updated economically if part of the structure changes. Since the system controller is synthesized from subcontrollers, if one substructure has a configuration or parameter change, the only subcontroller which needs to be redesigned is the one associated with that substructure. Therefore, the SCS controller is, in fact, an adaptable controller for structures with varying configuration and/or with varying mass and stiffness properties.

The organization of this paper is as follows. In Section 2, substructuring decomposition is defined for a general linear time-invariant system described by first-order equations. In Section 3, a substructuring decomposition for structural dynamics systems is developed. Then, based on the substructuring decomposition, a Substructural Controller Synthesis method is formulated in Section 4. Finally, in Section 5, a plane-truss example is used to illustrate the applicability of the proposed method.

2. Substructuring Decomposition

Consider a linear time-invariant system described by

$$\begin{aligned} S\dot{z} &= Az + Bu \\ y &= Cz \end{aligned} \tag{1}$$

where $z \in R^n$ is the state vector, $u \in R^l$ is the input vector, and $y \in R^m$ is the output vector. S, A, B, and C are the system matrices with appropriate dimensions.

Next, consider another linear time-invariant system described by

$$\begin{aligned} \tilde{S}\dot{\tilde{z}} &= \tilde{A}\tilde{z} + \tilde{B}u \\ y &= \tilde{C}\tilde{z} \end{aligned} \tag{2}$$

with the system matrices in the following block diagonal form

$$\tilde{S} = \begin{bmatrix} S_1 & & & \\ & S_2 & & \\ & & \ddots & \\ & & & S_\nu \end{bmatrix}, \quad \tilde{A} = \begin{bmatrix} A_1 & & & \\ & A_2 & & \\ & & \ddots & \\ & & & A_\nu \end{bmatrix}, \quad \tilde{B} = \begin{bmatrix} B_1 & & & \\ & B_2 & & \\ & & \ddots & \\ & & & B_\nu \end{bmatrix}, \quad \tilde{C} = \begin{bmatrix} C_1 & & & \\ & C_2 & & \\ & & \ddots & \\ & & & C_\nu \end{bmatrix}$$

and

$$\tilde{z} = \left[z_1^T,\ z_2^T,\ \ldots,\ z_\nu^T\right]^T, \quad u = \left[u_1^T,\ u_2^T,\ \ldots,\ u_\nu^T\right]^T, \quad y = \left[y_1^T,\ y_2^T,\ \ldots,\ y_\nu^T\right]^T$$

The dimensions of the variables are $z_i \in R^{n_i}$, $u_i \in R^{l_i}$,and $y_i \in R^{m_i}$. It is assumed that system (1) and system (2) have the same set of inputs ($\sum_{i=1}^{\nu} l_i = l$) and the same set of outputs ($\sum_{i=1}^{\nu} m_i = m$). Therefore, for this case it is appropriate to use u and y in Eq. (2) as well as in Eq. (1). Because of the block diagonal form of the system matrices, system (2) is, in fact, a collection of ν decoupled subsystems

$$\begin{aligned} S_i\dot{z}_i &= A_i z_i + B_i u_i \\ y_i &= C_i z_i \end{aligned} \qquad i = 1,\ 2,\ \ldots,\ \nu \tag{3}$$

Now let us define a substructuring decomposition. System (2) is said to be a *substructuring decomposition* of system (1) if there exists a *coupling matrix* $\tilde{T}$ such that the following relationships hold

$$S = \tilde{T}^T\tilde{S}\tilde{T} \qquad A = \tilde{T}^T\tilde{A}\tilde{T} \qquad B = \tilde{T}^T\tilde{B} \qquad C = \tilde{C}\tilde{T} \tag{4}$$

and if the states of the two systems can be related by

$$\tilde{z} = \tilde{T}z \tag{5}$$

The above relationships merely state that the system matrices of system (1) are assemblages of the system matrices of the subsystems in Eq. (3). Therefore, system (1) will be referred to as the *assembled system* and system (2) will be referred to as the *unassembled system.*

3. Substructuring Decomposition of Structural Dynamics Systems

In this section, the substructuring decomposition of a structural dynamics system is formulated. Without loss of generality, we will consider a structure composed of two substructures that have a common interface, as shown in Figure 2. It is assumed that the control inputs and the measurement outputs are localized. In the present context, "localized control inputs" means that the actuators are distributed such that u_α is applied to the α-substructure only and u_β is applied to the β-substructure only. "Localized measurements" means that y_α measures only the response of the α-substructure and y_β measures only the response of the β-substructure.

Let the equations of motion of the two substructures be represented by

$$\begin{aligned} M_i\ddot{x}_i + D_i\dot{x}_i + K_i x_i &= P_i u_i \\ y_i &= V_i x_i + W_i\dot{x}_i \end{aligned} \qquad i = \alpha,\ \beta \tag{6}$$

It is noted here that the above dynamics equations for the substructures do not have to be exact (full-order) models. They can be approximate (reduced-order) models obtained by any model reduction method, say a Component Mode Synthesis method [2]. The dynamics of the the assembled structure (the structure as a whole) is described by

$$\begin{aligned} M\ddot{x} + D\dot{x} + Kx &= Pu \\ y &= Vx + W\dot{x} \end{aligned} \tag{7}$$

Since the two substructures have a common interface and are parts of the assembled structure, the displacement vectors of the substructures and the displacement vector of the assembled structure are related. There exists a coupling matrix T which relates x_α, x_β, to x as follows:

$$\begin{Bmatrix} x_\alpha \\ x_\beta \end{Bmatrix} = Tx = \begin{bmatrix} T_\alpha \\ T_\beta \end{bmatrix} x \tag{8}$$

Given the coupling matrix T, it can be shown that the system matrices of the assembled structure and the system matrices of the substructures satisfy the following relations:

$$\begin{aligned} &M = T^T \begin{bmatrix} M_\alpha & 0 \\ 0 & M_\beta \end{bmatrix} T, \quad D = T^T \begin{bmatrix} D_\alpha & 0 \\ 0 & D_\beta \end{bmatrix} T, \quad K = T^T \begin{bmatrix} K_\alpha & 0 \\ 0 & K_\beta \end{bmatrix} T \\ &P = T^T \begin{bmatrix} P_\alpha & 0 \\ 0 & P_\beta \end{bmatrix}, \qquad V = \begin{bmatrix} V_\alpha & 0 \\ 0 & V_\beta \end{bmatrix} T, \qquad W = \begin{bmatrix} W_\alpha & 0 \\ 0 & W_\beta \end{bmatrix} T \end{aligned} \tag{9}$$

The above relationships can be proved by using the method of Lagrange's equation of motion [1]. Therefore, it is an inherent property of structural dynamics systems that the system matrices of the assembled structure can be obtained by assembling the system matrices of the substructures. This property is, in fact, the essence of all "matrix assemblage" methods, e.g., the Finite Element Method and Component Mode Synthesis. The above formulation is based on the matrix second-order equation of motion. For control design purposes, a first-order formulation which leads to a substructuring decomposition of the structural dynamics system is required.

Let us rewrite the equation of motion (6) in the following first-order form

$$\begin{aligned} &\underset{(S_i)}{\begin{bmatrix} D_i & M_i \\ M_i & 0 \end{bmatrix}} \underset{(\dot{z}_i)}{\begin{Bmatrix} \dot{x}_i \\ \ddot{x}_i \end{Bmatrix}} = \underset{(A_i)}{\begin{bmatrix} -K_i & 0 \\ 0 & M_i \end{bmatrix}} \underset{(z_i)}{\begin{Bmatrix} x_i \\ \dot{x}_i \end{Bmatrix}} + \underset{(B_i)}{\begin{bmatrix} P_i \\ 0 \end{bmatrix}} u_i \\ &y_i = \underset{(C_i)}{[V_i \ \ W_i]} \underset{(z_i)}{\begin{Bmatrix} x_i \\ \dot{x}_i \end{Bmatrix}} \end{aligned} \qquad i = \alpha,\ \beta \tag{10}$$

where the symbol under each matrix denotes that this equation corresponds to Eq. (3). Similarly, Eq. (7) can be rewritten as

$$\begin{aligned} &\underset{(S)}{\begin{bmatrix} D & M \\ M & 0 \end{bmatrix}} \underset{(\dot{z})}{\begin{Bmatrix} \dot{x} \\ \ddot{x} \end{Bmatrix}} = \underset{(A)}{\begin{bmatrix} -K & 0 \\ 0 & M \end{bmatrix}} \underset{(z)}{\begin{Bmatrix} x \\ \dot{x} \end{Bmatrix}} + \underset{(B)}{\begin{bmatrix} P \\ 0 \end{bmatrix}} u \\ &y = \underset{(C)}{[V \ \ W]} \underset{(z)}{\begin{Bmatrix} x \\ \dot{x} \end{Bmatrix}} \end{aligned} \tag{11}$$

where the symbol under each matrix denotes that this equation corresponds to Eq. (1).

Combination of the two substructure equations in Eq. (10) gives the first-order equation of motion of the unassembled system in the form of Eq. (2).

$$\begin{aligned} &\underset{(\tilde{S})}{\begin{bmatrix} S_\alpha & 0 \\ 0 & S_\beta \end{bmatrix}} \underset{(\dot{\tilde{z}})}{\begin{Bmatrix} \dot{z}_\alpha \\ \dot{z}_\beta \end{Bmatrix}} = \underset{(\tilde{A})}{\begin{bmatrix} A_\alpha & 0 \\ 0 & A_\beta \end{bmatrix}} \underset{(\tilde{z})}{\begin{Bmatrix} z_\alpha \\ z_\beta \end{Bmatrix}} + \underset{\tilde{B}}{\begin{bmatrix} B_\alpha & 0 \\ 0 & B_\beta \end{bmatrix}} \begin{Bmatrix} u_\alpha \\ u_\beta \end{Bmatrix} \\ &y \equiv \begin{Bmatrix} y_\alpha \\ y_\beta \end{Bmatrix} = \underset{(\tilde{C})}{\begin{bmatrix} C_\alpha & 0 \\ 0 & C_\beta \end{bmatrix}} \begin{Bmatrix} z_\alpha \\ z_\beta \end{Bmatrix} \end{aligned} \tag{12}$$

It can be shown that the unassembled system (12) is a substructuring decomposition of the assembled system (11). That is, $(S,\, A,\, B,\, C)$ in Eq. (11) and $(\tilde{S},\, \tilde{A},\, \tilde{B},\, \tilde{C})$ in Eq. (12) satisfy the relations in Eq. (4). The state vector of the assembled structure and the state vectors of the substructures are related by a coupling matrix $\tilde{T}$ as

$$\underset{(\tilde{z})}{\begin{Bmatrix} x_\alpha \\ \dot{x}_\alpha \\ x_\beta \\ \dot{x}_\beta \end{Bmatrix}} = \underset{(\tilde{T})}{\begin{bmatrix} T_\alpha & 0 \\ 0 & T_\alpha \\ T_\beta & 0 \\ 0 & T_\beta \end{bmatrix}} \underset{(z)}{\begin{Bmatrix} x \\ \dot{x} \end{Bmatrix}} \tag{13}$$

Physically, the coupling matrix $\tilde{T}$ that relates the state vectors of the substructures and the state vector of the assembled structure simply describes the compatibility conditions which must be imposed on the interface degrees of freedom. Let x_i represent the physical displacement coordinates of substructures i, and let the physical coordinates of the substructures be partitioned into two sets: Interior coordinates (I-set) and Boundary coordinates (B-set), as shown in Figure 2.

The displacement compatibility condition requires that $x_\alpha^B = x_\beta^B$. If the displacement vector of the assembled structure is represented by

$$x = \begin{Bmatrix} x_\alpha^I \\ x^B \\ x_\beta^I \end{Bmatrix}$$

where x^B is the vector of interface degrees of freedom, then the three displacement vectors x_α, x_β, and x are related by

$$\begin{Bmatrix} x_\alpha \\ x_\beta \end{Bmatrix} \equiv \begin{Bmatrix} x_\alpha^I \\ x_\alpha^B \\ x_\beta^I \\ x_\beta^B \end{Bmatrix} = \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & I \\ 0 & I & 0 \end{bmatrix} \begin{Bmatrix} x_\alpha^I \\ x^B \\ x_\beta^I \end{Bmatrix} \equiv \begin{bmatrix} T_\alpha \\ T_\beta \end{bmatrix} x \tag{14}$$

with

$$T_\alpha = \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \end{bmatrix}, \qquad T_\beta = \begin{bmatrix} 0 & 0 & I \\ 0 & I & 0 \end{bmatrix}$$

The velocity compatibility condition requires that $\dot{x}_\alpha^B = \dot{x}_\beta^B$, which leads to

$$\dot{x}_\alpha = T_\alpha \dot{x}, \qquad \dot{x}_\beta = T_\beta \dot{x} \tag{15}$$

Combination of Eqs. (14) and (15) shows that the state vectors of the substructures and the state vector of the assembled structure are related by a coupling matrix $\tilde{T}$ as in Eq. (13).

4. Substructural Controller Synthesis

The discussion in this section is based on the two-component structure in Section 3. The system is assumed to be subject to disturbance and observation noise. Therefore, the formulation is a stochastic one. At the end of this section, a control design procedure called the LQGSCS Algorithm is used to summarize the Substructural Controller Synthesis scheme. The method proposed can also be applied to a deterministic problem with only slight modification.

First, let the dynamics of the *assembled structure* (the structure as a whole) in Figure 2 be described by

$$\begin{aligned} S\dot{z} &= Az + Bu + N\varpi \\ y &= Cz + v \end{aligned} \tag{16}$$

where input disturbance ϖ and output disturbance v are assumed to be uncorrelated zero-mean white noise processes. For a linear stochastic system with incomplete measurement, optimal state feedback control design requires a state estimator, called a Kalman filter, to reconstruct the states for feedback. The state estimator of the plant described by Eq. (16) has the form

$$S\dot{q} = Aq + Bu + F^{o}(y - Cq) \tag{17}$$

where F^{o} is determined by solving a Riccati equation. If a feedback control scheme $u = G^{o}q$ is incorporated with Eq. (17) to control the plant, the estimator becomes a controller in the form

$$\begin{aligned} S\dot{q} &= (A + BG^{o} - F^{o}C)q + F^{o}y \\ u &= G^{o}q \end{aligned} \tag{18}$$

where superscript o denotes optimal design. The feedback gain matrix G^{o} is determined by minimizing a performance index

$$J = \lim_{t\to\infty} \frac{1}{2} E[z^T Q z + u^T R u] \tag{19}$$

For structural control problems, the weighting matrix Q is usually chosen to be

$$Q = \begin{bmatrix} K & 0 \\ 0 & M \end{bmatrix} \tag{20}$$

such that the first term in the performance index represents the total energy of the structure. Since u is assumed to have the form indicated in Eq. (2), it is appropriate to choose the control weighting matrix R to have the form

$$R = \begin{bmatrix} R_\alpha & 0 \\ 0 & R_\beta \end{bmatrix} \tag{21}$$

The above centralized design scheme for a linear optimal compensator is well known. Now, a decentralized controller synthesis method, called the *Substructural Controller Synthesis* (SCS) method, will be formulated. The development of the Substructural Controller Synthesis method, which is stimulated by the substructuring decomposition and the Component Mode Synthesis method, is described in detail in Ref. [11]. The plant to be controlled is first decomposed into several substructures by the substructuring decomposition method. Then, for each substructure a subcontroller is designed by using linear quadratic optimal control theory. The collection of all the subcontrollers is considered as the substructuring decomposition of a system controller that is to be employed to control the whole plant. Finally, the same coupling scheme that is employed for the plant is also used to synthesize the subcontrollers into a coupled system controller.

In order to show more clearly how the concept of substructuring decomposition is employed to assemble the subcontrollers, the collection of the two substructures is now considered as a single system, the *unassembled system.* The dynamic equation of the unassembled system can be written in a compact form

$$\begin{aligned} \tilde{S}\dot{\tilde{z}} &= \tilde{A}\tilde{z} + \tilde{B}u + \tilde{N}\varpi \\ y &= \tilde{C}\tilde{z} + v \end{aligned} \tag{22}$$

with

$$\tilde{S} = \begin{bmatrix} S_\alpha & 0 \\ 0 & S_\beta \end{bmatrix}, \quad \tilde{A} = \begin{bmatrix} A_\alpha & 0 \\ 0 & A_\beta \end{bmatrix}, \quad \tilde{B} = \begin{bmatrix} B_\alpha & 0 \\ 0 & B_\beta \end{bmatrix},$$
$$\tilde{N} = \begin{bmatrix} N_\alpha & 0 \\ 0 & N_\beta \end{bmatrix}, \quad \tilde{C} = \begin{bmatrix} C_\alpha & 0 \\ 0 & C_\beta \end{bmatrix}$$

and

$$\tilde{z} = \left\{ \begin{matrix} z_\alpha \\ z_\beta \end{matrix} \right\}, \quad u \equiv \left\{ \begin{matrix} u_\alpha \\ u_\beta \end{matrix} \right\}, \quad y \equiv \left\{ \begin{matrix} y_\alpha \\ y_\beta \end{matrix} \right\}, \quad \varpi \equiv \left\{ \begin{matrix} \varpi_\alpha \\ \varpi_\beta \end{matrix} \right\}, \quad v \equiv \left\{ \begin{matrix} v_\alpha \\ v_\beta \end{matrix} \right\}$$

The distribution of the input noise is assumed to be substructurally decomposable, i.e., $N = \tilde{T}^T\tilde{N}$, so that system (22) is a substructuring decomposition of system (16). This assumption is not a serious restriction since, in general, distribution and intensity of the noise are uncertain quantities.

The performance index of the unassembled system is simply the summation of the performance indexes of the substructures

$$\tilde{J} = J_\alpha + J_\beta = \lim_{t\to\infty} \frac{1}{2} E[\tilde{z}^T\tilde{Q}\tilde{z} + u^T\tilde{R}u] \tag{23}$$

with

$$\tilde{Q} = \begin{bmatrix} Q_\alpha & 0 \\ 0 & Q_\beta \end{bmatrix}, \quad \tilde{R} = \begin{bmatrix} R_\alpha & 0 \\ 0 & R_\beta \end{bmatrix}$$

The optimal controller for the unassembled system, which is the collection of the two independently designed subcontrollers , can be written in compact form as

$$\begin{aligned} \tilde{S}\dot{\tilde{q}} &= (\tilde{A} + \tilde{B}\tilde{G}^{\circ} - \tilde{F}^{\circ}\tilde{C})\tilde{q} + \tilde{F}^{\circ}y \\ u &= \tilde{G}^{\circ}\tilde{q} \end{aligned} \tag{24}$$

with

$$\tilde{G}^{\circ} = \begin{bmatrix} G_\alpha^{\circ} & 0 \\ 0 & G_\beta^{\circ} \end{bmatrix}, \quad \tilde{F}^{\circ} = \begin{bmatrix} F_\alpha^{\circ} & 0 \\ 0 & F_\beta^{\circ} \end{bmatrix} \tag{25}$$

The last step is to assemble the subcontrollers by using the same coupling scheme as used for assembling the substructures. The assembled controller for the assembled system is represented by

$$\begin{aligned} S\dot{q} &= (A + BG^{\oplus} - F^{\oplus}C)q + F^{\oplus}y \\ u &= G^{\oplus}q \end{aligned} \tag{26}$$

with

$$F^{\oplus} = \tilde{T}^T \tilde{F}^{O}, \qquad G^{\oplus} = \tilde{G}^{O} \tilde{T} \tag{27}$$

where superscript $\oplus$ denotes that the controller is not optimal but is considered as suboptimal. The control design matrices $F^{\oplus}$ and $G^{\oplus}$ for the assembled structure are obtained by assembling the optimal control design matrices F_i^O and G_i^O for the substructures by using the coupling matrix $\tilde{T}$. If the assembled controller is employed to control the assembled structure, Eq. (16), the following closed-loop equation is obtained

$$\begin{bmatrix} S & 0 \\ 0 & S \end{bmatrix} \begin{Bmatrix} \dot{z} \\ \dot{q} \end{Bmatrix} = \begin{bmatrix} A & BG^{\oplus} \\ F^{\oplus}C & A + BG^{\oplus} - F^{\oplus}C \end{bmatrix} \begin{Bmatrix} z \\ q \end{Bmatrix} + \begin{bmatrix} N & 0 \\ 0 & F^{\oplus} \end{bmatrix} \begin{Bmatrix} \varpi \\ v \end{Bmatrix} \tag{28}$$

Closed-loop stability of a Substructural Controller Synthesis design is, in general, not guaranteed. This is the same disadvantage that most indirect control design methods have. Indirect control design means that the controller is not designed based upon the exact full-order structure but is based on an approximate model or reduced-order model. From the form of Eq. (28), it is seen that the separation principle is applicable to the SCS control system. The closed-loop poles of the assembled system are the union of the regulator poles (eigenvalues of $S^{-1}(A + BG^{\oplus})$) and the observer poles (eigenvalues of $S^{-1}(A - F^{\oplus}C)$). Therefore, stability of the assembled closed-loop system can be checked by examining the locations of these two sets of eigenvalues.

One advantage of using Substructural Controller Synthesis to design a controller is that an SCS controller is highly adaptable. For a structure with varying configuration or varying mass and stiffness properties, like some space structures, the Substructural Controller Synthesis method may be especially efficient. The SCS controller can be updated economically by simply carrying out redesign of subcontrollers associated with those substructures that have changed. On the other hand, for a controller based on a centralized design scheme, a slight change of the structure may require a full-scale redesign. This favorable decentralized feature of the Substructural Controller Synthesis method is similar to that of the Component Mode Synthesis method in the application to model modification.

5. Example

A plane truss example is used to demonstrate the applicability of the Substructural Controller Synthesis method. The example consists of two identical substructures with almost co-located sensor and actuator allocations. The truss structure, which is shown in Figure 3, consists of six bays and has twenty degrees-of-freedom. Two force actuators and two displacement sensors are allocated symmetrically at f and d, respectively. The actuators are contaminated by disturbances with intensity 10^{-3}. The sensors are contaminated by noises with intensity 10^{-12}. These levels of noise intensities are chosen arbitrarily just for the purpose of example study, and are not justified by the experience of any real case. (In Ref. [13], there is an example with input noise intensity 10^{-4} and output noise intensity 10^{-15}.) All disturbances are assumed to be uncorrelated zero-mean white noise processes. The mass and stiffness matrices for the structure are obtained by the finite element method. The damping matrix is chosen to be 1/1000 of the stiffness matrix. The eigenvalues of the open-loop system have damping ratios ranging from 0.05% to 1.5%. The structure is divided into two substructures as shown in Figure 3.

In order to illustrate in some sense the "adaptable" feature of the method, SCS control design has been carried out and compared with the full-order optimal controller for three different cases. Conditions, assumptions, formulations, and results for the three cases studied are summarized in the following.

Case 1: (Two-input and two-output)

For this case, the two substructures are identical due to symmetry. Therefore, only one substructural level control design need be carried out. The other subcontroller can be obtained by using symmetry. The results are shown in Table 1 and Figure 4, in which R is the weighting of control cost in the performance index. It is seen that the SCS controller has a near-optimal performance. The performance value of the SCS controller is less than 4% higher than the performance value of the optimal controller. The substructures and subcontrollers for this case are symbolically represented by the following equations.

Left substructure

$$S_1\dot{z}_1 = A_1 z_1 + B_1 u_1 + B_1 \varpi_1$$
$$y_1 = C_1 z_1 + v_1$$

Left subcontroller

$$S_1\dot{q}_1 = (A_1 + B_1 G_1^{o} - F_1^{o} C_1) q_1 + F_1^{o} y_1$$
$$u_1 = G_1^{o} q_1$$

Right substructure

$$S_2\dot{z}_2 = A_2 z_2 + B_2 u_2 + B_2 \varpi_2$$
$$y_2 = C_2 z_2 + v_2$$

Right subcontroller

$$S_2\dot{q}_2 = (A_2 + B_2 G_2^{o} - F_2^{o} C_2) q_2 + F_2^{o} y_2$$
$$u_2 = G_2^{o} q_2$$

Case 2: (Two-input and single-output)

Assume that the right sensor has malfunctioned. In this case, the right substructure is not observable. The generalized subcontroller for the right substructure is defined to be a full-state feedback controller, although there is really no state estimator available. Comparisons of the SCS controller and the full-order optimal controller are summarized by Table 2 and Figure 5. It is seen that the performance of the SCS controller for this case is not as good as that for the previous case. The substructures and subcontrollers for this case are symbolically represented by the following equations.

Left substructure

$$S_1\dot{z}_1 = A_1 z_1 + B_1 u_1 + B_1 \varpi_1$$
$$y_1 = C_1 z_1 + v_1$$

Left subcontroller

$$S_1\dot{q}_1 = (A_1 + B_1 G_1^{o} - F_1^{o} C_1) q_1 + F_1^{o} y_1$$
$$u_1 = G_1^{o} q_1$$

Right substructure

$$S_2\dot{z}_2 = A_2 z_2 + B_2 u_2 + B_2 \varpi_2$$

Right generalized subcontroller

$$S_2\dot{q}_2 = (A_2 + B_2 G_2^{o}) q_2$$
$$u_2 = G_2^{o} q_2$$

Case 3: (Two-input and single-output; right substructure noise-free)

We suspect that the poor performance of the SCS controller in Case 2 is due to the fact that there is not an observer to filter the noise on the right substructure. Therefore, as another case for study, we consider the same actuator/sensor configuration as that of Case 2, but assume that the right substructure is free of disturbance. The results are summarized by Table 3 and Figure 6. The SCS controller for this case has a near-optimal performance. The substructures and subcontrollers for this case are symbolically represented by the following equation.

Left substructure

$$S_1\dot{z}_1 = A_1 z_1 + B_1 u_1 + B_1 \varpi_1$$
$$y_1 = C_1 z_1 + v_1$$

Right substructure

$$S_2\dot{z}_2 = A_2 z_2 + B_2 u_2$$

Left subcontroller

$$S_1\dot{q}_1 = (A_1 + B_1 G_1^{\mathrm{o}} - F_1^{\mathrm{o}} C_1) q_1 + F_1^{\mathrm{o}} y_1$$
$$u_1 = G_1^{\mathrm{o}} q_1$$

Right generalized subcontroller

$$S_2\dot{q}_2 = (A_2 + B_2 G_2^{\mathrm{o}}) q_2$$
$$u_2 = G_2^{\mathrm{o}} q_2$$

From the results of the above three cases, it is seen that, for this example, the performance of the SCS controller is, in general, near-optimal. The only situation where the SCS controller exhibited a poor performance is Case 2, in which the right substructure is subject to disturbance but has no output measurement as a feedback to filter the noise. Additional cases are presented in Ref. [11].

6. Conclusions

A decentralized linear quadratic control design method called Substructural Controller Synthesis is proposed for the control design of flexible structures. The SCS method presented in this paper is only a preliminary research result. It is not a fully-developed method, but rather a proposed controller design technique which requires further research. Although some numerical examples have shown promising results, theoretical aspects of the SCS method still need to be pursued in greater depth and other examples need to be considered. The example illustrated does not involve model reduction and controller reduction. However, the method is ready to be incorporated with component mode synthesis and controller reduction methods.

References

1. Craig, R. R. Jr., *Structural Dynamics - An Introduction to Computer Methods*, John Wiley & Sons, Inc. NY, 1981.
2. Craig, R. R. Jr., "A Review of Time-Domain and Frequency-Domain Component Modes Synthesis Methods," *Combined Experimental/Analytical Modeling of Dynamic Structural Systems*, AMD-Vol. 67, ASME, NY, pp. 1-30, 1985. Also *International Journal of Analytical and Experimental Modal Analysis*, Vol. 67, pp. 1-30.
3. Hurty, W. C., "Dynamic Analysis of Structural Systems Using Component Modes," *AIAA J.*, Vol. 3, pp. 678-685, 1965.
4. Craig, R. R. Jr. and Bampton, M. C. C., "Coupling of Substructures for Dynamic Analysis," *AIAA J.*, Vol. 6, pp. 1313-1316, 1968.
5. MacNeal, R. H., "A Hybrid Method of Component Mode Synthesis," *Comp. & Struct.*, Vol. 1, pp. 581-601, 1971.
6. Benfield, W. A. and Hruda, R. F., "Vibration Analysis of Structures by Component Mode Substitution," *AIAA J.*, Vol. 9, pp. 1255-1261, 1971.
7. Young, K. D., "A Distributed Finite Element Modeling and Control Approach for Large Flexible Structures," to appear *J. Guidance, Control, and Dynamics.*
8. Ikeda, M. and Šiljak, D. D., "Overlapping Decompositions, Expansions, and Contractions of Dynamic Systems," *Large Scale Systems*, Vol. 1, pp. 29-38, 1980.

9. Ikeda, M., Šiljak, D. D., and White, D. E., "An Inclusion Principle for Dynamic Systems," *IEEE Trans. Automat. Control*, Vol. 29, No. 3, pp. 244-249, 1984.

10. Yousuff, A., "Application of Inclusion Principle to Mechanical Systems," *Proceedings of 1988 American Control Conference*, Atlanta, GA, June 15-17, pp. 1516-1520, 1988.

11. Su, T. J., "A Decentralized Linear Quadratic Control Design Method For Flexible Structures," Ph.D. dissertation, The University of Texas at Austin, Austin, Texas, August 1989.

12. Su, T. J. and Craig, R. R. Jr., "Controller Reduction By Preserving Impulse Response Energy," *AIAA Guidance, Navigation and Control Conference,* Boston, MA, pp. 55-64, August 14-17, 1989.

13. Yousuff, A. and Skelton, R. E., "Controller Reduction by Component Cost Analysis," *IEEE Trans. Automat. Control*, Vol. 29, No. 6, pp. 520-530, 1984.

Table 1: Performance values of Case 1

R=	0.01	0.05	0.1	0.5	1	5	10
Optimal	1.1737E-4	1.7796E-4	2.1445E-4	3.3929E-4	4.1689E-4	6.7621E-4	8.3436E-4
SCS method	1.2155E-4	1.8168E-4	2.1856E-4	3.4522E-4	4.2385E-4	6.8522E-4	8.4451E-4
Difference	3.6%	2.1%	1.9%	1.7%	1.7%	1.3%	1.2%

Table 2: Performance values of Case 2

R=	0.01	0.05	0.1	0.5	1	5	10
Optimal	1.3742E-4	1.9240E-4	2.2709E-4	3.4887E-4	4.2544E-4	6.8283E-4	8.4029E-4
SCS method	5.3709E-4	6.6359E-4	7.0535E-4	7.9789E-4	8.4867E-4	1.0293E-3	1.1520E-3
Difference	291%	245%	210%	129%	99%	51%	37%

Table 3: Performance values of Case 3

R=	0.01	0.05	0.1	0.5	1	5	10
Optimal	5.9433E-5	8.9437E-5	1.0763E-4	1.6989E-4	2.0863E-4	3.3822E-4	4.1726E-4
SCS method	6.1968E-5	9.1607E-5	1.0989E-4	1.7296E-4	2.1219E-4	3.4275E-3	1.1520E-3
Difference	4.3%	2.4%	2.1%	1.8%	1.7%	1.3%	1.2%

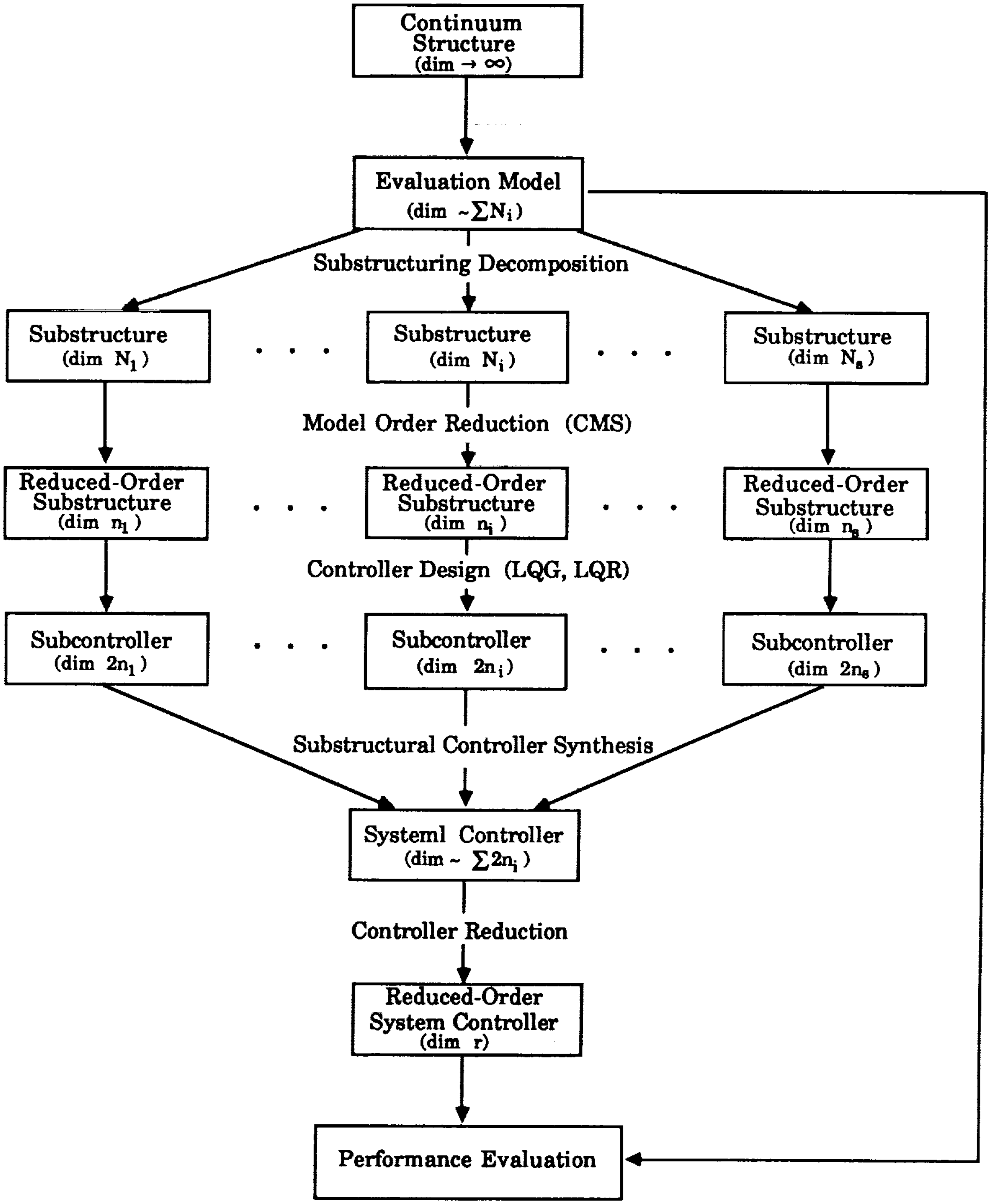

Continuum Structure (dim → ∞)
Evaluation Model (dim ~ΣNi)
Substructuring Decomposition
Substructure (dim N1)
. . .
Substructure (dim Ni)
. . .
Substructure (dim Ns)
Model Order Reduction (CMS)
Reduced-Order Substructure (dim n1)
Reduced-Order Substructure (dim ni)
Reduced-Order Substructure (dim ns)
Controller Design (LQG, LQR)
Subcontroller (dim 2n1)
Subcontroller (dim 2ni)
Subcontroller (dim 2ns)
Substructural Controller Synthesis
Systeml Controller (dim ~ Σ2ni)
Controller Reduction
Reduced-Order System Controller (dim r)
Performance Evaluation

Figure 1: Steps in Substructural Controller Synthesis method

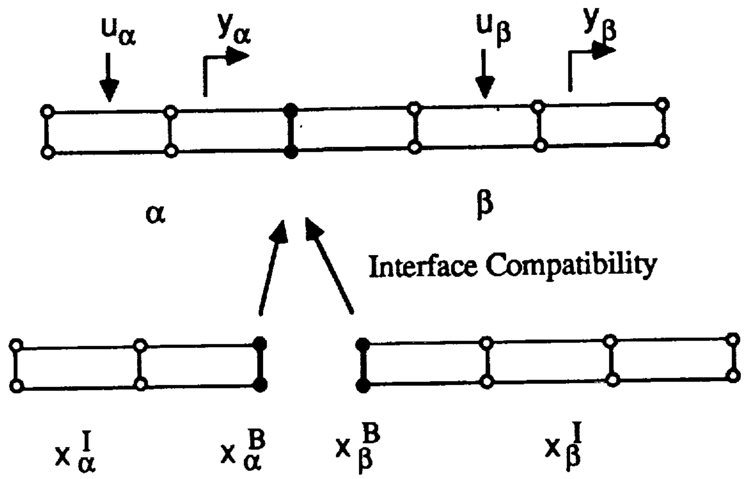


Figure 2: Two-component structure

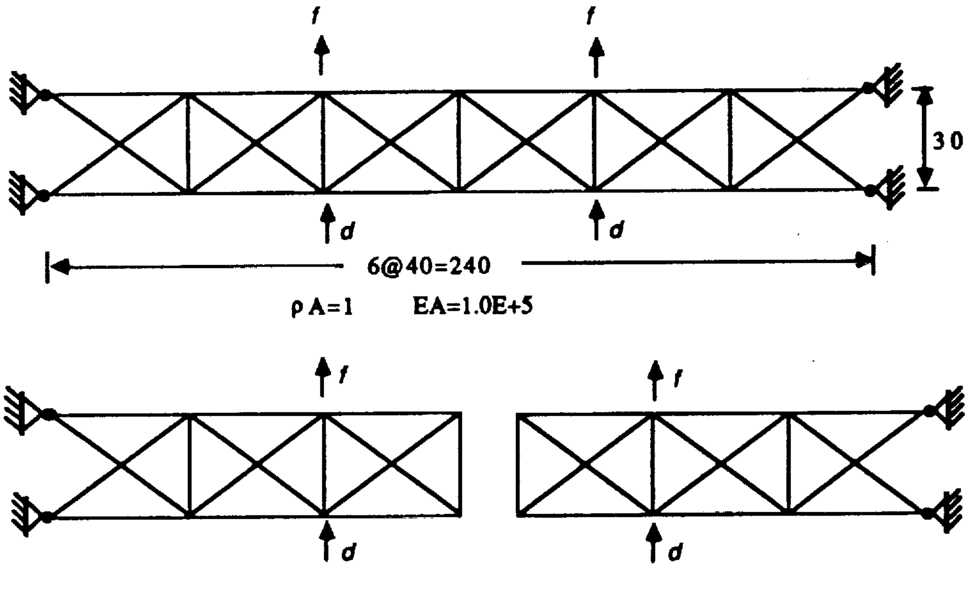


Figure 3: Details of the plane truss for the SCS design example

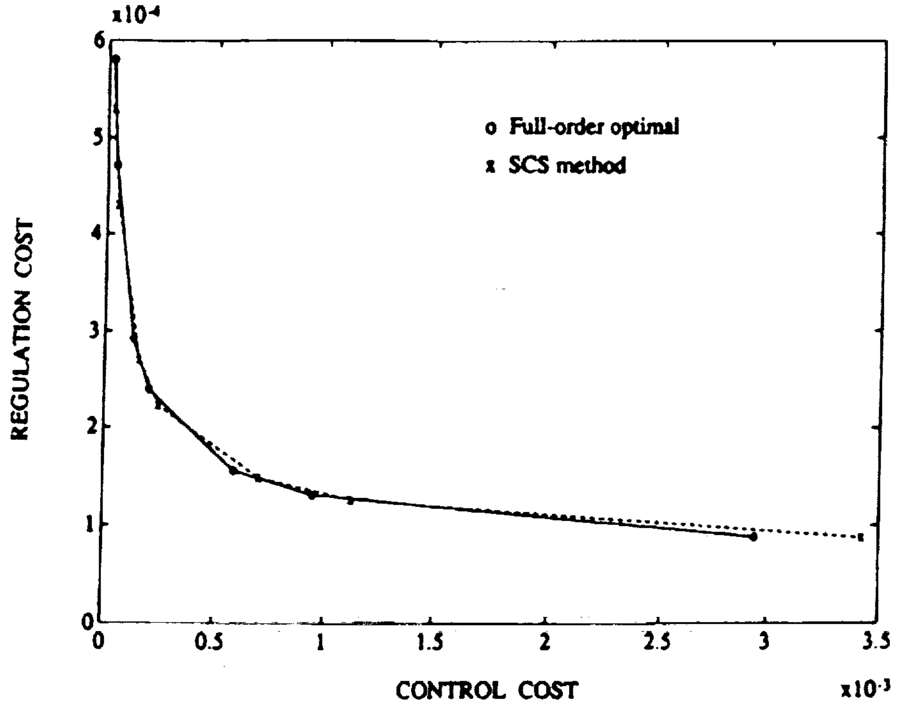


Figure 4: Performance plot of Case 1

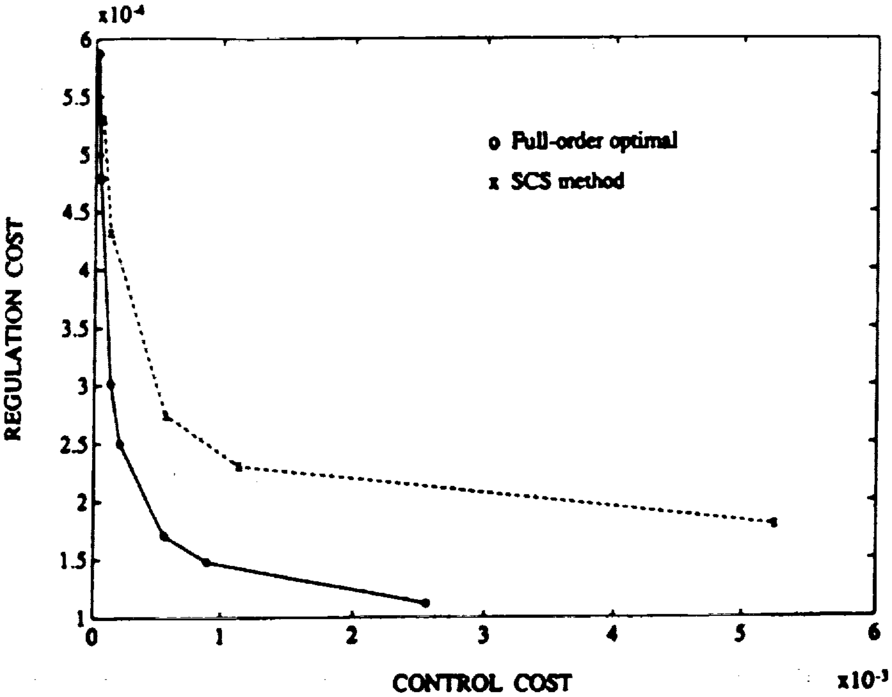


Figure 5: Performance plot of Case 2

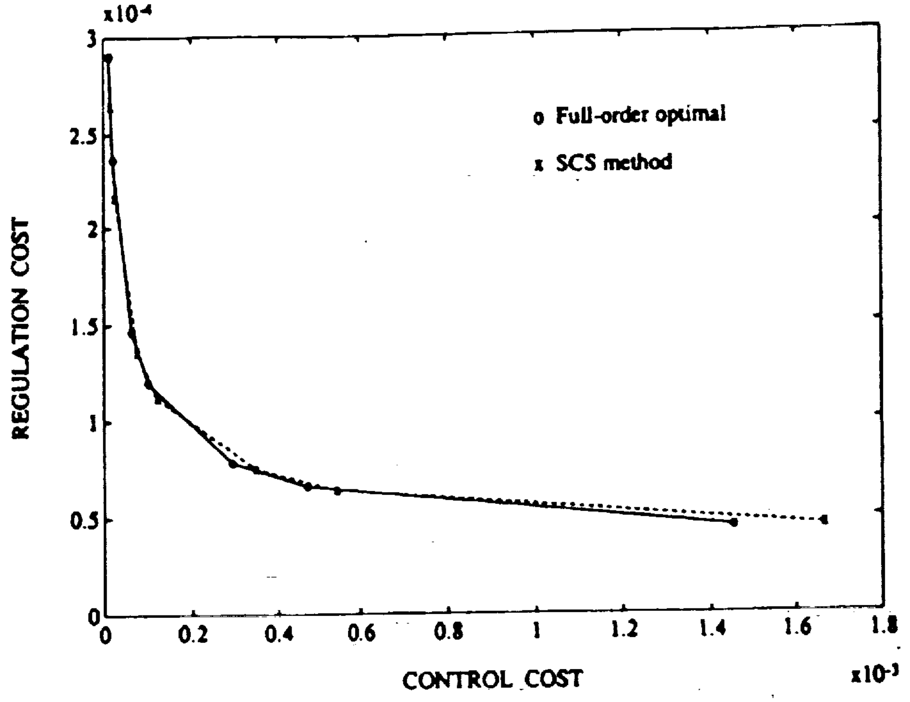


Figure 6: Performance plot of Case 3

N90-23056

Extensions of Output Variance Constrained Controllers to Hard Constraints

R. Skelton, G. Zhu
Purdue University

Abstract

Covariance Controllers assign specified matrix values to the state covariance. a number of robustness results are directly related to the covariance matrix. This paper illustrates with examples the conservatism in known upperbounds on the H∞, L∞, and L_2 norms for stability and disturbance robustness of linear uncertain systems using covariance controllers. These results are illustrated for continuous and discrete time systems.

If

$$\dot{x}_p = A_p x_p + D_p w + B_p u, \qquad \dot{x}_c = A_c x_c + Fz$$

$$z = M_p x_p,\ y = C_p x_p \qquad u = Gx_c + H_z$$

describes a stable controllable linear system then the L∞ bound

$$\frac{\hat{}y\hat{}_\infty}{\hat{}w\hat{}_2} \le \bar{s}(CXC^*) \quad , \qquad X \triangleq E\,[xx^*],\ x^* = (x_p^* x_c^*)$$

$$C = \{C_p O]$$

holds for all L_2 disturbances w(t). If ΔA is the perturbation in A then A + ΔA remains stable for all ΔA subject to

$$\hat{}\Delta A\hat{} \le \frac{\underline{s}(DWD^*)^{1/2}}{\bar{s}(X(DWD^*)^{-1/2})}$$

The three assignability conditions for the closed loop to have the given covariance X are added to the robustness goals to complete the constraints on X to allow a specified degree of disturbance rejection

$$\frac{\hat{}y\hat{}_\infty}{\hat{}w\hat{}_2} \le e_1$$

and parameter tolerance

$$(\hat{}\Delta A\hat{}) \le e_2$$

for specified e_1, e_2.

Minimal Complexity Control Law Synthesis

Dennis S. Bernstein
Harris Corporation
Melbourne, FL 32902
Wassim M. Haddad
Florida Institute of Technology
Melbourne, FL 32902
Carl N. Nett
GE Corporate Research and Development
Schenectady, NY

Abstract

In light of i) the increasingly complex nature of systems requiring controls and ii) the increasingly stringent accuracy required of control systems, the predominate considerations in control law design for modern engineering systems have become control law complexity and control law robustness, respectively. Indeed, with i) comes increasing and usually overriding concern with system cost, reliability, and maintainability, and with ii) comes increasingly complex control systems. Since, generally speaking, the more complex the control system, the more it costs, the less reliable it is, and the harder it is to maintain, it follows that i) and ii) conflict with each other through the specification of control system complexity. Similarly, with i) comes increasing levels of system/environmental uncertainty, and with ii) comes control systems which are increasingly robust relative to a fixed level of system/environmental uncertainty. Since the maximal achievable level of robustness decreases as the level of system/environmental uncertainty increases, it follows that i) and ii) are also in conflict with each other through the specification of control system robustness. correspondingly, control law complexity and control law robustness are, respectively, the predominant considerations in control law design for modern engineering systems.

In light of the above discussion, it seems both natural and appropriate to postulate the following paradigm for control law design for modern engineering systems. Minimize control law complexity subject to the achievement of a specified accuracy in the face of a specified level of uncertainty. correspondingly, the overall goal in this paper is to make progress towards the development of a control law design methodology which supports this paradigm. We achieve this goal by developing a general theory of optimal constrained-structure dynamic output feedback compensation, where here constrained-structure means that the dynamic-structure (e.g., dynamic order, pole locations, zero locations, etc.) of the output feedback compensation is constrained in some way. By applying

this theory in an innovative fashion, where here the indicated iteration occurs over the choice of the compensator dynamic-structure, the paradigm stated above can, in principle, be realized.

In this paper the optimal constrained-structure dynamic output feedback problem is formulated in general terms. an elegant method for reducing optimal constrained-structure dynamic output feedback problems to optimal static output feedback problems is then developed. This reduction procedure makes use of star products, linear fractional transformations, and linear fractional decompositions, and yields as a by-product a complete characterization of the class of optimal constrained-structure dynamic output feedback problems which can be reduced to optimal static output feedback problems. Issues such as operational/physical constraints, operating-point variations, and processor throughput/memory limitations are considered, and it is shown how anti-windup/bumpless transfer, gain-scheduling, and digital processor implementation can be facilitated by constraining the controller dynamic-structure in an appropriate fashion.

There are two principal contributions of this paper. First, many results on both full- and reduced-order optimal dynamic output feedback compensation obtained by other authors are readily shown to be but special cases of our results on optimal static output feedback compensation. As such, a significant unification of many known results in optimal control theory is achieved. Second, the general theory of optimal constrained-structure dynamic output feedback compensation provides a theoretical basis for the analytical design of optimal "industry standard" controllers, such as proportional-integral (PI) controllers and lead-lag compensators. consequently, the results presented in this series of papers will do much to help bridge the gap that currently exists between control theory and control practice.

OPTICON: Pro-Matlab Software for Large Order Controlled Structure Design

Lee D. Peterson
Sandia National Laboratories, Albuquerque, NM 87185

Abstract

A software package for large order controlled structure design is described and demonstrated in this paper. The primary program, called OPTICON, uses both Pro-Matlab M-file routines and selected compiled Fortran routines linked into the Pro-Matlab structure. The program accepts structural model information in the form of state-space matrices and performs three basic design functions on the model: 1) open loop analyses, 2) closed loop reduced order controller synthesis, and 3) closed loop stability and performance assessment. The current controller synthesis methods which have been implemented in this software are based on the Generalized LQG theory of Bernstein. In particular, a reduced order Optimal Projection synthesis algorithm based on a homotopy solution method has been successfully applied to an experimental truss structure using a 58-state dynamic model. These results will be presented and discussed. The paper will also discuss current plans to expand the practical size of the design model to several hundred states and the intention to interface Pro-Matlab to a supercomputing environment.

N90-23059

ROBUST FIXED ORDER DYNAMIC COMPENSATION FOR LARGE SPACE STRUCTURE CONTROL

Anthony J. Calise and Edward V. Byrns, Jr.
Georgia Institute of Technology
School of Aerospace Engineering
Atlanta, GA 30332

ABSTRACT

This paper presents a simple formulation for designing fixed order dynamic compensators which are robust to both uncertainty at the plant input and structured uncertainty in the plant dynamics. The emphasis is on designing low order compensators for systems of high order. The formulation is done in an output feedback setting which exploits an observer canonical form to represent the compensator dynamics. The formulation also precludes the use of direct feedback of the plant output. The main contribution lies in defining a method for penalizing the states of the plant and of the compensator, and for choosing the distribution on initial conditions so that the loop transfer matrix approximates that of a full state design. To improve robustness to parameter uncertainty, the formulation avoids the introduction of sensitivity states, which has led to complex formulations in earlier studies where only structured uncertainty has been considered.

INTRODUCTION

Linear Quadratic Regulator (LQR) design methods are easy to use and have guaranteed stability margins at the plant input. Unfortunately, this requires full state feedback for implementation. With Loop Transfer Recovery (LTR) techniques, the loop characteristics of an LQR design for square, minimum phase plants can be asymptotically realized using output feedback with a full order observer.[1] This design methodology can be used to improve the robustness of observer based controller designs to unstructured uncertainty. However, for control of large order plants, this approach may result in controllers that are computationally expensive to implement. Moreover, although there is good robustness to unstructured uncertainty at the plant input, the design may remain sensitive to structured uncertainty in the plant parameters.

Optimal output feedback of fixed order dynamic compensators[2] has received limited attention due to numerous difficulties associated with this approach. Initially, the proposed compensator representation was overparameterized, which means it lacked a predefined structure. To

C-3

√

overcome this obstacle, several canonical structures have been proposed which result in a minimal parameterization.[3,4] This study utilizes the observer canonical form since it yields a convenient form when the design is treated as a constant gain output feedback problem.

Another major objection to fixed order dynamic compensation is that there are no guarantees on the stability margins. This drawback was eliminated by a new methodology for designing fixed order compensators.[5] This technique approximates the LQR/LTR method, by appropriate selection of the plant and compensator state weighting matrices. Much like the full order observer design, a two step process is used. First, full state gains are computed for desirable loop properties, followed by the approximate LTR compensator design. The fundamental assumption of this approach is that if the closed loop return signals of the two designs are equal, then the loop transfer functions (with the loop broken at the plant input) must be equivalent. Unlike the LQR/LTR design, there is no requirement that the system be minimum phase or square.

A popular method of parameter sensitivity reduction consists of including a quadratic trajectory sensitivity term in the linear regulator formulation. In Ref. 6, the approximate LTR methodology for low order compensators was extended to include sensitivity reduction for real plant parameter variations. The sensitivity reduction is accomplished by a simple modification of the quadratic performance index. Unlike earlier studies in this area,[7,8,9] this formulation does not require increasing the dimension of the problem to include the sensitivity states. In addition, the parameter sensitivity reduction is achieved with minimal sacrifice in the loop transfer characteristics.

An outline of this paper is as follows. First, the approximate LTR methodology of Ref. 5 is reviewed. Then, the sensitivity reduction formulation is presented for the specific case of a scalar uncertainty in the state equation. The generalizations of the trajectory sensitivity approach, given in Ref. 6, are summarized afterwards. Several future extensions of this research are also discussed.

CONTROLLER DESIGN FORMULATIONS

Dynamic Compensation

Consider a linear system of the form

$$\dot{x}_p = A_p x_p + B_p u_p \qquad x_p \varepsilon \Re^n,\ u_p \varepsilon \Re^m \qquad (1)$$

$$y_p = C_p x_p + D_p u_p \qquad y_p \varepsilon \Re^p \qquad (2)$$

where y_p represents the measured outputs. A dynamic compensator can be parameterized in the following observer canonical form:[4]

$$\dot{z} = P^o z + u \qquad z_o \,\varepsilon\, \Re^{nc} \tag{3}$$

$$u = P_z u_p - N y_p \qquad u_o \,\varepsilon\, \Re^{nc} \tag{4}$$

$$u_p = - H^o z \tag{5}$$

where nc is the order of the compensator and P_z and N are the matrices containing the design parameters. The matrices P^o and H^o are specified by the choice of observability indices, and their structures are given in Ref. 4.

An equivalent, constant gain output feedback problem is obtained using the augmented system dynamics.

$$\dot{x} = Ax + Bu_c \tag{6}$$

$$y = Cx \tag{7}$$

$$u_c = -Gy \tag{8}$$

where $x^T = \{x_p{}^T, z^T\}$, $y^T = \{y_p{}^T, -u_p{}^T\}$ and

$$A = \begin{bmatrix} A_p & -B_p H^o \\ 0 & P^o \end{bmatrix} \qquad B = \begin{bmatrix} 0 \\ I_{nc} \end{bmatrix} \tag{9}$$

$$C = \begin{bmatrix} C_p & -D_p H^o \\ 0 & H^o \end{bmatrix} \qquad G = [N \quad P_z] \tag{10}$$

The performance index in this case is

$$J = E_{x_o}\left\{\int_0^\infty \left[x^T Q x + u_c^T R u_c\right] dt\right\} \tag{11}$$

It is shown in Ref. 5 that the loop transfer properties of a full state feedback design are approximately recovered at the plant input by choosing the following weighting matrices in (11).

$$Q = \begin{bmatrix} K^{*T}K^{*} & -K^{*T}H^{o} \\ -H^{oT}K^{*} & H^{oT}H^{o} \end{bmatrix} \qquad R = \rho B^{T}B = \rho I_{nc} \qquad (12)$$

$$E\{x_o x_o^T\} = \begin{bmatrix} B_p B_p^T & 0 \\ 0 & 0 \end{bmatrix} \qquad (13)$$

where K* is the full state feedback gain matrix, and repeating the design for decreasing values of ρ. Although the full state loop transfer properties are approximately recovered as ρ is reduced, the controller design may be sensitive to parameter uncertainty.

Uncertain State Equation

We first consider the system (1,2) with a scalar uncertainty parameter α in the state equation.

$$\dot{x}_p = A_p(\alpha)x_p + B_p(\alpha)u_p \qquad (14)$$

In the case of dynamic compensation, (5) becomes

$$\dot{x} = A(\alpha)x + Bu_c \qquad (15)$$

where

$$A(\alpha) = \begin{bmatrix} A_p(\alpha) & -B_p(\alpha)H^{o} \\ 0 & P^{o} \end{bmatrix} \qquad (16)$$

The trajectory sensitivity dynamics are obtained by differentiating (13) with respect to α.

$$\dot{\sigma} = A_\alpha x + (\overline{A} - BGC)\sigma; \quad \sigma(0) = 0 \qquad (17)$$

where $\sigma = \partial x/\partial\alpha|_{\alpha_0}$, $A_\alpha = \partial A(\alpha)/\partial\alpha|_{\alpha_0}$, $\bar{A} = A(\alpha_0)$ and α_0 denotes the nominal value for α. The standard approach would consider minimizing the following performance index.

$$J = E_{x_0}\left\{\int_0^\infty \left[x^TQx + u_c^TRu_c + \sigma^TS\sigma\right] dt\right\} \tag{18}$$

where the weighting matrix S is used to penalize sensitivity to parameter uncertainty. However, this increases the dimension of the problem to 2(n+nc). To avoid this drawback, we adopt the following viewpoint. Note that $A_\alpha x$ acts as a forcing function in (15), and that $\sigma(0) = 0$. Also, note that since G stabilized the nominal system, the the dynamics of $\sigma(t)$ are also stable. Thus, $||\sigma||$ can be reduced by penalizing $||A_\alpha x||$. This suggests that the following index of performance be used to introduce a penalty on sensitivity to parameter uncertainty

$$J = E_{x_0}\left\{\int_0^\infty \left[x^T(Q + \eta A_\alpha^T A_\alpha)x + u_c^TRu_c\right] dt\right\} \tag{19}$$

Thus, a second design parameter, η, is introduced which can be used to penalize sensitivity to parameter variations, without increasing the order of the dynamic system. When Q, R and X_0 are chosen in accordance with (12) and (13), then increasing η permits a design trade off between desirable loop transfer properties at the plant input and parameter sensitivity reduction. Equations (7), (8), (15) and (19), with $A(\alpha) = \bar{A}$, constitute a static optimal output feedback problem, whose necessary conditions for optimality are well known.[10]

This approach to parameter sensitivity reduction can be generalized to include an uncertain output equations. With uncertainty in the state and output equation, the trajectory sensitivity dynamics become

$$\dot{\sigma} = (A_\alpha - BGC_\alpha)x + (\bar{A} - BG\bar{C})\sigma; \quad \sigma(0) = 0 \tag{20}$$

where the input matrix B is specified by the observer canonical structure in (9). To minimize $\sigma(t)$, the logical extension is to penalize $||(A_\alpha - BGC_\alpha)x||$ in the performance index. Since this penalty depends explicitly on the gain matrix G, the standard necessary conditions for optimal output feedback no longer apply. In Ref. 6, these new necessary conditions are given as well as a generalization of this methodology to include a vector parameters. Also, the design approach is illustrated by a high bandwidth controller for a flexible satellite.

EXTENSIONS

There are several directions in which the current research on low order compensation is heading. First, the approximate LTR methodology used at the plant input has been extended to recovery of loop properties at the plant output. This design technique uses both the controller and the observer canonical forms and the duality which exists between these structures. Similar to the dual LTR formulation of Ref. 1, a full order observer is first designed for desirable loop properties at the dual system input. With the appropriate quadratic state and control penalties, the compensator design approximately recovers these loop characteristics at the plant output. This formulation can easily be extended to include a structured uncertainty penalty similar to the trajectory sensitivity dynamics used in this paper.

The next extension of this work is to develop a design methodology which will allow simultaneous approximate LTR at both the plant input and the plant output. This will allow the extension to H-Infinity design. Specifically, using the design approach in Ref. 11, the H-Infinity controller becomes a constant gain full state feedback design. This can be viewed as a design approach that simultaneously achieves loop shaping with the loop broken at both the plant input and the plant output. Thus, the loop characteristics of a H-Infinity design can be realized with a low order dynamic compensator if simultaneous LTR can be achieved.

SUMMARY

A method has been developed for designing fixed order dynamic compensators which are robust to both unstructured uncertainty at the plant input and structured uncertainty in the plant dynamics. The design approach specifies weighting matrices which allow the loop transfer properties to approximate that of a full state design. The robustness to real structured parameter variations is accomplished by a modification of the quadratic performance index. The approach avoids the introduction of the sensitivity states. Hence, it does not increase the dimension of the problem to acheive the robustness to real plant parameter variations. Extension to H-Infinity design are currently under development.

REFERENCES

1. Doyle, J. C. and Stein, G., "Multivariable Feedback Design: Concepts for a Classical/Modern Synthesis," *IEEE Trans. on Automatic Control*, Vol. AC-26, Feb. 1981, pp. 4-16.

2. Johnson, T. L. and Athans, M., "On the Design of Optimal Constrained Dynamic Compensators for Linear Constant Systems," *IEEE Trans. on Automatic Control*, Vol. AC-15, Dec. 1970, pp. 37-41.

3. Martin, G. D. and Bryson, A. E., "Attitude control of a Flexible Spacecraft," *Journal of Guidance and Control*, Vol. 3, Jan.-Feb. 1980, pp. 37-41.

4. Kramer, F. S. and Calise, A. J., "Fixed Order Dynamic Compensation for Multivariable Linear Systems," *Journal of Guidance, Control and Dynamics*, Vol. 11, Jan.-Feb. 1988, pp. 80-85.

5. Calise, A. J. and Prasad, J. V. R., "An Approximate Loop Transfer Recovery Method for Designing Fixed Order Compensators," AIAA Guidance, Navigation and Control Conf., Paper No. 88-4078, Minneapolis, MN, Aug. 1988. (To appear in AIAA J. of Guidance, Control and Dynamics).

6. Calise, A. J. and Byrns, E. V. Jr., "Robust Fixed Order Dynamic Compensation," AIAA Guidance, Navigation and Control Conf., Paper No. 88-4078, Boston, MA, Aug. 1989.

7. Kriendler, E., "On the Definition and Application of the Sensitivity Function," *Journal of the Franklin Institute*, Vol. 285, pp. 26-36, 1968.

8. O'Reilly, J., "Low Sensitivity Feedback Controllers for Linear Systems with Incomplete State Information," *International Journal of Control*, Vol. 29, No. 6, 1979, pp. 1042-1058.

9. Okada, K. and Skelton, R., "A Sensitivity Controller for Uncertain Systems," AIAA Guidance, Navigation and Control Conf., Paper No. 88-4077, Minneapolis, MN, Aug. 1988.

10. Mendel, J. M., "A Concise Derivation of Optimal Constant Limited State Feedback Gains," *IEEE Trans. on Automatic Control*, Vol. AC-19, Aug. 1974, pp. 447-448.

11. Petersen, I. R., "Disturbance Attenuation and H-Infinity Optimization: A Design Method Based on the Algebraic Riccati Equation," *IEEE Trans. on Automatic Control*, Vol. AC-32, May 1987, pp. 427-429.

N90-23060

Multivariable Frequency Weighted Model Order Reduction For Control Synthesis

David K. Schmidt
Department of Mechanical and Aerospace Engineering
Arizona State University

This paper was also presented as AIAA paper 89-3558.

Abstract

Quantitative criteria are presented for model simplification, or order reduction, such that the reduced order model may be used to synthesize and evaluate a control law, and the stability and stability robustness obtained using the reduced-order model will be preserved when controlling the full-order system. The error introduced due to model simplification is treated as modeling uncertainty, and some of the results from multivariable robustness theory are brought to bear on the model simplification problem. A numerical procedure developed previously is shown to lead to results that meet the necessary criteria. The procedure is applied to reduce the model of a flexible aircraft. Also, the importance of the control law itself, in meeting the modeling criteria, is underscored. An example is included that demonstrates that an apparently robust control law actually amplifies modest modeling errors in the critical frequency region, and leads to undesirable results. The cause of this problem is identified to be associated with the canceling of lightly-damped transmission zeroes in the plant.

Whether the engineer is developing a system model for dynamic analysis, control law synthesis, or simulation, a simple low-order model with the requisite validity is desirable for a variety of practice reasons. The question arises, therefore, as to how to obtain such a simple yet valid model. Even more fundamental is the question of what model characteristic are important such that one may strive to retain them. Although the initial question has been addressed for some time, from the attention still paid to model and controller order reduction (c.f. Refs. 1,2), it appears that the issues still remain unresolved.

In Refs 3-6, some previous offerings on the subject are presented. In this paper, discussion will continue, in the attempt to expand on some of the earlier results, to further clarify the theoretical basis behind the proposed methodology, and to reveal some important aspects of not only model-simplification, but also control-law synthesis for elastic vehicles.

1. Criteria for Modeling

The objective in model simplification, as with all system modeling, is to develop a fundamental understanding of the system in question. For the model to be useful, it should predict to the required engineering accuracy the behavior of the actual system. Note that it does not have to predict with perfect accuracy, and that is not possible anyway. The required accuracy depends on the application for which the model is intended.

In this paper, as in Refs. 3 - 6, the intended application of the model is to predict the behavior of the system when it is subject to feedback action, as shown, for example, in Fig. 1. Clearly, then, the critical characteristics of the actual system that must be adequately captured by the model are those characteristics important in a feedback system. (Note that the feedback action could represent an automatic control system, as well as that of a human, or manual controller.) Finally, the existence of a sufficiently valid, although perhaps complex model for the system is assumed to be available - admittedly a big assumption. Further, if this model is infinite-dimensional and/or non-linear, it is assumed that a locally linearized, finite-dimensional model may be obtained. The original (complex) model will be denoted as $\underline{G}$, while the linear model will be denoted as G.

As a result of any simplification process, differences between the more-accurate model and the simple model arise. Or conceptually, if G_R is a simpler model for G, the model-simplification error may be considered to be $\Delta G = G - G_R$. These errors are key to the research presented here. In contrast, model-simplification errors arising due to the development of G, or $\Delta\underline{G} = \underline{G} - G$, will be considered only indirectly.

The critical question then is what errors ΔG are critical, or should be minimized, and what procedure will do so? The answer to the first part of the question could be that ΔG's critical in a feedback loop should be minimized. Further, if these ΔG's are interpreted more generally as model uncertainty rather that model-reduction error, the recent research on multi-variable robustness theory may be bought to bear on the model-simplification problem. This is the main idea in this research.

2. Robustness and Model Reduction

In this section, some key results from robust control theory will be noted, and they will be interpreted in the context of the model reduction problem.

With reference to the system shown in Fig. 2, G_R is the transfer-function-matrix representation of this simplified model, $\Delta G(s)$ in the analogous representation of the model-simplification error, and the full-order linear model is $G = G_R + \Delta G$. Likewise, K(s) is the matrix of control compensators, perhaps to be designed using G_R. Clearly in this context, one desires that the K(s) so obtained will control the "true" G(s) as predicted through the use of G_R. Attention is now turned to exposing the critical ΔG's via multi-variable Nyquist theory.[7]

Let $\Phi(s)$ be an analytic function of the complex variable s, and let the number of zeros of $\Phi(s)$ in the open right half of the complex plane be denoted as z. Then the Principle of the Argument states that

$$\underset{R\rightarrow\infty}{N} (O, \Phi(s), D_R) = z$$

or the number (N) of clockwise encirclements of the origin made by the image of the contour D_R, under the mapping of $\Phi(s)$, as s travels clockwise around D_R , equals z. Here D_R is the "Nyquist D contour" that encloses the entire right-half of the complex plane. Clearly, with regards to stability, the $\Phi(s)$ of interest is the closed-loop characteristic polynomial of the feedback system, denoted by $\Phi_{CL}(s)$.

Now, as shown in Ref. 8, and elsewhere, and referring to Fig. 1, for example,

$$\begin{aligned}\Phi_{CL}(s) &= \Phi_{OL}(s) \det [I + GK] \\ &= \Phi_{OL}(s) \det [I + KG] \qquad (1)\end{aligned}$$

where $\Phi_{OL}(s)$ is the characteristic polynomial of the open-loop system KG(s) or GK(s). That is, if either the transfer function matrix GK(s) or KG(s) has the state-space realization

$$\begin{aligned}\dot{x} &= A_{GK}x+ B_{GK}e \\ y &= C_{GK}x\end{aligned}$$

then $\Phi_{OL}(s) = \det[sI - A_{GK}]$, and the zeros of $\Phi_{OL}(s)$ are the open-loop poles of the system. Note that Eqn. 1 may therefore be re-written as

$$\Phi_{OL}(s) = \det[sI - A_{GK}] \det [I + C_{GK} [sI - A_{GK}]^{-1} B_{GK}]$$

Now if the number of right-half-plane zeros of $\Phi_{OL}(s)$ is p, then the number of right-half-plane zeros of det [I + C_{GK} [sI - $A_{GK}]^{-1}$ B_{GK}] must be z-p. Furthermore, from the Principle of the Argument

$$\underset{R\rightarrow\infty}{N} (O, \det [I + C_{GK} (sI - A_{GK})^{-1} B_{GK}] ,D_R) = z\text{-}p$$

Consequently, if p is known, z may be deduced from

$$\begin{aligned}z &= p + (z - p) \\ &= p + [\underset{R\rightarrow\infty}{N} (O, \det [I + C_{GK} (sI - A_{GK})^{-1} B_{GK}] ,D_R)]\end{aligned}$$

or closed-loop stability is determined from knowledge of p and the examination of the Nyquist contour for det[I + GK] or det [I + KG]. Therefore, the closed-loop system is stable if and only if the Nyquist contour for det[I + GK](= det[I +KG]) encircles the origin counterclockwise exactly p times.

Of course the determination of z is possible from other means, and the real utility of the above fact is in defining the concept of relative stability, and in identifying factors that are critical to closed-loop system stability. These issues are of special import here.

Consider the model error, or uncertainty, to be ΔG (as in Fig. 2), and assume that K is such that KG_R leads to a stable closed-loop system with good stability margins. (Note this assumption should always be true as it involves a key objective in determining K(s) using G_R to begin with.) Then if (assumption 1) the number of right-half-plane poles of KG (= p) is identical to the number of right-half-plane poles of KG_R (= p_R), K will stabilize G if and only if (assumption 2)

$$\underset{R\to\infty}{N}(O, \det[I + GK], D_R) = \underset{R\to\infty}{N}(O, \det[I + G_RK], D_R)$$

or the number of encirclements of the origin made by the Nyquist contours associated with G and with G_R are identical.

Stability is guaranteed as follows:

Let z = no. of unstable closed-loop poles of the KG loop.
z_R = no. of unstable closed-loop poles of the KG_R loop
p, p_R = defined above

Then to show stability (or z = 0), note that if (assumption 1) $p_R = p$, then

$$z = z_R - (z_R - p_R) + (z - p)$$

By the assumption KG_R leads to a stable system, $z_R = 0$, and from assumption 2, $(z_R - p_R) = (z-p)$. Hence, z = 0.

This now establishes in a meaningful way, qualitative criteria for model simplification, the simplification must at least lead to ΔG's such that assumption 1 and 2 are satisfied. But the criteria goes further. Not only must stability of the KG loop be assured (i.e., z = 0) but the margins "designed" into KG_R should carry over to the closed-loop system associated with KG. Otherwise, the K so designed would not be satisfactory. It is for this reason that any model reduction technique that just assures stability of the full-order closed-loop system may not be good enough!

To satisfy assumption 2, or to assure that the number of encirclements of the origin is unchanged due to ΔG, requires that [9]

$$\det[I + G_RK + \varepsilon\Delta GK] \neq 0 \qquad \forall\ \omega>0,\ \varepsilon \in [0,1] \tag{2}$$

In other words, if as the Nyquist contour for det[I + G_RK] is continually warped to that for det [I + GK] the origin is never intersected, the number of encirclements of the origin cannot change. Furthermore, Eqn. 2 is assured if (c.f., Ref. 9)

$$\bar{\sigma}(\Delta GK) < \underline{\sigma}[I + G_RK] \quad \forall\ \omega>0 \tag{3}$$

Finally, it is known that an alternative to Eqn. 3 is

$$\bar{\sigma}(E_m) < \underline{\sigma}[I + (G_RK)^{-1}] = \underline{\sigma}\{[G_RK(I + G_RK)^{-1}]^{-1}\} \quad \forall\ \omega>0 \tag{4}$$

where $E_m = G_R^{-1}\Delta G$

The above expressions (Eqns. 2 - 4) may be extended by breaking the frequency domain ($0 \leq \omega < \infty$) into the domains ($0 \leq \omega \leq \omega^*$) and ($\omega^* < \omega < \infty$). Note that these domains are non-intersecting. Now it can be argued that Eqn. 2 will be satisfied if

$$\det [\, I + G_R K + \varepsilon \Delta G K] \neq 0 \qquad \begin{array}{c} (0 \leq \omega \leq \omega^*) \\ (0 \leq \varepsilon \leq 1) \end{array} \qquad (5)$$

and

$$\det [\, I + G_R K + \varepsilon \Delta G K] \neq 0 \qquad \begin{array}{c} (\omega^* < \omega < \infty) \\ (0 \leq \varepsilon \leq 1) \end{array} \qquad (6)$$

Further, Eqn. 5 is assured if Eqn. 3 is satisfied for $\omega \leq \omega^*$, while satisfying Eqn. 4 for $\omega > \omega^*$ assures that Eqn. 6 is satisfied. Hence, in such a situation, Eqn. 2 is satisfied.

By Eqns. 3 and 4, quantitative criteria on critical ΔG's are established. Further, the overall strategy for model simplification becomes apparent, and the interaction between model simplification and control law synthesis is underscored. Regarding the later, it should be clear that the allowable ΔG's (those that do not destroy closed-loop stability of the full-order system controlled by K(s)) depend on K itself. In other words, designing a "good" K(s) increases that allowable ΔG, while designing a bad one may put very strict limitations on the allowable ΔG, and hence model accuracy. The former K(s) is robust, the latter is not.

Regarding the model simplification strategy, then, first observe the right side of Eqn. 3. When $\underline{\sigma}\ (G_R K) >> 1$, $\underline{\sigma}\ [I + G_R K] \approx \underline{\sigma}\ (G_R K)$. Conversely, when $\bar{\sigma}(G_R K) << 1$, $\underline{\sigma}\ [I + G_R K] \approx 1$. Finally, the $\underline{\sigma}\ [I + G_R K\}$ will take on its minimum value in the frequency range where $\sigma_i\ (G_R K) \approx 1$. The frequency range where the latter occurs is of course the (multi-variable) gain crossover region. Consequently, it is this frequency range where the ΔG must be the smallest, and this can be assured if each element of the ΔG matrix is small in this frequency range.

Also, noting the above discussion, Eqn. 3 may be satisfied by rather large ΔG in any frequency range where $\underline{\sigma}\ [I + G_R K]$ is large, and this will occur when $\underline{\sigma}\ (G_R K)$ is large. If K is designed to give a good classical Bode loop shape, $\underline{\sigma}\ (G_R K)$ will be large for frequencies below crossover.[9]

Now consider Eqn. 4. When $\bar{\sigma}\ (G_R K) << 1$, $\underline{\sigma}\ (G_R K)^{-1} >> 1$, and $\underline{\sigma}\ \{I + (G_R K)^{-1}] \approx \underline{\sigma}\ (G_R K)^{-1} >> 1$. Hence the allowable ΔG may also be rather large in this case. Further, if K yields a good loop shape, or is well attenuated, at high frequencies, $\sigma\ (G_R K)$ will be small for frequencies above crossover. So clearly, the ΔG must be smallest in the region of multi-variable crossover, while if K yields a good bode loop shape, rather large ΔG elsewhere may be acceptable and Eqns. 3 and 4 may be satisfied. The above discussion is summarized in Fig. 3.

The final issue to be addressed is that of satisfying assumption 1, or the number of unstable poles of KG_R must be identical to the number of unstable poles of KG. First note that this is equivalent to requiring the number of unstable poles of G and G_R to be the same, since only one K is involved. Then observe that the poles of G are the poles of G_R + ΔG, which consists of the poles of G_R plus the poles of ΔG. Hence to satisfy assumption 1, ΔG must be stable.

Attention will now turn to some additional criteria arising from performance considerations rather that from stability robustness. The system to be considered is that shown in Fig. 4. The vector of responses Y(s) is given by

$$\begin{aligned} Y = [I + (G_1 + \Delta G_1)K]^{-1} (G_1 + \Delta G_1) K (Y_C - N) \\ + [I + (G_1 + \Delta G_1)K]^{-1} (G_2 + \Delta G_2)\, D \end{aligned}$$

Here G_1 is the reduced-order model for the response of G to control inputs, where G_2 is the reduced-order model for the response of G to disturbances being considered. ΔG_1 and ΔG_2 are the analogous model-simplification errors.

The first observation to be made is that stability and stability robustness depends on G_1 and ΔG_1, not on G_2 and ΔG_2. Note that the poles of $(G_1 + \Delta G_1)$ are the poles of the "true" plant G, as are the poles of $G_2 + \Delta G_2$. Hence if K stabilizes G, which will be assured if G_1 and ΔG_1 satisfy the criteria developed previously, K must therefore stabilize $(G_2 + \Delta G_2)$. This is significant since some (stable) poles of G may be approximately cancelled by some zeroes for the transfer functions governing responses to control inputs, but not cancelled in those governing responses to disturbances. Cancelling these poles to obtain G_1 has raised questions by some as to whether those poles so cancelled could lead to problems later in analysis. The answer appears to be that they will not if G_2 is obtained such that those poles are retained. But from the above discussion on stability, the only reason to keep these poles in G_2 (that by assumption are not approximately cancelled) is such that the disturbance-rejection performance predicted using G_2 (when designing K, for example) will be reasonably accurate.

Finally, noting that the disturbance response due to ΔG_2 is

$$Y_{D_2} = [I + (G_1 + \Delta G_1)K]^{-1} \Delta G_2 D$$

for good performance prediction (Y_{D_2} small), ΔG_2 should tend to be small whenever D is large and $(G_1 + \Delta G_1)K$ is small. But here again, if K is designed to obtain a "good loop," it will be designed such that G_1K (and by implication $(G_1 + \Delta G_1)$ K) will be large over the frequency range where D is large. Consequently, this should not pose stringent requirements on ΔG_2.

In ending this section, it is worth noting that assuming K is designed properly has been critical. By doing so, one takes advantage of one of the basic advantages of a good feedback system, reduction in sensitivity to plant (or plant model) variations. This allows the development of a modeling procedure that focuses on the really critical problem of obtaining a good model in the crossover region.

3. Methodology and Sample Results

The procedure offered was discussed in detail in Ref. 5, and the computational technique is summarized again in Table 1. The technique is a frequency weighted internally-balanced approach, with stable factorization in the case of an unstable plant G. The stable factorization procedure sets the unstable subsystem of G aside via partial fraction expansion, leaving the remaining subsystem G_S stable. This stable subsystem is then reduced, such that a stable reduced order model G_{RS} is guaranteed. The unstable subsystem is then rejoined with G_{RS} to obtain the final reduced-order model G_R. By this procedure, the number of unstable poles of G are preserved. In fact the unstable poles in G are exactly retained in G_R.

The internally balanced technique [10] requires the frequency-weighting extension[5] since the basic technique leads to small model-simplification errors ΔG where the elements of G have large magnitude, which is not necessarily the crossover region. Further, a very poor model may be obtained where the elements of G have small magnitude. As will be shown later, this can be totally unacceptable.

In Ref. 11, a frequency-weighted approach was also suggested, but the weighting required the knowledge of the compensator K, obtained using the full-order plant. Since designing a simple K using the simpler plant G_R is the typical design objective, the above weighting is undesirable. In Ref. 5, it was noted that simply adding a weighting filter obtainable by inspection of the Bode plots of G and knowledge of the desired crossover frequency range let to excellent results. This filter is easily discarded after G_R is retained. In the example presented later, it will be shown that this approach again appears quite acceptable.

The key to the concept is the knowledge of the fact that the internally balanced approach yields a small ΔG where the elements of G have large magnitude. Heuristically, if a filter W(s) is used such that W(s)G(s) has large magnitude in the required frequency range, and if WG reduced such that WG_R is obtained, then G_R will have the desired properties.

Table 1. Frequency Weighted Internally Balanced Reduction

Given: System State space description A, B, C and weighting filter state space description A_W, B_W, C_W.

Find: r^{th} order system

Step 1: Solve for X and Y

$$\begin{bmatrix} A & BC_W \\ 0 & Z_W \end{bmatrix}\begin{bmatrix} X & X_{12} \\ X_{21} & X_{22} \end{bmatrix} + \begin{bmatrix} X & X_{12} \\ X_{21} & X_{22} \end{bmatrix}\begin{bmatrix} A^T & 0 \\ C_W^TB^T & A_W^T \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & B_WB_W^T \end{bmatrix} = 0$$

$$\begin{bmatrix} A^T & 0 \\ C_W^TB^T & A_W^T \end{bmatrix}\begin{bmatrix} Y & Y_{12} \\ Y_{21} & Y_{22} \end{bmatrix} + \begin{bmatrix} Y & Y_{12} \\ Y_{21} & Y_{22} \end{bmatrix}\begin{bmatrix} A & BC_W \\ 0 & A_W \end{bmatrix} + \begin{bmatrix} CC^T & 0 \\ 0 & 0 \end{bmatrix} = 0$$

Step 2: Find T and Σ where $XY = T\Sigma^2T^{-1}$, $T = [T_r\ T_{n-r}]$, $T^{-T} = [U_r, U_{n-r}]$

$$\Sigma^2 = \begin{bmatrix} \Sigma^2 & 0 \\ 0 & \Sigma_{n-r} \end{bmatrix} \qquad \text{where}$$

$$\Sigma r = \text{diag}(v_{ci}\ v_{oi}) \quad i = 1, \ldots, r$$

$$\Sigma_{n-r} = \text{diag}(v_{ci}\ v_{oi}) \quad i = r + 1, \ldots, n$$

$$v_{ci}\ v_{oi} \geq \ldots \geq v_{ci}\ v_{oi} \geq 0$$

Step 3: r^{th} order system is

$$A_r = U_r^TATr$$

$$B_r = U_r^TB$$

$$C_r = CT_r$$

As the example, consider an elastic aircraft identical to the configuration investigated in Refs. 3 and 6. This configuration is of reasonably conventional geometry with a low-aspect ratio swept wing, conventional tail, and canard. A numerical model for the longitudinal dynamics is available from the above references. Both rigid-body modes and four elastic modes (resulting in a 11th order model) are included. The in-vacuo vibration frequencies are 6.3, 7.0, 10.6, and 11.0 rad/s, and are representative for a supersonic/hypersonic cruise vehicle. These frequencies, furthermore, are all near the anticipated frequencies at crossover for the control systems to be designed.

Control inputs are elevator deflection δ_B and canard deflection δ_C, while the disturbance is the perturbation in angle of attack due to atmospheric turbulence α_g. Selected responses are vertical acceleration a_z' measured at the cockpit and pitch rate q measured at the antinode of the first bending mode. Therefore, the flight and structural mode control loops in the context of Figure 4, might correspond to the following, for example

$$Y = [a_z'\ q]^T$$
$$U = [\delta_B\ \delta_C]^T$$
$$D = \alpha_g$$

Obtaining the reduced order model G_1 was the subject of Ref. 6. An anticipated crossover frequency range (for G_1K) was assumed as 1 to 10 rad/s. In that reference, it was also noted that a fourth-order for G_1 was sought based on the observation that the full order model has two oscillatory models in this frequency range.

Attention is now turned to the requirements for G_2. As a realistic example, the Dryden gust spectrum for turbulence is used to describe the disturbance. A fourth-order model for G_2 is sought based on the observation that the full order model has two oscillatory models in the frequency range where the spectrum of D is largest. This frequency range is coincidentally also 1 to 10 rad/s.

The reduced order models for G_1 and G_2 were then obtained simultaneously from the frequency-weighted internally-balanced reduction technique [5] which was specifically developed to meet the criteria in Section 3. The frequency-weighting filter used was a band pass filter of unity magnitude in the 1 to 10 rad/s frequency range with 40 db/dec roll off on either side of this frequency range.

Table 2 contains the reduced order state space matrices A, B, C and D. Figures 5 through 10 show the reduced order and full order frequency response magnitudes for G_1 and G_2. Observe that the reduced order model accuracy approximates the full order model in the 1 to 10 rad/s frequency range as desired. To complete this example, a simple control law, consisting of three constant gains was synthesized using the model G_1. The synthesis objective was to augment the damping of the first aeroelastic mode with acceleration feedback to the canard, to augment the short period damping with pitch-rate feedback to the elevator, and to provide some response decoupling with a cross feed from the elevator to the canard. The resulting control law is of the following form

$$\begin{bmatrix} \delta_C \\ \delta_B \end{bmatrix} * \begin{bmatrix} K_1 & K_2K_3 \\ 0 & K_3 \end{bmatrix} \begin{bmatrix} a_Z' \\ q \end{bmatrix} + \begin{bmatrix} 1 & K_2 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \delta_{C_{com}} \\ \delta_{B_{com}} \end{bmatrix}$$

Actuation effects were modeled with simple first-order lags, with corner frequencies at 20 r/s for both the canard and the elevator.

Table 2 Reduced Order Model

$$\begin{array}{c|c} A & B \\ \hline C & D \end{array}$$

-.9932	. 8294	-.0138	-.0507	-31.67	14.48	13.59
-2.013	-.0137	.0121	.0329	35.92	-21.42	-24.38
-5.593	-.6638	-.3175	-9.658	-593.7	-420.0	700.1
4.934	.2098	3.739	-.5171	-281.4	-175.2	342.5
.0665	-.03471	.0017	.0015	0	0	0
8.762	.7218	.9287	-2.038	52.01	-244.5	333.0

$y =$ q (r/s) $\quad$ $u = \delta_B$ (rad) $\quad$ $D = \alpha_g$ (rad)
az' (ft/s) $\quad$ δ_C (rad)

Shown in Fig. 11 is the plot of Eqn. 3, while Eqn. 4 is shown in Fig. 12. Note that although this control law did not result in high gain (large G, K) at low frequencies, Eqn. 3 was still satisfied below crossover region. Conversely, Eqn. 4 is satisfied, although barely, in the frequency range above crossover. Hence, from the argument in Section 2, if ω^* in Eqns. 5 and 6 is in the crossover region, stability is assured. For reference, the pitch-rate to elevator transfer function is

$$\frac{q(s)}{\delta_{E_C}(s)} = \frac{50\,(0.33)[.13,4.84][.01,10.6][.03,11.0][.21,13.](45.)}{[.53,1.81][.15,4.78][.02,10.8][.03,11.][.19,13.3](19.)(69.)}$$

4. An Additional Criteria

As noted in Section 3, the ΔG arising from the model simplification must satisfy stringent criteria in the crossover region, and if Eqn. 3 and/or 4 (or 5 and 6) is satisfied, closed-loop stability is assured. To be discussed here is the fact that the controller K should not be such that small ΔG is amplified such that $\bar{\sigma}\ (\Delta GK)$ becomes large. It will be shown by example that this can easily occur where the magnitudes of G (or of the g_{ij}'s) are small. Hence, the example will demonstrate why obtaining a good model in this situation is important (recall that unweighted balanced reduction has a problem here), and some implications regarding control-law synthesis will also arise.

Consider the simple scalar plant

$$g(s) = \frac{(s^2 + .04s + 1^2)}{s(s^2 + .032s + 0.8^2)}$$

The plant is stable and minimum phase, so a robust control law should be obtainable. Using LQG/LTR or H_∞, for example, the following compensator could be obtained.

$$k(s) = \frac{8(s^2 + .032s + 0.8^2)}{(s^2 + .04s + 1^2)\,(s + 8)}$$

It can be easily verified that the loop shape kg is very good, yielding infinite gain margin, 90 degree phase margin, and good roll off above 8 rad/s.

Now assume that the "true" plant is

$$g_{\mathrm{true}} = \frac{0.69\,(s^2 + .048s + 1.2^2)}{s(s^2 + .032s + 0.8^2)}$$

or the numerator "frequency" is in error by 20% (1.0 → 1.2). Note that this could occur, for example, if a vibration mode shape was slightly off in the modeling. Shown in Fig. 13 is the plot of Eqn. 3 for this example, and clearly $\bar{\sigma}(\Delta gk) > \underline{\sigma}(1 + gk)$ at 1 r/s (the designed crossover frequency). Further, a quick check would show the kg_{true} loop to be unstable. But the $|\Delta g| = |g - g_{\mathrm{true}}|$ (not shown) would be found to be rather modest at ω = 1r/s, with much larger $|\Delta g|$ at lower frequencies. The problem could be interpreted as one of the control law k amplifying the $|\Delta g|$ at $\omega = 1$ r/s, and this is confirmed from the plot of $|k(j\omega)|$ in Fig. 14.

Stability of the kg_{true} loop would result, and Eqn. 3 satisfied, if the $|k(j\omega)|$ at $\omega = 1$ rad/sec were simply reduced. This is accomplished with the following compensator

$$k_{\mathrm{mod}}(s) = \frac{8(s^2 + .032s + 0.8^2)}{(s^2 + 1.2s + 1^2)\,(s + 8)}$$

or the damping of the complex compensator poles is increased, and the plant model zeroes close to the imaginary axis are not exactly cancelled. Clearly the loop shape with this compensator is not as "optimal" as the original, but this control law is more robust against this Δg.

Noting that the problem arose with a modeling error that is associated with lightly-damped zeroes, the critical Δg was at a frequency ($\omega = 1$ r/s), where $|g(j\omega)|$ was relatively small as shown in Fig. 15. Hence, obtaining a good model at this frequency is important. Furthermore, by attempting to cancel those lightly-damped zeroes in the plant, the original controller was very sensitive to their location. Increasing the damping of the compensator poles, as in a classical notch filter, made the loop more robust against the uncertainty in the location of these plant zeroes. (Incidentally, this can be accomplished with a modified LTR procedure, as noted in Ref. 12 and in another paper in preparation.)

As a final remark, it is observed that lightly-damped zeroes in the compensator are different from similar zeroes in the plant since through the design and implementation of the compensator, the location of its zeroes may be more accurately defined.

5. Conclusions

Quantitative criteria are presented for model (or controller) simplification. The reduced order model (or controller) must well approximate the full-order system in the (multivariable) crossover region for stability, and stability robustness, to be assured. Bounds on the model-simplification error were noted, and if the bounds are satisfied, stability is assured. It was also noted that the model reduction criteria were functions of the control law, and by synthesizing a robust control law, the criteria could be easier to satisfy.

A numerical procedure, consisting of stable factorization with weighted balancing of coordinates has been shown, by example, to meet the above criteria. The example involved reducing an eleventh order linear model of an elastic aircraft to obtain a fourth-order model leading to the desired six transfer functions.

Finally, another example demonstrated the importance of obtaining good agreement between the full- and reduced-order model in the crossover region, even where the transfer function (or functions) have relatively small magnitude. Furthermore, the example demonstrated that an apparently robust controller could in fact amplify small errors, and lead to unstable results. The problem would occur with any control law that had the effect of cancelling lightly-damped transmission zeroes of the plant model.

6. Acknowledgement

This research was supported by NASA Langley Research Center under Grant NAG1-758. Mr. Douglas A. Arbuckle has served as the technical monitor. This support is appreciated.

7. References

1. Liu, Y. and Anderson, B.D.O., "Controller Reduction Via Stable Factorization and Balancing," Int. Jrn. of Auto. Cont., Vol. 44, No. 2, 1986.
2. Anderson, B.D.O., "Controller Reduction," Plenary Session, 1987 ACC, Minneapolis.
3. Waszak, M. R. and Schmidt, D. K., "Flight Dynamics of Aeroelastic Vehicles," Journal of Aircraft, Vol. 25, No. 6, June, 1988.
4. Bacon, B. J. and Schmidt, D. K., " A Fundamental Approach to Equivalent Systems Analysis," Journal of Guidance, Control, and Dynamics, Vol. 11, No. 6, Nov. - Dec., 1988.
5. Bacon, B. J. and Schmidt, D. K., "Multivariable Frequency-Weighted Order Reduction," Journal of Guidance, Control, and Dynamics, Vol 12, No. 1, Jan. - Feb., 1989.
6. Schmidt, D. K. and Newman, B., "Modeling, Model Simplification, and Stability Robustness With Aeroelastic Vehicles," 1988 AIAA Guidance And Control Conference, Minneapolis.
7. Stein, G., Lecture Notes on Control System Design, MIT, 1979.
8. Kwakernaak, H. and Sivan, R., Linear Optimal Control Systems, New York: Wiley-Interscience, 1972.
9. Doyle, J. C. and Stein, G., "Multivariable Feedback Design: Concepts for a Classical/Modern Synthesis," IEEE Transactions Automatic Control, Vol. AC-26, Feb., 1981.
10. Moore, B.C., "Principal Component Analysis in Linear Systems: Controllability, Observability, and Model Reduction," IEEE Transactions On Automatic Control, Vol. AC-26, February 1981.
11. Enns, D.F., "Model Reduction for Control System Design," Ph.D. dissertation, Department of Aeronautics and Astronautics, Stanford University, June, 1984.
12. Schmidt, D.K., "Research on Control System Design to Meet Aircraft Handling Qualities Requirements," Final Report for MCAIR, July, 1989.

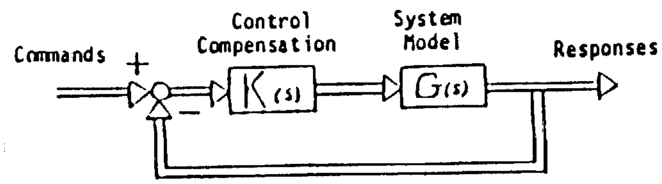


Figure 1

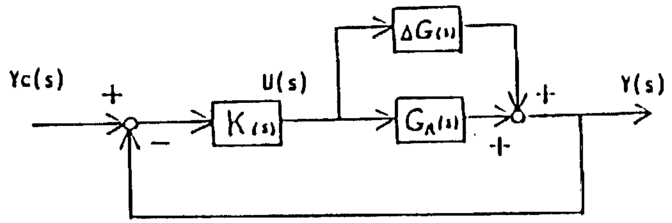


Figure 2

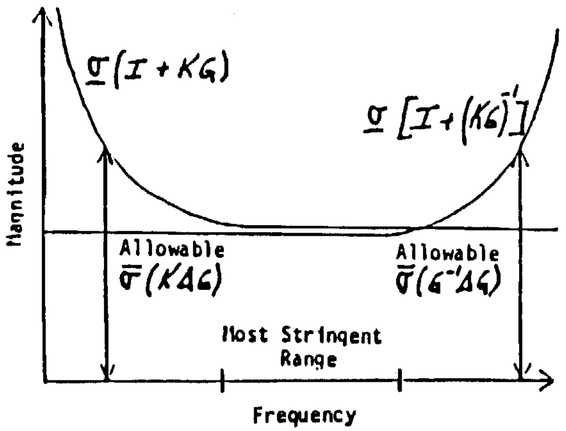


Figure 3

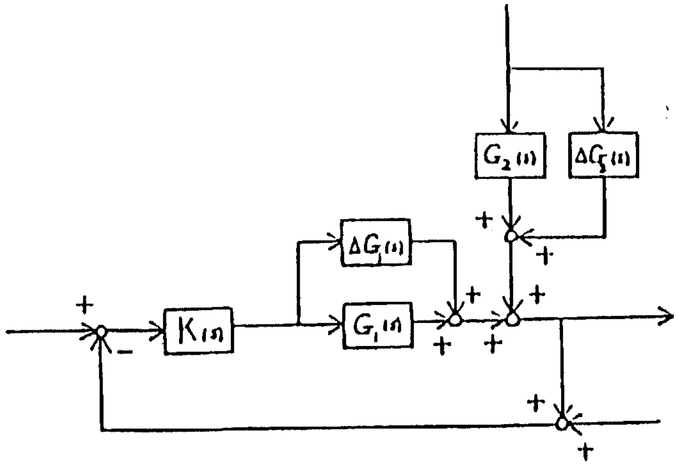


Figure 4

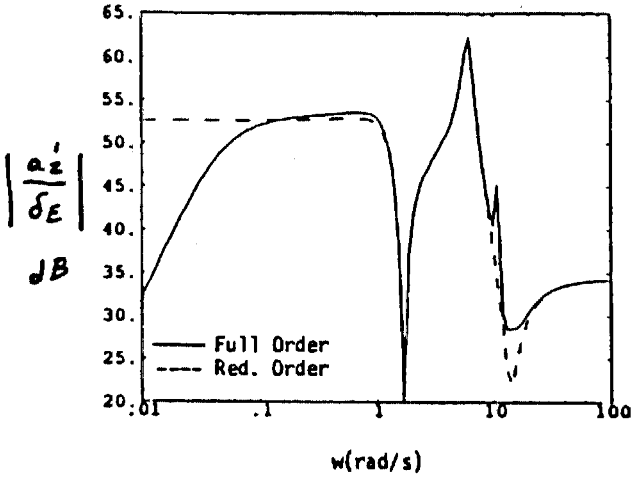


Figure 5

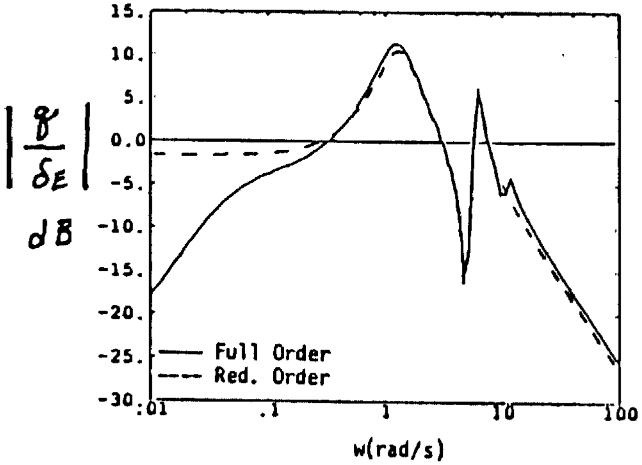


Figure 6

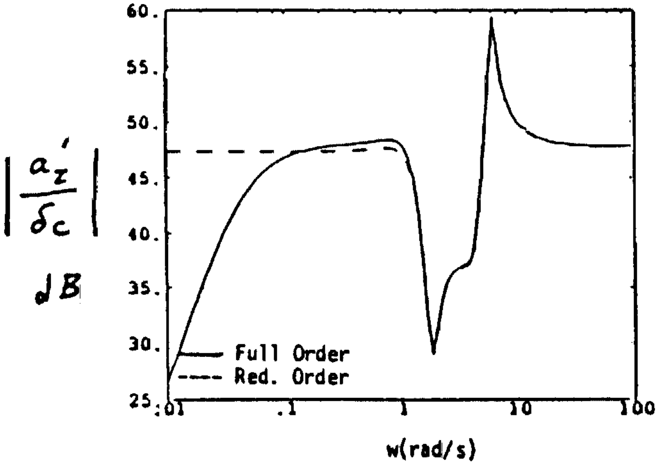


Figure 7

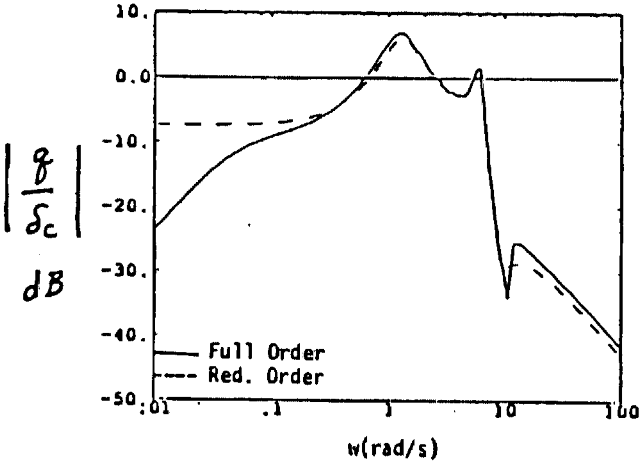


Figure 8

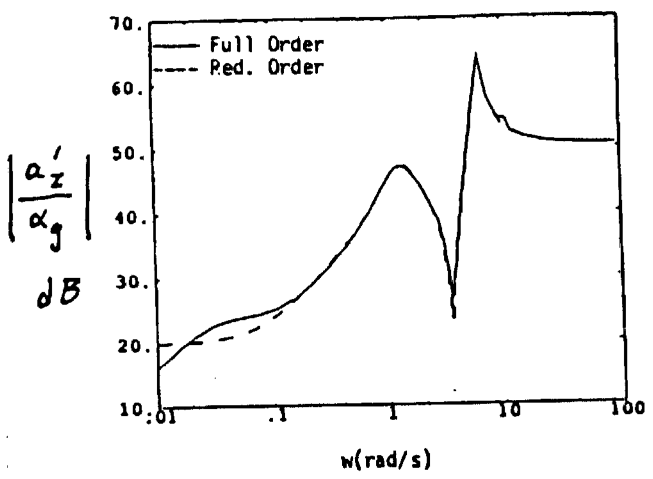
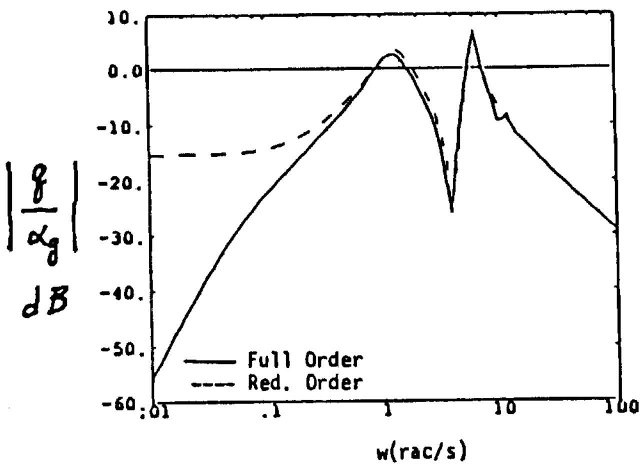


Figure 9



Fiqure 10

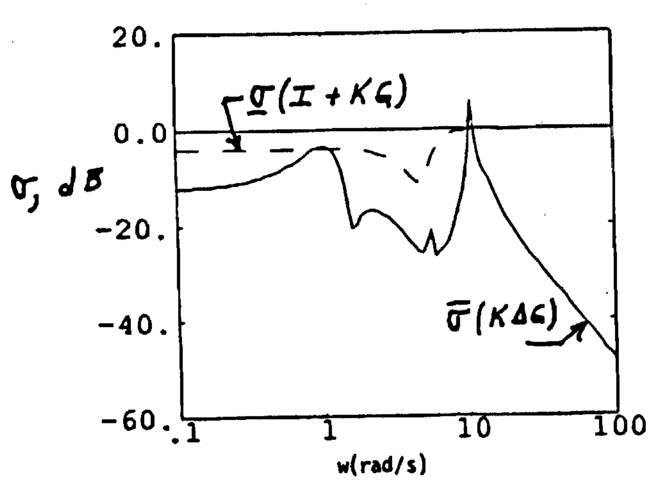


Figure 11

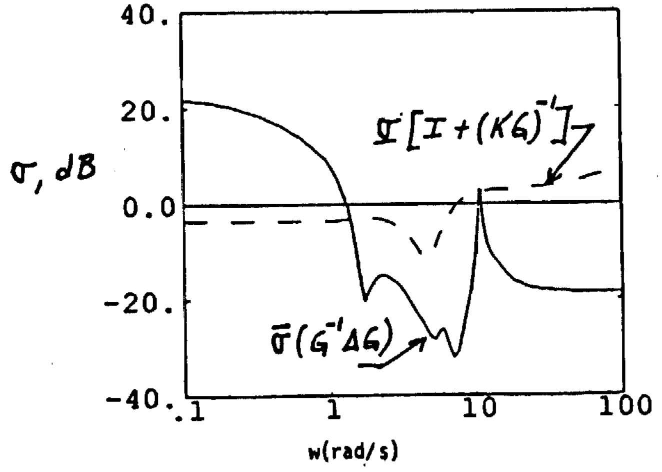


Figure 12

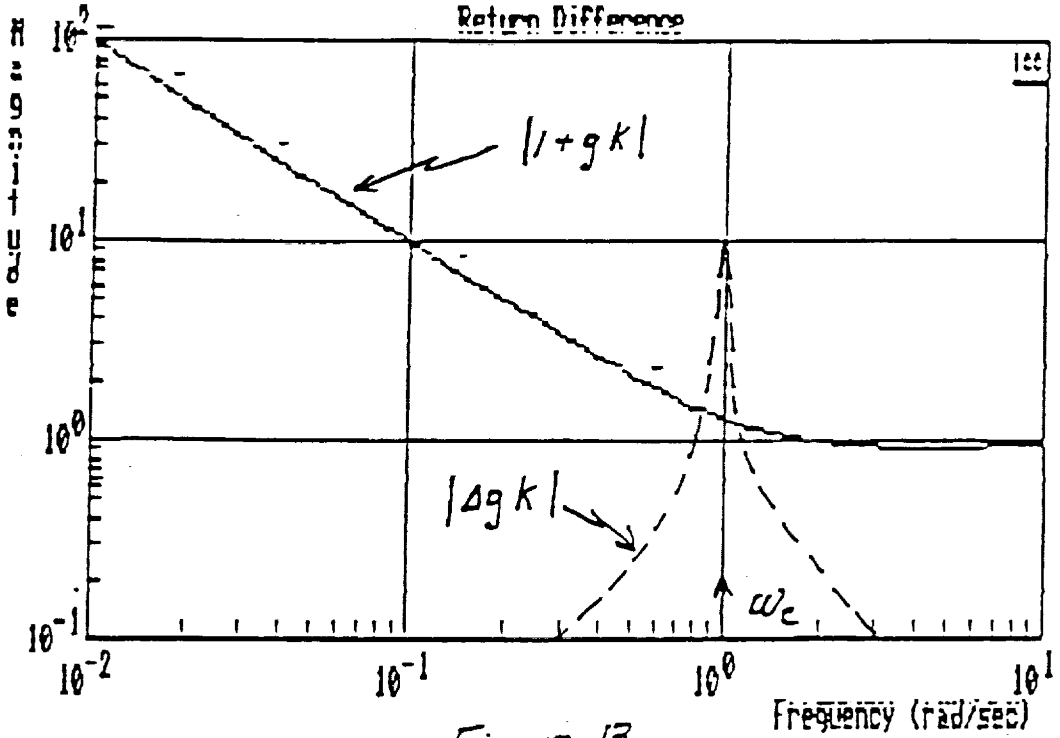


Figure 13.

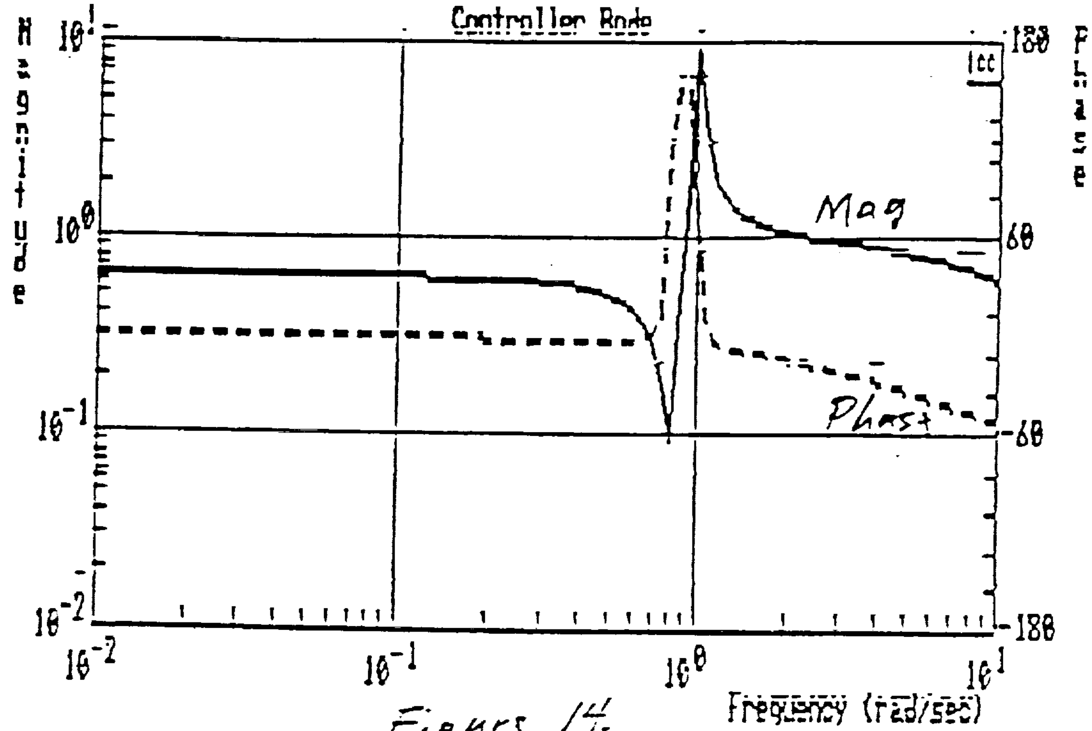


Figure 14.

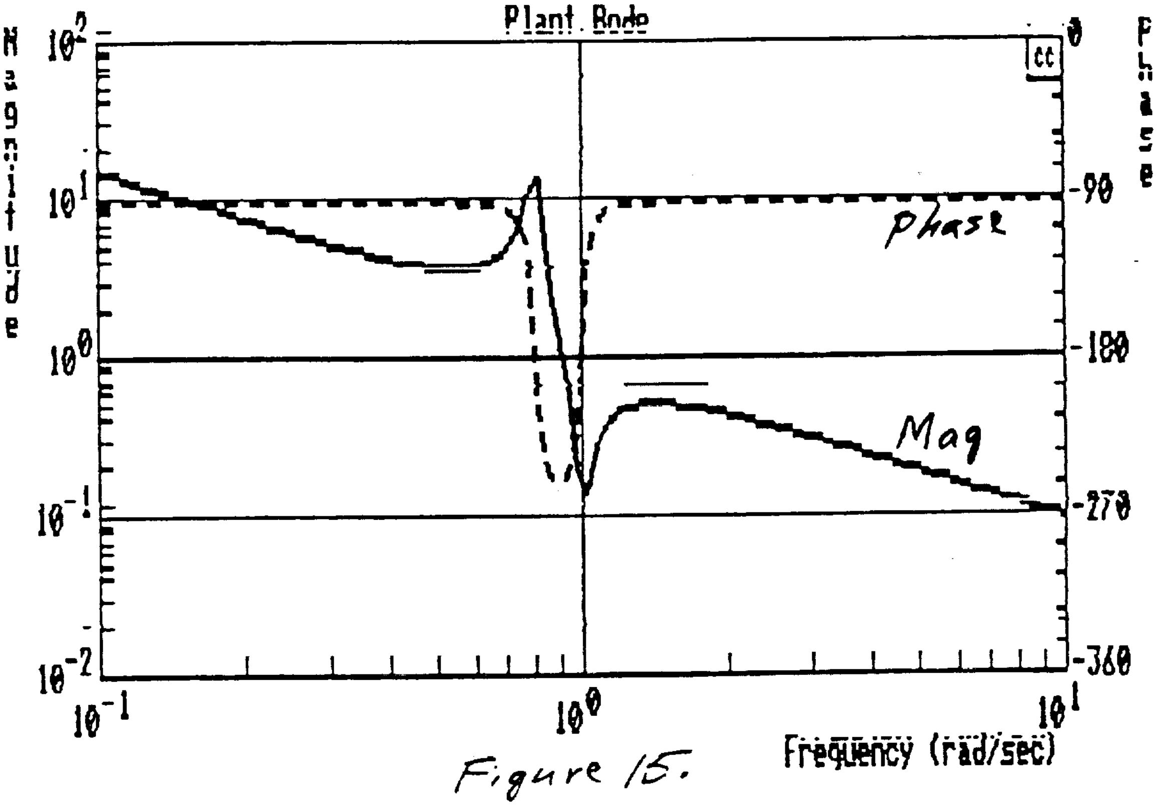

Plant Bode
Magnitude
Phase
cc
10^2
10^1
10^0
10^-1
10^-2
0
-90
-180
-270
-360
10^-1
10^0
10^1
Frequency (rad/sec)
phase
Mag

Figure 15.

N90-23061

DISTRIBUTED NEURAL CONTROL OF A HEXAPOD WALKING VEHICLE

R. D. Beer
L. S. Sterling
Computer Engineering and Science

R. D. Quinn
Mechanical and Aerospace Engineering

H. J. Chiel
R. Ritzmann
Biology

Case Western Reserve University
Cleveland, Ohio

ABSTRACT

There has been a long-standing interest in the design of controllers for multilegged vehicles. Our approach is to apply distributed control to this problem, rather than using parallel computing of a centralized algorithm. We describe a distributed neural network controller for hexapod locomotion which is based on the neural control of locomotion in insects. The model considers the simplified kinematics with two degrees of freedom per leg, but the model includes the static stability constraint. Through simulation we have demonstrated that this controller can generate a continuous range of statically stable gaits at different speeds by varying a single control parameter. In addition, the controller is extremely robust, and can continue to function even after several of its elements have been disabled. We are building a small hexapod robot whose locomotion will be controlled by this network. We intend to extend our model to the dynamic control of legs with more than two degrees of freedom by using data on the control of multisegmented insect legs. Another immediate application of this neural control approach is also exhibited in biology: the escape reflex. Advanced robots are being equipped with tactile sensing and machine vision so that the sensory inputs to the robot controller are vast and complex. Neural networks are ideal for a lower level safety reflex controller because of their extremely fast response time. Our combination of robotics, computer modelling, and neurobiology has been remarkably fruitful, and is likely to lead to deeper insights into the problems of real-time sensorimotor control.

1. INTRODUCTION

In rough terrain multi-legged walking machines promise much greater mobility than their wheeled counterparts. Walking vehicles are being researched and developed for hazardous rough environments such as battlefields, nuclear irradiated facilities and remote planetary exploration. Examples of these vehicles include some Mars Rover configurations (Ref. 1), the six-legged OSU-DARPA vehicle (Ref. 2), and various machines in research labs throughout the world (Ref. 3).

The major problems encountered in walking vehicle development are hardware (especially sensor) reliability and control. The controller must process all of the sensory data and coordinate the motions of the multiple legs with their multiple joints while maintaining stability. Most control approaches require position and rate feedback from all of the joints as well as tactile or force feedback from contact surfaces. The sensory data may be conflicting especially in the presence of sensor failure.

Centralized control, where all control decisions are made based on all sensory information and all performance requirements, has proven inefficient and cumbersome. Centralized control requires that the computational speed be extremely fast relative to the walking speed in order to process the large quantity of complex sensory data and choose an acceptable joint motion in real time. With centralized control, safety requires that the machine stop when sensory information conflicts or hardware failure occurs. Parallel processing can increase computational speed but in itself does not alleviate the basic flaws of centralized control.

Distributed control approaches, where some control decisions are made based on localized information, promise to speed the overall system. Mechanical subdivisions such as individual joints or legs are to be controlled by local dedicated processors. Some sensory and system information must be shared among these parallel processors and a central processor. The central processor is responsible for coordination of the subdivisions. A hierarchical approach to system control permits distributed control of basic (low-level) functions freeing the central processor for higher level control decisions. The parallelism of distributed control promotes robustness in the presence of malfunctions. The difficult questions encountered in applying this approach are: What control architecture is suitable, how are the subdivisions chosen, what information should be shared, and how much authority must the central processor have?

Artificial neural networks offer the possibility of highly distributed control. Each neuron can be viewed as a processor working in parallel with the other neurons. The synapses which connect the neurons permit the sharing of information. Most research in artificial neural nets has emphasized homogeneous architectures where all neurons are of the same design despite their function. Learning is the process where the synaptic weights are adjusted so that the nervous system exhibits the desired input/output characteristics. A synaptic weight of zero nullifies the synaptic connection between two neurons. Learning is generally required for even the most fundamental tasks.

Even relatively primitive animals such as insects have nervous systems which are orders of magnitude more complex than the most advanced artificial neural nets. Yet, biologists are now studying certain insect nervous systems in detail with the intent of understanding their architecture and input/output characteristics.

In nature insects solve the problem of coordination of six multi-segmented legs in real time in the presence of variations in terrain and developmental changes. Also, insects display robustness, that is, they continue to function, although less efficiently, after suffering mechanical and electrical damage (Ref. 4). Biologists have found that nervous systems are heterogeneous, that is, a neurons structure is closely tied to its function. As a consequence, insects display remarkable coordination at birth. Their neural architecture is such that they do not require learning to perform basic functions. However, learning permits the insects to adapt to their environment and become more efficient.

An artificial neural network was developed to control the *kinematic* problem of locomotion of a hexapod walking machine in the presence of the static stability constraint. The hexapods six legs each had two degrees of freedom: foot up/down and leg swing front/back. The controller architecture was inspired by neurobiology; The artificial neural net was heterogeneous and learning was not necessary.

A computer simulation was performed displaying locomotion of the hexapod.

Changing a single input caused the hexapod to change its gait. The gaits are very similar to those observed in nature. The robustness of the controller in the presence of malfunctions was investigated through "lesion studies". For this purpose particular synapses in the artificial neural network were severed rendering a particular neuron ineffective during simulations. These lesion studies demonstrated that the artificial neural controller is robust to damage to any neuron.

The neural controller described in this paper was developed by Beer, Chiel and Sterling and has been published in other forums (Refs. 5,6). We (the entire list of authors) have since been working together on the project. The pupose of this paper is to report what we believe are important findings to the Aerospace community and highlight direct applications of this type of neural control to the Aerospace field.

2. HEXAPOD MECHANICAL MODEL

The mechanical model is a six-legged walking vehicle with two degrees of freedom per leg. It is loosely based on Periplaneta americana, the American Cockroach and Fig. 1 is a top view of the model. The legs can swing back and forth and the foot can be raised and lowered. The small black squares in Fig. 1 denote the feet in the down position. The simulated locomotion of the hexapod takes place in a horizontal plane on a smooth surface. The dashed lines connecting the squares form what is known as the static stability polygon. When the center of mass of the hexapod lies inside this polygon, the system is statically stable. Other than satisfying the static stability constraint, the simulated model considers only simplified kinematics where the leg swing and foot up/down motions are considered to be independent.

The natural insect actually has what can considered to be four revolute degrees of freedom per leg for a total of 24 joint degrees of freedom. The "hip" joint where the leg attaches to the body has two revolute degrees of freedom permitting swing along the body axis and away from the body. The lower two joint degrees of freedom, the "knee" and "ankle", joint axes are aligned. The "foot" is long and relatively flexible. When a foot is down, consider that the foot translation is constrained to zero. Hence, ignoring flexibility, when all six feet are down, there are 18 constraints leaving 6 degrees of freedom. The insect can then move its body rigidly with all its feet down.

Our hexapod model, on the other hand, has only two degrees of freedom per leg. Furthermore, we make the simplification that the swing and foot up/down motions are independent. Actually, the joint motions must be coupled and the joints must move simultaneously to accomplish the desired walking motion. The desired walking motion involves only translation of the body forward or backward without unnecessary pitching or other body motions. With all feet down and constrained to zero translation, our model permits the body to translate forward or backward with appropriate coupled motion of all the joints.

3. NEURAL CONTROLLER MODEL

A schematic showing the electrical circuit of the most complex neuron used in the heterogeneous network is shown in Fig. 2. Weighted synaptic currents are input/output from/to connected neurons. The synapses are weighted to establish a hierarchy for the input information. Intrinsic currents are internal to the neuron and permit "self stimulation." The parallel RC circuit mimics biological cell membrane electrical characteristics. The output firing frequency of the cell is a nonlinear

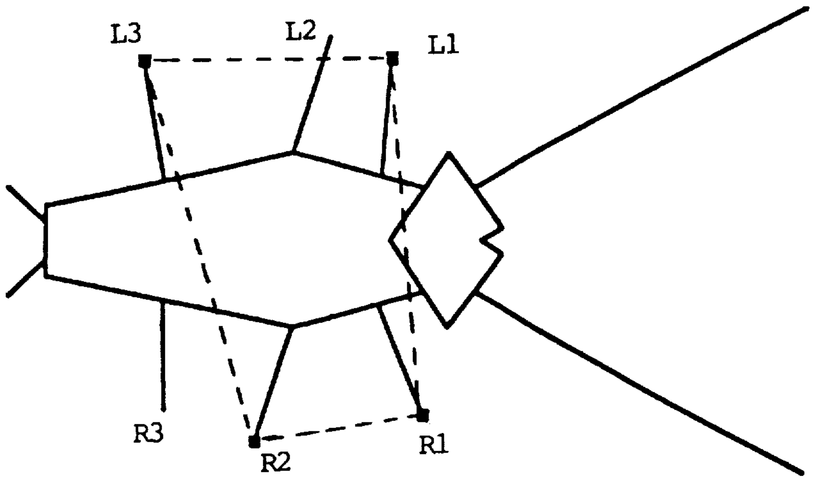


Fig. 1 Periplaneta computatrix (Simulated model)

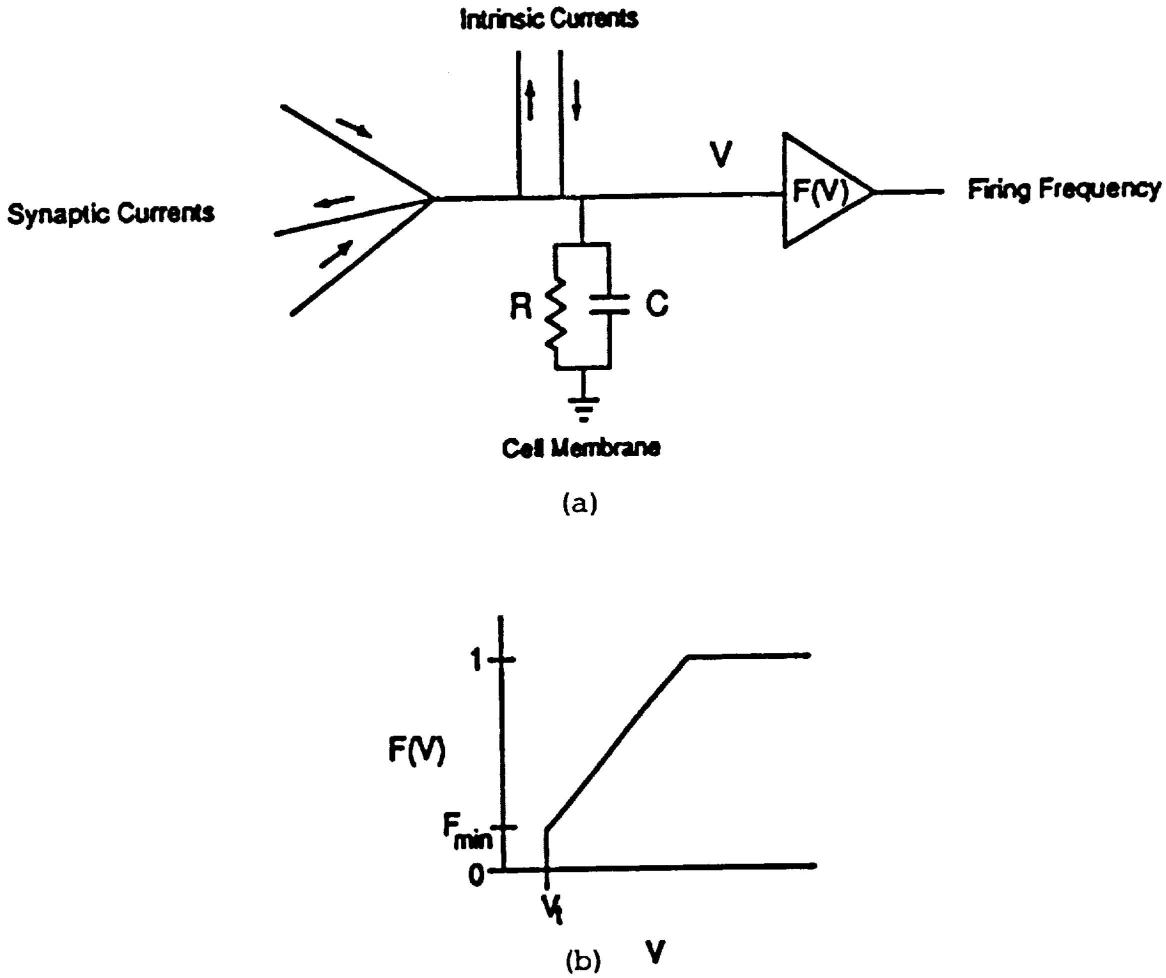


Fig. 2 Neural Model

function f(V) of the resulting neural potential V. Saturating linear threshold functions were used as shown in Fig. 2b.

Summing the currents in the network, the state equation for the ith neuron can be expressed as

$$C_i \frac{dV_i}{dt} = \sum_{j=1}^{n} S_{ij} F_j(V_j) + \sum_{k=1}^{m} \bar{I}_k(V_i,t) - \frac{V_i}{R_i}$$

where S_{ij} is the weight for the synapse carrying input current from the jth neuron to the ith neuron. If this weight is zero, there is no electrical connection between these neurons. In general the model includes n neurons in the network and m intrinsic currents for the ith neuron. $\bar{I}_k$ is the kth intrinsic current for the ith neuron and it is in general a function of the neuron potential and time.

Biological nervous systems display heterogeneous architectures. In particular, some natural neurons exhibit intrinsic stimulation characteristics and some do not. In fact, intrinsic currents have proven to be important neural components underlying many behaviors. A "pacemaker" cell is capable of intrinsically producing rhythmic bursting and can be externally inhibited or excited by other neurons. In this way the frequency and phase of the internal bursting rhythm can be changed by other neural inputs. As described by Kandel (Ref. 7), a pacemaker cell exhibits the following characteristics: 1) when it is inhibited below its threshold, it does not fire, 2) when it is excited beyond saturation, it fires continuously, 3) between these extremes, the firing frequency is a continuous function of the membrane potential, 4) transient excitation or inhibition can shift the phase of (reset) the intrinsic firing rhythm.

Pacemaker cells play a crucial role in our locomotion controller. In our model two intrinsic currents permitted a neuron to act as a pacemaker cell. One current I_H tended to raise the neural potential above firing threshold and the other intrinsic current I_L tended to lower the potential below threshold. The "control law" for these currents obeyed the following rules: 1) I_H is triggered or I_L is terminated when the cell potential goes above threshold, and remains active for a fixed time period, 2) I_L is triggered when I_H terminates, and then remains active for a variable time period which is a linear function of membrane potential.

4. NEURAL NETWORK CONTROLLER

The kinematic locomotion controller consists of a network of 6 neurons controlling each leg and 1 central command neuron for a total of 37 neurons. Figure 3 shows the controller for a single leg including the common command neuron. There are three motor neurons per leg: stance, foot, and swing. The stance neuron swings the leg backward and, if the foot is down, propels the body forward. When the foot motor neuron fires, the foot is lowered. The swing neuron swings the leg forward and, if the foot is down, propels the body backward. The level of the outputs of the motor neurons determines the speed of the motor actions.

The pacemaker P natural rythmic firing inhibits the foot and stance neurons and excites swing. The command neuron C excites the pacemaker and stance neurons. This excitation influences pacemaker burst rate and stance

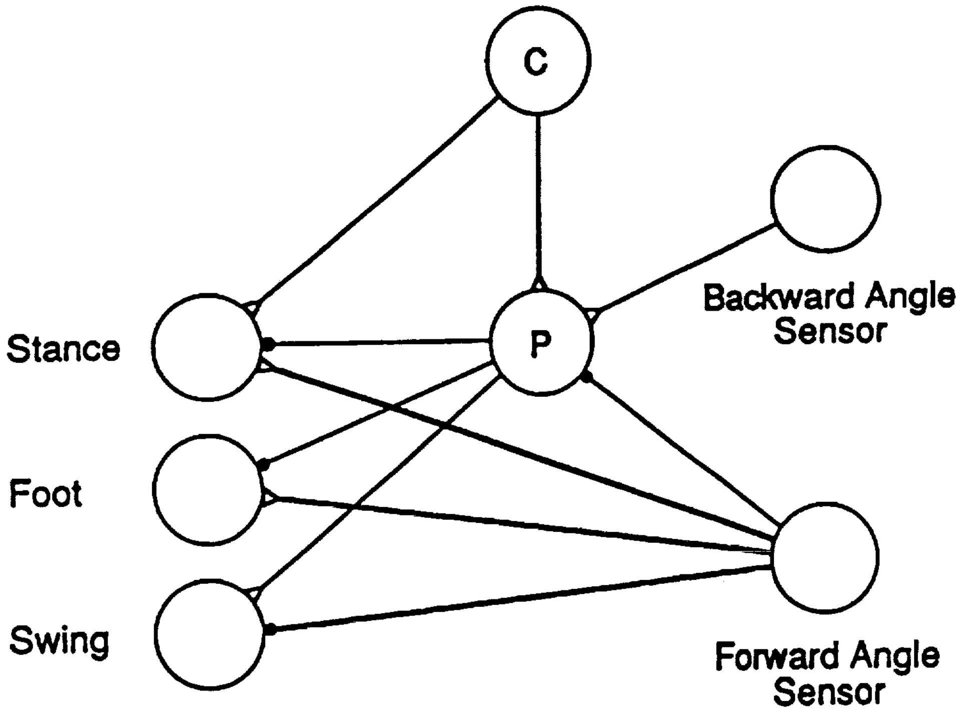


Fig. 3 Neural Network Controller for a Leg

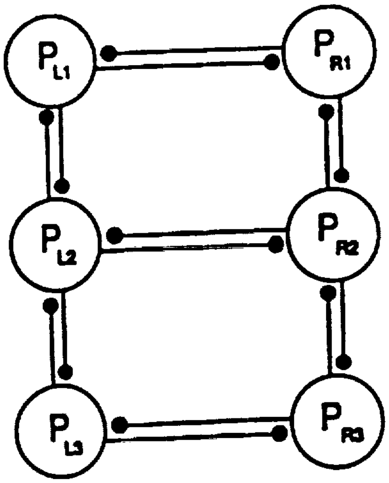


Fig. 4 Adjacent Leg Pacemaker Inhibition

speed so that the command neuron may be thought of in simple terms as the throttle.

The locomotion controller can function open-loop based on the "natural" pacemaker rhythm. However, in order to smooth and coordinate the swing/stance transitions, limit-switch sensor neurons were added to sense when the legs reached extreme backward and forward angle positions. This information is fedback to the pacemaker neuron. The forward angle sensor information is also fedback to the motor neurons which provides a biologically inspired "stance reflex" (Ref. 8). The stance reflex compensates for the delay at the end of each swing caused by the RC characteristics and smooths the jerky movements otherwise caused by the delay, and increases stability.

The backward angle sensor neuron excites the pacemaker which in turn excites the swing. The forward angle sensor inhibits the pacemaker and swing and excites the stance and foot (stance reflex). Hence, the sensors reinforce the controller strategy and coordinate the leg motors. These sensors were inspired by the hair plate receptors observed on the natural insect.

If we ended the controller development at this point, the legs would function independently except for the input of the common command neuron. The resulting walking gaits show arbitrary leg movements and are awkward, uncoordinated and often statically unstable. Again, inspired by observed natural insect walking gaits (Ref. 8), we observe that adjacent legs do not swing simultaneously which is clearly a good rule of thumb for maintaining static stability. This controller strategy was implemented through adjacent pacemaker inhibition shown in Fig. 4.

Stability remains a problem for the controller because there is no device to order the stepping sequence. The gaits were found to depend on the initial angles of the legs. Turning to biology for inspiration once more, we note that insects tend to walk with their legs in a particular sequence: the "metachronal wave" of stepping progresses from back to front (Ref. 9). This sequence was achieved by our controller by slightly increasing the leg angle ranges of the rear legs, lowering their stepping frequency for a given constant swing/stance angular rate.

The rear legs angle range increase along with the pacemaker coupling of Fig. 4 results in the rear legs entraining the middle legs as illustrated in Fig. 5. In this simplified example, R3 and R2 denote the right rear and right middle legs. The square impulses drawn with dashed lines show the coupling pacemaker inhibition by the other leg. The bold lines denote pacemaker firing. Note the longer stroke of R3. In this example the fourth R2 pacemaker firing is delayed through inhibition by R3. The entrainment is then complete, the middle leg swings immediately after the back leg.

5. SIMULATION RESULTS

When the neural controller was implemented on the simulated hexapod in a smooth environment, the hexapod walked successfully. The walking speeds and gaits changed when the firing frequency of the central command neuron was varied. A continuum of statically stable gaits was observed from the "wave gait" to the "tripod gait." Very similar gaits are observed in biological insects locomotion. Noteably, these gaits "naturally" occur in the simulation environment as a result of the interaction between the neural controller and the mechanical model.

Figure 6 is a comparison of gaits of biological insects (Ref 10) with the simulated gaits of the model hexapod. The legs are labeled as in the top of Fig. 6. A black bar denotes the swing phase of each leg; during the space between the swings, the legs are in the stance phase. The simulated gaits

(Fig. 6b) were chosen from the continuum of possible gaits to most closely match the displayed natural gaits. These gaits were obtained by merely increasing the firing rate of the command neuron from the lowest (top figure) to the highest (bottom figure).

The bottom-figure (Fig. 6a) natural gate is statically unstable and so could not be obtained our model. In the wave gate, the metachronal waves on each side of the body are nearly separated (top comparative figures in Fig. 6a and 6b). In the tripod gate (bottom comparative figures in Fig. 6a and 6b) the front and back legs on one side of the body step with the middle leg on the other side.

Lesion studies were conducted to determine the robustness of the controller to particular neurons being disabled (Ref. 6). Because of its highly distributed architecture, the controller was found to be robust to damage to any individual element. For instance, when the command neuron was disabled during a stable gait, there was no effect. When the command neuron was disabled initially, a stable gait displaying the metachronal wave was slowly reached. This illustrates the value of the "self stimulating" pacemaker neurons. In another case, the rear sensors were disabled during the tripod gait with no effect. When the rear sensors were disabled during a slower gait, a stable gait ensued.

6. CONCLUSIONS AND ONGOING WORK

An artificial neural network was designed for the purpose of controlling a simulated hexapod walking vehicle. The neural model and network architecture were inspired by observed natural insect nervous systems. The simulation addressed the kinematic problems of locomotion of a six legged walking vehicle with two degrees of freedom per leg subject to the static stability constraint. The neural control "strategy" includes feedback from sensor neurons which fire when the legs reach their extremes angles. Pacemaker neurons which have an intrinsic firing rhythm play a crucial role in the controller.

The hexapod walked successfully exhibiting a continuum of statically stable gaits. The walking gait and speed depended on the central command neuron firing frequency so that the command neuron could be thought of as a throttle. The gaits appear "naturally" in simulation because of the interactions between the controller and the mechanical model. The gaits are very similar to those exhibited by natural insects.

The controller is highly distributed; The only coupling between the legs is through adjacent leg pacemaker inhibition and the command neuron is the only common central neuron. Furthermore, the pacemakers natural rhythm enables stable walking gaits even when the command neuron is silent. This high degree of control distribution (or parallelism) produces an extremely robust controller. In fact, the controller is robust to removal of any individual neuron.

The high degree of distribution also yields a controller with extremely quick reflex-like responses. A clear application of this type of system is for safety-reflex control of all types of robots and telerobots with advanced sensing capabilities. A great deal of complex sensing information can be provided by tactile and force sensors as well as machine vision. When a dangerous situation arises, the time delays associated with a centralized controller will not permit the robot to recognize the danger and react in the short amount of time that may be needed to avoid catastrophe. The danger could be for the robot itself, for humans, or for precious cargo. A lower level neural reflex controller can be preprogrammed to recognize dangerous situations and reside in the robot control hierarchy. When the situation arises, the reflex controller can then take command to move the robot to a predetermined safe configuration.

The reflex safety response is also biologically inspired. For instance, the American Cockroach senses wind from a suddenly approaching predator, turns away and begins to run in approximately 50 milliseconds. Biologists at CWRU are presently studying this phenomenon and the detailed nervous system of this insect.

We are also constructing a three dimensional simulated hexapod model including dynamics so that we can further validate and improve the controller. A small mechanical hexapod machine is being constructed with the intention of applying our controller to a working machine. The vehicle initially will have two degrees of freedom per leg , but a third degree of freedom is to be eventually added to permit smooth turning and climbing.

7. REFERENCES

1. Allen, L. (1988) "Mars Rover Sample Return: Rover Challenges," Presented at *AIAA/NASA First International Symposium on Space Automation and Robotics*, November 29-30, Arlington, VA.
2. Song, S. M. and Waldron, K. J. (1989), *Machines that Walk*, MIT Press, Cambridge, MA.
3. Raibert, M. H. (1986), *Legged Robots that Balance*, MIT Press, Cambridge MA.
4. Graham, D. (1985), "Pattern and Control of Walking Insects," *Advances in Insects Physiology*, Vol. 18, pp. 31-140.
5. Beer, R. D., Chiel, H. J., and Sterling, L. S. (1989), "Heterogeneous Neural Networks for Adaptive Behavior in Dynamic Environments," to appear in *Advances in Neural Information Processing Systems*, Volume 1, Morgan Kaufman.
6. Chiel, H. J. and Beer, R. D. (1989), "A Lesion Study of a Heterogeneous Artificial Neural Network for Hexapod Locomotion," *Proceedings of the International Joint Conference on Neural Networks (IJCNN89)*, Washington, D. C. June 1989, pp. 407-414.
7. Kandel, E. R. (1976), "*Cellular Basis of Behavior: An Introduction to Behavioral Neurobiology*, W. H. Freeman.
8. Pearson, K. (1976), "The Control of Walking," *Scientific American*, Vol. 235, pp. 72-86.
9. Graham, D. (1977), "Simulation of a Model for the Coordination of Leg Movements in Free Walking Insects," *Biological Cybernetics* ,Vol. 26, pp. 187-198.
10. Wilson, D. M. (1966), "Insect Walking," *Annual Review of Entomology*, Vol. 11, pp. 103-122.

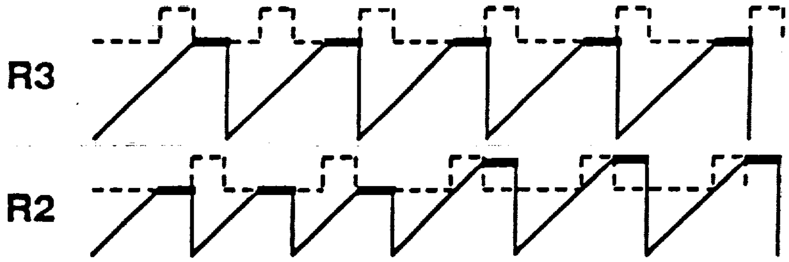
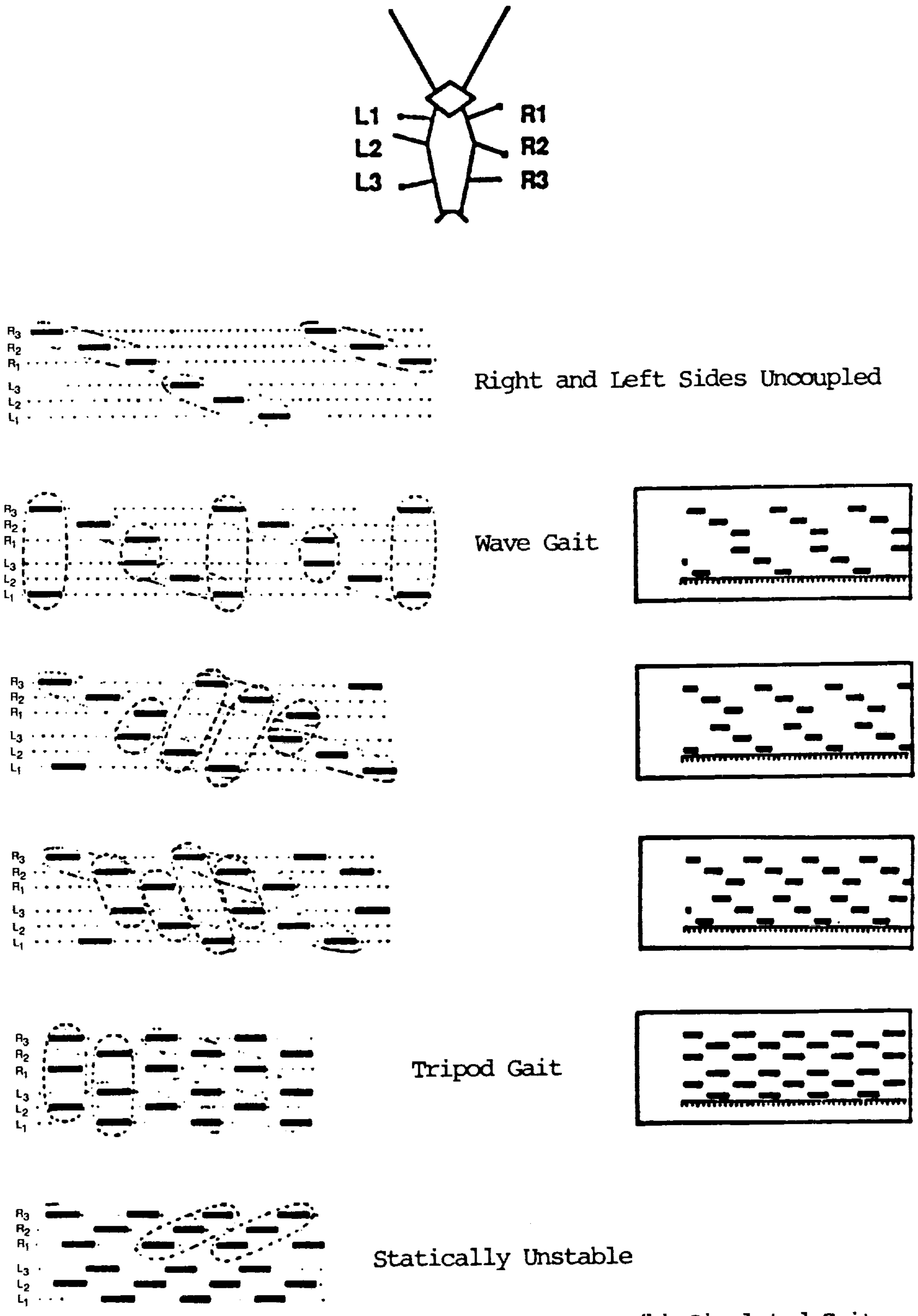


Fig. 5 Entrainment of Middle Leg by Rear Leg



(a) Natural Gaits (Ref. 10)

(b) Simulated Gaits

Fig. 6 Comparison of Natural Gaits with Simulated Gaits

N90-23062

Combining High Performance Simulation, Data Acquisition, and Graphics Display Computers

by

Dr. Robert J. Hickman
21 Brittany Ct.
Newport Beach CA, 92660

ABSTRACT

For a growing class of simulation problems, the generation of the motion and signal environment for testing hardware-in-loop requires high speed computing along with data transfer, mass storage, and graphic display rates sufficient to save and display the data generated. This typically requires a complex of specialized processors that are specifically selected for their processing tasks, along with a specialized communication computer system for fast inter-processor communication and data transfer. This computer complex can be employed in different roles as the test hardware-in-loop interface matures during an advanced development or full scale engineering development program. Early in such a program, all elements of the system to be studied (and their environments) are simulated. Then as in-loop elements are developed, they are inserted into the complex, and the simulation computers concentrate on increasing the fidelity of the test environment dynamics that the in-loop elements experience.

This paper discusses issues involved in the continuing development of an advanced simulation complex. This approach provides the capability to perform the majority of tests on advanced systems, non-destructively. The controlled test environments can be replicated to examine the response of the systems under test to alternative treatments of the system control design, or test the function and qualification of specific hardware. Field tests verify that the elements simulated in the laboratories are sufficient.

The digital computer complex is hosted by a Digital Equipment Corp. MicroVAX computer with an Aptec Computer Systems Model 24 I/O computer performing the communication function. An Applied Dynamics International AD100 performs the high-speed simulation computing and an Evans & Sutherland PS350 performs on-line graphics display. A Scientific Computer Systems SCS40 acts as a high-performance Fortran program processor to support the complex, by generating numerous large files from programs coded in Fortran that are required for the real-time processing.

Four programming languages are involved in the process, FORTRAN, ADSIM, ADRIO, and STAPLE. FORTRAN is employed on the MicroVAX host to initialize and terminate the simulation runs on the system. The generation of the data files on the SCS40 also is performed with FORTRAN programs. ADSIM and ADRIO are used to program the processing elements of the AD100 and its IOCP processor. STAPLE is used to program the Aptec DIP and DIA processors.

INTRODUCTION

As developers of complex systems that include sensors, computers and actuators we must continually examine the need to maintain and improve our capability to design and test such systems. Advances in technology have encouraged our customers to seek more advanced systems that involve increasingly complex on-board control. While laboratory tests do not totally replace field testing, the laboratory can provide a controlled test environment wherein semi-physical testing of a number of alternative systems and components can be replicated in complex computer generated environments. This is difficult to achieve in field tests as many elements are not under the control of the experimenter and flight tests, for non-recoverable systems, typically result in destruction of the test hardware. The modern concept is to perform the majority of tests in non-destructive testing in simulation laboratories and utilize field tests to verify that the elements simulated in the laboratories are sufficient. The reduction in the number and duration of field tests leads to a direct reduction in costs and an improved competitive position in the high-technology development business. This paper discusses an approach for the phased introduction of target sensor and signal processor elements into a hardware-in-the-loop simulation that is designed to provide the capability to test and evaluate such systems.

During the initial stages of development the entire system is simulated by computer. As sensor and signal processing hardware elements are developed they are inserted into the control loop, producing a hybrid hardware-computer system. This hardware-in-loop simulation allows significantly improved non-destructive evaluation of design performance. Typically breadboard flight controllers or processors, with their embedded software algorithms and special purpose hardware, computational accuracy and latency can be tested early in a development program to evaluate the adequacy of the candidate approaches for control. As long-lead items become available (e.g. sensors, signal processors) they are integrated into the loop and their simulated characteristics are replaced by actual physical performance leading to improved confidence in the results.

A generic block diagram for the simulation of a guided vehicle system employing a target sensor, flight motion sensors and a control servos for aerodynamic fin deflection is presented in Figure 1. These elements appear in the double lined boxes and have the same interface in the simulation facility as they have in flight tests. The target sensor interfaces with the target signal spectrum in the isolation chamber and the flight motion sensors interface with inertia from the motion simulator. The aerodynamic and kinematic models of the vehicle and the target kinematic model are handled by the simulation computer with the target signal model generated in a computer complex that is re-structured as the sensor hardware is inserted into the loop. The 3-way switch under the target signal model illustrates the following options (from right to left):

1. Simulate the entire target sensing and signal processing function. The goal is to provide realistic inputs to the control processor so that performance trades can be evaluated in the conceptual design phase of an advanced development program.

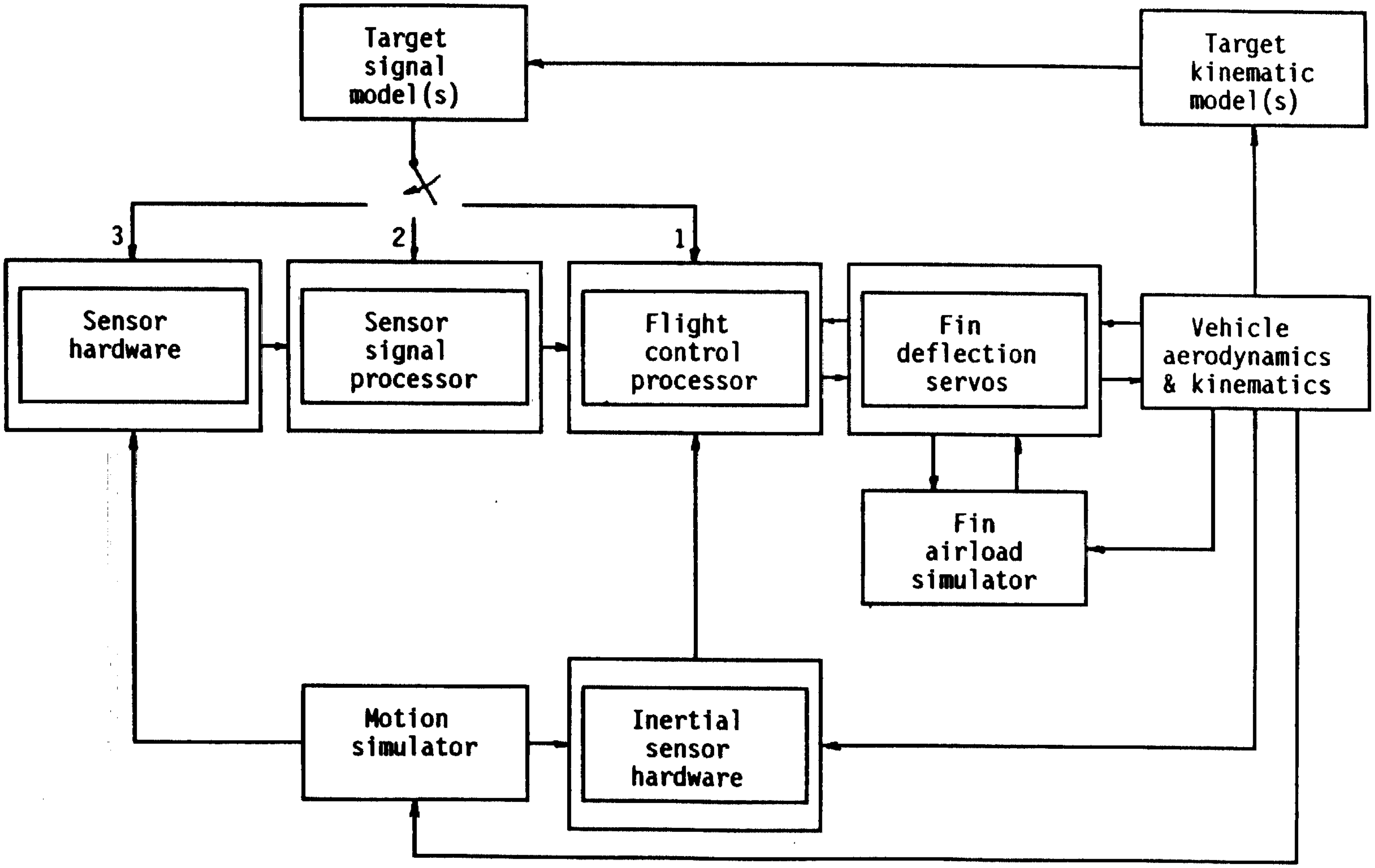


Figure 1, Hardware-in-Loop Block Diagram

2. Simulate the collection of energy from the target and the conversion of this energy into detector or pre-amplifier output signals that are processed by the signal processor. Here the concern is to exercise elements of the signal processing associated with the target sensor. This can involve automatic gain control for the amplifiers, signal thresholding for false alarm control, and target scene element state estimation for the flight control processor.

3. Encode the simulated target scene into an energy distribution of the scene and transmit this in the proper spectrum to the sensor, or in an analogous spectrum for surrogate detectors that are designed to provide "equivalent" response. This option is intended to exercise sensor pointing, energy collection, detectors and amplifiers as well as the elements discussed above.

The third option represents the goal that system developers may wish to achieve with their non-destructive semi-physical testing, however the cost of reaching the level of fidelity in target scene generation that can exercise the resolution and dynamic range of the sensor may inhibit that approach and lead to a compromise on the second option. While this may seem like a reasonable concession the fact that many unsuccessful attempts to develop target sensing systems have been associated with problems in the "front end" of the system involving energy collection, and conversion to signals for processing. Simulating the sensor and not including it as hardware-in-the-loop may really be whistling past the graveyard for the development program as the real need is to identify the anomalous behavior in the target sensor in dynamic non-destructive testing and develop signal processing and flight control "work arounds" until future versions of the sensor, that address fixes for the problems encountered, can be available for test.

REQUIREMENTS

Figure 2 presents a summary of desired features for the specialized computing functions necessary to implement the evolution of sensor hardware-in-the-loop testing. The major blocks in this figure are identified as specialized computing "servers", and may by themselves be a complex of computers serving to produce the specialized function in the overall simulation system. The requirements listed in the blocks are based on attaining high performance, mutual compatibility, and reasonable costs for peripheral devices and components with this distributed processing approach.

Development Server – The development server is the host computer for the system and acts as such for the other computers in the system. This leads to the requirement for open bus peripherals (to reduce costs) and configuration management software tools on this element as the majority of the software for the other computers will be written, cross-compiled, stored in libraries, linked, and down loaded at execution time by the host. The configuration management software is quite important as the complexity created by the inter-dependence of software across the system can become formidable. The VMS operating system from Digital Equipment Corp. (DEC) is an excellent choice for this element of the system as it provides the framework for achieving these requirements. This operating system executes on a DEC VAX computer and most potential users already have considerable experience with this combination. However it is probably not a good idea to use a VAX heavily loaded with unrelated use as the host for this system when a dedicated MicroVAX can handle this work-load for a very reasonable cost.

Development Server

- open bus peripherals
- configuration management
- VMS operating system

Communication Server

- high-speed communication
- transparent mode option
- high-capacity data logging
- modular organization
- VMS compatible

Simulation Server

- high-speed
- 64 bit arithm.
- high-level code
- real-time I/O
- on-line control
- VMS compatible

Fortran Server

- high-speed
- 64 bit arithm.
- open bus I/O
- on-line control
- VMS compatible

Graphics Server

- high-resolution
- high-speed
- high-level graphics
- local pan/zoom
- VMS compatible

Telemetry Server

- high-speed
- on-line gather, smooth & tag
- modular organiz.
- on-line control
- VMS compatible

Figure 2, Desired Processing Features

Communication Server -- The communication server provides high-speed communication among the specialized servers of the system and as a result requires modular organization to accomodate the stepwise growth as this simulation system is developed. An Aptec Computer Systems I/O computer performs the communication function as it provides a high-speed data bus architecture with tightly coupled parallel controllers that supports simultaneous data transfers at the full input/output data rate of the attached processors, thus permitting real-time capture of the simulated environment and test data for storage on high-speed high-capacity disk drives and rapid interpretation of results through on-line graphic displays. An important feature of this computer is its "transparent mode" in which it mimics the DEC Unibus interface. This mode is very useful in reducing the software impact of connecting a set of different computers and/or devices, especially when these elements have been designed for use as attached processors to a DEC VAX host computer. Typically, in distributed systems, the majority of the communication software is involved with non-time intensive initializing and terminating the elements of the distributed complex as it is applied to a test problem. The real-time portion of the communication software may comprise only 10 to 20 percent of the total, so that the major portion of the software can be high-level language calls (e.g. Fortran) to libraries provided by the suppliers of these elements for execution on the VAX host processor for their processor or device. The real-time communication involves the "program mode" where a sub-set of the routines are required to be programmed for the Aptec controllers for high throughput. While this involves re-programming in many cases the form of the code is the same as that provided for the host VAX computer.

Simulation Server -- An Applied Dynamics AD100 simulation computer is employed to perform the table interpolation for aerodynamic coefficients and mass properties as well as the numerical integration of the set of non-linear ordinary differential equations that describe the motion of the flight vehicles and the sightlines. The performance of the AD100 on simulation problems has been compared to a number of general purpose computers, and it takes the capability of super-computers to match its results (Applied Dynamics, 86). The AD100 is a tightly coupled set of high-speed parallel pipelined processors that provide not only high-speed and high-precision but also real-time control of I/O to the attached hardware-in-the-loop, and on-line interactivity through the VAX host. Support software for the AD100, the ADSIM and ADRIO compilers and the INTERACT run-time program significantly reduce the programming and checkout burden and run-time inflexibility that has been associated with special purpose computers.

Sensor "front end" simulation can be performed using an Applied Dynamics AD10 computer to simulate sensor scene scanning and detector/pre-amplifier response. This response is decomposed into table interpolation of simulated detector/pre-amplifier response to the principal elements of the target scene and combined as a composite signal from this simulated portion of the sensor. The scene elements are defined by: 1) Response to the target as a function of 2 relative sightline angles, target aspect angle, and relative range; 2) Response to countermeasures as a function of one relative sightline angle and relative range; 3) Response to background as a function of 2 inertial sightline angles; 4) Atmospheric transmission coefficient as a function of relative range and altitude. The entries for the interpolation are determined from flight vehicle, sensor, and target motion variables that are determined in the AD100 and sensor scanning and signal processing variables simulated in the AD10. The AD10 and AD100 communicate at high rates through dual-ported memories on a frame to frame basis to provide the inputs for very high speed sensor response interpolation. The AD10 design was specifically optimized for table interpolation an its performance on this function is without equal. The AD10 is a set of tightly coupled high-speed parallel pipelined processors that operate on 16 bit data. This data format is more in keeping with the target scene data as the intensity data from sensors is quantified much less than this.

Target scene simulation is required for the third option of sensor-in-the-loop simulation. This requires the generation of a video bandwidth signal of the encoded energy distribution of the simulated target scene. A Pixar image computer provides the processing throughput and bus bandwidth to dynamically update a target scene from stored scene components as a function of inertial positions determined in the AD100. The scene components being generated by image processing off-line and transferred to the Pixar during initialization. A key decision in the selection of components to implement the third option for sensor simulation involving scene generation is the selection of the video format standard. The finite resolution and synchronous nature of the frame format presented to the sensor can lead to artifacts in the response of the sensor, especially if the sensor signal processing is in some sense differentiating elements of the observed scene. The conventional National Television Standards Code (NTSC) video format (also known as RS-170) callsfor a data frame of 480 lines of 525 element resolution being presented in an interlaced fashion at 60 fields a second. These interlaced fields contain either the 240 odd or even lines of a frame, so the total 480 line frame is presented 30 times a second in two successive interlaced fields. The RS-343 standard provides for increased lines per frame and more resolution elements per line (both in excess of 1000) however this limits the use of commercially available video equipment (cameras, recorders, monitors, control panels, mixing, special-effects, etc.). Many computer generated raster graphic displays exceed these standards by providing more lines, higher resolution, and higher frame rates without interlacing, however these displays are usually conventional cathode ray tubes driven by non-standard video interfaces. Use of these video displays is potentially acceptable as a medium for the presentation of a target scene to a sensor when the spectrum for the sensor is near the visible spectrum, very bright elements (countermeasures, sun glints, etc.) are not part of the scenario, and the combination of frame spatial resolution being high enough and tube phosphor time constant being slow enough is such that artifacts in the sensor response are not excited.

The techniques for theater projection of video signals offer potential to overcome these limitations by employing a video signal to modulate a deformable reflective surface for radiant energy produced by an intense source (over 4000 lumens). The xenon arc lamp is used as a source for visible or near infra-red energy, but the Nernst glower can generate intense radiant energy over the band of interest for a number of infra-red sensors. A key issue for further examination are the compatibility of the method of generating the deformable surface (scanning an oil film with an electron beam modulated by the video signal) with the longer wavelength energy from the glower and the spatial and temporal resolution necessary to avoid artifacts with the sensor and its signal processing.

Fortran Server -- This computer is provided to handle the portions of the simulation problem that are either difficult to implement on the specialized computers or have already been implemented in Fortran and represent an investment that either cannot be replaced (nobody really understands the "rats nest of old code" but the experts believe the results) or the cost in time to re-program the code is out of the question. These types of problems can involve real-time processing that would be best handled by the simulation server but due to the above mentioned reasons cannot be re-programmed, but they are more likely to impact the initialization phase rather than the real-time mode and typically are involved in the generation of aerodynamic tables, mass properties, atmospheric properties, fuse function, warhead effects, etc. A number of this class of computing problems have been coded for the Cray supercomputers, taking advantage of the high-speed and high-precision to numerically evaluate Lagrangian or Eulerian integration of non-linear partial differential equations over finite element grids. These codes will probably run on the host MicroVAX but the problem is that they are long running on a general purpose computer and become the principal delay in the use of a very high performance simulation system. The addition of a high-speed high-precision Fortran server can have a major impact in the turn-around time for setting-up runs on the system. The requirement is to include a computer that runs "Cray code" in a reasonable elapsed time (and a reasonable cost) has high-speed open bus I/O, on-line control, and is compatible with the VMS nature of the communication server. A significant feature of the CTSS operating system for Cray computers is a process recovery feature. This allows special-user processes to seize the processor for high-priority runs and then return control to numerous other users with no loss of data. An important aspect of the Cray architecture is the superposition and integration of a scalar (SISD) and an array (SIMD) processor on the same bus and memory structure. Several computer manufacturers offer systems that have been designed along these lines with some actually copying the Cray instruction set architecture but implementing the hardware on more affordable electronic components that can be operated in the laboratory environment. The SCS40 from Scientific Computer Systems meets these requirements as its architecture is a clone of the Cray XMP/24, it executes public domain software, and operates as an attached processor to a VAX host.

Graphics Server -- On-line high-resolution graphics displays of both simulated and hardware-in-loop variables are produced by an Evans & Sutherland PS350 graphics station which performs graphics computing in an attached processor to the VAX host. However with the Aptec communications server this graphics station is actually attached to an Aptec programmable controller that mimics the VAX interface in "transparent mode" and provides data at the limit of the E&S interface during "program mode". Like the other systems attached to the Aptec the PS350 is actually a complex of processors working together and dedicated to a particular type of processing. The task of generating on-line graphics and subsequent hard-copy of the display is simplified by E&S graphics support software. This allows the use of the graphics control processor during initialization and termination of a real-time run and bypasses this processor during the real-time portion to improve throughput. This graphics control processor is commanded by Fortran subroutines from a support software library that E&S provides for VAX host computers. Subsequent hard-copy of the on-line display or off-line post-run graphics hard-copy of data saved on the run log-file can be easily accessed by multiple users via network communications on the VAX host computer.

Telemetry Server -- Collecting data from the test hardware-in-the-loop requires telemetry processing. A Fairchild Weston EMR 8715 on-line telemetry processor provides a modular capability to gather selected variables from the telemetry stream, tag the data quality, smooth the data, and transform the values to engineering units on the fly. Thus the on-line data logging can include hardware based data combined with simulated data so that direct comparisons and causality can be examined. The EMR 8715 is a complex of modular processors than can be flexibly re-configured for a variety of formats and data rates, with parallel or pipelined processing. This system is supported by MicroVAX host resident software that allows the user to dynamically configure the hardware and on-line processing.

IMPLEMENTATION

Figure 3 presents an element interconnect block diagram of the simulation system. The upper right hand portion of the diagram illustrates the MicroVAX host computer and its associated open bus peripherals. Note that the AD100, Aptec, and SCS40 all have parallel interfaces. The AD100 interface connects the host to the supervisor processor for initialization and on-line control of the AD100. The SCS40 interface is a DR11 emulator used for concentrated terminal I/O and file transfers to the host. The Aptec host I/O controller (HIA) is a Unibus interface that operates through a Unibus to Q-bus converter in a Unibus BA-11 module. The Aptec computer consists of the row of I/O controllers (DIA's, DIP's), connected to both the Unibus and the Aptec data interchange bus. The Aptec high-speed scatter/gather access memory (MEM) appears to the VAX host as an RMS disk device and data is stored here during initialization by the host using VAX Fortran statements for file output. The STAPLE code device drivers and procedures are executed by an interpreter in the local memory of the I/O controllers during the real-time mode of operation. The I/O controllers also have micro-coded procedures stored in their local memory that are called by the STAPLE procedures to perform the time intensive functions, so that high throughput can be achieved.

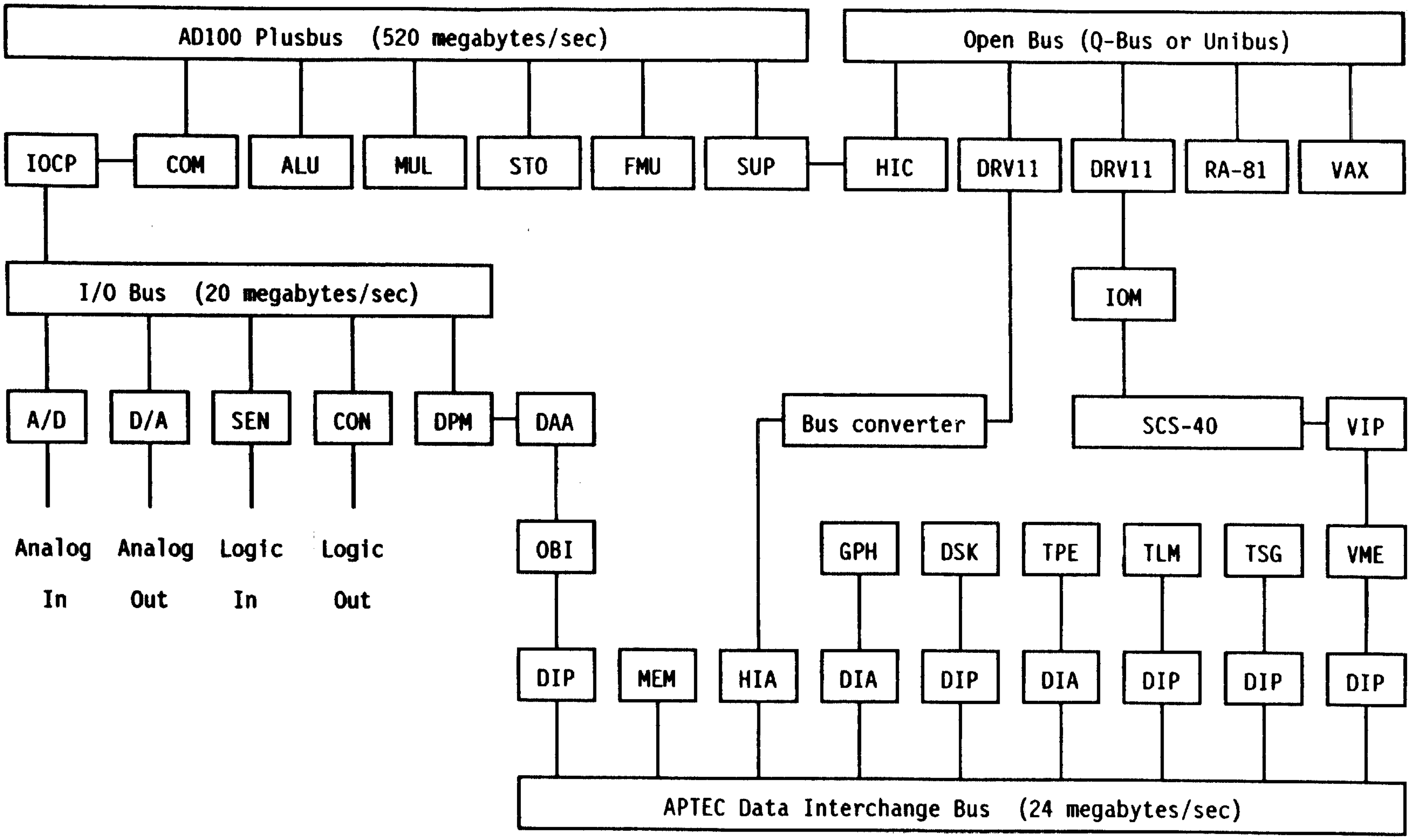


Figure 3. Simulation Interconnect Block Diagram

The I/O controllers connect to the specialized elements of the simulation system through either a DEC Unibus interface or an Aptec private bus interface. The private bus interface can be driven at nearly four times the data rate of the Unibus, but the throughput depends on the total chain of devices in the path, so the Unibus interface may be adequate if the port controllers on the specialized elements are not able to support the higher data rate. The IOCP and DPM on the AD100 and the Ibis disk controller can both support private bus rates, whereas the GPIO controller on the PS350 supports Unibus rates. The IOM controller on the SCS-40 supports Unibus rates whereas its VIP processor supports VME bus I/O and private bus rates. The EMR 8715 (TLM) and the VLDS (TPE) will support either Unibus or VME bus interfaces, but the data rates are limited to that of the Unibus.

The maximum I/O rates are required for data transfer from the AD100 and EMR 8715 to the Ibis disk in the real-time mode, as this is the data logging path with the objective of saving as much data as possible. Data transfer to the PS350 (GPH) and Pixar (TSG) are much lower with the PS350 requiring at most 24 data sets (each involving five 16 bit transfers) every nth frame of the real-time run, where n depends on Nyquist sampling considerations for the particular simulation. The Pixar, when used for on-line real-time target scene simulation requires the following slow moving parameters from the AD100 (they are functions of the inertial position of the scene elements): 1) relative range, target aspect angle, and two inertial reference angles from the target; 2) two inertial reference angles from the background; 3) relative range with one inertial reference angle from each countermeasure that can affect the sensor; 4) atmospheric transmission coefficients for each of these elements.

Figure 3 also presents a block diagram of the ADI interface to the Aptec. The ADI high speed adapter (DAA) interfaces the dual-ported memories (DPM) in the I/O rack to an Aptec open bus interface (OBI) which in turn interfaces to an Aptec high-speed DIP programmable controller. This path can support the high speed private bus data rates. Figure 4 displays a block diagram of the modules of FORTRAN, ADSIM, ADRIO, and STAPLE code that control the execution of the combined system and the flow of data. Frame data is buffered in the COM memory by ADSIM region sync5 code and flags control the transfer of the frame buffers to larger buffers in the DPM by ADRIO main-loop code executing on the I/O control processor (IOCP). Initialization of the communication between the AD100 and IOCP is performed in ADSIM region sync3 and ADRIO pre-run modules while termination occurs in ADSIM region terminal module which signals ADRIO main-loop to terminate. FORTRAN sections start and finish control the execution of the ADSIM code and also performs graphics initialization and termination for the PS350 in the transparent mode.

Figure 5 presents a software block diagram of the Aptec STAPLE code for the I/O controllers (IOC's). The Aptec code is initialized by the FORTRAN routine Aptec-init which is called from the FORTRAN section start. Aptec-init initializes the STAPLE programs for the DIA's controlling the DPM and PS350 interfaces as well as the DIP controlling the Ibis disk interface. Data is transferred from ADSIM region sync5 to the STAPLE main-loop code where it is buffered in the Aptec ram memory. The PS350 DIA gather reads specified data from the buffer for display on the screen and the Ibis disk DIP transfers the entire buffer to the disk for long-term storage.

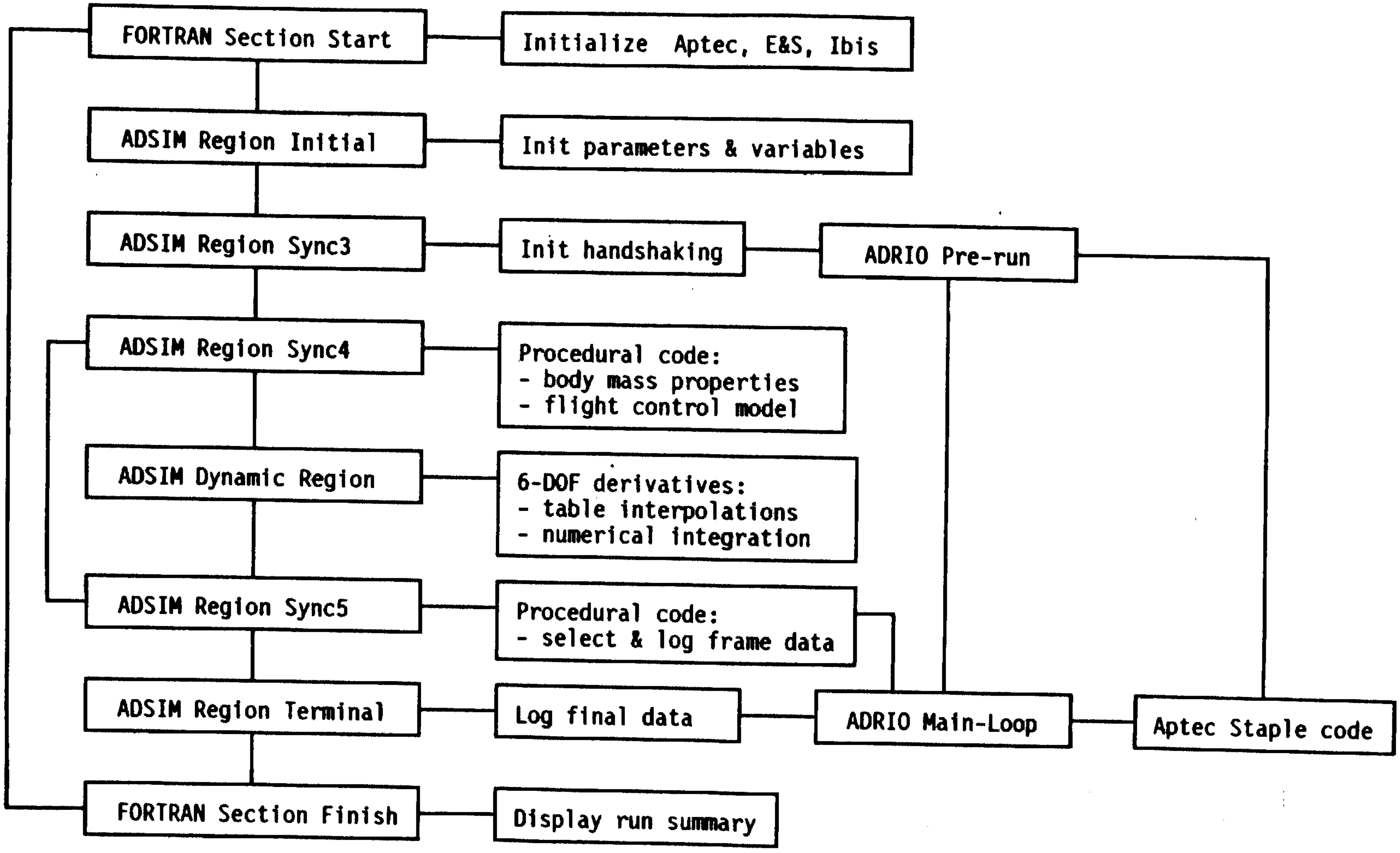


Figure 4, Run Control for the Combined System

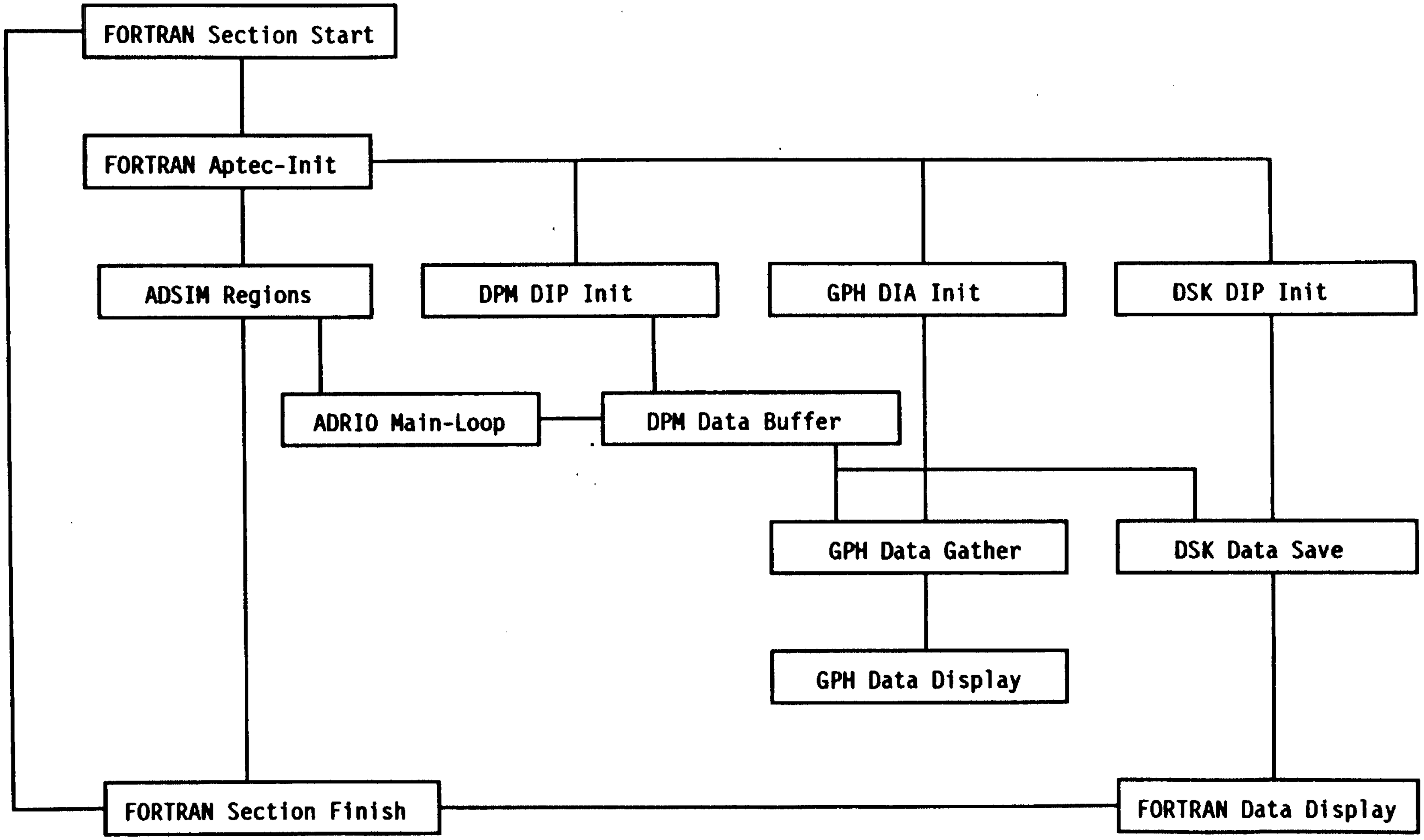


Figure 5, Aptec Software Block Diagram

OPERATIONAL USE

Use of this simulation facility in a typical hardware-in-loop test involves five phases; pre-run, initialization, real-time, termination, and post-run. These phases are described here for illustrative purposes.

Pre-run -- This phase of operation is concerned with the generation of data tables for the interpolation in the AD100 and AD10, and target scene elements for the scene generation in the Pixar. The SCS40 is used to generate the aerodynamic and mass property tables for the ADSIM program on the AD100. These tables are then transferred to the host VAX and processed by the FUNGEN utility provided by Applied Dynamics to generate interpolation tables in the proper format for the micro-coded routines on the AD100. When an AD10 is used to simulate the sensor "front end" the SCS40 is used to generate scene element tables that are also transferred to the host and processed by the INPBDA utility for the assembly coded routines on the AD10. These tables are loaded through the host interface during the initialization phase.

Initialization -- This phase of operation is controlled by the host processor with the I/O controllers for the AD100 and PS350 initially in the transparent mode. The host attaches and loads the AD100 through the Q-bus supervisor interface, whereas the PS350 is initialized with Fortran subroutines through its parallel interface. The final step of this phase is to open Files-11 format data files in Aptec memory and on the Ibis disk for subsequent data storage during the real-time phase, and write files in Aptec memory for the STAPLE procedures and control parameters (addresses, pointers, word counts, offsets, scaling factors, etc.) that the I/O controllers require in the real time mode. Control is then transfered to an ADRIO program in the IOCP of the AD100, which monitors the AD-I/O equipment for the real-time start signal that initiates the execution on the AD100. The I/O controllers then begin processing their real-time programs, which check for control flags to change value and indicate the availability of data to be handled.

Real-time -- The IOCP starts the real-time run of the AD100 when it receives the real-time start signal from an external switch attached to the AD-I/O. The AD100 processes a specified number of simulation frames and sets communication control flags with the IOCP so it can move data from COM memory to the DPM of the AD-I/O. When a buffer is ready for transfer to the Aptec, control flags are set for the I/O controller attached to the DPM. This controller can then move data from the DPM buffer to an Aptec memory buffer and then set flags to indicate that this buffer is ready for the controller attached to the PS350 GPIO to gather-read data from this buffer. The PS350 controller can then scale and offset the selected data, according to the parameters stored in Aptec memory, and transfer this data to the PS350 main memory for on-line display. Thus the AD100 controller moves data in response to ADRIO generated control flags and sets flags for the PS350 controller to select, format, and transfer graphics data, and also for the Ibis disk controller to transfer the entire buffer to the log-file on the disk.

Termination -- Termination results from either an external switch on the AD-I/O or the end-run condition being satisfied on the AD100. This results in the present simulation frame being the last frame to be generated on the AD100 and the execution of the "terminal" region of ADSIM code and the "finish" Fortran section on the host. The terminating phase places the last frame in the COM buffer and sets flags so that the IOCP can transfer the terminal parameters to the DPM and signal the Aptec AD100 controller that the another buffer is ready for transfer. The IOCP will then wait for the Aptec to signal it has completed the handling of this last buffer before it signals the Aptec to shut-down processing. This avoids a pre-mature shutdown and loss of the last buffer of data. The Aptec controllers then return to the transparent mode for the initialization of the next run.

Post-run -- This phase of operation is concerned with the analysis and interpretation of on-line graphic data and the generation of off-line additional graphic and tabular data from the log-file to document the test. The log-file may then be archived to the VLDS tape for long-term storage, thereby providing adequate space on the Ibis disk for the generation of large contiguous block files that are necessary for the high throughput real-time data capture.

SUMMARY

Phased implementation can be accomplished by initially integrating the AD100, AD10, and PS350 into the VAX host, then adding the Aptec controller, Ibis disk, motion simulator, and SCS40. Initial operation for application software development typically can proceed prior to obtaining real-time I/O capability with the Aptec. Then as test hardware availability dictates, the simulation of hardware function is replaced by physical operation and the simulation equipment concentrates on more detailed models of the environment that the hardware experiences.

The phased insertion of sensor hardware into the loop certainly depends on the nature of each particular sensor, its signal processing, and also on the mission scenarios, which dictate target scene generation. It may prove impractical to simulate each detector's response for sensors with large focal plane arrays of detectors, and the logical second step for this type of system may be target scene generation for the optics and focal plane. This may also prove to be the case for systems involving reticle modulation of the target scene. In the same sense, when surrogate detectors are not credible, low temperature background and target scenarios can prove difficult to present to sensors, so that scene generation for these scenarios may not be practical.

REFERENCES

Applied Dynamics International. Dec. 1986. "Applications Summary, Structural Dynamics Benchmark Simulation". 3800 Stone School Rd., Ann Arbor, MI 48104.

N90-23063

Man-In-the-control-loop Simulation of Manipulators

J. L. Chang, T. C. Lin, and K. H. Yae

Center for Simulation and Design Optimization
Department of Mechanical Engineering
The University of Iowa
Iowa City, Iowa 52240

ABSTRACT

A method to achieve man-in-the-control-loop simulation is presented. Emerging real-time dynamics simulation suggests a potential for creating an interactive design workstation with a human operator in the control loop. The recursive formulation for multibody dynamics simulation is studied to determine requirements for man-in-the-control-loop simulation. High speed computer graphics techniques provides realistic visual cues for the simulator. Backhoe and robot arm simulations are implemented to demonstrate the capability of man-in-the-control-loop simulation.

1. INTRODUCTION

Man-in-the-control-loop simulation has been used in vehicle design and pilot training. The first visual vehicle simulation was done by Sheridan, Paynter, and Coons in 1964 [1]. In the early 1970's, Volkswagen built a motion-base driving simulator to test the influence of vehicle parameters on safety [2]. General Motors had its own fixed-base driving simulator to test driver-vehicle performance in the late 1970's [3-5]. Daimler-Benz built an advanced driving simulator in 1984 [6-7]. This system has a 180-degree-wide image on a dome, and its motion platform allows a driver to use his peripheral vision in a natural way and to make judgements about handling qualities of new vehicle designs.

The flight simulator arrived at its modern form at the end of the 1960's. Fixed-base simulators with simple dynamics models were established before then. The flight simulators have assumed an increasingly crucial role in the development and testing of new product lines in the aviation industry. For instance, the Boeing Flight Systems Laboratory has its advanced flight simulators [8]. They are used to design, develop, and validate a newly developed aircraft as well as to train pilots.

Man-in-the-control-loop simulation has an enormous potential for dynamic system design. The simulator provides realistic environmental cues to allow the operator to control the simulated system, as he would in the real world. This simulator provides a very natural and realistic simulation environment for the teleoperation. It also provides a low-cost environment for man/machine interaction, thus leading us to human factor research. With the recent development of a highly efficient multibody dynamic modelling method, a quantum leap in computer hardware and software, and a substantial cost reduction in high-speed computer graphics, the real-time man-in-the-control-loop simulation appears to be within our reach.

In comparison to the conventional design procedure that typically requires a long development time from a prototype to the final product, "simulation-aided design" appears to be an efficient design tool that would make it possible that the dynamic characteristics of a system are predicted in the early design stage. The designer can observe the behavior of a mechanical system from the simulation and can correct any design flaw even before the fabrication and test of prototypes. At the same time, the operator can provide valuable feedback to the designers about the system's characteristics that require further improvement. Therefore, the quality of the system can be improved and the cost of prototype fabrication and test can be reduced. In this way, the machine can be ergonomically and economically designed to be adapted to the human, instead of forcing the human to adapt to the machine after the final design.

A low-cost network-based simulator is used in this research. In this simulator, the dynamic simulator in a parallel-processing computer is integrated with high-speed computer graphics through the Network Computing System (by Apollo Computer, Inc.). And the operator comes in the loop through a serial port in the IBM PC that digitizes operator's control action on the joysticks. Here, we are focusing on two special systems: backhoe and robot arm. The backhoe has four degrees of freedom with a human operator in the loop all the time. Thus it makes a perfect example for this research. The same technique is applied to teleoperation in which the seven degree-of-freedom robot arm is controlled by the human operator in the loop.

2. METHODOLOGY

The real-time dynamic simulation demands a more efficient and accurate mathematical formulation. The recent development in the recursive formulation, in which the relative coordinates are used to yield a minimal set of differential and algebraic equation, makes the computation much more efficient when implemented with the parallel-processing algorithms. The numerical results from the simulation then have to be displayed in real-time. The quality of the graphics at the same time must be high enough to provide the human operator with high-fidelity realism. The operator's workstation that encompasses a host computer, a graphics computer, a control console, and interfaces between each other is introduced in this section.

2.1 NETWORK-BASED SIMULATOR

Figure 1 shows the setup of the network-based simulation. Two computers, an Alliant eight-processor computer and an Iris silicon graphics computer, are used in this simulation. They communicate through the Network Computing System(NCS). We use real-time dynamics formulation and Visualization of Dynamics System(VDS) to simulate and display the system. The dynamics code is in the Alliant/FX8 and the graphics package is in the Iris graphics workstation. The operator gives the control input by using a pair of joysticks, thus initiating the simulation. The joystick interface samples and digitizes the signal from the joysticks and sends it to VDS through serial communication. VDS sends the input command to the dynamics program. Then the dynamics program simulates the system and sends the updated position and orientation to VDS. VDS displays graphics either on the 19" Iris screen or on a large projection screen. Finally, the operator gets the visual feedback and issues a new command to complete the simulation cycle. In this application, a pair of joysticks is used to drive the mechanical systems. For the backhoe, each joystick has two axis controllers. The two joysticks have four controllers to drive four degrees of freedom of the backhoe. For the robot arm, each joystick has three axis controllers. Thus, six degrees of freedom of the robot arm can be driven by two joysticks.

2.2 DYNAMICS MODELLING

The absolute coordinate and relative coordinate are used to derive equations of motion for mechanical systems; however, the absolute coordinate is not suitable for high speed simulation due to its inefficiency. The recursive formulation of relative coordinates is applied to derive equations of motion to achieve high speed simulation. Haug and McCullough developed a systematical approach to derive equations of motion by using a variational-vector calculus formulation [9]. Bae and Haug employed the cut joint method and variational equations of motion to derive the recursive Newton-Euler equations of motion for constrained mechanical systems [10-11]. Recently, Bae, Hwang, and Haug refined this work by introducing a state vector notation [12]. This approach simplified the derivation and reduced the computing time for numerical simulation in certain classes of applications. Thus it is suitable for real-time simulation. The resulting form of the equations of motion from recursive formulation can be expressed as

$$\begin{bmatrix} \mathbf{M} & \mathbf{\Phi_q}^T \\ \mathbf{\Phi_q} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{q}} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathbf{Q} \\ \boldsymbol{\gamma} \end{bmatrix} \qquad (1)$$

where the mass matrix $\mathbf{M}$ is obtained from the inertia properties of bodies and $\mathbf{\Phi_q}$ is Jacobian submatrices of cut joint constraints. $\mathbf{Q}$ is the vector that evaluates the inertia force and external force, and $\boldsymbol{\gamma}$ is the right side of constraint acceleration equations. A linear solver is used to obtain $\ddot{\mathbf{q}}$ that includes the acceleration vector of base body and the relative joint accelerations, and the Lagrange multipliers $\boldsymbol{\lambda}$. Cartesian accelerations of bodies can be recovered forward from the base body to tree end bodies. Relative joint velocities and accelerations are numerically integrated to obtain relative joint coordinates and velocities for the next time step.

The joint relative coordinate is used to derive the equations of motion for the backhoe and robot arm. The backhoe is the first application. For this given model, we chose swing tower, boom, dipper, and bucket as major components. The joint definitions and local vectors are shown in Figure 2. This backhoe is driven by hydraulic actuators. The computer model of robot arm is shown in Figure 3. The robot arm has seven major components and seven joints. Four of the seven joints are roll joints, the other three are pitch joints. So we have roll-pitch combinations. We used the same technique as in the backhoe; the only difference is that the robot arm is driven by a servo motor and harmonic drive.

The execution time is the major concern because of the real-time constraint. Figure 4 shows the relationship between execution time per one function evaluation and the number of processors for backhoe simulation. The straight line shows the largest integration stepsize. Below this line, the execution time is faster than real-time. The execution time for two processors is very close to real-time. Therefore, if we use more than three processors, we can achieve real-time or even faster than real-time. Meanwhile, fine grain parallelism is used to tune the dynamics program. The utility factor for four processors is 60%, but it drops to 40% for eight processors. That is because the backhoe system is too small: even though we assigned eight processors to the simulation, they were not fully utilized.

2.3 COMPUTER IMAGE GENERATION

The computer image generation plays an important role in man-in-the-control-loop simulation. Since computer graphics generates environmental cues, it should be realistic to the extent that the operator could have a similar experience with the simulator as he would with the actual system. It is required to provide the operator with the essential visual information in the simulator. The visual information should include a general perspective view of system, color of objects, surface texture, shading, and lighting. The graphics animation displays spatial position and orientation of bodies in a system at a high frame rate. Recently, the low cost simulation goal has become more realizable, owing to hardware and software improvement in graphics computers. Dubetz, Kuhl, and Haug presented an approach for interactively animated graphics for real-time dynamic simulation [13]. They implemented a network between workstation and host computer to display simulation graphics both in an interactive way and in a batch mode. Dubetz developed an interactive graphics package called the Visualization of Dynamic Systems that is capable of animating three-dimensional multibody systems with real-time rendering, viewing, and lighting operations [14].

VDS is used to generate realistic three-dimensional graphics either in batch mode or in interactive mode for man-in-the-control-loop simulation purposes. Three groups of data support VDS to display graphics: they are geometric data, visualization data, and system state data. Geometry data is used to describe geometry of components in their own fixed reference frame. Visualization data defines light, color, shading, texture, center of projection, view reference point, and so on. Both are pre-defined once for one system. System state data is defined for each frame, so it should contain the updated position and orientation of components in the interactive display. In the batch mode, we have to first create all the frames and then display. In the interactive mode, we create and display one frame after another. Parameters that are modified at a VDS interface and sent to a simulation server are called "valuators". Parameters that are modified by the simulation server and set to a VDS interface are called "displays". The valuators are controlled by the interface that VDS provides, such as knobs, dials, and mouse. In this way, VDS allows the user to display or control the simulated system interactively.

Using the batch-mode VDS, we obtained the realistic computer graphics of the backhoe. It has fourteen hundred polygons and eighteen bodies. Then we found that the display speed was too slow, about seven frames per second. For high speed display purposes, we reduced polygons to five hundred. As a result, we were able to display ten frames per second on Iris 4D/70 and twenty frames per second on the upgraded Iris, which has two processors.

In this research, VDS is the main controller for data communication and graphics display. VDS controls the operator's input, dynamics simulation, and graphics display. A couple of pictures from the interactive graphics display are shown in Figure 5. The pictures show that the backhoe pick up and dump an object. That is the quality we had in the interactive simulation. However, slow graphics display and network time delay made it difficult at this point to assess the realism of the simulated operation.

3. SUMMARY

The recursive dynamic formulation with parallel processing algorithms is capable of simulating dynamics of a mechanical system in real-time or sometimes even faster than real-time. But when it is integrated with high-fidelity graphics and an operator in the loop, the overall performance is not up to real-time yet. The slowdown was mostly caused by slow graphics display and communication time delay. To resolve these problems, the structure of the simulation is changed from network-based to workstation-based as shown in Figure 6. This new setup with one parallel-processing computer that houses both the dynamic simulation and graphics will be eventually able to eliminate the network time delay. And the visualization system is continually upgraded to increase display speed. The simulation cycle of the new setup is similar to that of the workstation-based simulation, except that the dynamic simulation, instead of VDS, controls the overall process.

In addition, the realism of the simulator will be enhanced by adding different types of feedback such as motion, auditory, and tactile feedback. When the actual operator console is linked up with the backhoe simulation and the Kraft mini-master with the robot arm simulators these simulators will possess a great potential for human factor research.

REFERENCES

1. Sheridan, T. B., Paynter, H. M., and Coons. S. A., "Some Novel Display Techniques for Driving Simulation," IEEE Transaction on Human Factor in Electronics, Vol. HFE-5, pp. 29-32, September 1964.

2. Richter, B., "Driving Simulator Studies: The Influence of Vehicle Parameters in Safety in Critical Situations," SAE Technical Paper, No. 741105.

3. Wierwille, W. W., "Driving Simulator Design for Realistic Handling," Proceedings of the Third International Conference on Vehicle System Dynamics, Swets and Zeitlinger, B. V., Amsterdam, 1975, pp. 186-199.

4. Repa, B. S. and Wierwille, W. W., "Driver Performance in Controlling a Driving Simulator with Varying Vehicle Response Characteristics," SAE Technical Paper, NO. 760779, October 1976.

5. Repa, B. S., Alexandridis, A. A. and Howell, L. J., "Study of Vehicle Steering and Response Characteristics in Simulated and Actual Driving," SAE Technical Paper, No. 780011, 1978.

6. Drosdol, J., and Panik, F., "The Daimler-Benz Driving Simulator: A tool for Vehicle Development," SAE Technical Paper, No. 850334, 1985.

7. Hahn, S., and Kalb. E., "The Daimler-Benz Driving Simulator Set-Up and Results of First Experiments," 1987 Summer Computer Simulation Conference Proceeding, pp. 993-998, 1987.

8. Fraser, T. L., and Phillips, C. E., "Engineering Flight Simulation-A Revolution of Change", SAE Technical Paper, No. 851901.

9. Haug, E. J., and McCullough, M. K., "A Variational-Vector Calculus Approach to Machine Dynamics," Journal of Mechanisms, Transmissions, and Automation in Design, Vol. 108, pp. 25-30, March 1986.

10. Bae, D. S., and Haug, E. J., "A Recursive Formulation for Constrained Mechanical Systems, Part I - Open Loop," Mechanics of Structures and Machines, Vol. 15(3), pp. 359-382, 1987.

11. Bae, D. S., and Haug, E. J., "A Recursive Formulation for Constrained Mechanical Systems, Part II - Closed Loop," Mechanics of Structures and Machines, Vol. 15(4), 1987.

12. Bae, D. S., Hwang, R. S., and Haug, E. J., "A Recursive Formulation for Real-Time Dynamic Simulation," submitted to Journal of Mechanisms, Transmissions, and Automation in Design.

13. Dubetz, M. W., Kuhl, J. G., and Haug, E. J., "A Network Implementation of Real-Time Dynamic Simulation with Interactive Animated Graphics," submitted to Journal of Mechanisms, Transmissions, and Automation in Design.

14. Dubetz, M. W. , et al., User Documentation for Visualization of Dynamic System, Version 1.0, Center for Simulation and Design Optimization, The University of Iowa, 1988.

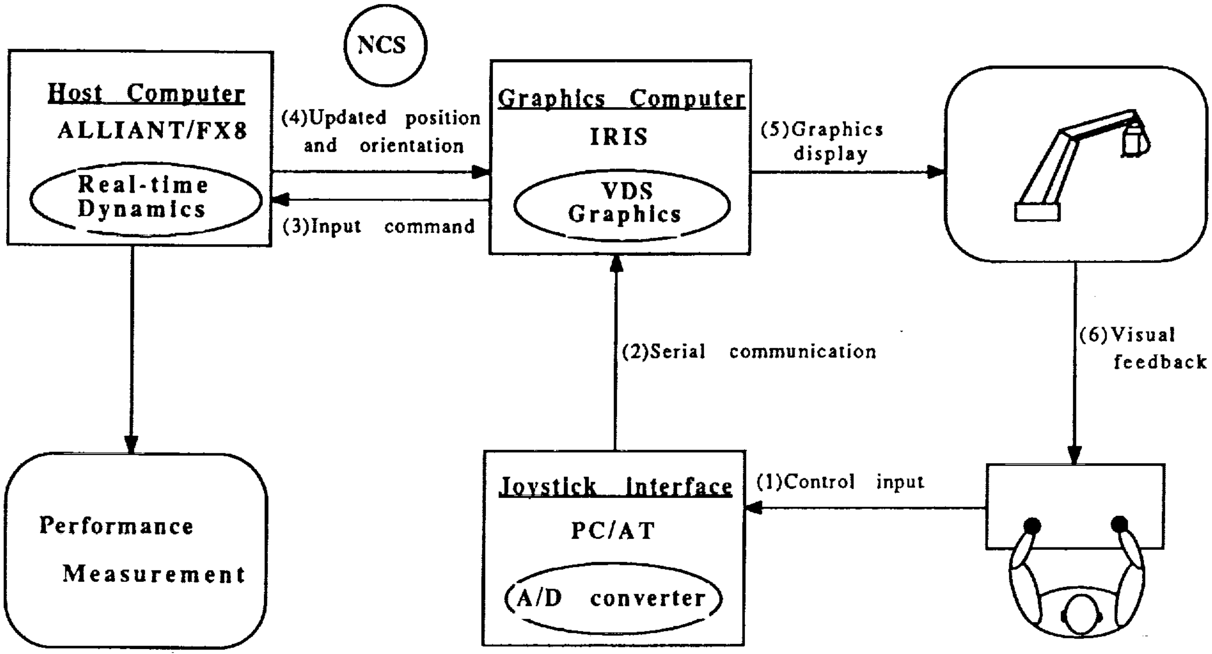


Figure 1. Network-based simulator

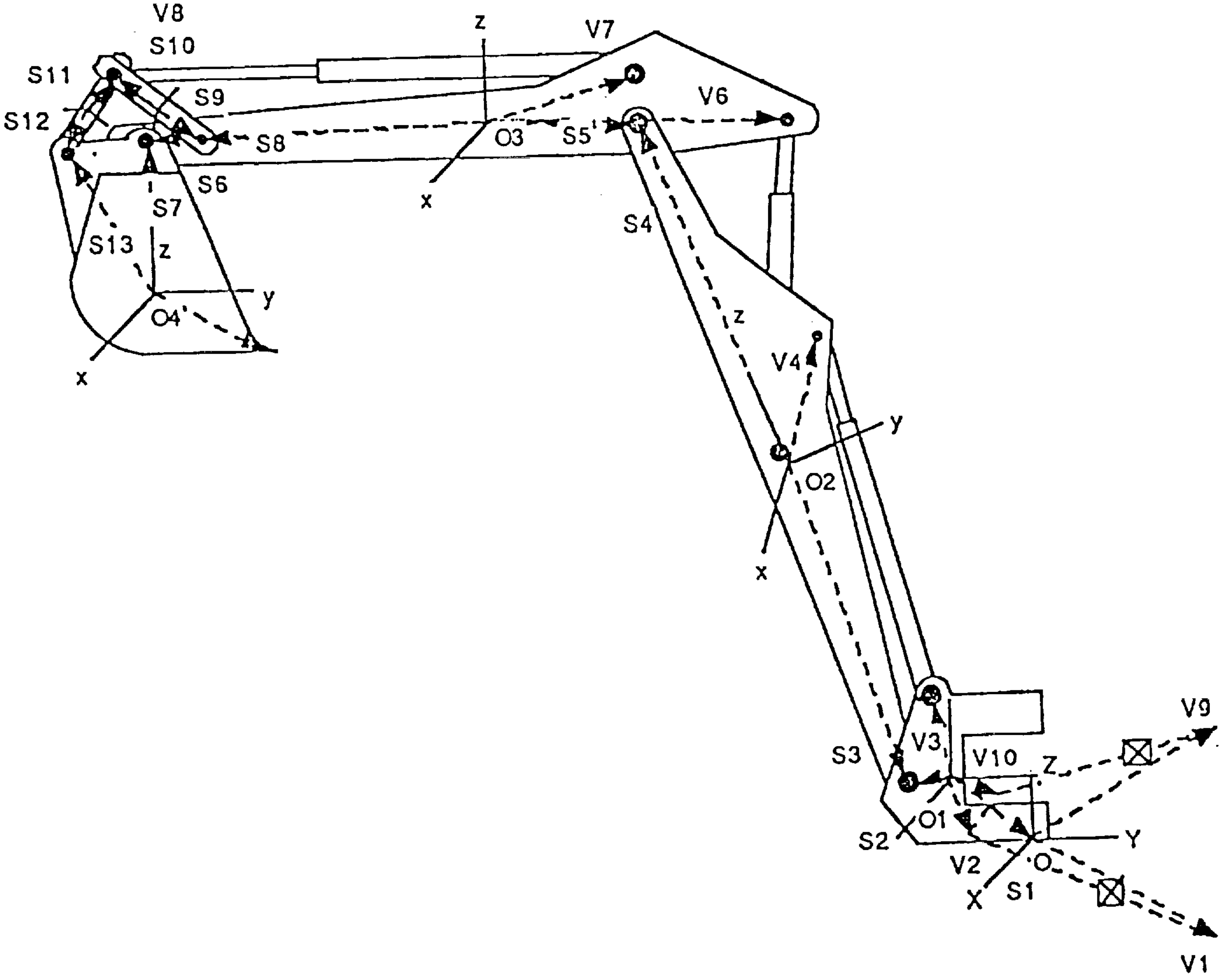


Figure 2. Computer model of backhoe

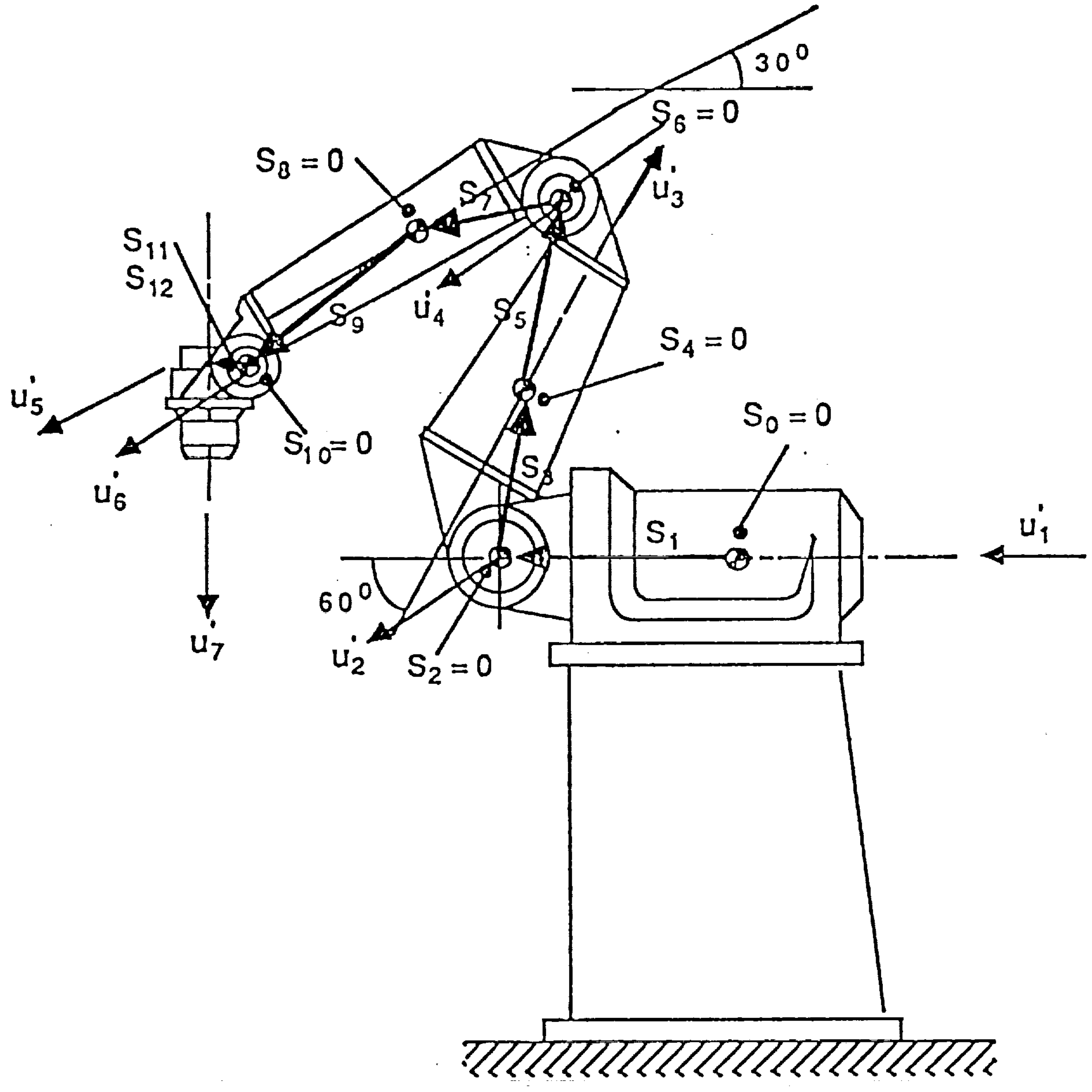


Figure 3. Computer model of robot arm

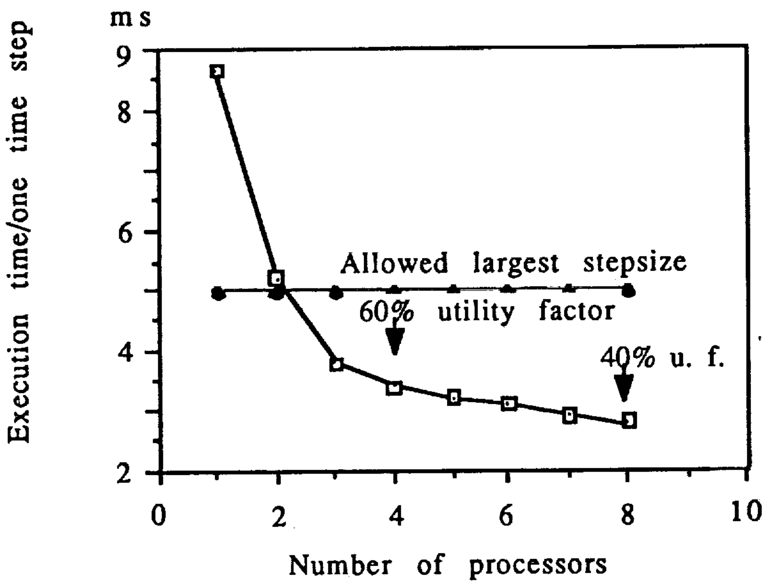


Figure 4. Execution time vs. number of processors for backhoe simulation

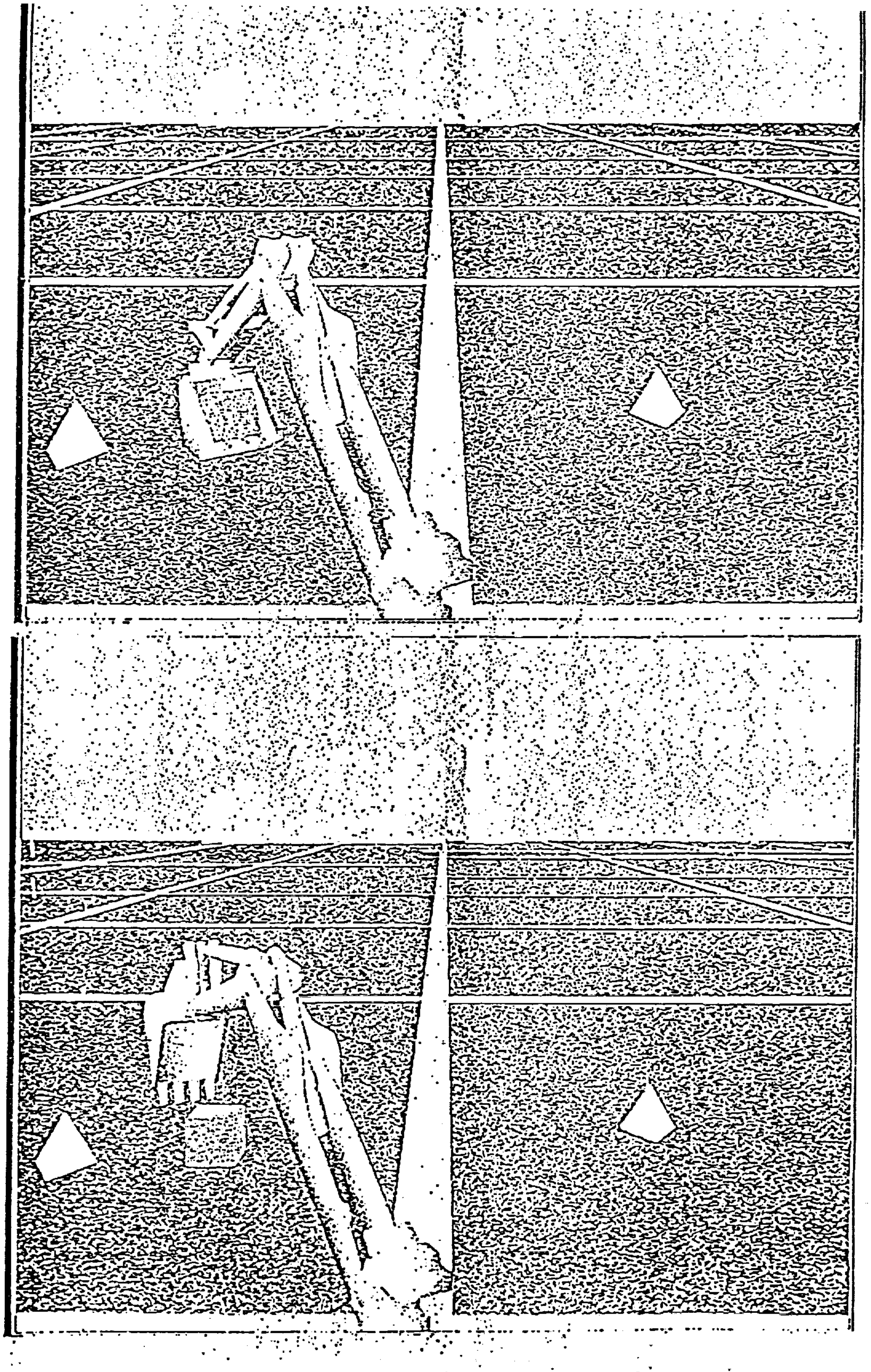


Figure 5. Interactive graphics display of backhoe

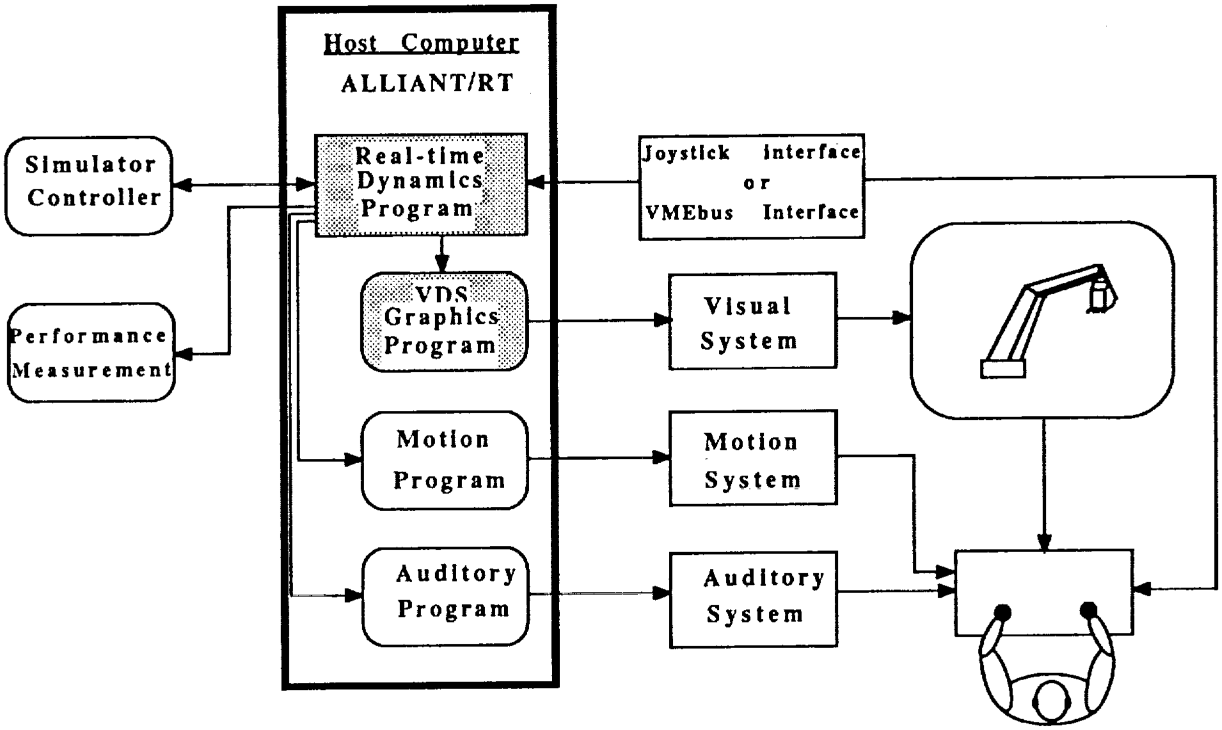


Figure 6. Workstation-based simulator

A New Second-order Integration Algorithm for Simulating Mechanical Dynamic Systems

R.M. Howe

Department of Aerospace Engineering, The University of Michigan, Ann Arbor, Michigan
and
Applied Dynamics International, Ann Arbor, Michigan

ABSTRACT

A new integration algorithm which has the simplicity of Euler integration but exhibits second-order accuracy is described. In fixed-step numerical integration of differential equations for mechanical dynamic systems the method represents displacement and acceleration variables at integer step times and velocity variables at half-integer step times. Asymptotic accuracy of the algorithm is twice that of trapezoidal integration and ten times that of second-order Adams-Bashforth integration. The algorithm is also compatible with real-time inputs when used for a real-time simulation. It can be used to produce simulation outputs at double the integration frame rate, i.e., at both half-integer and integer frame times, even though it requires only one evaluation of state-variable derivatives per integration step. The new algorithm is shown to be especially effective in the simulation of lightly-damped structural modes. Both time-domain and frequency-domain accuracy comparisons with traditional integration methods are presented. Stability of the new algorithm is also examined.

1. Introduction

In the simulation of mechanical dynamic systems described by ordinary differential equations the required dynamic accuracy is often modest, especially when the real-time computation is utilized as part of a hardware-in-the-loop simulation. Accuracies ranging between 0.1 and 1 percent are considered adequate in many cases. For this reason, lower-order numerical integration algorithms are often employed. Also, fixed integration time steps are invariably used in real-time simulations in order to assure that the simulation outputs for each integration step occur at a fixed rate that can be synchronized with real time. In fact, the Adams-Bashforth second-order predictor algorithm, hereafter referred to as AB-2, is perhaps the most widely used method for real-time simulation.

In this paper we consider a modified form of Euler integration which is well suited to the dynamic simulation of mechanical systems. It is especially effective in the simulation of systems with lightly-damped oscillatory modes, such as flexible structures. The method has the simplicity of conventional Euler integration but exhibits dynamic errors that are second order rather than first order in the integration step size h. Also, the dynamic error coefficients associated with the method are smaller than those for any other second-order method. In the next section we introduce the basic concept behind the modified Euler method as it is used in the dynamic simulation of mechanical systems. This is followed by a discussion of dynamic error measures with emphasis on the frequency domain. Several example simulations are then introduced to demonstrate the accuracy improvement achieved when using the modifed Euler method instead of conventional algorithms. The stability boundaries for different versions of the modified Euler method are also compared with those for conventional methods.

2. The Modified-Euler Method

The simulation of mechanical dynamic systems normally requires the integration of a time-dependent acceleration A(t) to obtain a velocity V, followed by a second integration to obtain a displacement D. This is illustrated diagramatically in Figure 1 One method for implementing the required integrations is to use the forward Euler formula for the first integration and the backward Euler formula for the second integration. The required difference equations are the following:

$$V_{n+1} = V_n + hA_n \ , \qquad D_{n+1} = D_n + hV_{n+1} \tag{1}$$

Here h is the integration step size and A_n, V_n and D_n represent the respective variables at the time $t = nh$, where n is an integer. Eq. (1) has been used in real-time simulation to achieve dynamic accuracy improvement over that obtained when using the forward Euler formula for both integrations. In Eq. (1) the first-order error associated with the forward Euler formula cancels the equal and opposite first-order error associated with the backward Euler formula. As a result the displacement D exhibits second-order accuracy with respect to the input acceleration A.

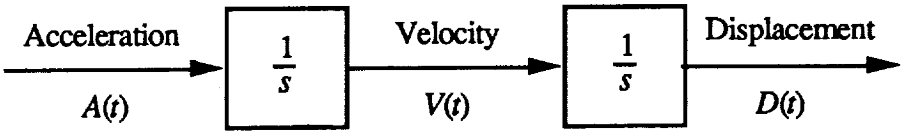


Figure 1. Paired integration to obtain velocity and displacement from acceleration.

Both integrations become second order if we consider the velocity to be represented at a half-integer frame. In this case the difference equations become

$$V_{n+1/2} = V_{n-1/2} + \mathrm{h}\,\mathrm{A}_\mathrm{n} \ , \qquad D_{n+1} = D_n + hV_{n+1/2} \tag{2}$$

The acceleration A will, of course, usually be a function of both velocity V and displacement D, as well as an explicit time-dependent input $U(t)$. In this case we can write the system state equations as

$$\dot{V} = A[D,V,U(t)] \ , \qquad \dot{D} = V \tag{3}$$

where in general the variables will be vectors rather than scalars. In Eq. (3) we see that the acceleration A_n at the nth frame depends on the velocity V_n at the nth frame, which is not available in the half-integer representation for V as utilized in Eq. (2). The best we can do is to employ an estimate $\hat{V}_n$ for V_n based on half-integer values $V_{n-1/2}$, $V_{n-3/2}$, etc. Then the modified Euler difference equations are given by

$$V_{n+1/2} = V_{n-1/2} + hA(D_n, \hat{V}_n, U_n)\ , \qquad D_{n+1} = D_n + hV_{n+1/2} \tag{4}$$

Table 1 lists some possible candidate formulas for estimating V_n. In the first formula in Table 1 we let $\hat{V}_n = V_{n-1/2}$. This is equivalent to using conventional Euler integration rather than modified Euler integration for the V dependent portion of $A(D,V,U)$, with the corresponding dynamic error proportional to h. The second formula for $\hat{V}_n$ uses a linear extrapolation based on $V_{n-1/2}$ and $V_{n-3/2}$. It is equivalent to using AB-2 integration for the V dependent portion of A, with the corresponding dynamic error proportional to h^2. In the third formula $\hat{V}_n$ is derived from averaging $V_{n+1/2}$ and $V_{n-1/2}$. It is equivalent to trapezoidal integration for the V dependent portion of A and represents an implicit formulation, since $V_{n+1/2}$ now appears on both sides of the left equation in (4). Later in this section we will see how this can be turned into an explicit formulation in many cases. Finally, the last formula in Table 1 uses a second-order predictor integration method to obtain $\hat{V}_n$ from $V_{n-1/2}$ and the derivatives $\dot{V}_{n-1}$ and $\dot{V}_{n-2}$. It produces a local truncation error in $\hat{V}_n$ proportional to h^3 and therefore permits the full accuracy of the modified Euler method to be realized.

Table 1. Methods for Estimating $\hat{V}_n$ in $A(D_n, \hat{V}_n, U_n)$

Euler	$\hat{V}_n = V_{n-1/2}$
AB-2	$\hat{V}_n = \frac{3}{2}V_{n-1/2} - \frac{1}{2}V_{n-3/2}$
Trapezoidal	$\hat{V}_n = \frac{V_{n+1/2} + V_{n-1/2}}{2}$
Predictor Integrator	$V_n = V_{n-1/2} + h(\frac{7}{8}\dot{V}_{n-1} - \frac{3}{8}\dot{V}_{n-2})$

Before considering some examples of the application of the modified-Euler methods described here, we consider some dynamic error measures for examining comparitive accuracy of different integration algorithms.

3. Integrator Error Measures

Consider the solution of the state equation $dy/dt = f(t)$ using a numerical integration formula for y_{n+1} in terms of y_n and the derivative f. Furthermore, let $y[(n+1)h]$ and $y[nh]$ represent the exact solution of the continuous system at the times $t = (n+1)h$ and nh, respectively. Then we can then write

$$y_{n+1} - y_n \cong y[(n+1)h] - y[nh] - e_I f_n^{(k)} h^{k+1} \tag{5}$$

Here the term $- e_I f_n^{(k)} h^{k+1}$ represents the local truncation error associated with the integration method of order k and $f_n^{(k)}$ is the kth time derivative of f at $t = nh$ [1]. For example, $k = 1$ and $e_I = 1/2$ for Euler integration; for AB-2 integration $k = 2$ and $e_I = 5/12$. We now take the Z transform of Eq. (5) and divide by $z - 1$ to obtain

$$Y^*(z) \cong Y_{ref}^*(z) - \frac{e_I h^{k+1} F^{(k)}*(z)}{z - 1} \tag{6}$$

Here $Y_{ref}^*(z)$ is the Z transform of the exact solution, $y[nh]$. Next we consider the case of sinusoidal data sequences by replacing z with $e^{j\omega h}$. We also note that $F^{(k)}*(e^{j\omega h}) = (j\omega)^k F^*(e^{j\omega h})$, i.e., the Fourier transform of the kth derivative of a function is equal to the Fourier transform of the function multiplied by $(j\omega)^k$. After dividing the resulting expression by F^*, we have

$$\frac{Y^*(e^{j\omega h})}{F^*(e^{j\omega h})} \cong \frac{Y_{ref}^*(e^{j\omega h})}{F^*(e^{j\omega h})} - \frac{e_I h (j\omega h)^k}{e^{j\omega h} - 1} \tag{7}$$

The term Y^*/F^* is simply the sinusoidal transfer function, $H_I^*(e^{j\omega h})$, of the numerical integrator. The term $Y_{ref}^*/F^* = 1/j\omega$, the transfer function of an ideal integrator. If we now approximate $e^{j\omega h} - 1$ by $j\omega h$, Eq. (7) becomes the following:

$$H_I^*(e^{j\omega h}) \cong \frac{1 - e_I (j\omega h)^k}{j\omega} \cong \frac{1}{j\omega[1 + e_I (j\omega h)^k]}, \quad \omega h << 1 \tag{8}$$

Here e_I is the integrator error coefficient and k is the algorithm order. To illustrate the application of our integrator transfer function model, we consider the simulation of a linearized dynamic system with transfer function $H(s)$. For the case of sinusoidal inputs of frequency ω, the transfer function becomes $H(j\omega)$. When the continuous system is simulated with a single-pass integration method, the sinusoidal transfer function of the digital simulation is simply given by $H(1/H_I^*)$, where H_I^* is the transfer function of the digital integrator. For $\omega h << 1$, $1/H_I^*$ can be approximated by $j\omega[1 + e_I(j\omega h)^k]$ in accordance with the integrator model of Eq. (8). Thus the formula for the transfer function of the digital system in simulating the linear system with transfer function $H(s)$ is given by

$$H^*(e^{j\omega h}) = H(1/H_I^*) \cong H\{j\omega[1 + e_I (j\omega h)^k]\}, \quad \omega h << 1 \tag{9}$$

For example, consider a first-order linear system with eigenvalue λ and transfer function given by $H(s) = 1/(s - \lambda)$. The transfer function for sinusoidal inputs becomes

$$H(j\omega) = \frac{1}{j\omega - \lambda} \tag{10}$$

Then the digital system transfer function for sinusoidal input data sequences is given approximately by

$$H^*(e^{j\omega h}) \cong \frac{1}{j\omega[1 + e_I(j\omega h)^k] - \lambda} \quad , \quad j\omega << 1 \tag{11}$$

We note that the characteristic root (eigenvalue) λ for the continuous system is given by the value of $j\omega$ in Eq. (10) which makes the denominator vanish. It follows that the equivalent characteristic root λ^* for the digital system is given by the value of $j\omega$ which makes the denominator of Eq. 11) vanish. Replacing $j\omega$ by λ^* in Eq. (11) and setting the denominator equal to zero, we can write

$$\lambda^* = \lambda - \lambda^* e_I(\lambda^* h)^k$$

For $|\lambda h| << 1$, $\lambda^* \cong \lambda$ to order h^k. Then we can replace λ^* by λ on the right side of the equation and obtain

$$e_\lambda = \frac{\lambda^* - \lambda}{\lambda} \cong -e_I(\lambda h)^k \quad , \qquad |\lambda h| << 1 \tag{12}$$

Here e_λ represents the fractional error in the digital system characteristic root. We recall that the transfer function for any finite order linear system with distinct roots can be represented as the sum of first-order transfer functions of the form $1/(s - \lambda)$, where the charcteristic roots may be real or complex. It follows that Eq. (12) can be used to estimate the error in each characteristic root in the digital system simulation of any order linear system.

From the digital transfer function formula in Eq. 11) we can write

$$H^*(e^{j\omega h}) \cong = \frac{1}{(j\omega - \lambda)\left[1 + \frac{e_I j\omega(j\omega h)^k}{j\omega - \lambda}\right]} = \frac{H(j\omega)}{1 + \frac{e_I j\omega(j\omega h)^k}{j\omega - \lambda}}$$

from which

$$\frac{H^*(e^{j\omega h}) - H(j\omega)}{H(j\omega)} \cong -\frac{e_I j\omega(j\omega h)^k}{j\omega - \lambda} \quad , \qquad \omega h << 1 \tag{13}$$

Here $(H^* - H)/H$ represents the fractional error in digital system transfer function. For $\omega h << 1$ it is evident that this fractional error will be small in magnitude compared with unity. In this case it can be shown that the real part of $(H^* - H)/H$ is approximately equal to the fractional gain error of the digital transfer function and the imaginary part is approximately equal to the phase error [2]. We note that the transfer function for any finite-order linear system can be written as the product of individual pole and zero factors of the form (s - λ), where again λ can be either real or complex. It is then straightforward to show that the fractional error in the overall digital transfer function is approximately the sum of the individual errors given by Eq. (13) for each factor [2]. It follows that both gain and phase errors of the overall digital system transfer function for sinusoidal inputs are proportional to $e_I(j\omega h)^k$.

Thus for single-pass integration methods Eq. (12) and (13) represent simple approximate formulas for both characteristic root and transfer function errors. For a given integration algorithm the errors are directly proportional to the integrator error coefficient e_I for that algorithm. Table 2 lists e_I and k for the algorithms considered in this paper, including the modified-Euler method, which has the smallest error coefficient (e_I = 1/24).

Table 2. Error Coefficients for Integration Methods

$$\text{Integrator transfer function} = H_I{}^*(e^{j\omega h}) \cong \frac{1}{j\omega[1 + e_I(j\omega h)^k]}, \quad \omega h << 1$$

	e_I	k
Euler	$\frac{1}{2}$	1
AB-2	$\frac{5}{12}$	2
Trapezoidal	$-\frac{1}{12}$	2
Modified Euler	$\frac{1}{24}$	2

All of the above algorithms in Table 2 are explicit except trapezoidal, which is implicit. An explicit method with the same asymptotic accuracy can, however, be realized with the two-pass Adams-Moulton (AM-2) algorithm. In this method the first pass employs AB-2 integration to obtain an estimate $\hat{y}_{n+1}$ of the next state. This is then used in the trapezoidal formula to compute the corrected y_{n+1}. The local truncation error associated with $\hat{y}_{n+1}$ is of order h^3, which ensures that the asymptotic accuracy of order h^2 for the corrected y_{n+1} will be the same as that for implicit trapezoidal integration.

4. Specific Examples

We now turn to some specific examples to compare the accuracy of modified Euler integration with traditional algorithms of second order. We consider first a simple linear dependence of the acceleration A in Eq. (3) on the displacement D and velocity V. This leads to the following state equations, which for convenience have been written in terms of the undamped natural frequency ω_n and damping ration ζ of the second-order system:

$$\dot{V} = \omega_n^2(U_n - D_n) - 2\zeta\omega_n V \quad , \quad \dot{D} = V \tag{14}$$

From Eqs. (2) and (4), with the trapezoidal formula from Table 1 used for $\hat{V}_{n+1}$, we obtain the following difference equations for the modified Euler formulation:

$$V_{n+1/2} = V_{n-1/2} + \omega_n^2 h(U_n - D_n) - \zeta\omega_n(V_{n+1/2} + V_{n-1/2}) \quad , \quad D_{n+1} = D_n + hV_{n+1/2}$$

After solving the first equation for $V_{n+1/2}$, we have the following explicit equations:

$$V_{n+1/2} = C_1 V_{n-1/2} + C_2(U_n - D_n) \quad , \quad D_{n+1} = D_n + \mathrm{h}V_{n+1/2} \tag{15}$$

where

$$C_1 = \frac{1 - \zeta\omega_n h}{1 + \zeta\omega_n h} \quad , \quad C_2 = \frac{\omega_n^2 h}{1 + \zeta\omega_n h} \tag{16}$$

Here the constants C_1 and $C2$ can be precomputed. From Eq. (15) it follows that the ongoing simulation run then requires only 3 adds and 3 multiples per integration step.

Figure 2 shows plots of the solution error when using modified Euler integration to compute the response of a second-order system with $\zeta = 0.25$ to a unit step input. The initial conditions are given by $x(0) = y(0) = 0$. Shown in the figure are error plots for three of the damping methods listed in Table 1, including the trapezoidal damping used to derive Eqs. (15) and (16). For comparison Figure 2 also shows the step-response errors when AB-2 integration is used. In all cases the step size is given by $\omega_n h = 0.25$. The startup problem associated with AB-2 integration (the initial states at $t = -h$ are not specified) is solved by using Euler integration for the first step. In the case of modified Euler integration the first step which computes $y_{1/2}$ from y_0 uses a step equal to $h/2$. The figure clearly shows the superior accuracy of the modified Euler method, with the scheme using second-order predictor integration to estimate $\hat{y}_n$ producing the smallest errors.

The second example considered in this section is the simulation of the full nonlinear flight equations of an aircraft. Since the largest characteristic roots for the rigid airframe are normally those associated with the short-period pitching motion, we will only consider symmetric flight, i.e., the longitudinal equations of motion, in our example simulation. The conclusions regarding dynamic errors can be safely extrapolated to the full six-degree-of-freedom case. For this simulation the translational equations of motion are written with respect to flight-path axes, while the rotational equations of motion are written with respect to body axes [3]. Then the velocity state variables become total aircraft velocity V_p, angle of attack α, and pitch rate Q. The displacement state variables are altitude H, pitch angle Θ, and horizontal distance X. The velocity state equations are given by

$$\dot{V}_p = \frac{F_{wx}}{m} \quad , \quad \dot{\alpha} = Q + \frac{F_{wz}}{mV_p} \quad , \quad \dot{Q} = \frac{M}{I_{yy}} \tag{17}$$

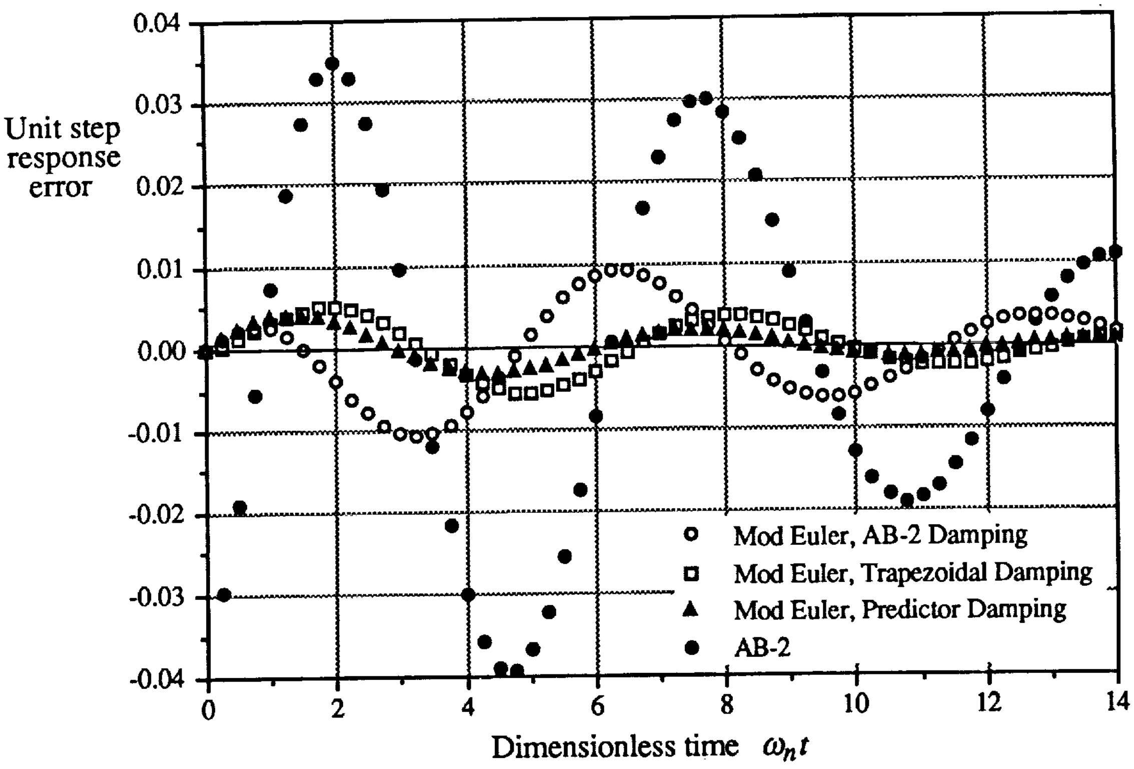


Figure 2. Unit step response errors in simulating second-order system, $\zeta = 0.25$, $\omega_n h = 0.25$.

and the displacement state equations by

$$\dot{\Theta} = Q \quad , \quad \dot{H} = V_p \sin(\Theta - \alpha) \quad , \quad \dot{X} = V_p \cos(\Theta - \alpha) \tag{18}$$

Here F_{wx} and F_{wz} are the external force components along the x and z flight-path axes, respectively, and M is the moment about the y body axis; m and I_{yy} represent, respectively, the aircraft mass and pitch-axis moment of inertia. The following formulas were used to represent the external forces and moment:

$$F_{wx} = - qS\,(C_{D_0} + C_{DC_L^2} C_L^2) - g\sin(\Theta - \alpha) + \frac{T}{m}\cos\alpha \tag{19}$$

$$F_{wz} =^{-} qS\,(C_L + C_{L_{\delta_e}}\delta_e) + g\cos(\Theta - \alpha) - \frac{T}{m}\sin\alpha \tag{20}$$

$$M = qcS(C_{M_0} + C_{M_\alpha}\alpha + C_{M_Q}\frac{c}{2V_p}Q + C_{M_{\dot{\alpha}}}\frac{c}{2V_p}\dot{\alpha} + C_{M_{\delta_e}}\delta_e) \tag{21}$$

where

$$q = \text{dynamic pressure} = \frac{1}{2}\rho V_p^2 \tag{22}$$

and

$$C_L = \text{lift coefficient} = C_{L_0} + C_{L_\alpha}\alpha \tag{23}$$

In these equations S is the aircraft wing area, g is the gravity acceleration, T is powerplant thrust, δ_e is elevator displacement, and c is the mean aerodynamic chord. The various C's represent aerodynamic coefficients and stability derivatives in accordance with the subscripts. In a full flight-envelope simulation these will be nonlinear functions of other variables such as V_p (through Mach number dependence), α, δ_e, and h.

Based on the way in which modified-Euler integration was introduced in Section 2, the velocity states V_p, α and Q in Eq. (17) would be represented at half-integer frames, with the position states Θ, H and X represented at integer frames. For the nth integration frame this results in the computation of the n+1/2 velocity state from the n-1/2 velocity state, followed by computation of the n+1 position state from the n position state using the n+1/2 velocity state just obtained. However, from Eq. (17) it is apparent that it would be better to represent the angle of attack α at integrer frames, even though it is derived from a velocity state equation. This is because the dominant term on the right side of Eq. (17) affecting the high-speed dynamics is the pitch-rate Q, which is represented at half-integer frames. The other term in Eq. (24), F_{wz}/mV_p, is the negative of the flight-path-axis pitch rate, and is generally much smaller in magnitude than Q. For this reason we have chosen to represent α at integer frames in the modified Euler mechanization of the flight equations. Since the force (and hence acceleration) term F_{wz} is computed and therefore represented at integer frames, it is necessary to compute an estimate of F_{wz} at the n+1/2 frame in the modified Euler integration of α_n to obtain α_{n+1}. This is easily accomplished using the first-order extrapolation formula $F_{wz_{n+1/2}} = (3/2)F_{wz_n} - (1/2)F_{wz_{n-1}}$. The actual difference equations used to solve (17) through (23) with modified Euler integration are presented in a previous paper by the author [4].

As a specific example we consider a business jet flying at 40,000 feet at a speed of Mach 0.7 [5]. For the above flight condition the undamped natural frequency of the short-period mode is about 3 rad/sec and the damping ratio is 0.4. In the example simulation we let the step size h = 0.1 second. This makes $\omega_n h \cong 0.3$ for the short-period motion, which should yield the moderate accuracies normally associated with a real-time simulation. We consider the aircraft response to the input function shown in Figure 3, which is a step elevator displacement with a one second rise time. Use of this input function tends to reduce the large transient errors caused by step inputs when predictor integration algorithms are used. It is also probably more typical of an actual transient input. The simulation is started at t = 0 with the aircraft in level equilibrium flight. In order to make the example more representative of an ongoing simulation, the step input

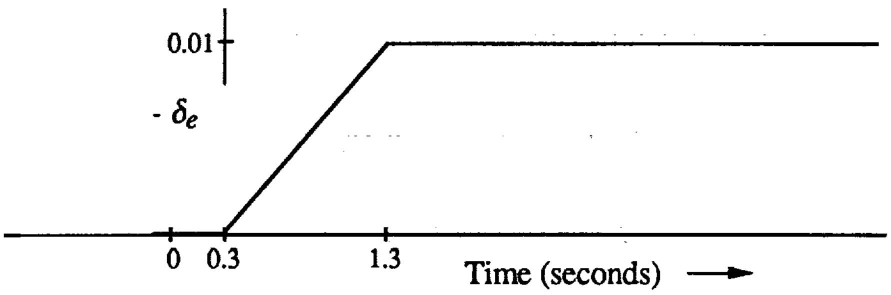


Figure 3. Delayed, finite rise-time step input.

is delayed for 0.3 seconds (three integration steps for $h = 0.1$) after the initial time $t = 0$. Figure 4 shows the error in pitch angle versus time for AB-2 integration and for modified Euler integration using the predictor integration of Table 1 for the integer velocity estimate. We note that the modified Euler method is an order of magnitude more accurate.

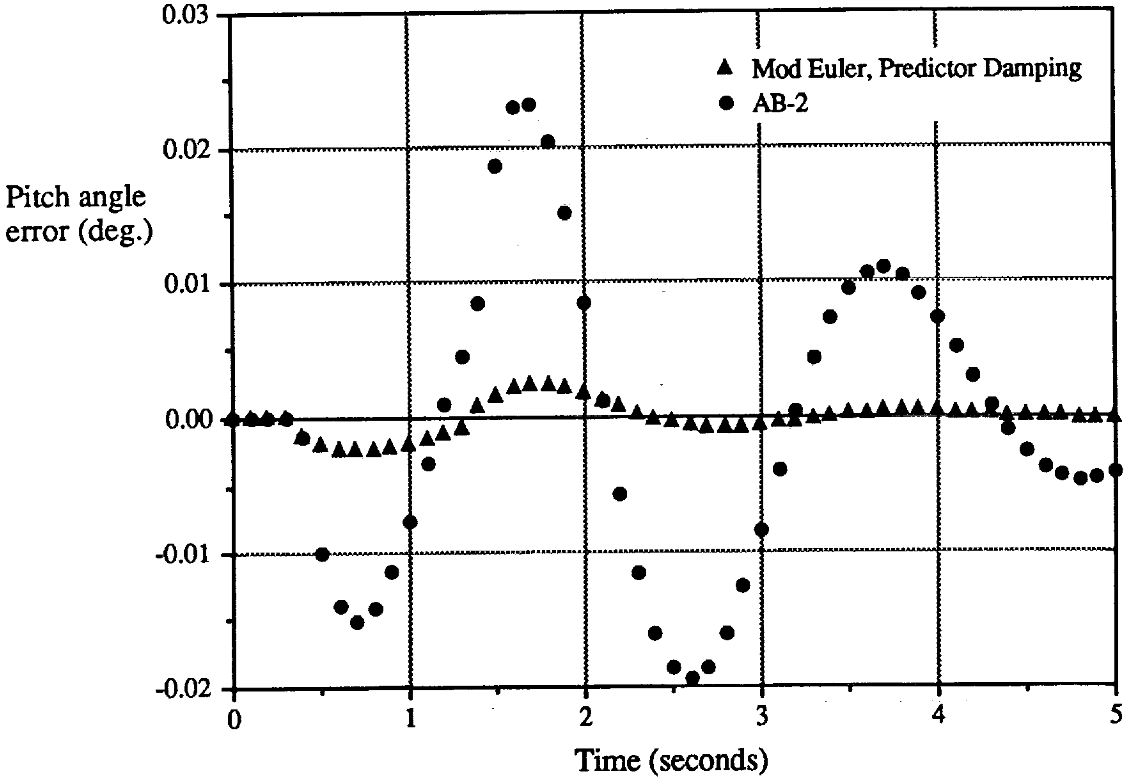


Figure 4. Aircraft pitch angle error for the input function of Figure 3; h = 0.1 seconds.

5. Stability Considerations

In addition to considering the dynamic accuracy associated with different numerical integration methods, it is important to consider the stability of the methods. This is usually done by considering the stability boundary in the complex λh plane. These boundaries are shown in Figure 5 for modified Euler integration used to solve Eq. (14) with the various methods for computing the velocity estimate, $\hat{V}_n$, as presented in Table 1. Also shown in Figure 5 is the stability boundary for the AB-2 predictor method, as well as that for the AM-2 predictor-corrector method. In the latter case the stability region has been reduced by a factor of two to take into account that AM-2 is a two-pass method. Any values of λh lying outside the boundary shown for a given method (the boundaries are symmetric with respect to the real axis) will lead to instability. From the figure it is evident that the modified Euler method with trapezoidal integration for the damping term exhibits the largest stability boundary. Note also that the stability boundary for all of the modified Euler methods lies on the imaginary axis. This means that modified Euler integration, when used to simulate systems with pure imaginary roots, as in the case of undamped oscillatory modes, will also exhibit pure imaginary roots corresponding to zero damping. This is true regardless of the integration step size h and is the reason why the modified Euler method is especially effective in simulating lightly-damped dynamic systems.

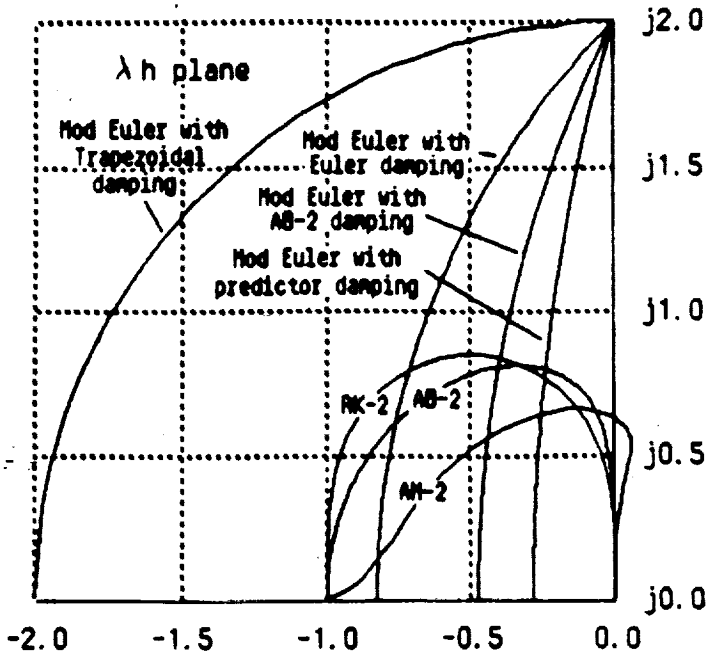


Figure 5. Stability boundaries for modified Euler and other second-order integration methods.

In the second-order system example considered in Section 4 we were able to use trapezoidal integration for the damping term in the modified Euler mechanization because the damping was linear. This in turn permitted us to construct an explicit, single-step formulation represented by Eqs. (15) and (16). When the the dependence of acceleration on velocity is nonlinear, this is no longer possible. Yet it would be advantageous for stability reasons to still use a trapezoidal implementation.

The nonlinear dependence of the acceleration A on the velocity V in Eq. (3) can often be expressed in terms of $V\partial A/\partial V$, where $\partial A/\partial V$ is not a function of V, or at worst is only slightly dependent on V. For example if A represents dQ/dt, the time derivative of pitch rate Q in the flight equations, then $\partial A/\partial Q$ is proportional to the aerodynamic stability dericvative C_{MQ}, i.e., the dimensionless pitching moment due to dimensionless pitch rate. C_{MQ} is normally independent of Q, although it may be dependent on other variables such as Mach number. Also, the overall $\partial A/\partial Q$ in this case will be independent of Q. Letting V be a scalar which represents the angular velocity Q, we can rewrite Eq. (14) as follows:

ORIGINAL PAGE IS
OF POOR QUALITY

$$\dot{V} = C_0[D,U(t)] + C_1[D,U(t)]V \tag{24}$$

where $C_0 + C_1V = A$ and $C_1 = \partial A/\partial V$. Now, when mechanizing the modified-Euler difference equations (4) we can compute $\hat{V}_n$, the estimate of V at the nth frame, by the formula

$$\hat{V}_n = \frac{1}{2}(V_{n+1/2} + V_{n-1/2}) \tag{25}$$

From Eqs. (24) and (25) the difference equation for $V_{n+1/2}$ in Eq. (4) then becomes

$$V_{n-1/2} = V_{n-1/2} + h[C_0(D_n,U_n) + C_1(D_n,U_n)\frac{V_{n+1/2}+V_{n-1/2}}{2}] \tag{26}$$

With respect to the velocity state V this equation clearly represents implicit trapezoidal integration. However it can be solved to obtain the following explicit formula for $V_{n+1/2}$:

$$V_{n+1/2} = \frac{(1+hC_1/2)V_{n-1/2} + hC_0}{1-hC_1/2} \tag{27}$$

This formulation, i.e., the use of trapezoidal integration for the damping term, expands very substantially the stability region in the λh plane compared with the use of the predictor formula for $\hat{V}_n$, as we have seen in Figure 5. It can also reduce appreciably the dynamic errors following transient inputs. The extra required computation is modest and consists mainly of an additional division.

In deriving Eq. (27) we have assumed that V is a scalar, whereas V will in general be a vector. In this case $\partial A/\partial V$ will be a matrix, which must be inverted to obtain the explicit formula for $V_{n+1/2}$. Fortunately, the critical terms in this matrix in the case of the flight equations are the diagonal terms, in which case simple formulas similar to Eq. (27) involving only the diagonal terms can be derived. In the longitudinal flight equations (17) through (23), for example, an equation similar to (27) can be written for $Q_{n+1/2}$, where C_1 is proportional to the stability derivative C_{M_Q}.

6. Conclusions

We have shown that mechanical dynamic systems are well suited to a modified Euler integration method which computes displacement and acceleration variables at integer frame times and velocity variables at half-integer frame times. Examination of asymptotic formulas for characteristic root and transfer function errors associated with a linearized version of any nonlinear mechanics problem shows that the modified Euler method is at least twice as accurate as any other known second-order algorithm. For the usual case where the acceleration is a function of velocity, there are a number of candidate methods for computing the required velocity estimates at integer frames from the velocity as computed at half-integer frames. A second-order predictor integration formula produces the most accurate integer-frame velocity estimate; an estimate based on the equivalent of trapezoidal integration produces the most stable simulation. Neither estimate requires any additional derivative evaluations, and the predictor formula can be used to produce output displacements at half-integer as well as integer frame times in a real-time simulation, i.e., at double the integration frame rate. The modified Euler method is particularly effective in simulating systems with lightly damped modes, since modes with zero damping in a continuous system generate modes with zero damping in the modified Euler mechanization, regardless of the integration step size. The modified Euler method also has a simple and accurate startup procedure and is completely compatible with real-time inputs. Two examples, a second-order linear system and a sixth-order nonlinear flight simulation, have been used to demonstrate the superior accuracy of the modified Euler method.

7. References

1. Gear, William G., "*Numerical Initial Value Problems in Ordinary Differential Equations,*" Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1971.

2. Howe, R.M., "Transfer Function and Characteristic Root Errors for Fixed-Step Integration Algorithms," *Transactions of the Society for Computer Simulation,* 2 (4): 293-320, 1985.

3. Fogarty, L.E., and R.M. Howe, "Computer Mechanization of Six-Degree-of-Freedom Flight Equations," *Simulation,* 11(4): 187-193, 1968.

4. Howe, R.M., "An Improved Numerical Integration Method for Flight Simulation," *Proceedings of the AIAA Flight Simulation Technologies Conference*, Boston, Aug. 14-16, 1989, pp 310-316.

5. Roskam, J., "*Airplane Flight Dynamics and Automatic Flight Controls,*" Vol. 1, 1982, pp. 616-624, Roskam Aviation and Engineering Corporation, Route 4, Box 274, Ottawa, Kansas 66067.

N90-23065

The Use of Real-Time, Hardware-in-the-Loop Simulation in the Design and Development of the New Hughes HS601 Spacecraft Attitude Control System

Loren I. Slafer
Laboratory Scientist, Guidance and Control Systems Laboratory
Hughes Aircraft Company, Space and Communications Group

Abstract

Realtime simulation and hardware-in-the-loop testing is being used extensively in all phases of the design, development, and testing of the attitude control system (ACS) for the new Hughes HS601 satellite bus. Realtime, hardware-in-the-loop simulation, integrated with traditional analysis and pure simulation activities is shown to provide a highly efficient and productive overall development program. Implementation of high fidelity simulations of the satellite dynamics and control system algorithms, capable of real-time execution (using applied Dynamics International's System 100), provides a tool which is capable of being integrated with the critical flight microprocessor to create a 'mixed simulation' test (MST). The MST creates a highly accurate, detailed simulated on-orbit test environment, capable of open and closed loop ACS testing, in which the ACS design can be validated. the MST is shown to provide a valuable extension of traditional test methods. a description of the MST configuration is presented, including the spacecraft dynamics simulation model, sensor and actuator emulators, and the test support system. Overall system performance parameters are presented. MST applications are discussed - supporting ACS design, developing on-orbit system performance predictions, flight software development and qualification testing (augmenting the traditional 'software-based' testing), mission planning, and a cost-effective subsystem-level acceptance test. The MST is shown to provide an ideal tool in which the ACS designer can 'fly the spacecraft on the ground'.

N90-23066

"A Real Time, FEM Based Optimal Control Algorithm and its Implementation Using Parallel Processing Hardware (Transistors) in a μ Processor Environment"

William Neff Patten
School of Aerospace and Mechanical Engineering
University of Oklahoma

Abstract

There is an evident need to discover a means of establishing reliable, implementable controls for systems that are plagued by nonlinear and, or uncertain, model dynamics. There is reported here the development of a generic controller design tool for tough-to-control systems. the method utilizes a moving grid, time finite element based solution of the necessary conditions that describe an optimal controller for a system.

The technique produces a discrete feedback controller. Real time laboratory experiments are now being conducted to demonstrate the viability of the method. The algorithm that results is being implemented in a microprocessor environment. Critical computational tasks are accomplished using a low cost, on board, multiprocessor (INMOS T800 Transputers) and parallel processing. Progress that has been made to date to validate the methodology will be presented. applications of the technique to the control of highly flexible robotic appendages will be suggested.

N90-23067

Six–Degree–of–Freedom Aircraft Simulation with Mixed–Data Structure Using the Applied Dynamics Simulation Language, ADSIM

Clare Savaglio
Applied Dynamics International
Ann Arbor, Michigan

Abstract

This paper presents a realistic simulation of an aircraft in flight using the AD 100 digital computer. We discuss specifically the implementation of three model features: (1) a large aerodynamic data base (130,000 function values) which is evaluated using function interpolation to obtain the aerodynamic coefficients, (2) an option to trim the aircraft in longitudinal flight, and (3) a flight control system which includes a digital controller. Since the model includes a digital controller the simulation implements not only continuous time equations but also discrete time equations, thus the model has a mixed–data structure.

Introduction

Real–time simulation of a realistic model is a cost effective way to design and test hardware. Model simulation is faster and safer than testing the actual system. Obviously, the computer system used to simulate a model is an important factor when considering simulation speed and simplicity of implementation, but the simulation language is often equally important. The ease with which a system can be modelled is strongly dependent on the simulation language.

In this paper we present a realistic model using the System 100. We try to show how conveniently certain modelling problems can be handled using ADSIM. We discuss three major features of the aircraft model which are common to many simulations. The first feature is the evaluation of aerodynamic functions with a large data base. This type of situation occurs in simulations where an analytic function does not exist and where function files dependent on several independent variables are given to describe a model. The second feature, the longitudinal trim technique, is applicable to systems where useful simulations must be run at an equilibrium condition. Finally, the case of mixed–data systems occurs whenever a digital and a continuous system are simulated. Although z–transform theory can be used to estimate errors of a dynamic plant with a digital controller, it must be assumed that the plant is linear. If this is not the case, the plant must be approximated by a linear system. A thorough analysis of the error and system dynamics can be gained through simulation. Even if the plant is linear, oftentimes a knowledge of the inter–sample behavior is desired, and simulation of the continuous plant and the digital controller is again required.

In the next section a brief discussion of the System 100 is given. The following section gives an overview of the aircraft model; the reference systems used, the orientation method, the external force model, and the control system model are each described. The last sections are devoted to three specific features of the aircraft model, demonstrating how they are modelled using ADSIM.

The System 100

The System 100 is an integrated simulation environment which consists of a general purpose computer and a digital simulation computer. The general purpose computer is one of the DEC VAX systems. The VAX front–end computer serves as a host for the Applied Dynamics AD 100 and the ADSIM compiler. Program preparation is done on the host computer. The AD 100 is the digital simulation compute engine and consists of up to seven parallel processors. The AD 100 is a totally synchronous, bus–oriented, multiprocessor system capable of performing 20 million floating–point operations per second (20 MFlops).

The simulation language, ADSIM, is equation driven and block oriented. Many key elements of a typical simulation, such as integration techniques, function generation, and control system nonlinearity functions, are built into the language. A control executive consisting of two programs provides the basis for implementing a model. The two programs are INTEXEC which runs on the host computer and SIMEXEC which runs on the AD 100. The executive controls such parameters as simulation time, frame time, and integration step size. The user's ADSIM program, consisting of blocks of code, is inserted into the SIMEXEC code at compile time. In this paper we mention two types of ADSIM blocks, the **REGION** block and the **DYNAMIC** block. **REGION** blocks may contain procedural code. The number of times and the order in which these blocks are executed are dependent on the specific name of the block. For instance, the code of **REGION sync2** is executed before the simulation run is begun while that of **REGION sync4** is executed before each step of the simulation run. Eight optional **REGION** blocks are available. The **DYNAMIC** blocks contain the model dynamics. The code must be nonprocedural. The model differential equations reside here, written as a series of scalar and first order differential and difference equations.

ADSIM offers an interactive environment which allows the program to be modified without recompiling. This environment is named **INTERACT** and allows instantaneous changes to simulation elements at run time which include all numerical values of the program, integration algorithms, integration step size, end time, sample time of a digital system and the speedup ratio with respect to real time.

In this paragraph we note some ADSIM integration terminology since we will need the concepts to describe the implementation of the model. These terms are referred to as runspecs of the ADSIM program([6]). The frame time is the amount of time needed for the computer to solve the differential model equations. The step size, T, is the integration algorithm step used to calculate the system solution. The step time is the step size divided by the number of passes through the dynamical equations needed for the specific integration algorithm. For instance the fourth order Runge-Kutta method, (RK–4), requires four passes through the differential equations and thus the step time is equal to $\frac{T}{4}$. In order for the simulation to run at real–time the actual time it takes to integrate the equations should be equal to the simulated time, thus the step time should equal the frame time. As an example, suppose the model uses the RK–4 integration method, thus for real–time simulation the step size divided by four should be equal to the frame time. If the step time is larger than the frame time the program is running at faster than real–time.

Aircraft Model

The aircraft model is representative of a business jet. The model can represent many other types of aircraft simply by changing some of the simulation parameters. The jet aircraft is modelled as a rigid body, the center of mass chosen as the reference point so that the translational and rotational aspects of the motion can be analyzed separately. The state variables chosen to describe the translational motion are: total velocity relative to the atmosphere, angle of attack, sideslip angle, distance north, distance east and altitude ([1]). The rotational motion is described using the conventional aircraft Euler angles and the components of the aircraft angular velocity along body axes. Figure 1 shows a block diagram of the overall six-degree-of-freedom flight equations. Starting on the left are the state variables and ending on the right are the time rates of change of the state variables. The time rates of change are integrated over each integration step to obtain the state variables used for the next step.

A total of four reference systems is used in the model to conveniently describe the aircraft motion. The model orientation is best described using a Body reference system, a coordinate set fixed to the aircraft. A nonrotating flat-earth atmospheric reference is defined to enable the translations and rotations of the aircraft to be related to an absolute reference. The translational equations of motion are written using Flight Path axes since the total velocity, angle of attack, and sideslip angle are easily described in that frame. (The Flight Path frame is defined such that the total aircraft velocity is along the X axis.) The Stability frame, an intermediate axis set between the Body and the Flight Path frames, is used to describe the aerodynamic equations since force and moment data are usually obtained in that frame.

Only three angles are needed to describe every possible orientation of the rigid body model. Of the choices of well known methods to describe rigid body orientation, this aircraft model implements the method of quaternions. The method of Euler angles was not used since a singularity occurs when the aircraft is pitched ±90 degrees. The method of direction cosines was not chosen since the method uses 9 parameters to define the model orientation, requiring 6 constraint equations. The method of quaternions only requires 4 parameters to describe the system orientation, and thus only one constraint equation is needed. ADSIM offers a convenient quaternion model, which, given the initial Euler angles (defined for a $3-2-1$ rotation), and the aircraft's present angular velocities, computes the present Euler angles and the direction cosines of the system.

The external forces acting on the jet aircraft are modelled as a constant gravity force, aircraft thrust, and aerodynamic forces. Fourteen aerodynamic coefficients are used to describe the aerodynamics of the aircraft. The coefficients are dependent on the aircraft angle of attack, sideslip angle, deflections of the control surfaces, (aileron, elevator, rudder and flaps), and mach number.

The aircraft flight control system is described with a pitch, roll, and a yaw-damper control loop. Each control loop contains an actuator described by a proportional plus-rate feedback second-order system. In the case of the pitch and roll loops, a proportional plus rate feedback is subtracted from the autopilot angle input to drive the controller. In the case of the yaw-damper loop, the yaw rate is sent through a high-pass filter and subtracted from the autopilot rudder input to drive the system.

Aerodynamic Function Interpolation

Function evaluation is an important task for real-time simulation. In the aircraft model there is a large aerodynamic data base. Two variables, density and the reciprocal of the speed of sound, are dependent on only one variable, the aircraft altitude, and 14 aerodynamic coefficients, (such as the lift, side-force

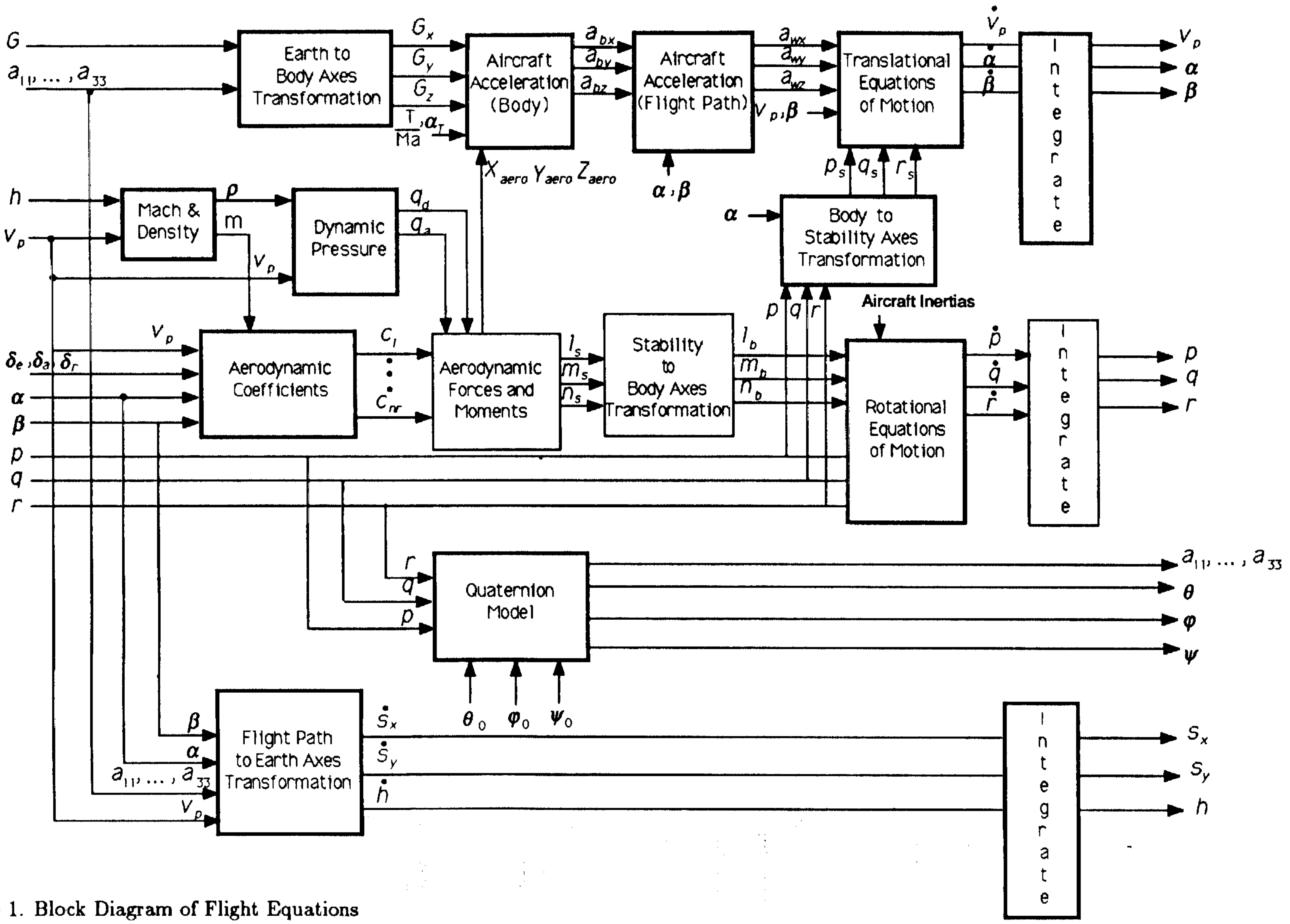


Figure 1. Block Diagram of Flight Equations

and drag coefficients), are dependent on up to four independent variables. ADSIM provides function generation capabilities for both equally and arbitrarily spaced functions on the AD 100. ADSIM allows multivariable functions with up to seven independent variables. Basically the way ADSIM evaluates a function of some independent values is to first use a binary search technique to determine between which given independent data points each input variable lies. Next interpolation techniques are used to evaluate the function at the given independent variable values. In order to perform function evaluation, independent variable and function tables must be declared. The aircraft simulation code containing function declarations for the density, and drag and lift coefficients is shown below ([4]).

```
(* Function generation breakpoint
 table definitions *)

INTERPOLATION_INTERVALS
           h_data(129 OF 129)
INTERPOLATION_INTERVALS
              a_data(33 OF 33)
INTERPOLATION_INTERVALS
              m_data(65 OF 65)
INTERPOLATION_INTERVALS
               de_data(3 OF 3)
INTERPOLATION_INTERVALS
               df_data(3 OF 3)

(* Function generation
              definitions *)

INTERPOLATION_FUNCTIONS
       air_density(h_data)
INTERPOLATION_FUNCTIONS
lift_coeff(a_data,m_data,
          de_data,df_data)
INTERPOLATION_FUNCTIONS
drag_coeff(a_data,
    m_data,de_data,df_data)
```

```
(* Assign data files to
  breakpoint tables and functions *)

FILES
 h_data   =     'h_sixdof.bpt',
 a_data   =     'a_sixdof.bpt',
 m_data   =     'm_sixdof.bpt',
 de_data  =     'de_sixdof.bpt',
 df_data  =     'df_sixdof.bpt'

FILES
 air_density = 'rho_sixdof.fun',
 lift_coeff  = 'liftc_sixdof.fun',
 drag_coeff  = 'cd_sixdof.fun'


(* The dynamic block containing the
continuous time equations *)

DYNAMIC Continuous
.
.
(* Evaluate air density *)

rho     =       air_density(h)
.
.
(* Aerodynamic coefficient
        evaluation *)

cl      =    lift_coeff(a,m,de,df)
cd      =    drag_coeff(a,m,de,df)
.
.
END DYNAMIC continuous
```

Longitudinal Trim

The model contains an option to trim the aircraft in longitudinal flight. The aircraft is assumed to be in symmetric level flight. Longitudinal trim is accomplished with the function, **NEWTON_RAP**, supplied by ADSIM. For a given flight regime, (aircraft altitude and speed), **NEWTON_RAP** determines the angle of attack, thrust magnitude, and elevator deflection to cause the longitudinal accelerations and pitch moment to be zeroed.

The function **NEWTON_RAP** uses the Newton Raphson method, a successive approximation process, to determine the trim values. This method has quadratic convergence. Although only a local method, the Newton Raphson algorithm works very well for the aircraft flight model, because the aircraft equations are well behaved and an inital guess for the trim values with some physical insight causes the method to converge within about four iterations.

The function **NEWTON_RAP** is called in the **REGION sync2** block of ADSIM, thus the equilibrium values can be determined before the simulation run and the aircraft control is initialized for a trimmed flight.

Simulation of Mixed–Data System

The simulation of both a continuous and a digital system presents special modelling problems, such as simulation of the computational delay times for the Analog to Digital (A/D) and Digital to Analog (D/A) conversions, and the simulation of sampling times. A special execute pair exists in ADSIM so that these modelling questions need not be re–solved for each model containing digital and continuous systems ([3]). The special execute pair named **EXECUTE mixed_data** controls the execution of the integration, the implementation of the delay times, and the order in which the dynamic systems are solved. Note that the continuous time equations should be solved on a frame-by-frame basis while the discrete time equations should be solved only every integer multiple of the frame time. This integer is dependent on the sampling rate of the digital system.

The execute pair for mixed–data system simulation is used in ADSIM programs along with two dynamic blocks. Simulations of this type are easily partitioned into two or more subsystems. One subsystem consists of algebraic and difference equations that represent the digital/discrete-time control laws. The other subsystem consists of the algebraic and differential equations that describe the behavior of the continuous time system. When creating the control structure for simulation of mixed–data systems, the following factors were considered:

- Numerical integration of discontinuities introduced by the digital controller residing in the continuous time subsystem.
- Selection of numerical integration method(s).
- Adjustment of either the sample period or integration step size.
- Proper padding of simulation frames in order to maintain real-time.
- Representation of computational delay in the digital subsystem.
- Representation of errors introduced by A/D and D/A conversions.

Let us discuss the selection of the integration method. Single-step predictor type methods which use the current and past derivatives to predict the new state values do so under the assumption of continuous derivatives. This assumption does not hold when simulating mixed-data systems. Strong discontinuites are introduced by the control command as determined by the discrete time system. Self-starting methods, such as the classical, multistep Runge-Kutta second, third and fourth order variety, do not rely on this assumption and thus these methods could be used.

However, the classical Runge-Kutta methods are not suited for real-time simulation since they require external inputs to the integration method before they become available in time. Applied Dynamics developed Runge-Kutta Real-Time methods which have similar accuracy and stability characteristics as the traditional versions but can be used for real–time simulations. Assume that we wish to solve the following vector differential equation:

$$\dot{x} = f(x, u(t)) \qquad (1)$$

Where x is the state vector and u is the input vector. The following difference equation defines the second order Runge-Kutta Real-Time (RKRT2) integration method:

$$x_{n+1} = x_n + Tf(x_{n+\frac{1}{2}}, u_{n+\frac{1}{2}}) \qquad (2)$$

where

$$x_{n+\frac{1}{2}} = x_n + \frac{T}{2}f(x_n, u_n) \qquad (3)$$

We note from these equations that when using the RKRT2 integration method for the simulation of mixed data systems, we must assure the sample time is an integer multiple times the integration step size, T. The control structure of **EXECUTE mixed_data** will automatically adjust the sample time to insure this condition (only if the sample time is not an integer multiple of the frame time).

We now consider the simulation of the digital controller's computational delay. Actual microprocessor hardware does not produce control action to the continuous system instantaneously. The most common method is to compute the control effort and output as soon as the results become available. However, when simulating delay time, the lag must be an integer multiple of the continuous system integration step time. If an estimate of the computational delay for a particular piece of hardware is known, **EXECUTE mixed_data** will allow the effect of both fixed and time-varying computational delays to be simulated on the continuous system model. The error introduced by the limited accuracy of the A/D and D/A conversions can be modelled using a nonlinear control ADSIM function named **Quantizer** which generates a symmetric staircase function.

The general format for ADSIM programs using the control structure of **EXECUTE mixed_data** follows:

```
TITLE {Simulation Title}

DYNAMIC discrete

{ Difference and algebraic equations
     to represent digital controller.}

END DYNAMIC discrete

DYNAMIC continuous

{ Differential and algebraic equations
    to represent continuous-time plant.}

END DYNAMIC continuous

EXECUTE 'mixed_data'
```

We now describe the implementation of the digital controller in the aircraft model. In order to design the digital pitch control algorithm, knowledge of aircraft pitch behavior is needed. The longitudinal dynamics of aircraft are in general dominated by two phenomena; phugoid motion and short period pitching motion. Phugoid motion is a very slow and lightly damped oscillatory motion in the vertical plane induced by mismatch between lift and drag of the airframe for a particular flight condition. The short period pitching motion occurs when the aircraft pitch angle is not aligned with the flight path. The aircraft dynamics will drive the pitch angle back toward the flight path. The short period pitching mode is much faster than the phugoid mode. The continuous system pitch dynamics were simulated and the frequency and damping ratio of the two motions were estimated from the transients. It was found that the phugoid motion exhibits a natural frequency of about 0.012 Hertz with a damping ratio of 0.1, while the short period pitching mode frequency is about 0.45 Hertz with a damping ratio of 0.35. The actuator is modelled by a proportional plus rate feedback second order system, its dynamics are characterized by a natural frequency of 5.0 Hertz and a damping ratio of 0.5. Using these estimates a transfer function representation was used to approximate the continuous system dynamics, where the input is the elevator deflection command and the output is the aircraft pitch angle.

A proportional–integral–derivative (PID) type of digital controller is implemented in the aircraft simulation. The digital controller transfer function, D(z), can be defined using z–transform notation and has the following form,

$$D(z) = \frac{a_1 + a_2 z^{-1} + a_3 z^{-2}}{1 + b_1 z^{-1} + b_2 z^{-2}}. \qquad (4)$$

The controller constants, a_1, a_2, a_3, b_1, b_2, were chosen using the transfer function approximation of the continuous system. These constants are easily refined at run–time using **INTERACT**.

As discussed in a previous section, the ADSIM language supports both continuous time derivatives and discrete time difference equations. Therefore the D(z) transfer function is simply converted to ADSIM notation and placed into the discrete dynamic block. The continuous dynamic block contains the equations for the continuous model equations. The prime symbol denotes the differential operator, $\frac{d}{dt}$, and the pound symbol denotes a shift in time, or a next–state variable. We note that the transfer function for a zero order hold need not be represented in the ADSIM program since the simulation control structure of the **Execute mixed_data** pair supples this effect implicitly. The relevant ADSIM code is shown below.

```
TITLE 'Six Degree-of-Freedom
          Aircraft Simulation'

COMMENT

This is an ADSIM program for simulating an
aircraft in six degrees of freedom.
The program runs on the Applied Dynamics
AD 100. Sixdof performs all the computations
required for the six-degree-of-freedom
aircraft equations of motion.
...

END COMMENT

...

(* Equations of motion of the
                 aircraft*)

DYNAMIC Continuous

(* Flight control systems &
                  actuators: *)

(* Elevator actuator *)

dei      =   Deilim*SAT(deicom*
             Ideilim)
udec     =   Kde*(dei-des-Cdedot*
             dedot)
dedot'   =   Udelim*SAT(udec*
             Iudelim)
des'     =   dedot
de       =   des+detrim

(* Roll control system *)
```

```
daicom  =   Kphi*(phii-phi-
            Cp*p)
dai     =   Dailim*SAT(daicom*
            Idailim)

(* Aileron actuator *)

udac=Kda*(dai-das-Cdadot*dadot)
...
...
...
r’=Inv_1mIxzsqxInv_IxxIz*(mz+
   IxzxInv_Iz*mx)

END DYNAMIC Continuous

DYNAMIC Discrete

(* Pitch control system *)

e_theta  = -(thetai-theta
           -Cq*q)
f_theta# = e_theta
g_theta# = f_theta

deicom  =
B1*u_theta + B2*v_theta +
A1*e_theta + A2*f_theta +
A3*g_theta

u_theta# = deicom
v_theta# = u_theta

END DYNAMIC Discrete

EXECUTE 'mixed_Data'
```

Speed of Simulation

The time needed for a single pass through the highly nonlinear dynamic equations is 138.1 microseconds, corresponding to a frame rate of about 7000 frames per second. Since accurate real–time simulations of aircraft require about a 10 to 30 second frame rate([5]), it would be possible to run this model up to 700 times real time.

Conclusions

We have presented an ADSIM model for the real–time simulation of an aircraft. We have described the modelling of three specific features of the simulation which are common to many simulation models. These features are a large aerodynamic data base requiring function interpolation, an option to trim the aircraft, and a digital controller. We have shown the model can be implemented and executed conveniently using ADSIM.

References

[1] Fogarty, L.E., Howe, R.M., *Computer Mechanization of Six-Degree-of-Freedom Flight Equations,* Simulation 11, October 1968.

[2] Wright, M., *System 100 Simulation Computer Architecture,* European Simulation Multiconference, June 1989.

[3] Zammit, S., Zwaanenberg, K., *Simulation of Mixed-Data Systems Using the Applied Dynamics Simulation Language ADSIM,* European Simulation Multiconference, June 1989.

[4] *Six-Degree-of-Freedom Flight Simulation,* ADI Application Report, Ann Arbor, MI, 1988.

[5] *Six-Degree-of-Freedom Flight Simulation Using the System 10,* ADI Application Report, Ann Arbor, MI, 1983.

[6] *ADSIM Reference Manual (Version 6.1),* ADI, Ann Arbor, MI, 1989.

N90-23068

Six-Degree-of-Freedom Missile Simulation Using the ADI AD 100 Digital Computer and ADSIM Simulation Language

Koos Zwaanenburg
Applied Dynamics International
Ann Arbor, Michigan

Abstract

This paper illustrates the use of an AD 100 computer and the ADSIM language in the six-degree-of-freedom digital simulation of an air-to-ground missile. The missile is launched from a moving platform, typically a helicopter and is capable of striking a mobile target up to 10 kilometers away. The missile could be any tactical missile.

Introduction

Real-time simulation with hardware in the loop is used extensively in the missile development business. It is used primarily to validate designs prior to the actual flight, to avoid experiments with actual flight hardware, and to reduce costs of flight trials by simulating a large variety of scenarios and conditions instead of performing these tests in real life. Missile manufacturing companies are an important section of Applied Dynamics' customer base. Therefore, Applied Dynamics International(ADI) developed this particular missile model so that it can function as a realistic example of the type of models that most of our customers work with.

Six-Degree-of-Freedom Missile Model

The missile is controlled by four fins mounted in cruciform configuration at the rear of the missile. The fins are independently controlled by pneumatic servos. The servos are activated by commands from the autopilot, which processes the sensor and seeker guidance outputs before issuing these commands. The missile roll, pitch, and yaw attitudes are sensed using two-degree-of-freedom gyros. Two such gyros are required. The target is tracked using an inertially stabilized seeker (laser or RF/IR) mounted at the front end of the missile inside a radome. Imperfect attitude sensing and target tracking are included in the simulation. The user can exclude, include or amplify these extraneous effects by selecting the proper options switches.

The missile is thrusting during part of the flight. Thrust is a prespecified function of time. Burnout occurs about five seconds from launch time. As the missile burns the propellant, the rotational inertias of the vehicle change and the center of gravity moves forward along the missile longitudinal axis. The missile aerodynamic center remains aft of the missile center of gravity at all times. Imperfect c.g. location, thrust offset, and misalignment are also modeled in the simulation. Atmospheric wind velocity is modeled as the sum of a steady state and a gusting component. The gusting component is assumed to be a zero-mean, uniformly distributed random variable in all directions. Vehicle translational equations of motion are expressed in earth axes, while the missile body axes are used for the rotational dynamic equations.

Aerodynamic coefficients are described as multivariable functions of typically the missile angle of attack, angle of sideslip, control surface displacements and Mach number. Furthermore, rotor downwash (the effects of which are confined to the neighborhood of the helicopter), instantaneous thrust, gyro drift angles, and seeker tracking errors are described as nonlinear functions.

The simulation is terminated when one of the following conditions occurs: when the simulation end time exceeds the user-specified end time, when the missile crashes, when the missile Mach number exceeds a specified limit, or when the relative position component along the relative velocity vector x_{rv} changes sign. When x_{rv} changes sign, the miss distance is computed by linearly interpolating the separation between the missile and the target to the point where x_{rv} is zero.

The complete system of equations has been partitioned into different modules for simplicity. These modules can be looked upon as various "generic" subsystems of the missile model. They can be used separately for other missile simulations.

The missile coordinate conversion module (MCC) describes the various inertial and body axes coordinate transformations required for every tactical missile model, the missile attitude, quaternions, direction cosines, the effect of steady and gusting wind components, the rotor downwash, the velocity components, and angle of attack and sideslip.

The line-of-sight module (LOS) describes the missile-target geometry and velocities in absolute and relative terms and describes the target tracking errors, radome aberrations, and glint.

The seeker module describes the tracking errors in azimuth and elevation and the seeker dynamics. Furthermore, it describes the guidance commands for the autopilot module. This module allows insertion of specifics for laser, RF, or IR seekers.

The autopilot module describes an analog autopilot that accepts guidance commands from the seeker module and body attitude information from the MCC module to implement a proportional navigation scheme that will guide the missile to the target. The autopilot module features separate autopilot channels for roll, pitch, and yaw. Gyro drift is modeled as well. The autopilot module generates four fin position commands for the four actuators. The actuator module will be used four times for this particular missile simulation. The actuators are described as nonlinear fourth-order systems with nonlinearities like running and breakaway friction, torque limiting, and fin angular traveling limiting.

The aerodynamics module describes all aerodynamic force and moment coefficients acting on the missile airframe as multivariable functions. Inputs to the functions are typically the missile angle of attack, angle of sideslip, control surface displacements and Mach number. This module is dependant on the types of wind-tunnel measurements performed on the missile; therefore, this module is the least "generic" of all missile modules. For this particular missile there are 14 functions of one variable, 17 functions of two variables and 9 functions of three variables that describe the aerodynamics. Furthermore, some functions are included to represent thrust and mass variations due to propellant burn.

The missile equations of motion module combines the aerodynamic coefficients with air density, dynamic pressure, and gravity to form the total forces and moments acting on the airframe. Dividing these forces and moments by mass or moments of inertia provides the missile translational and rotational accelerations.

SYSTEM 100 Architecture

Applied Dynamics International has been involved in solving time-critical simulation of continuous dynamic systems since its founding in 1957. The SYSTEM 100 simulation computer system was introduced by ADI in 1984. It consists of an AD 100, a high-speed, floating-point compute engine; a host controller, a general-purpose digital computer of the VAX family; and ADSIM, a user-friendly simulation language designed specifically for the AD 100. The SYSTEM 100 hardware and software work together to form a complete simulation environment.

The AD 100

The AD 100, a synchronous, bus-oriented multiprocessor compute engine designed for time-critical digital simulation, is the basic fundamental building block of the SYSTEM 100. The AD 100 is a single-user system without an operating system. It is controlled by a multiprocessing VAX host computer running the VMS operating system. Acting as the controller and user interface, the host computer relieves the AD 100 of these interrupt-based tasks. The compute engine needs to be isolated from the overheads and restrictions associated with an operating system in order to achieve and maintain its optimum computation speed or frame rate.

The AD 100 is capable of 20 million floating-point operations per second. The basic system consists of four processors and a host interface tied to a common bus, the PLUSBUS. The PLUSBUS is 105 bits wide, 65 bits of data and 40 bits of address and control. Emitter-Coupled Logic (ECL) devices are used to obtain high computational speed. The four basic processors are the Communication and Control Processor (COM), the Arithmetic Logic Unit (ALU), the Multiplier (MUL), and the Storage Processor (STO). Each processor has its own program memory, program counter, and instruction decoder. Each processor has a 64-bit instruction. The COM Processor has a 64K program memory, and the other processors have 4K program memories. Timings in the AD 100 are expressed in terms of a master clock period of 25 nanoseconds. The instruction cycle of the AD 100 consists of four of these phases or periods.

Every arithmetic operation performed on the AD 100 is done in floating-point arithmetic. Calculations are performed using either a 53-bit short-word format or a 65-bit long-word format. Both formats contain 1 sign bit, 12 exponent bits, and either 40 or 53 significand bits. The long-word format is used where additional accuracy is needed, such as in the case of numerical integration to minimize roundoff and truncation errors.

The STO Processor provides 64K of 65-bit high-speed data storage. Memory accesses and address arithmetic can take place two per instruction cycle. Some simulation tasks such as function generation and memory buffers require large amounts of data storage. It is for these purposes that an optional processor, the Function Memory Unit (FMU), was introduced, which has data memory of 2 million words by 64 bits.

ADSIM Environment

With a specific application area in mind, namely real-time simulation, ADI was able to design the AD 100's hardware and ADSIM to handle the necessities of the simulation environment. ADSIM is made up of a high-level simulation language and a run-time interactive control environment.

The ADSIM language is mathematically oriented. Many key elements of a typical simulation are built into the language, such as integration techniques, function generation, and control-system nonlinearity functions. A control executive consisting of two programs, one that runs on the host controller and one that runs on the AD 100, provides the basis for implementing a model. The control structure built into this executive controls such parameters as system time (simulation time), frame time, and integration step size. A dynamic section is provided for the model's differential equations. ADSIM allows the model to be implemented as a series of scalar and first-order differential equations. If conditional code is included as part of the model, the control executive takes care of padding the frame such that each frame is consistent with real time. Sorting of the dynamic equations and identifying algebraic loops are examples of some of the capabilities of the ADSIM compiler. Integration is handled using Runge-Kutta, Adams-Bashforth, or Adams-Moulton system-level routines. The model development time is much less when a simulation language such as ADSIM is used since many of these standard simulation techniques are built into the language.

ADSIM program development, including editing, compiling, and debugging, is performed on the host computer. There are two ADSIM compilers: one that produces code to be executed on the AD 100 for time-critical work and one that produces code to be executed on the host computer for non-time-critical experimentation and debugging. The same ADSIM source can be processed by either compiler.

Running a program on the AD 100 involves loading the executable code from the host computer into the AD 100 at run time. The user run-time interface consists of a program called INTERACT running on the host computer. INTERACT provides a user-friendly interface for debugging and experimentation, allowing constants to be changed, variables to be displayed, integration methods changed, breakpoints to be set, etc. This environment **reduces the time it takes to get the simulation into a state where it can be integrated into the design and testing phase.**

FORTRAN

The AD 100 is also able to run FORTRAN programs. ADI developed FORTRAN/AD, a subset of the FORTRAN 77 standard, to run on the AD 100 computer. In general, FORTRAN programs will not be as computationally efficient as ADSIM programs, but they allow the user to benefit from investments already made in FORTRAN simulations.

Implementation in ADSIM

The interactive nature of ADSIM, together with the INTERACT utility, make the task of implementation, verification, and validation easier and allows one to develop a feel for the system being simulated. A rich set of INTERACT commands allows the user to change any simulation variable or to change the integration time step and method; an on-line data logger and graphics package allow the capability to verify simulation results. Various researchers have estimated that the verification and validation portion of a simulation can consume 30 to 60 percent of a particular project's schedule and budget. The implementation of the missile model in ADSIM, together with its interactive environment, is very straightforward. It requires about 1150 lines of ADSIM source code. The implementation runs on the AD 100 with a frame time of 444 μs, allowing the model to run more than four times faster than real time. The same ADSIM model runs on VAXes as well. The frame times on various VAXes allow the model to run between 5 and 20 times slower than real time, depending on the type of VAX. On the AD 100, the entire model requires about 5 percent of the hardware resources.

Implementation in FORTRAN

To compare the accuracy and performance of the model, ADI also implemented the missile model in FORTRAN. This FORTRAN code runs on the AD 100 as well as on many general-purpose digital computers. The implementation in FORTRAN requires about 3950 lines of source code. On the AD 100, the frame time of the model is about 1100 μs, allowing the model to run just a bit faster than real time. On a VAX computer, the same FORTRAN code runs from 10 to 50 times slower than real time, depending on the type of VAX.

Comparing ADSIM and FORTRAN

The implementation in FORTRAN does not support an interactive environment. As a result, the development of the FORTRAN version of the simulation turned out to be orders of magnitude more tedious than the corresponding ADSIM implementation. Such issues as functions, models, sorting of the equations, and optimizing were major hurdles for the FORTRAN implementation, while they were trivial for the ADSIM implementation. The numerical accuracy of the two models is essentially the same. The two implementations provide answers that are similar up to seven or eight decimal places. Both models, when running on the AD 100, can be extended, to hardware-in-the-loop studies.

Conclusions

The performance numbers of the AD 100 show that it is possible to implement a high-performance missile model in a real-time simulation without the problems associated with an implementation on a general-purpose computer using FORTRAN.

References

[1] Wright, M. *System 100 Simulation Computer Architecture*. European Simulation Multiconference, June 1989.

[2] *Six-Degree-of-Freedom Missile Simulation*. Applied Dynamics Application Report, March 1989.

Real-Time Closed-Loop Simulation and Upset Evaluation of Control Systems in Harsh Electromagnetic Environments

Celeste M. Belcastro
NASA Langley Research Center
Hampton, Virginia 23665

ABSTRACT

Digital control systems for applications such as aircraft avionics and multibody systems must maintain adequate control integrity in adverse as well as nominal operating conditions. For example, control systems for advanced aircraft, and especially those with relaxed static stability, will be critical to flight and will, therefore, have very high reliability specifications which must be met regardless of operating conditions. In addition, multibody systems such as robotic manipulators performing critical functions must have control systems capable of robust performance in any operating environment in order to complete the assigned task reliably. Severe operating conditions for electronic control systems can result from electromagnetic disturbances caused by lightning, high energy radio frequency (HERF) transmitters, and nuclear electromagnetic pulses (NEMP). For this reason, techniques must be developed to evaluate the integrity of the control system in adverse operating environments. The most difficult and illusive perturbations to computer-based control systems that can be caused by an electromagnetic environment (EME) are functional error modes that involve no component damage. These error modes are collectively known as "upset", can occur simultaneously in all of the channels of a redundant control system, and are software dependent. Upset studies performed to date have not addressed the assessment of fault tolerant systems and do not involve the evaluation of a control system operating in a closed-loop with the plant. This paper presents a methodology for performing a real-time simulation of the closed-loop dynamics of a fault tolerant control system with a simulated plant operating in an electromagnetically harsh environment. In particular, the paper discusses considerations for performing upset tests on the controller. Some of these considerations are the generation and coupling of analog signals representative of electromagnetic disturbances to a control system under test, analog data acquisition, and digital data acquisition from fault tolerant systems. In addition, the paper presents a case study of an upset test methodology for a fault tolerant electronic aircraft engine control system.

I. Introduction

Advanced aircraft designs reduce aerodynamic drag via relaxed static stability and, therefore, control systems that are critical to the flight of the aircraft are required to maintain stability. In addition, fuel efficiency is greatly improved in advanced designs by using light-weight nonmetallic (composite) aircraft structures, rather than the metal ones currently in use. The trend in avionics technology is the implementation of control laws on digital computers that are interfaced to the sensors and control surfaces of the aircraft. Since digital computers are highly susceptible to transient electrical signals, the use of digital controls compounds the problem already incurred through the use of composite structures which do not provide the electrical shielding inherent in metal. As the function of the control system becomes more flight critical and the use of composite materials becomes

C-4

more widespread, the problem of verifying the integrity of the control in adverse, as well as nominal, operating environments becomes a key issue in the development of a control system. The use of digital computers to implement control laws is also evident in other areas of application. For example, in multibody systems such as robotic manipulators, control laws are implemented on digital computers. Performance considerations of these systems when operating in electromagnetically harsh environments are also of concern. Performance evaluation techniques presented in this paper are applicable to the assessment of any digital control system, regardless of the specific application. For simplicity, this paper will concentrate on techniques for evaluating the performance of avionic control systems in adverse operating environments.

One particularly harsh operating environment results from the presence of electromagnetic fields caused by sources such as lightning, high energy radio frequency (HERF) transmitters, and nuclear electromagnetic pulses (NEMP). As shown in Fig. 1, sources such as lightning, HERF, and NEMP generate electromagnetic fields outside of the aircraft which are dependent on the aircraft's geometry and structural material. These exterior electromagnetic fields penetrate the aircraft by leaking through joints, seams, and apertures so that interior electromagnetic fields are present. The interior fields cause analog electrical transients to be induced on the aircraft's wiring, and these signals can propagate to the onboard electronic equipment despite shielding and protective devices such as filters and surge suppressors. There are two types of effects to digital computer systems that can be caused by transient electrical signals. The first is actual component damage that requires repair or replacement of the equipment. The second type of effect to a digital system is characterized by functional error modes collectively known as "upset" which involve no component damage. In the case of upset, normal operation can be restored to the system by corrective action such as resetting/reloading the software or by an internal recovery mechanism, such as an automatic rollback to a system state just prior to the disturbance. The subject of effective internal upset recovery mechanisms is another current topic for research. See reference [1] for a more detailed account of the electromagnetic threat to advanced digital avionics systems.

To date, there are no comprehensive guidelines or criteria for performing tests or analyses on digital control systems to evaluate upset susceptibility or verify control integrity in electromagnetically adverse operating environments. Therefore, the objective of this research is to develop a methodology whereby a digital computer-based control system can be evaluated for upset susceptibility as well as control integrity when subjected to analog transient electrical signals like those that would be induced by an electromagnetic source. In particular, the electromagnetic source under consideration in this research is lightning. This paper discusses various issues in the design and implementation of upset tests which can be performed in the laboratory on a candidate fault tolerant control system during a real-time simulation of the closed-loop dynamics of the controller and plant operating in an adverse electromagnetic environment. A case study is described involving the upset test design of a full-authority electronic engine controller (EEC).

II. Upset Test Design for Fault Tolerant Control Systems

Most upset studies conducted to date have involved general-purpose systems executing a generic application code during testing [2] - [6]. One upset study involved the evaluation of an Inertial Navigation System that was subjected to transient signals like those that could result from NEMP [7]. Since none of these studies involved a control system that has closed-loop dynamics with a plant, it is desirable that an upset methodology be formulated for such a system. The general laboratory test configuration for the upset evaluation of a control system is shown in Fig. 2. As shown in the figure, the test configuration involves two control units - the unit under test and an unperturbed reference unit. The controller under test is perturbed by transient signals like those that could be induced by lightning. Each controller is interfaced to a simulation (hardware or software) of the plant in such a manner as to represent the closed-loop dynamics of the system. The operation of the two plant simulations are compared during tests so that cases in which acceptable control is not maintained by the faulted controller can be flagged in real time. Data obtained from the controllers during tests are stored for post processing and analysis. An alternative to having a faulted and reference controller is to have one controller which would be run with the plant simulation without faults for a period of time in a so-called "gold run". Unfaulted data would be recorded from the controller as well as the nominal operating parameters of the plant. Then, the plant parameter data obtained during faulted runs would be compared after testing to the nominal data and a determination made regarding the control integrity of the faulted controller. Since use of the two controllers would save a step in data processing, it is advantageous to use this configuration if two prototype controllers are available.

A. Generation of Analog Transients in the Laboratory

The waveform, shown in Fig. 3, that is most representative of those that occur on internal aircraft wiring due to lightning is a 1 - 50 MHz damped sinusoid which decreases in amplitude 50 - 75 % after four cycles [8]. This waveform can be generated by a capacitor discharge circuit with light damping [9]. However, use of a simple RLC circuit is awkward because components must be changed in order to generate key frequencies in the 1 - 50 MHz range. Three pulse generators have been designed to fulfill the electromagnetic test requirements of the Royal Aerospace Establishment [10]. One pulse generator produces damped sinusoidal waveforms from 2 - 30 MHz, one is a fixed-frequency 100 kHz generator, and the third produces two waveforms for ground voltage lightning effects simulation.

The most versatile way to generate the transient signal, and the technique presented here, is a polynomial waveform synthesizer which generates the waveform that corresponds to the entered equation. The output of the waveform generator can then be scaled to the proper amplitude via a wideband power amplifier. In this way, transient signals can be easily generated that cover a frequency range of interest and represent the induced effects of any electromagnetic source.

B. Coupling Analog Transients to the Controller Under Test

The mechanism for coupling analog signals into the digital controller must be such that the controller is not loaded down by mismatched impedance. In addition, the coupling mechanism must be representative of that which would occur in the natural operating environment, depicted in Fig. 1. The two most widely used coupling techniques are resistive and inductive coupling. An advantage to resistive coupling is that no special equipment is needed. In addition, it is very easy to inject transient signals into integrated circuit pins as well as printed circuit board test points. The coupling method which best satisfies the above criteria is to induce voltage into a cable or cable bundle using a ferrite coupling transformer that can be clamped around the cable. Details of performing such tests are given in [11] and [12].

Another consideration is whether the transient signal injection should be synchronized with the operation of the controller or whether the transient should be injected asynchronously. If the transient is injected synchronously, it must be introduced into the controller during each operational state of the processor. Since the number of states in a digital control system is very large, the required amount of testing for this approach is impractical. For this reason, asynchronous injection of a statistically significant number of transient signals is more advantageous. In addition, asynchronously injected transients can occur during the transition between states and, therefore, more realistically represent the threat that could occur in a natural environment.

C. Controller Monitoring Strategies

In single channel systems, upset modes can be fairly easily detected using comparison monitoring techniques on a test unit and reference unit executing identical software and operating in bit synchronism. Any time the data bus, address bus, or control lines of the test unit differ from those of the reference unit, data can be recorded and analyzed. In this way, data is only recorded for transient injections from which errors have occurred. (It was established in [2] and [3] that the occurrence and type of upset depends on the relative timing of the transient signal injection and the state of the processor. For this reason, upset does not occur each time the transient signal enters the system.) An advantage to this technique is that, since error-free data is not recorded, the amount of required data reduction is reduced.

Conversely, upset detection in fault tolerant systems is much more complex. Fault tolerant controllers usually employ one of two basic redundancy strategies - voting or primary/secondary channels. Comparison monitoring techniques cannot be used in upset testing of fault tolerant systems since reconfigurations in the test unit would cause miscomparisons to be generated without faulty operation being present. For these types of systems, upset detection criteria must be carefully selected since they effectively define upset for the test unit.

D. Data Acquisition

It is recommended that both analog and digital data be recorded during upset tests. The analog data to be recorded are the waveforms induced in the digital controller. In this way, various threshold characteristics of transient signals that cause upset can be determined. Norms such as peak absolute amplitude, maximum absolute rate of rise, peak absolute impulse, rectified impulse, and root action integral have been suggested in the literature for measuring NEMP stress waveform attributes [13]. These norms were used in an NEMP upset study and found to be inadequate [7]. Therefore, appropriate frequency-dependent norms for characterizing upset stress attributes of electromagnetically induced transient signals from sources such as lightning, HERF, and NEMP remains a topic for further research.

Digital data to be acquired from the controller should include the calculated control commands obtained from the data bus, the internal status word of the processor, as well as the address bus and appropriate processor control lines. Range checks can be used to determine if the calculated control commands are appropriate for the control regime in progress. Commands that would be acceptable in one control mode could be devastating in another, so calculated command data can only be evaluated in the context of the application. The internal status word of the processor should be monitored for the results of self tests, parity checks, and other fault tolerance strategies that might be present in the digital controller under test. Monitoring the results of the processor's own self-health evaluation can signal the beginning of a functional error mode or upset. Upset modes that occur without indication from self-health checks may suggest self tests that could be effective against upset in future processor designs. Monitoring address bus activity establishes cases in which the processor accesses invalid or nonexistent memory space. When this happens, the processor executes whatever data word it finds there as a valid instruction and often never returns to the correct memory space or correct operation until the system is reinitialized. Monitoring the control lines of the processor establishes the operational mode of the processor and, therefore, enables the experimenter to determine if invalid memory space data has been decoded as an instruction. Exact details of the digital data acquisition are dependent on the controller under test.

In redundant systems with voting, the digital data described above must be obtained from all processors as well as the voter, and reconfiguration data must also be obtained. In redundant systems with primary/secondary channels, the digital data described above as well as the flags and signals related to which channel is in primary control and which is commanding the various control loops must be recorded. Digital data recorded from multiprocessor systems should be time-stamped so that concurrent activities of processors in the system can be correlated for post processing.

III. Case Study: Upset Test Set-up for a Fault Tolerant Engine Controller

The upset test methodology for digital controllers described in Section II is planned to be applied to an electronic engine control (EEC) unit. The EEC is a commercial controller manufactured by the Hamilton Standard Division of United Technologies, which provides electronic controls for Pratt & Whitney engines. The EEC is a full-authority engine controller and is a dual-channel system which operates with a primary/secondary channel strategy. A block diagram of the EEC is shown in Fig. 4. As shown in the diagram, the EEC receives signals from the airframe, actuator position sensors, and engine parameter sensors. The inputs to each channel are also available to the other channel so that the best inputs can be selected by both channels. The control commands are calculated with the selected inputs and one output is selected to be sent to the actuators. In addition to its control function, the EEC performs a comprehensive self-health evaluation during background activity.

The EEC to be used in the test set-up is a modified version of the commercial unit. Modifications to the EEC include access to the data bus, address bus, and control lines of the microprocessors of each channel to enable measurements in the laboratory. In addition, nominal flight parameter values for eight different flight conditions are stored in Read Only Memory (ROM) as well as the nominal values for all but three of the engine parameters. The eight flight conditions to be used during tests are take-off, cruise, acceleration, deceleration, reverse, idle, partial power, and climb. The variable inputs to the EEC are Throttle Resolver Angle (TRA), Inlet Air Temperature (T2), and Engine Speed (N1). These inputs can be varied for the eight flight cases during testing, and will be initially generated as shown in Fig. 5. The TRA input will be generated using a resolver, T2 will be generated using a resistive potentiometer, and N1 will be generated using a pulse generator. Therefore, for initial tests, the EEC will be running open-loop and the calculated commands will be stored in memory. In the next testing phase, these three loops will be closed so that the dynamics of the controller and plant can be simulated in real time. Subsequent plans are to modify the EEC so that additional variable inputs are provided.

During testing, each processor in both the test unit and reference unit will be monitored for activity on the data bus, address bus, and control lines. Upset for the EEC will initially be defined as :

(i) Selected parameter values for N1, T2, TRA is out of range for n cycles;

(ii) Calculated control commands are out of range for the given flight mode for n cycles;

(iii) Invalid memory space is accessed for n cycles.

Indiction of the occurrence of any of these activities on the data bus, address bus, and control lines of the processors in the test unit will result in the data being recorded for that test run. As testing proceeds, the list of activities defining upset for the EEC will be expanded as necessary.

A block diagram of the upset test instrumentation is shown in Fig. 6. The damped sinusoidal waveform of Fig. 3 is generated by a polynomial waveform synthesizer and amplified by a wideband power amplifier with a maximum output power of 1000 W and a frequency range of 10 kHz - 220 MHz. This analog signal is inductively coupled into the

EEC and the induced waveform is recorded on a waveform digitizer/analyzer on which some analysis, such as FFT and energy/power spectrum, can be performed directly. Digital data from the EEC is recorded on a digital analysis system with 240 input lines that can capture data from four microprocessors simultaneously with time correlation. Data can be displayed on the digital analysis system in timing, state, or graphical format. Analog and digital data from the waveform digitizer/recorder and the digital analysis system is then transferred via IEEE 488 bus to a personal computer, which is used for some of the analysis, display of data, and transmission to a VAX 11/750 for further analysis.

IV. Future Work

Upset tests will be performed on the EEC in both an open-loop and a closed-loop configuration in order to compare upset characteristics relative to each of these modes. The analog signals induced on the EEC will be recorded and appropriate norms will be defined which characterize upset stress thresholds. Digital data recorded from the EEC will be scrutinized for selected inputs that are out of range, calculated commands which are inappropriate for the given flight regime, accesses to invalid memory space, and problems which are flagged by self-health tests.

The objectives of initial testing are to demonstrate the methodology, establish an upset data base for a fault tolerant control system, define characteristic induced waveform threshold norms for upset stress, and obtain statistical information about upset in a fault tolerant controller. Long range goals include the development of on-line upset detection and correction strategies, upset tolerant design techniques, an upset assessment tool for data analysis, and an upset reliability estimation procedure.

SUMMARY

An upset test methodology is being developed for fault tolerant control systems and will be applied to the upset test design of an electronic engine controller. The methodology involves generating electrical transients like those that would occur naturally in a lightning environment, coupling these signals into a controller under test, and collecting both analog and digital data from the controller during tests. The primary objective of this methodology is to develop assessment techniques for fault tolerant control systems operating in electromagnetically harsh environments due to lightning, HERF, and NEMP. The primary motivation for the development of assessment techniques is the trend in the aeronautics industry towards flight-critical digital control systems onboard advanced composite aircraft. However, techniques and considerations presented in this paper are applicable to digital control systems for any application.

REFERENCES

1. Hess, R. F., et al., "Sharing the Protection of Aircraft Electronic Systems Against the Effects of High-Level Electromagnetic Environments Between Traditional Protection and System Architecture", Proceedings of the 8th Digital Avionics Systems Conference, October, 1988, pp. 294-307

2. Belcastro, C. M. , "Digital System Upset - The Effects of Simulated Lightning-induced Transients on a General-Purpose Microprocessor", NASA Technical Memorandum 84652, April, 1983

3. Belcastro, C. M. , "Data and Results of a Laboratory Investigation of Microprocessor Upset caused by Simulated Lightning-Induced Analog Transients", NASA Technical Memorandum 85821, June, 1984

4. Hanson, R. J. , "Conducted Electromagnetic Transient-Induced Upset Mechanisms: Microprocessor and Subsystem Level Effects", EOS/ESD Symposium Proceedings, 1987

5. Glaser, R. E. , Masson, G. M. , "The Containment Set Approach to Upsets in Digital Systems", IEEE Transactions on Computers, Vol. C-31, No. 7, July, 1982

6. Schmid, M. E. , et al. , "Upset Exposure by Means of Abstraction Verification", FTCS 12th Annual Symposium Fault-Tolerant Computing, June, 1982

7. Hanson, R. J. , "Subsystem EMP Strength Verification Methods: Upset Detection and Evaluation for Military Subsystems", Kaman Sciences Corp. DC-FR-4088.330-1 (Revised Draft), March 24, 1988

8. Protection of Aircraft Electrical/Electronic Systems Against the Indirect Effects of Lightning, SAE AE4L Committee Report AE4L-87-3, Revision A, October, 1988

9. Test Waveforms and Techniques for Assessing the Effects of Lightning-Induced Transients, SAE AE4L Committee Report AE4L-81-2, December 15, 1981

10. Hobbs, R. A. , "The Design, Construction, Performance and Calibration of Pulse Generators to Fulfill the Requirements of Specifications Defense Standard 59-41, FS(F) 510 and FS(F) 457", RAE TM FS(F) 550, September, 1988

11. Ketterling, George W. , "EM Transient Protection Requirements for Avionics LRUs", IEEE AES Magazine, April, 1986

12. Environmental Conditions and Test Procedures for Airborne Equipment, EUROCAE ED 14, Proposal to SAE AE4L Committee for DO 160C, Section 22, April, 1989

13. Thomas, R. E. , Diloreto, A. G. , "EMP Upset: Overview and Test Methodologies", AFWL-TR-84-60, March, 1985

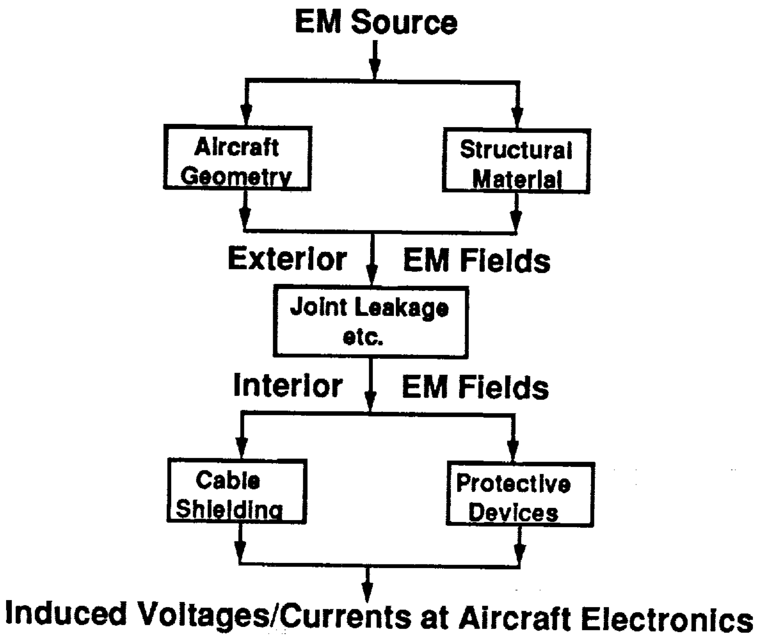


Figure 1: Coupling of Electromagnetic Fields into Aircraft

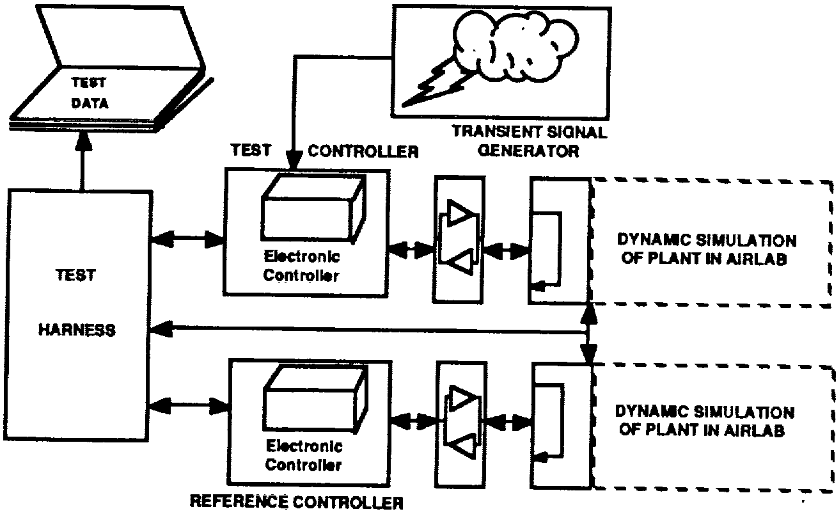
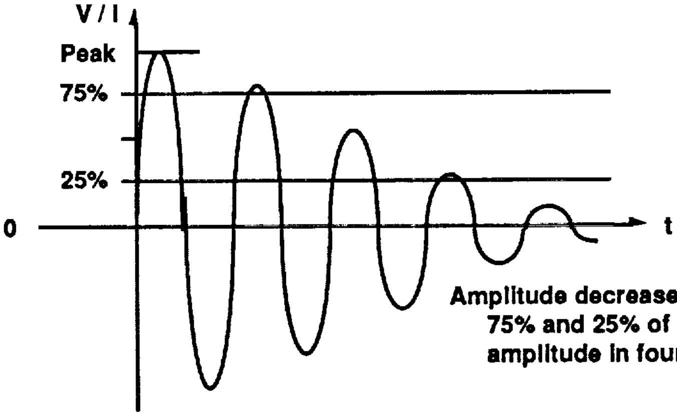

TEST DATA
TRANSIENT SIGNAL GENERATOR
TEST CONTROLLER
TEST HARNESS
Electronic Controller
DYNAMIC SIMULATION OF PLANT IN AIRLAB
Electronic Controller
DYNAMIC SIMULATION OF PLANT IN AIRLAB
REFERENCE CONTROLLER

Figure 2: Laboratory Configuration for Control System Upset Testing



Amplitude decreases to between 75% and 25% of peak amplitude in four cycles

Frequency (MHz)	Method	Purpose
10 (± 20%)	Pin/Bulk Cable	Damage/Upset
1 (± 20%)	Pin/Bulk Cable	Damage/Upset
1 - 50	Bulk Cable	Upset

Figure 3: SAE-AE4L Committee Damped Sinusoidal Voltage/Current Waveform

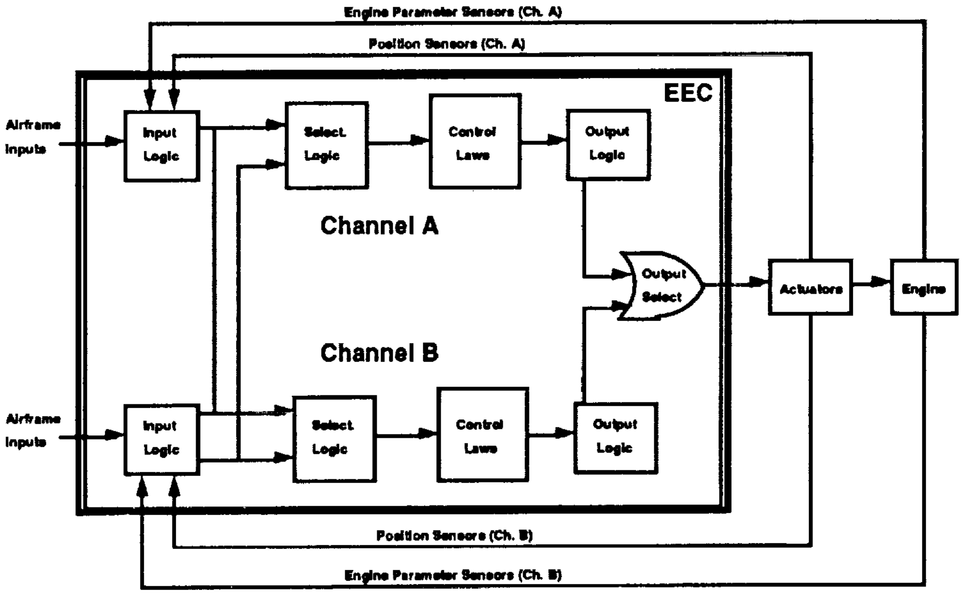


Figure 4: Block Diagram of the Electronic Engine Controller (EEC)

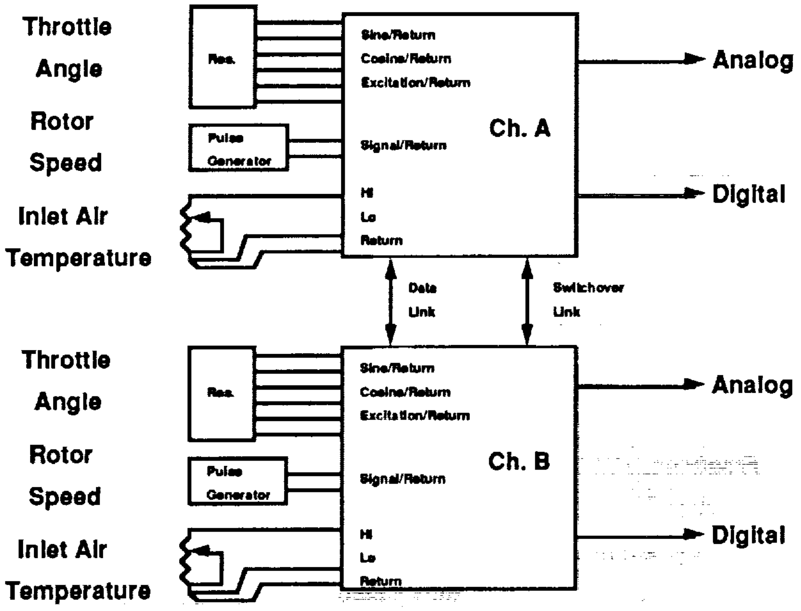


Figure 5: Simulated EEC Inputs for TRA, N1, T2

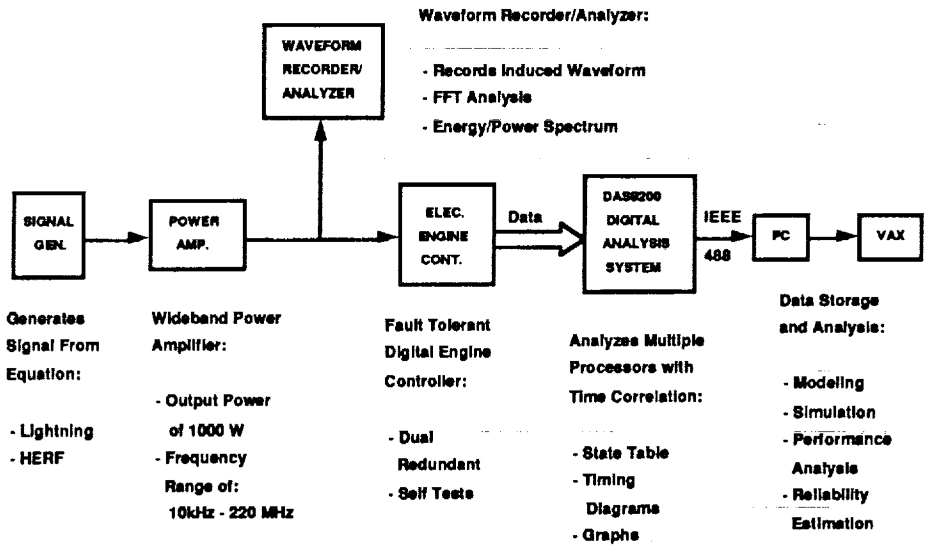

WAVEFORM RECORDER/ ANALYZER
Waveform Recorder/Analyzer:
- Records Induced Waveform
- FFT Analysis
- Energy/Power Spectrum
SIGNAL GEN.
POWER AMP.
ELEC. ENGINE CONT.
Data
DAS9200 DIGITAL ANALYSIS SYSTEM
IEEE 488
PC
VAX
Generates Signal From Equation:
- Lightning
- HERF
Wideband Power Amplifier:
- Output Power of 1000 W
- Frequency Range of: 10kHz - 220 MHz
Fault Tolerant Digital Engine Controller:
- Dual Redundant
- Self Tests
Analyzes Multiple Processors with Time Correlation:
- State Table
- Timing Diagrams
- Graphs
Data Storage and Analysis:
- Modeling
- Simulation
- Performance Analysis
- Reliability Estimation

Figure 6: Upset Test Instrumentation for EEC Assessment

ORIGINAL PAGE IS
OF POOR QUALITY

N90-23070

JPL Control/Structure Interaction Test Bed

Real-Time Control Computer Architecture

Presented at the

3rd Annual Conference on

Aerospace Computational Control

Oxnard, CA

28-30 August, 1989

By

Dr. Hugh C. Briggs, Deputy Manager

JPL CSI Program

Jet Propulsion Laboratory

California Institute of Technology

Pasadena, CA

Control/Structure Interaction Test Bed
Real-Time Control Computer Architecture

Hugh C. Briggs, Deputy Technical Manager
Jet Propulsion Laboratory
4800 Oak Grove Drive
Pasadena, CA 91109

Abstract

The Control/Structure Interaction Program is a technology development program for spacecraft that exhibit interactions between the control system and structural dynamics. The program objectives include development and verification of new design concepts - such as active structure - and new tools - such as a combined structure and control optimization algorithm - and their verification in ground and possibly flight test. A focus mission spacecraft has been designed based upon a space interferometer and will be the basis for design of the ground test article. The ground test bed objectives include verification of the spacecraft design concepts, the active structure elements and certain design tools such as the new combined structures and controls optimization tool. In anticipation of CSI technology flight experiments, the test bed control electronics must emulate the computation capacity and control architectures of space qualifiable systems as well as the command and control networks that will be used to connect investigators with the flight experiment hardware.

The Test Bed facility electronics have been functionally partitioned into three units: a laboratory data acquisition system for structural parameter identification and performance verification; an experiment supervisory computer to oversee the experiment, monitor the environmental parameters and perform data logging; and a multilevel real-time control computing system. The design of the Test Bed electronics is presented along with hardware and software component descriptions. The system should break new ground in experimental control electronics and will be of interest to anyone working in the verification of control concepts for large space structures.

The NASA Control/Structure Interaction Program

The NASA CSI Program is an element of the Control of Flexible Structures Task in the NASA Civilian Space Technology Initiative. Three NASA Centers participate in the CSI Program: Langley Research Center, Marshall Space Flight Center and the Jet Propulsion Laboratory. This multiyear program to develop and validate new design technologies is organized around five elements: Systems and Concepts, Analysis and Design, Ground Test Methods, Flight Experiments and Guest Investigation Program. The CSI program goal is to develop validated technology that will be needed to design, verify and operate interactive control/structure systems to meet the ultraquiet structure requirements of 21st century NASA missions.

The CSI Program will integrate the advances made in other discipline technology programs to make the new spacecraft design methodology (see Figure 1). Controls programs such as Computational Control will develop a new generation of tools for multibody simulation, multibody component representation, and control analysis and synthesis. Structures technology programs such as Computational Mechanics are developing advanced finite element analysis codes. CSI will integrate these tools into a multidisciplinary environment and develop additional tools such as simultaneous structure and control optimization methods and conceptual design tools for flexible spacecraft structure/control architectures. New CSI systems and concepts, such as active structure, will be developed and integrated into the focus mission design example.

Other developments that will enable high-performance, flexible spacecraft design include an investigation of microdynamics and development of ground test methods for controlled flexible spacecraft structures. Microdynamic characterizations of spacecraft components such as joints and struts will identify the linearity of typical elements when dynamic motions are restricted to the submicron regimes required for future spacecraft. In addition, disturbance sources will be characterized at the microdynamic level to support analysis of ultraquiet spacecraft systems.

Implementation of the CSI Methodology - The Design Environment

The design environment is an important element of the methodology and consists of several elements. The following section will address the computer systems and the laboratory testing facilities. The software and analytical tools are described in the companion paper on the design methodology. [1]

The CSI computer system is a distributed network-based system consisting of workstations and servers (Figure 2). Laboratory testing computers are attached to the network to support the close integration of verification tests to the development of systems concepts and tools. Sufficient commercial technology exists to support a heterogeneous equipment set based upon standard network interfaces. For example, systems from Apple, DEC, Sun, Apollo, HP and others can all participate in an Ethernet network using TCP/IP. (For a brief description of terms, see the glossary.) This capability supports various user preferences and capabilities and provides the mechanism to protect existing corporate investments in computer systems.

The distributed system utilizes servers for those functions not allocated to the per-engineer workstations. Large computers, such as a CRAY or departmental VAX, function as compute servers to provide an execution site for large, computationally intensive jobs. Other servers might provide specialized capabilities for animation, data base management or communications. Most workstation companies make it financially attractive to collect most of the system disk resources in one or more file servers that support some form of a network disk system (eg. Sun's NFS). These file servers are repositories for large data sets, system executables and application libraries.

The workstations must support the interactive design environment with excellent speed and graphics. The CSI methodology requires computation of intermediate sized (ie. 100+ states)

[1] "Control/Structure Interaction Design Methodology," H.C. Briggs and W.E. Layman, Proceedings 3rd Annual Conference on Aerospace Computational Control, Oxnard, CA, 28-31 August, 1989.

problems and presentation of solid models on the workstations. The derived requirements for workstations are: 3-10 MIP 32 bit CPU, 12-16 Mb memory, Unix operating system, 200 Mb disk, Ethernet interface, 3-D vector graphic accelerator and windowed presentation manager with a mouse.

The network environment also extends into the laboratory where verification and validation experiments are executed on the CSI Test Bed. The computing environment internal to the lab is shown in Figure 3. The four functions are: real-time control, experiment supervision, modal analysis and software development. Individual systems can be readily purchased to perform each function although it is possible to configure certain commercial systems to perform multiple duties. In any case, the software development system will probably not be instantiated in the laboratory, using individual analyst workstations and the experiment supervisory computer instead.

The real-time control computer system will be described in more detail in the following sections but a summary is presented here to support the environment description. The control computer will be a distributed, multiprocessor computer based upon commercial VMEbus products. The operating system supports remote consoles, software loading, a prioritized scheduler and shared memory message passing. An excellent example is VxWorks from Wind Rivers although the underlying kernel requires additional multiprocessor extensions. Analysts will prepare simple control subroutines on their workstation and produce a load module just as they would any program for execution. Remote login facilities are provided for access to any real-time CPU and a C-like shell provides the operator interface. Products such as DbxWorks provide source level symbolic debugging.

The experiment supervisory computer provides the laboratory operator console and overall control of the Test Bed. This system monitors and logs environmental variables such as temperature and air velocity, monitors a panic button during experiment execution and collects measurements from the external truth sensor. Remote access from any network workstation allows remote execution of experiments.

The modal analysis and data acquisition system is a standard commercial product and supplies a necessary function found in all dynamics laboratories. To characterize the structural dynamics of the test article, a modal survey can be performed utilizing a large number of accelerometers distributed over the structure. This is typically done to verify open loop system models but should also be an integral part of closed loop system performance measurement. Results are available to any analyst via the network.

The laboratory environmental requirements are quite severe. Noise and seismic disturbance constraints will require all personnel and actively cooled electronics to be in an adjacent control room. During tests, the test chamber must be unoccupied, closed, and carefully maintained at constant temperature. This will require development of control procedures for remote experiments and forms the basis for emulation of on-orbit flight experiments. The essential Shuttle command, communication, and control features can be readily emulated with the network-based computer system and the computational capabilities of space-qualified computers can be replicated in the ground test hardware. Figure 4 illustrates the scale and complexity of a test bed that models a space-based interferometer.

CSI Testing Requirements

CSI will validate the system concepts, components and tools in realistic ground tests. Where the ground environment precludes acceptable verification due to such effects as the gravity field, seismic and acoustic disturbances and size limitations, flight tests will be proposed to complete the development and validation of the technology.

Testing is recognized as an essential component of the design process. The design methodology will include close coupling of the analysis with the testing and evaluation of results. This will foster verification of new system concepts and designs as well as provide analytical support for new ground test techniques. In addition, the CSI flight experiments will be designed to develop techniques for extending ground testing methods to on-orbit flight tests.

As a result of integrating testing into the design process, several capabilities must be built into the ground test facility. Interactive evaluation of control system performance must be provided to explore system phenomena and to enable reconciliation of measured behavior with predicted behavior. To validate the new optimization methods and to evaluate system robustness properties, substitution of any structural element will be provided without dismantling large subsections of the test article. Support for remote investigation of system performance via the electronic network, already mentioned as a requirement for CSI analysts, will also include support for off-site Guest Investigators. This access includes all test measurement data as well as the control programs of the real-time control computer. Finally, emulation of all essential Shuttle command, communication, and control features that impact proposed flight experiments will be provided.

RTC Requirements

Given the CSI program goals and testing requirements, several requirements for the real-time controller are presented in the following. Most are functional requirements that have shaped the system architecture although the section will be closed with a general statement of the computational requirements.

The real-time controller and the Test Bed are an integral part of the CSI design environment. As such they are part of a distributed network system that must support remote access and remote software development. This has been stated for the experiment supervisor and the data acquisition system but also applies to individual control CPUs. The Test Bed control electronics will execute selected portions of the FMI control functions including path length control, wavefront streering and active structure control. In addition, the system may be called upon to generate certain disturbance profiles for actuators. Because the system must serve as a validation host for spacecraft system designs, analysis tools, and methods, it must have a reconfigurable topology, flexible software, and a range of speed capabilities. For example, in addition to supporting ground test with the best possible execution speed, the system must emulate the restricted computational capabilities expected in the flight experiment environment.

The software development environment for the real-time control system must be particularly simple. Transparent cross-compilation and mainstream languages such as C and

Fortran are required. Since the system connectivity will be quite complex, remote symbolic source level debugging tools will be required. Libraries are required for I/O interface routines and kernel access functions. In summary, software development for the verification of new control techniques must be well supported to minimize costs.

The computational requirements are readily stated in terms of control loop parameters as follows. Although converting these to speed and size requirements for system components is not straightforward, a certain amount of experience with similar systems exists and a prototype of the controller is proposed to verify system performance issues. The Test Bed control system consists of many loosely coupled control loops. These typically utilize less than 10 states and 5 sensors and actuators. The fastest loop operates at a maximum of 600 Hz. The exception to the typical loop is the active structure controller that drives the active struts. This loop can control up to 20 struts, each having one actuator and two sensors. This requirement is made significantly more demanding by the possibility of loop rates to 1000 Hz although only one instance of an active structure controller will exist at any time. Single precision (32 bit) floating point arithmetic is acceptible and analog interfaces operate with 16 bit integers.

System Functional Architecture

Figure 5 shows the functional units of the laboratory in the top level structure chart. Compare this figure with Figure 3, Test Bed Computing Environment. The functions are shown as boxes and the real-time controllers are shown as the object labelled 2.0. Signal paths are shown as directional arrows and the signal content is indicated with text labels. At this high level, the controllers and the test article are simply shown exchanging the data "Controls - Actuators & Sensors."

This figure also shows the network connections between the analysts (represented by the stub "World Access") and the supervisory computer, real-time controllers and the modal analysis & data acquisition system. The analysts access each system as remote consoles using the remote login services of the network. The experiment supervisory computer and the data acquisition system each have local operator consoles.

The second level structure charts provide further definition of unit functions. For example, Figure 6 shows more details of the real-time controllers and contains objects labeled 2.x to show the heirarchy. The six functions in Figure 6 are taken from the FMI Control System Functional Diagram reproduced as Figure 7. Most of the information exchanged between functions in the FMI diagram represent mechanical or optical mechanisms and have been deleted in the structure chart to leave only information to be transmitted by the real-time control system.

Each of the six functions shown in Figure 6 has been further defined in lower level structure charts and one of these is shown in Figure 8. The active struts used in the active structure control might have local controllers which can command the strut motor and read the strut sensors. Each strut loop might be designed to present an idealized actuator to the block labeled " 2.31 Active Structure Control" which provides the strut commands.

Implementation Concept

The functional structure charts break the real-time control into progressively smaller pieces which can be assigned to hardware units for execution. Prior to this, however, an implementation concept must be selected. This will consist of choices for compute elements, busses, interconnects and software.

The CSI control system will be purchased as commercial items to the maximum extent possible. This policy should lead to reasonable costs, short development time and minimum work force requirements. Economical products with sufficient capacity exist at the board level, although custom integration of the desired components does not seem to be offered at economical prices. This also applies to the system software where operating systems and device interfaces are commercially available.

The implementation (see Figure 9) will provide separate communication channels for operator access and real-time data. Each CPU will be logically attached to the network with Ethernet connections to each chassis. This channel will be used for operator login, software loads, status display and debugging. Within a chassis, the network will be carried over the back plane with one CPU designated as the gateway.

The real-time data will be passed over the back plane as shared memory messaging. This leads to a need to locate CPUs with large message volumes in the same chassis and to extend chassis with bus bridges where chassis capacities are exceeded. Bus bridges that utilize the multi-master capabilities of the back plane are required to avoid degrading overall system performance.

The software implementation will utilize a commercial real-time kernel and development environment such as VxWorks. The kernel requires multiprocessor extensions and support for a wide variety of CPUs. VRTX with MPV meets these needs. Analog I/O interface routines will be based upon vendor supplied libraries and typically require only simple services such as single sampling of sequential channels. A standard control module will be written based upon a linear, constant coefficient update law and supplied to analysts. This template can be sized as required and filled with data to implement most common control laws.

RTC Typical Crate

This implementation concept leads to a standard chassis to host the functions contained in the structure charts. Approximately ten chassis might be required to house all of the units shown in Figure 9. Each chassis - or "crate" - will be configured similarly but execute a particular control function and be connected to different sensors and actuators.

The typical chassis is shown in Figure 10 and utilizes a 21 slot VMEbus rack mounted cabinet. Each chassis will have a CPU to allocate crate resources and handle network communications. This CPU will drive the chassis Ethernet controller and serve as a gateway, providing TCP/IP access to other chassis CPUs over the back plane. Chassis resources might consist of one or more array processors and 4 to 16 Mbytes of memory.

Each chassis might contain 3 to 5 additional CPUs to host control functions. Analog I/O devices are provided for each CPU to achieve maximum system speed and avoid contentions. These CPUs will initially utilize fast MC68030 processors with MC68882 floating point units and fast local memory. Future upgrades will augment this with DSP or RISC based single board computers supported by the development system and the real-time kernel.

The software on each SBC consists of the VRTX kernel with MPV, the VxWorks shell, device drivers, and the control loop code. The kernel supports task scheduling and message passing while the shell handles the operator interface and the symbol table. SBCs with at least 1 Mbyte of memory provide sufficient storage to meet these needs.

Software Implementation

Software development for complex real-time systems such as the CSI Test Bed can pose many problems. This system is clearly multirate and consists of many loosely coupled control loops. Without care, the system software can prove to be unmanageable.

The objectives of the software implementation design are all oriented toward simplifying the resulting system. The modularity of the structure charts will be followed by coding individual functions. This will promote structure, independence and simplicity in the resulting software system. The Test Bed is a research and development vehicle and must support evolving and untried algorithms. To meet throughput requirements, the software must execute in multiprocessor hardware and achieve maximum possible speed consistent with a simple, high level language software development environment. Finally, selected sections of the software must be hosted in the future on advanced DSP or RISC processor hardware.

The selection of a common real-time kernel with an operator shell and development environment supports these objectives. Standard procedures for kernel access, message passing and resource allocation are provided. Beyond this, analog I/O routines and other locally written standard modules will be maintained in libraries. Initially, a simple control law will be written and supplied as a template. The coefficients and size of this law can be changed but the module interface can be standardized.

Control Law Examples

The constant coefficient, linear update control law is illustrated in Figure 11. The module consists of four separate functions for input, state estimation, control generation and output. The modules pass prearranged data messages via the kernel messaging services. Each module is seen by the kernel scheduler as a process that, after initialization, is sleeping while waiting for the arrival of the messages. Each function executes sequentially but asynchronously with the exception of the input which waits for the next time slice to begin. If synchronous output is required, the final module could also be scheduled on time slice boundaries.

The software for each module in this control law can be readily standardized. The device interface library can store drivers that have been parameterized by base address and number of channels. The estimater and control routines require the actual coefficients of individual control laws but are based upon the standard template. With this structure, future changes should be localized to a few modules.

For certain control architectures, this software can be readily extended to a multiple processor hardware system. An example based upon decentralized control is shown in Figure 12. The estimator and control routines have been replicated and instantiated with the specific coefficients of two controllers. The message passing services of the kernel are used to pass the real-time data and control the execution sequencing of the routines. In this case, the message addresses include the CPU number but are otherwise unchanged from the single processor example. The execution sequence is again initiated by the input routine which is scheduled for the next time slice.

The capability to easily prepare multiprocessor (and multirate) systems is based upon the multimaster features of the VMEbus hardware and the multiprocessor services of the real-time kernel. Complex systems can be written, checked out in a modular fashion on a single CPU and expanded to multiple CPU systems after the logic and data structures have been verified. With care, loosely coupled control loops should realize near-linear increase in speed with the addition of CPUs.

Closing Remarks

A system cost model has been built based upon a spread sheet program. Hardware is allocated to crates at the board level and chassis capacity and total costs are calculated. A separate spread sheet contains a library of board components with note annotations documenting vendor, configuration, and discounts. This cost model was used to support quantity, capability and scope trade studies in a very effective, interactive manner.

A system prototype will be constructed prior to a critical design review in March 1990. This prototype will be used to verify performance and implementation issues such as CPU capacity, interconnect speeds and the complexity of the operator interface. The real-time control system must be fabricated prior to the initial turn-on of the Test Bed in October 1991 for system check-out.

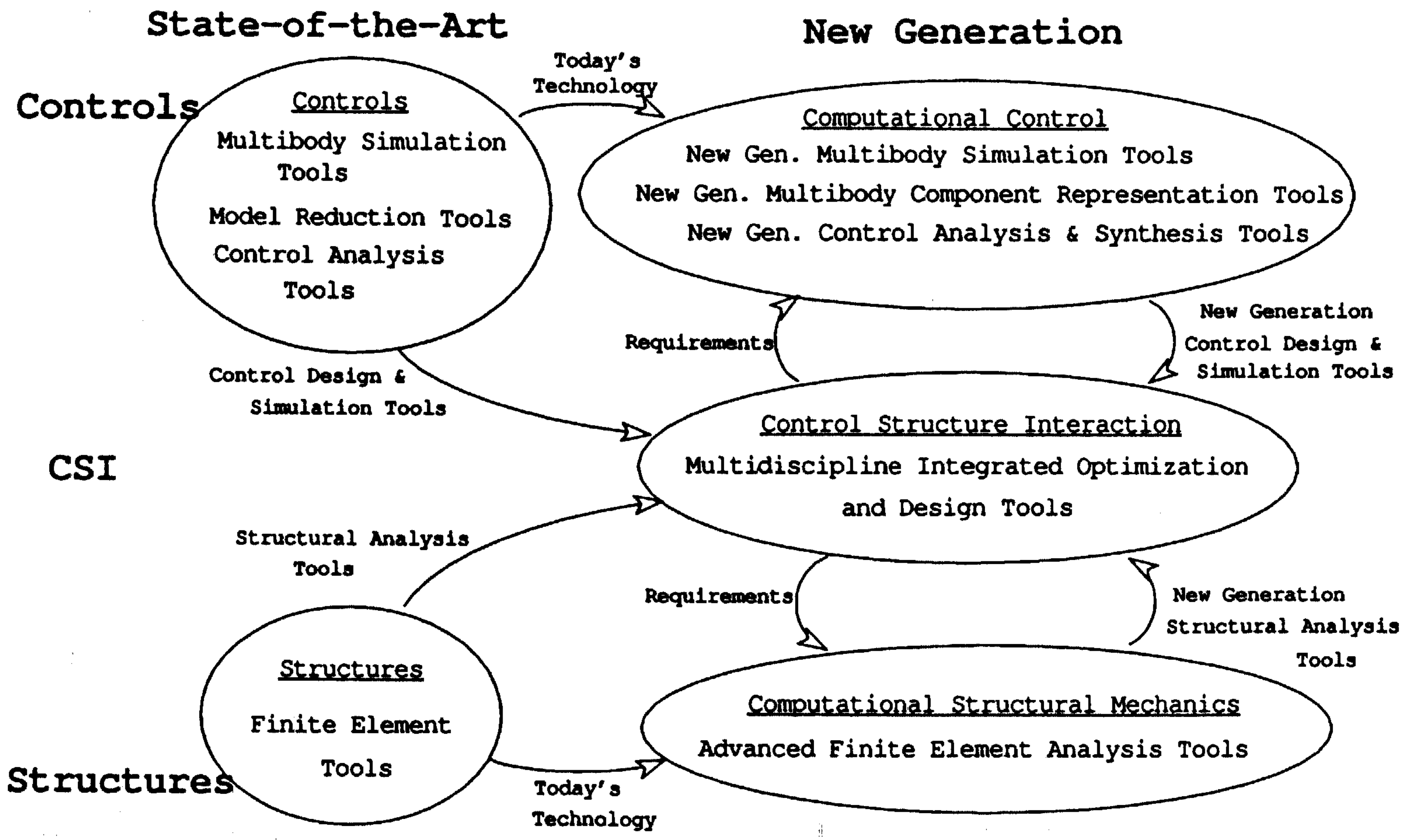
Acknowledgement

The research described in this paper was performed as part of the Control Structure Interaction (CSI) program at the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

Glossary

CPU - Central Processing Unit, the core of a computer.
DSP - Digital Signal Processor, a small fast CPU typically for embedded control applications.
FMI - Focus Mission Interferometer, a CSI design example.
MPV - Multiple processor extensions to the VRTX kernel.
RISC - Reduced Instruction Set Computer - new fast CPU.
SBC - Single Board Computer.
TCP/IP - Ethernet protocol for local area networks.
VRTX - Real-time kernel from Ready Systems.
VxWorks - Real-time operating system from Wind Rivers.

Figure 1. Relationship Between Controls, Structures, and CSI Tools


State-of-the-Art
New Generation
Controls
Controls
Multibody Simulation Tools
Model Reduction Tools
Control Analysis Tools
Today's Technology
Computational Control
New Gen. Multibody Simulation Tools
New Gen. Multibody Component Representation Tools
New Gen. Control Analysis & Synthesis Tools
Requirements
New Generation Control Design & Simulation Tools
Control Design & Simulation Tools
CSI
Control Structure Interaction
Multidiscipline Integrated Optimization and Design Tools
Structural Analysis Tools
Requirements
New Generation Structural Analysis Tools
Structures
Structures
Finite Element Tools
Today's Technology
Computational Structural Mechanics
Advanced Finite Element Analysis Tools

ORIGINAL PAGE IS
OF POOR QUALITY

Figure 2. CSI Computing Network

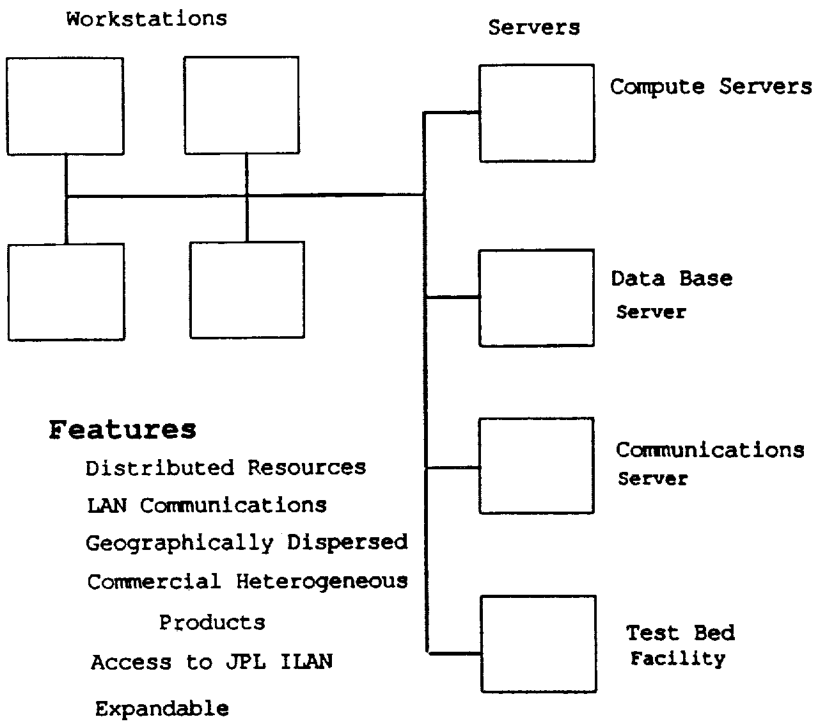
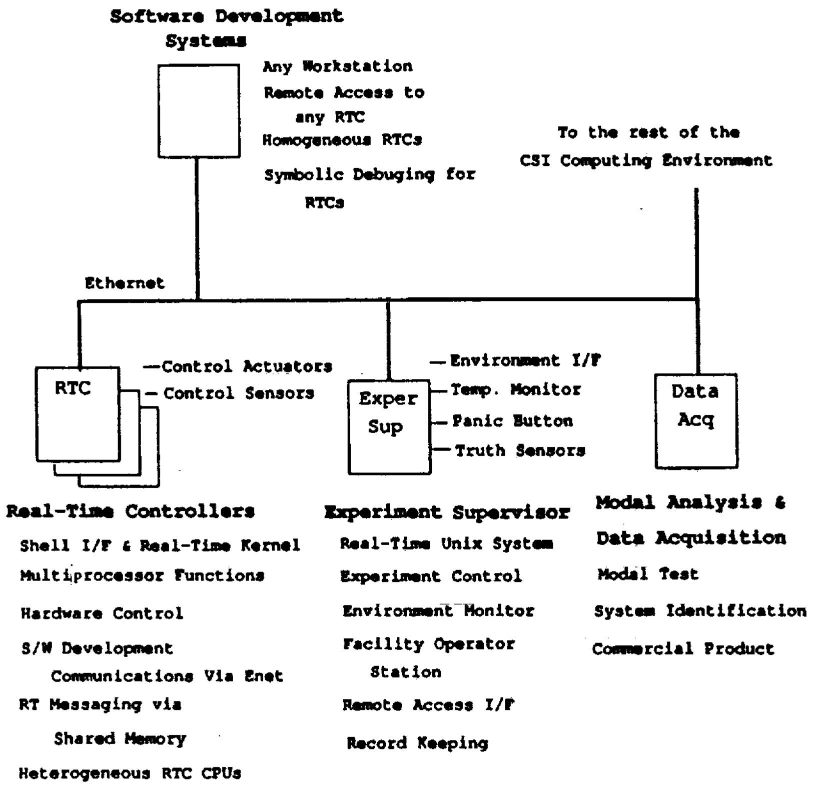


Figure 3. Test Bed Computing Environment



JPL

Figure 4. Integrated Controls and Structures Laboratory

CARL

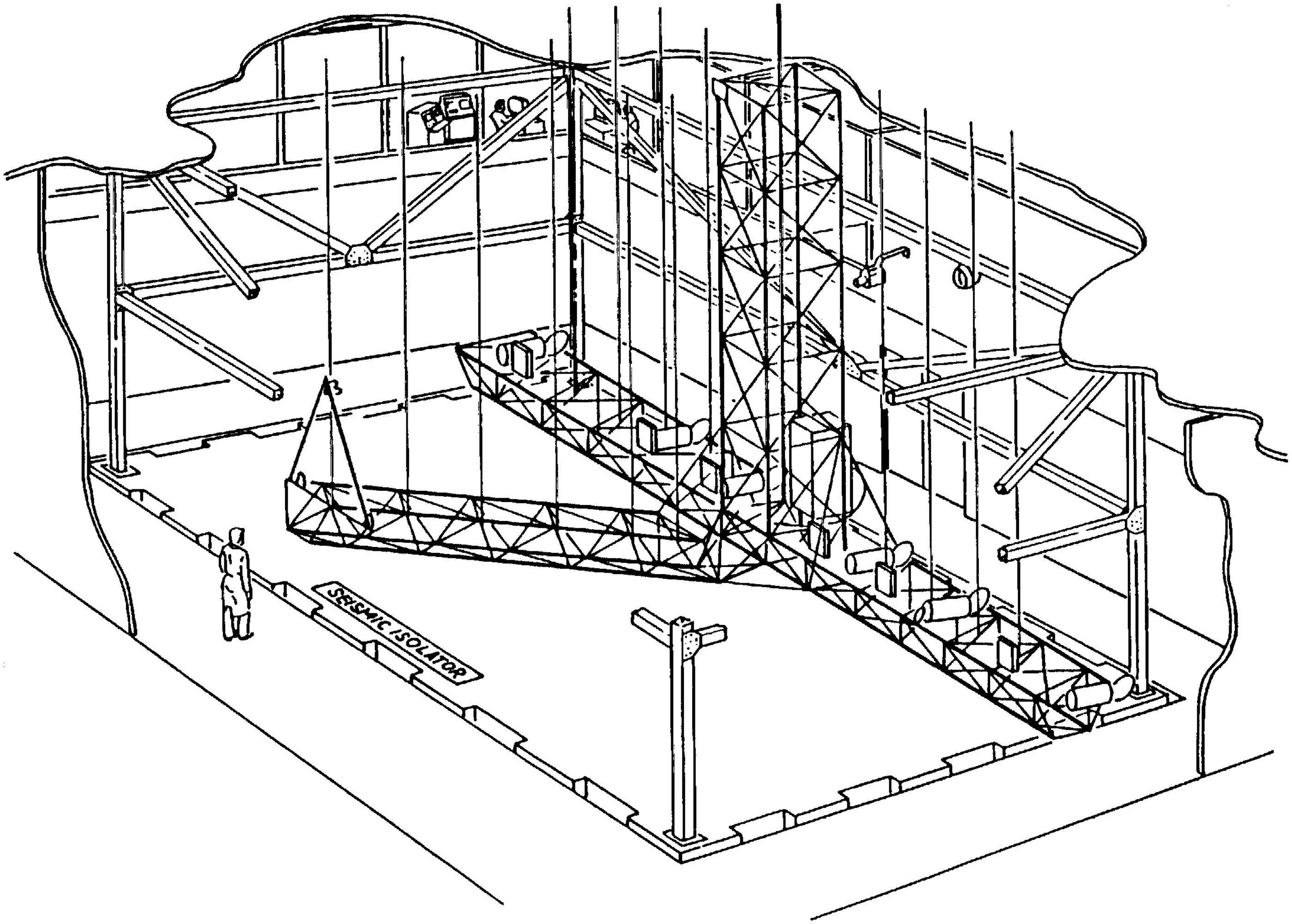


Figure 5. RTC Laboratory Environment

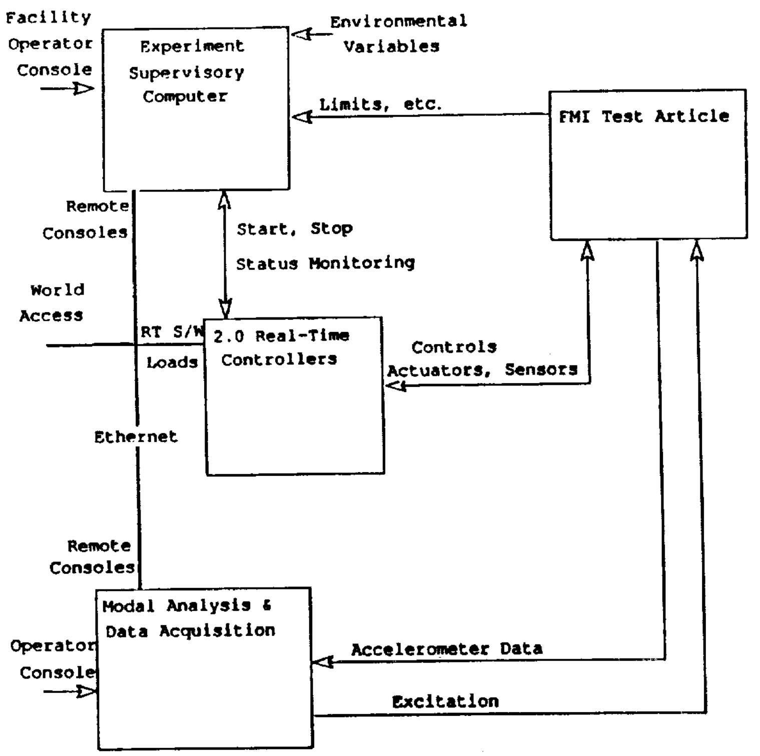
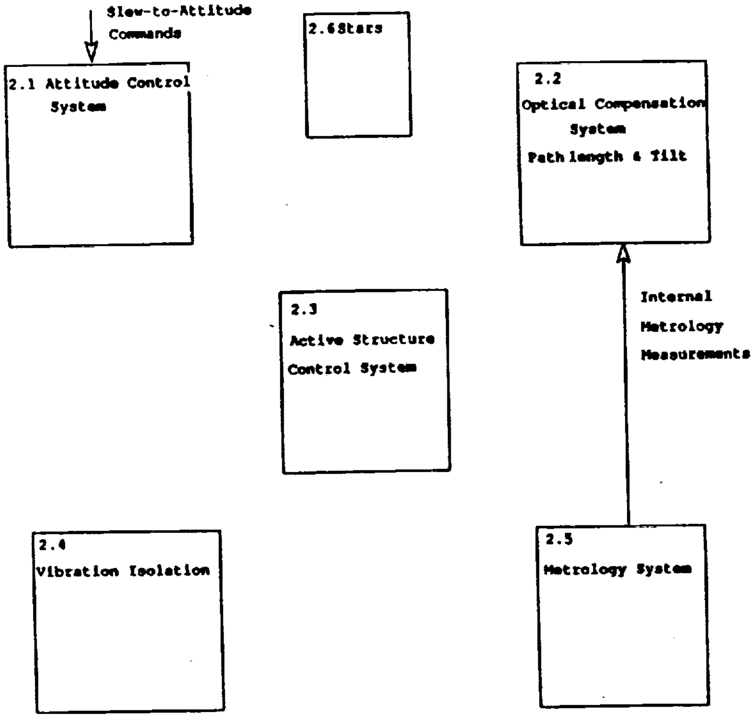


Figure 6. 2.0 Real-Time Controller*


Slew-to-Attitude Commands
2.1 Attitude Control System
2.6Stars
2.2 Optical Compensation System Path length & Tilt
2.3 Active Structure Control System
Internal Metrology Measurements
2.4 Vibration Isolation
2.5 Metrology System

*Based on the FMI Control System Functional Diagram

JPL

Figure 7. Focus Mission Interferometer Top Level Functional Block Diagram

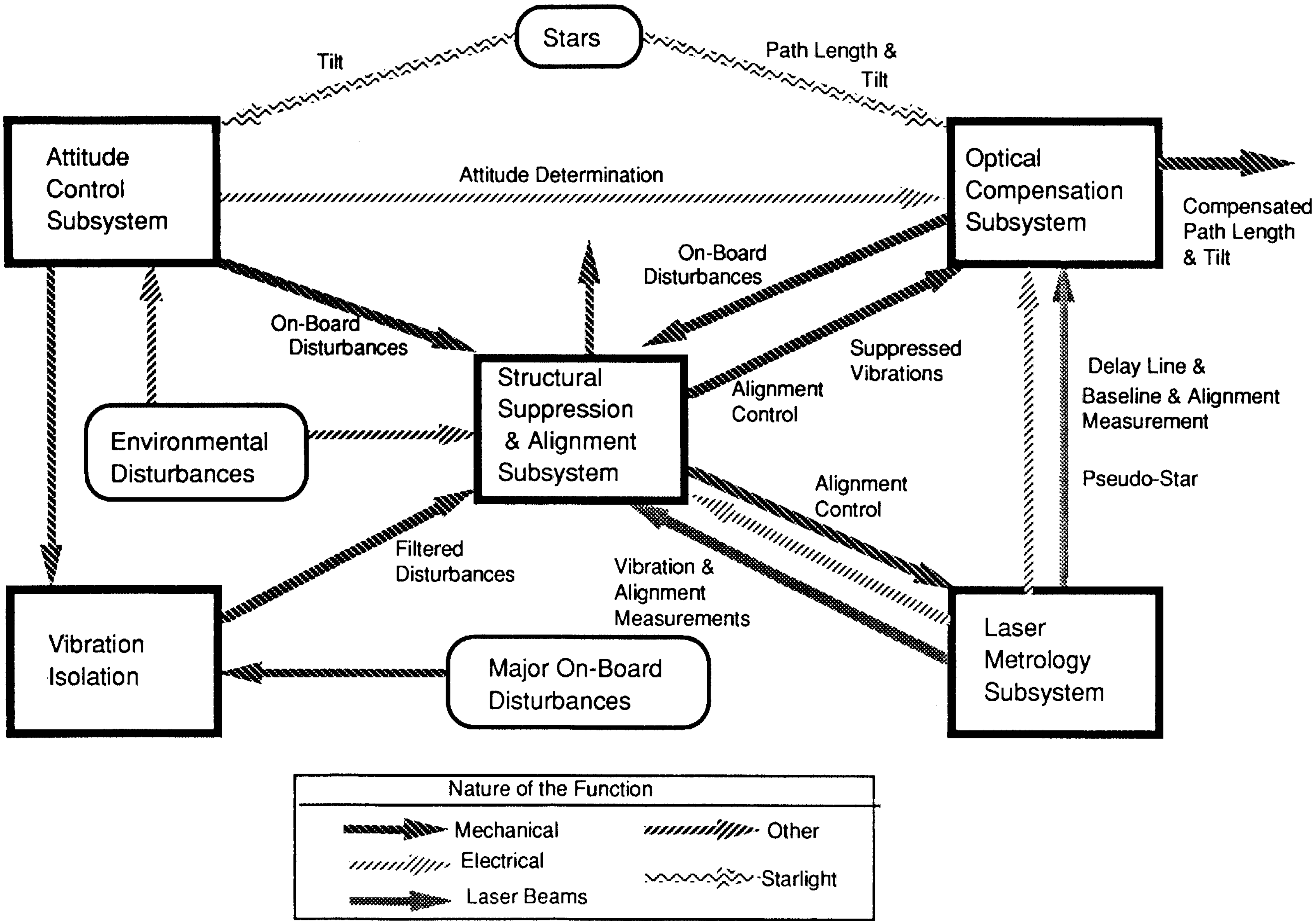

Stars
Tilt
Path Length & Tilt
Attitude Control Subsystem
Attitude Determination
Optical Compensation Subsystem
Compensated Path Length & Tilt
On-Board Disturbances
On-Board Disturbances
Suppressed Vibrations
Alignment Control
Delay Line & Baseline & Alignment Measurement
Pseudo-Star
Environmental Disturbances
Structural Suppression & Alignment Subsystem
Alignment Control
Filtered Disturbances
Vibration & Alignment Measurements
Vibration Isolation
Major On-Board Disturbances
Laser Metrology Subsystem
Nature of the Function
Mechanical
Electrical
Laser Beams
Other
Starlight

Figure 8. 2.3 Active Structure Control System

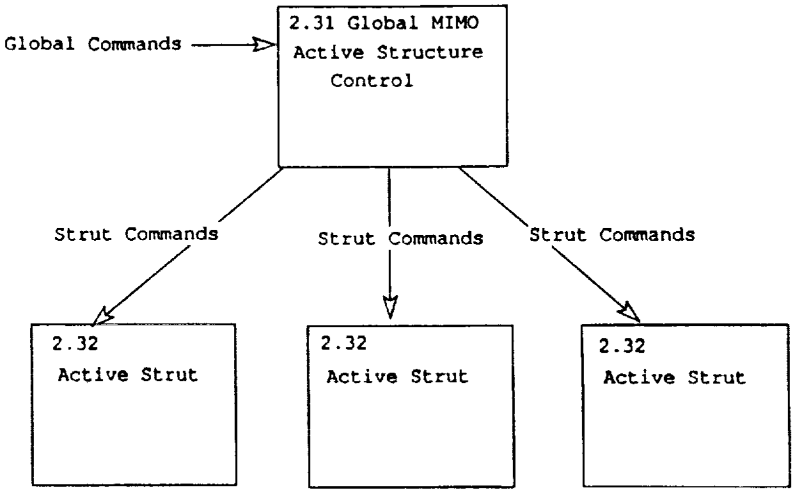


Figure 10. RTC Typical Crate

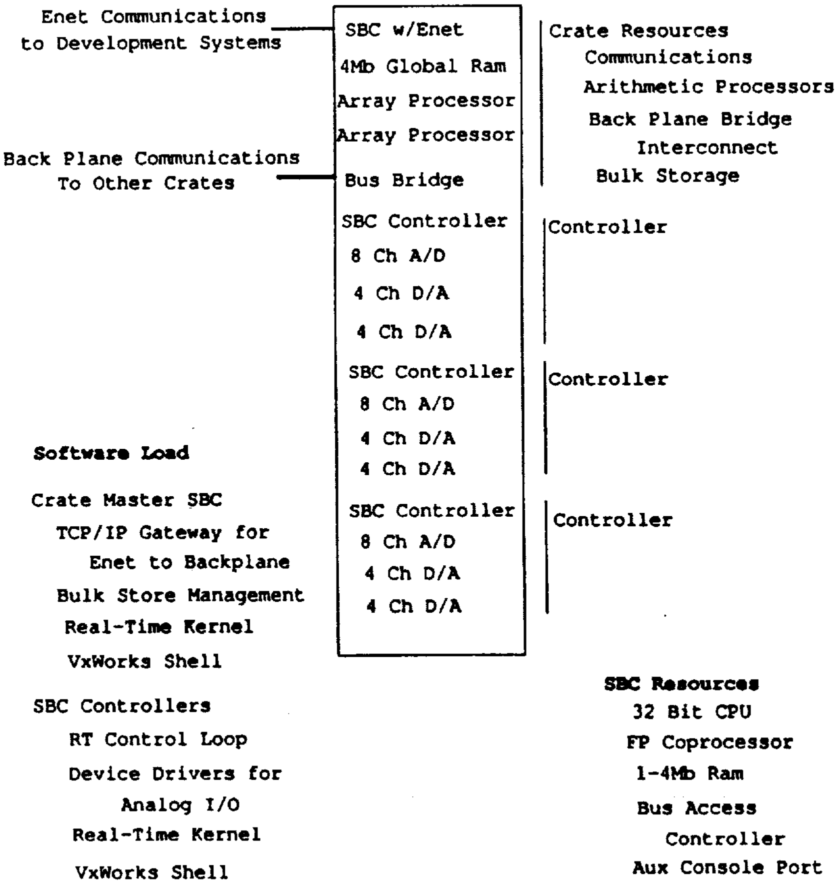


Figure 9. RTC Implementation Concept

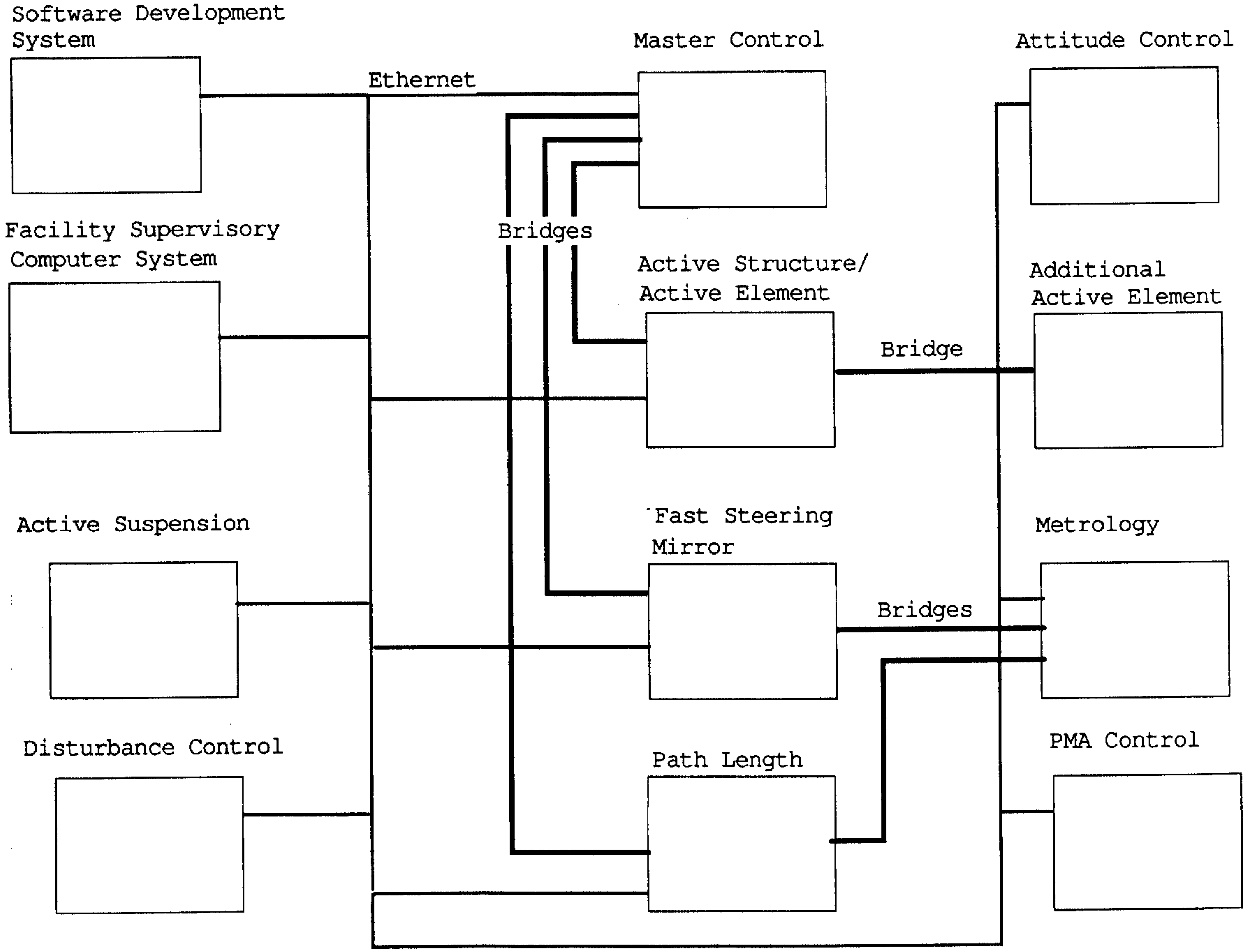


Figure 11. RTC Control Software
Functional Structure

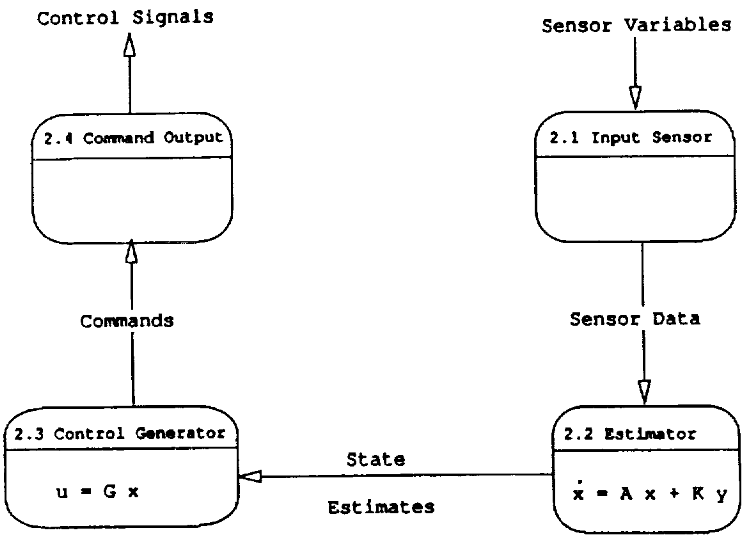
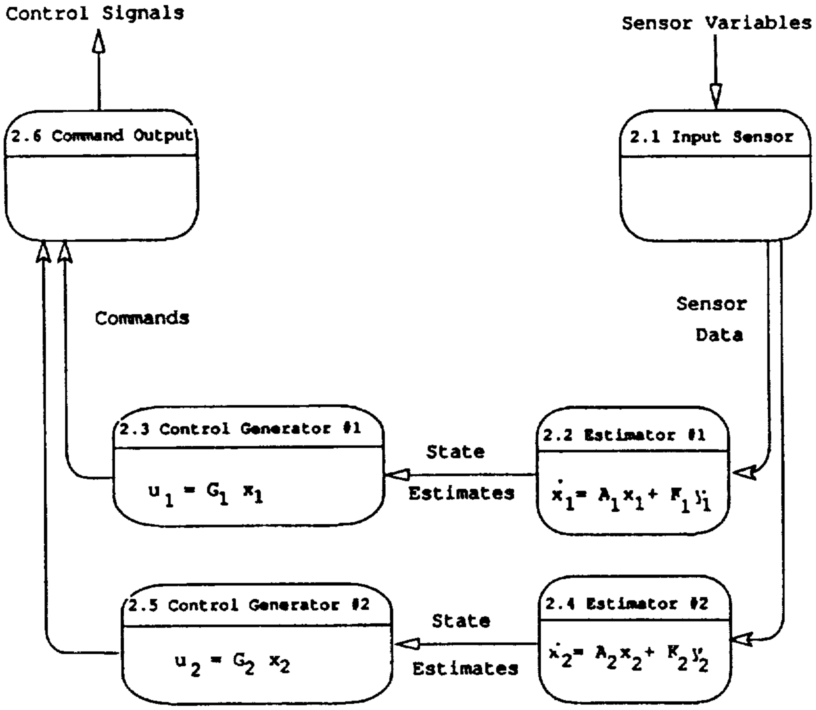


Figure 12. RTC Control Software
Decentralized Control Example



ORIGINAL PAGE IS
OF POOR QUALITY

N90-23071

Faster Simulation Plots

Richard A. Fowell
Hughes S&CG
SC S12 V362
PO Box 92919
Los Angeles
CA 90009

Abstract

Most simulation plots are heavily oversampled. Ignoring unnecessary data points dramatically reduces plot time with imperceptible effect on quality. The technique is suited to most plot devices.

We tripled our department's laser printer's speed for large simulation plots by data thinning. This reduced printer delays without the expense of a faster laser printer. Surprisingly, it saved computer time as well. We now thin all plot data, including PostScript and terminal plots.

We describe the problem, solution, and conclusions. Our thinning algorithm is described and performance studies are presented. To obtain Fortran 77 or C source listings, mail a SASE to the author.

Introduction

Post-processed simulation plots have hundreds or thousands of points. Plotting time ranges from noticeable to intolerable, but faster plots are universally desirable. This paper describes how we reduce plot delays when most of the plot time is used to plot the data curves. Our approach is to thin out uninformative data points, and plot an approximate curve with fewer points. The thinned curve will be indistinguishable from the original if the approximation tolerance is set to a sub-pixel level. When the time required to plot the data is significant, and the data is transmitted to the plot device as coordinate pairs (non-raster) this approach is well worth considering. For our simulation plots, specifying a maximum approximation error of a quarter pixel typically allows elimination of two-thirds of the points. We use a fast, (>10,000 points/sec on a VAX 11/780) compact, (55 lines of FORTRAN) graphical data compression algorithm related to those used for image processing and spacecraft telemetry.

Problem Background

As a spacecraft attitude control systems department, we produce hundreds of post-processed plots each month from our dynamics simulations, as well as telemetry and frequency domain plots. The plot data is generated on our IBM 3090, AD-100 simulation processor, VAXs, Sun workstations, test equipment and spacecraft hardware. It is plotted on graphics terminals, workstation screens and laser printers. The plots are time series, parametric, linear, polar and logarithmic. We now use data thinning on almost all of these plots.

The long time required for simulation plots was originally taken for granted. It was not until we started using laser printers for plots that the problem became severe enough to receive serious attention. In 1985 our department bought a QMS Lasergraphix 800 printer. It was first used to print letter-quality text. Since it was the only plotter connected to our local computers, we soon used it for plots.

All went well at first. We got crisp, clean 300 dot per inch cut-sheet plots, just down the hall. Plotting took a few minutes, but was faster than before. The text users were unhappy. Their documents had been ready when they arrived. Now they waited for plots to finish. Large plot jobs took a quarter hour or more. For jobs with the same number of print file bytes, plots were much slower than text. Therefore, the bottleneck was in the printer's ability to process plot commands, not the communications line speed or printer mechanism.

Many responses were possible. We could have ignored the issue. We could have reprogrammed the print queue to print text as first priority, interrupting plot jobs between pages, if necessary. We could have restricted use of the laser printer as a plotter (other departments did). We could have bought another laser printer, which would have been expensive and slow to arrive. Some advantages of our solution were that it was fast, cheap, compatible with alternative solutions and reduced waiting for all types of users. Additionally, it increased computer and communications line throughput as well as printer throughput.

Why Throw out Perfectly Good Data?

We don't, actually. The thinning algorithm is invoked on a plot by plot basis, after the scaling of the data to the physical plot units has been determined. The algorithm does not alter the data points, but simply scans them and returns the indices of the points to be plotted for that particular plot.

The deviations in the resulting curve from the full curve are negligible compared to the errors introduced by the plot device quantization. One of the algorithm's inputs is an allowable approximation tolerance. Every point on the polyline defined by the vertices returned by the algorithm lies within the specified tolerance of the polyline whose vertices are the full set of data, and vice versa. Due to the pixel quantization of the plot device, the plotted vertices will be rounded by up to half a pixel in any case. If the tolerance is set at a quarter pixel, the tolerance will be dominated by the pixel quantization noise.

I can't believe my plots have lots of redundant points!

Perhaps they don't - but they probably should. Usually, plot data is sampled at a uniform rate for each variable and at the same rate for all variables. This requires that the sampling be set at a fine enough rate to capture the most active portions of the data, which is wasteful for the rest. Even for a simple sine wave, the sampling rate required near the peaks is unnecessary near the axis crossings. And sampling at the rate required for the most active variable will also be wasteful for the others.

The sampling strategy is usually set before the run. If it does not take enough data, important phenomena will not be captured. This may require that a lengthy and expensive run be repeated, or, worse yet, mislead the simulator. Since these potential consequences of undersampling are usually far worse than those for oversampling, erring on the side of oversampling is to be preferred.

Our experience is that our average uniformly-sampled simulation plot data can be compressed by 3:1 using a quarter-pixel tolerance. To evaluate our algorithm, we used a suite of 29 plots from Loren Slafer, one of our heaviest plot users. Each curve had 3929 points from a real-time satellite simulation. The curves included both quantized and non-quantized data, and most were quite irregular.

Figure 1 shows the effect of the tolerance on the resulting compression. Three points are worthy of note. First, the mean number of points required is inversely proportional to the square of the tolerance. This is largely due to the fact that the plot is a piecewise linear approximation to the curve, so the error (tolerance) is second-order. This also implies that even with very tight tolerances, considerable compression is shown. Second, although the curves were of widely different character, considerable compression was achieved for 90% of them, even for tight tolerances. Third, 3929 points was not excessive. These were time history plots rendered for a laser printer with a time axis of 9 inches at 300 dpi, for a total of 2700 pixels. A solid black plot would have required even more samples (5400).

Figure 2 is another presentation of the results. The five curves that showed the least compression were plots of noisy signals - "hash" that looked as though it would require the full 5400 point capacity of the plot to render. The missing 29th bar is the curve with the most compression - a ramp that was compressed to two points. The next three short bars correspond to curves that were pulse trains or sawtooth signals.

The Algorithm

There are many published algorithms for representing a curve by a piece-wise linear curve with a relatively small number of points. The basic trade-off is between processing speed and amount of compression. We evaluated many such algorithms and variants before choosing our fan algorithm. Eight methods were coded and run on our 29 plot benchmark suite on a VAX 11/780. Table 1 compares our fan algorithm with the fastest (strip) and most effective (minimax) alternatives considered. The reference discusses some of the alternatives in more detail, and references a number of survey papers covering such algorithms.

Table 2 shows the effect of thinning on plot and CPU time. Note that total CPU time was reduced by thinning - the time used to thin the data is far outweighed by the reduction in CPU time caused by fewer points being formatted for the plotter. The reduction in plot time is less than the reduction in points (17% as many points take 33% as long to plot). This is because the time for plot gridding, axes, annotation and paper handling is not reduced by data thinning,

Conceptually, the fan algorithm proceeds as follows. Every point on the approximate curve is to lie within a specified distance, ε, of the original curve, and vice versa. Label the original points from first to last, 1 to n. Save point 1. Find the largest index, m, such that all points less than m lie within ε of the line segment from 1 to m. Delete points 2 to m-1, and save point m. If m is less than n, relabel the points starting with the mth point, and repeat; else, exit. The new curve is constructed by connecting the saved points. As described, the number of comparisons is proportional to n^2 (abbreviated as "$O(n^2)$").

The fan algorithm obtains O(n) performance by calculating a region from points 1 through i, such that point i can be deleted if point i+1 lies in this "feasible" region. The region is a truncated cone, or fan, hence the name. Point i is kept or rejected based on this test, then

Figure 1: Fraction of Points Retained vs. Allowable Error in Pixels

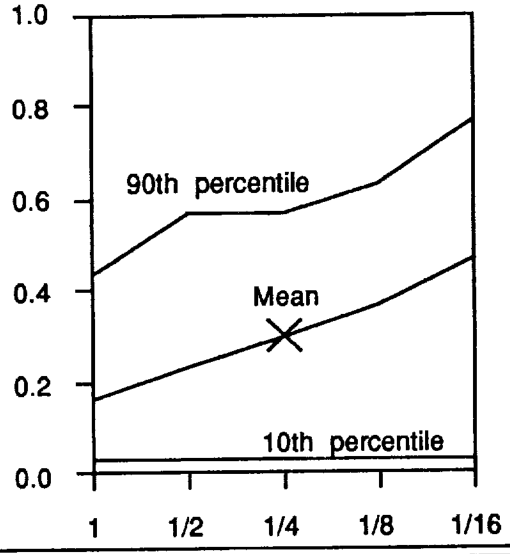


Figure 2: Fraction of Points Retained (1 Pixel Tolerance)

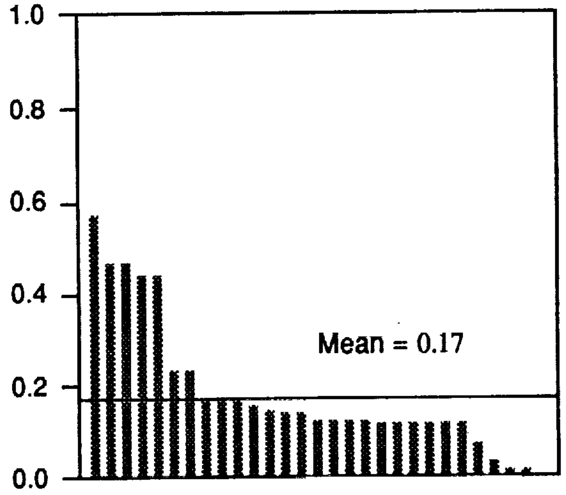


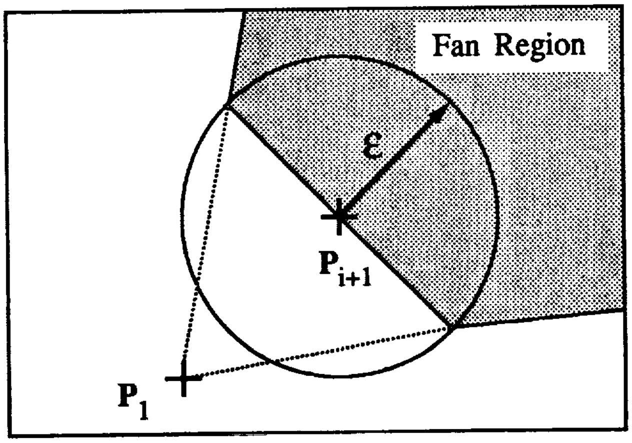
Table 1: Comparison of Methods

Method	Speed (Theory)	Speed on VAX 11/780 (Points/sec)	Effectiveness (Compression)	Complexity (Lines of FORTRAN)
Strip	O(n)	14,000	0.50	40
Fan	O(n)	10,000	0.30	55
Minimax	O(n•log n)	< 5,000 (est.)	0.25 (est.)	>150 (est.)

Table 2: Average Times (thinned and unthinned) on 29 Plots (ε = 1 pixel)

	Fan CPU time (sec/plot)	Total CPU time (sec/plot)	Plot time (sec/plot)	
Unthinned	N/A	10.5	155	(On Vax 11/780 with Apple LaserWriter)
Thinned	0.26	6.2	50	

Figure 3: Fan Region Relative to Tolerance Circle



the feasible region is updated by taking the intersection between the fan determined by points 1 and i+1 (Figure 3) with the previous fan.

The reference devotes three pages to presenting the algorithm, with equations, narrative, three figures and a pseudocode listing, so the interested reader is referred there for a detailed explanation. For Fortran and C source, send the author a SASE.

The algorithm has been validated by peer review, program proof and extensive testing. The algorithm itself is essentially a proof that the omitted points can safely be omitted. The automated test suite includes test curve generators and an automated checking routine which compares every point of the input curve against the thinned curve to make sure the tolerance was not exceeded. Six test curves comprising thousands of points are used which included such boundary cases as empty and one-point curves, vertical lines, repeated points and direction reversals. The algorithm and its validation suite have been run on our Suns, VAXs and IBM 3090.

Conclusions

We are quite enthusiastic about this approach to faster plotting. It began as a practical solution to a problem with plot speed. It improved performance with existing hardware, on several computer systems and plot devices. If plot time is a concern, and the time required to plot the curves is significant, we highly recommend this method. Several commercial simulation vendors have shown interest in this algorithm, and more are being contacted.

Acknowledgments

Richard Straka proposed data thinning to speed up plots, Loren Slafer provided the benchmark plots, and Dave McNeil ran the benchmarks and coded the C version.

Reference

R.A. Fowell and D.D. McNeil, "Faster Plots by Fan Data-Compression," IEEE Computer Graphics and Applications, V9 N2, March 1989, pp. 58-66.

MULTIBODY DYNAMICS: Modeling Component Flexibility with Fixed, Free, Loaded, Constraint, and Residual Modes

John T. Spanos† and Walter S. Tsuha‡
Jet Propulsion Laboratory, California Institute of Technology
Pasadena, California

The assumed-modes method in multibody dynamics allows the elastic deformation of each component in the system to be approximated by a sum of products of spatial and temporal functions commonly known as modes and modal coordinates respectively. This paper focuses on the choice of component modes used to model articulating and non-articulating flexible multibody systems. Attention is directed toward three classical Component Mode Synthesis (CMS) methods whereby component normal modes are generated by treating the component interface (I/F) as either fixed, free, or loaded with mass and stiffness contributions from the remaining components. The fixed and free I/F normal modes are augmented by static shape functions termed "constraint" and "residual" modes respectively. In this paper a mode selection procedure is outlined whereby component modes are selected from the Craig-Bampton (fixed I/F plus constraint), MacNeal-Rubin (free I/F plus residual), or Benfield-Hruda (loaded I/F) mode sets in accordance with a modal ordering scheme derived from balanced realization theory. The success of the approach is judged by comparing the actuator-to-sensor frequency response of the reduced order system with that of the full order system over the frequency range of interest. A finite element model of the Galileo spacecraft serves as an example in demonstrating the effectiveness of the proposed mode selection method.

INTRODUCTION

The general class of dynamical systems known as flexible multibody systems are assemblages of rigid and elastic bodies including spacecraft, robotic manipulators, and industrial machinery. The equations describing the motion of such systems are so complex that, in most situations, information from them can only be obtained via simulation. In 1987, the state-of-the-art in flexible multibody simulation was reviewed and assessed at a workshop hosted by NASA's Jet Propulsion Laboratory.[1] A number of open issues were raised including the issue of modeling component flexibility.

Most of the current simulation algorithms[2–6] addressing flexible multibody systems employ a formulation based on the classical assumed-modes method.[7] The method is summarized in Figure 1. For each component in the multibody chain, a moving coordinate frame $\{\underline{b}_1, \underline{b}_2, \underline{b}_3\}$ is introduced with respect to which the elastic deformation $\underline{u}$ is measured. Consequently, the overall motion of the component is described in part by the "large" motion of the frame $\{\underline{b}_1, \underline{b}_2, \underline{b}_3\}$ and in part by the "small" elastic deformation $\underline{u}$. The underlying assumption of the method is that the deformation $\underline{u}$ can be expanded in a finite sum of products of spatial and temporal functions. The spatial functions are often referred to as mode-shapes or simply modes while the corresponding temporal functions are termed generalized or modal coordinates. Accepting

† Member of Technical Staff, Guidance & Control Section

‡ Member of Technical Staff, Applied Technologies Section

that the deformation can be expanded in this form, one is confronted with the problem of having to select the modes such that the effects of flexibility are properly captured. In engineering practice, the modes are selected from a set of component eigenfunctions which are computed by commercial finite element codes (i.e., NASTRAN) after free or fixed interface* conditions are imposed. Once the modes are selected, they are entered into the multibody simulation program which assembles the system equations of motion and proceeds with their numerical integration.

Clearly, the two most important aspects of the mode selection problem is model accuracy and model order. Ideally, one would like to have a highly accurate system model of very low order. The problem is that these goals are generally at a conflict with each other. Qualitatively, the larger the number of modes used to describe the flexibility of each component, the more accurate the simulation results are expected to be. However, as the number of modes per component increases so does the time required to perform the simulation. Consequently, one is confronted with the problem of having to select a minimal set of modes for each component while maintaining acceptable accuracy in the simulation results. Therefore, the challenge is to find that set of *component* modes which makes the solution of the *system* equations to converge the fastest.

In order to improve convergence, an augmented fixed interface (I/F) mode set was first proposed in the 1960's by the pioneering work of Hurty[8] in connection with the now well known Component Mode Synthesis (CMS) method.** In the aerospace community, this mode set has long been known as the "Craig-Bampton" mode set (in attribute to the refinement of Hurty's work made by Craig and Bampton[9]) and will be referred to as such in this paper. The Craig-Bampton mode set is generated by augmenting the low frequency subset of fixed I/F normal modes with a set of static shape functions termed "constraint" modes. Hurty's work opened up a new area of research in structural dynamics as a number of new CMS methods appeared in the literature since.[10–18] In particular, two new mode sets proposed in the early 1970's were shown to have excellent convergence properties in the sense of CMS. First, the MacNeal-Rubin mode set, attributed to the works of MacNeal[10] and Rubin,[11] is formed by augmenting the low frequency subset of free I/F normal modes with a set of shape functions termed "residual" modes. Second, the Benfield-Hruda mode set proposed by Benfield and Hruda[12] consists entirely of normal modes referred to as "loaded" I/F. In this case, the component is loaded at its interface with mass and stiffness contributions from the remaining components and the loaded I/F normal modes are obtained from the solution of the "loaded" eigenvalue problem. Employing fixed, free, and loaded I/F modes respectively, the Craig-Bampton, MacNeal-Rubin, and Benfield-Hruda methods have been used extensively in connection with CMS-related component model reduction problems.

The problem of reducing the order of a mechanical system by reducing the order of its components is shared by both the structural dynamicist confronted with eigenvalue problems of thousands of degrees-of-freedom (dof) and the multibody dynamicist faced with days or weeks of nonlinear computer simulations for articulating systems of much lower order. This was recognized by a number of researchers in articulated multibody dynamics who transferred the CMS approaches to component model reduction into the large-motion multibody arena.[19–22] Sunada and Dubowsky[19,20] used the Craig-Bampton method to reduce computation time associated with the simulation of flexible linkages and robotic manipulators. Similarly, Yoo and Haug[21,22] adopted the Craig-Chang[15,16] version of the MacNeal-Rubin approach in their treatment of articulated flexible structures. Other researchers addressing component mode selection in multibody dynamics include Singh et al.[5] who along with Macala[23] advocate the use of augmented-body modes, a special case of mass-loaded modes in the Benfield-Hruda method. Other relevant studies include the residual mass concept of Bamford,[24] the modal identities of Hughes,[25,26] and the parallel work of Hablani.[27]

However, a disadvantage of the CMS methods is that they do not directly consider the control system

* The collection of all points where a component attaches to other components is referred to as "interface" or simply "I/F".

** To provide some background, CMS is a Rayleigh-Ritz based approximation method born out of need to analyze linear structural dynamics problems of unusually high order. The large order structure is broken down into a number of components or substructures and a Ritz transformation is employed in reducing the order of each substructure. Subsequent coupling of the reduced order substructures results in a low order system model amenable to linear analysis.

or the location of actuators and sensors when reducing component order. More specifically, a large number of low frequency appendage modes, characteristic of complex spacecraft components, do not contribute to control-structure interaction and consequently these should be discarded as they unnecessarily complicate the multibody simulation model. In view of the control elements, how does one then identify and truncate the non-participating component modes such that the system dynamics remain intact? With the exception of two recent papers,[28,29] this question has received little attention in the multibody literature. Eke and Man[28] proposed a system based modal selection technique where the significant system modes are first identified via a suitable method, then projected down to the components, and finally orthogonalized with respect to the component mass and stiffness matrices. Skelton[29] advocates Component Cost Analysis (CCA) to component mode selection. It should be noted that, in the case of articulating structures, both of these approaches are sensitive to inter-component articulation since mode selection is done after the multibody system equations have been linearized about a particular equilibrium configuration.

Outside multibody dynamics, order reduction of linear system models has been a topic of research by the controls community. Here, the primary motivation behind model reduction is the design of low order controllers which are in turn based on low order models of the system under control. In 1980, a new model reduction approach was introduced by Moore[30] known as "balanced" model reduction. The approach takes into account the system inputs and outputs and suggests that yet another set of modes (i.e., balanced modes) be used in coordinate truncation. Moore employs a coordinate transformation to bring the system into the balanced form whereby the reachability and observability gramians are equal and diagonal.[30] In the balanced form, the coordinates corresponding to small elements on the diagonal of the gramians are candidates for truncation since they can be interpreted as least controllable from the actuators and least observable from the sensors. Application of balancing to structural systems showed that, as damping approaches zero asymptotically, truncation of balanced modes is equivalent to truncation of normal modes.[31–33] This special result is used in the component mode selection method proposed in this paper.

In this paper a two-stage component model reduction methodology is proposed complementing CMS with balancing. First, CMS mode sets are generated and used to reduce the order of each component in the Rayleigh-Ritz sense. The methods of Craig-Bampton, MacNeal-Rubin, and Benfield-Hruda provide alternate Ritz transformations for component model reduction. After the reduced component models are brought to diagonal form, a second reduction is performed via balancing. In particular, Gregory's[32] modal ranking criterion derived for lightly damped structures with sufficiently separated modal frequencies is used to identify and further truncate "insignificant" modes from each component. In this stage, the component interface locations are treated as additional inputs and outputs of interest. The component model is thus reduced as a separate entity without having to assemble the system model.

The paper is organized as follows. First, the three component mode sets of Craig-Bampton, MacNeal-Rubin, and Benfield-Hruda are briefly described. Then, the component Ritz reduction and diagonalization procedure are presented. Next, the balanced reduction procedure is discussed in the context of component mode selection. Finally, the effectiveness of the proposed end-to-end model reduction methodology is demonstrated with an example of a complex spacecraft.

COMPONENT MODE SETS

Consider a structural system consisting of several interconnected elastic components. Each component (see Fig. 2) can be described by a second order matrix differential equation of the form

$$M_{nn}\ddot{x}_n + K_{nn}x_n = f_n \tag{1}$$

where x_n, f_n denote the $n \times 1$ displacement and force vectors respectively and M_{nn}, K_{nn} represent the $n \times n$ mass and stiffness matrices respectively. This n-dof component model is typically obtained from a commercial finite element program such as NASTRAN.

Before proceeding with the description of the mode sets, the reader should be clear on the special notation used in this section. That is, vectors and matrices carry single and double subscripts indicating their respective dimension. The only non-subscripted vectors and matrices are those whose elements are all zeroes.

Craig-Bampton Mode Set[8,9]

The component finite element model of Eq(1) can be partitioned as follows

$$\begin{bmatrix} M_{ii} & M_{ij} \\ M_{ji} & M_{jj} \end{bmatrix} \begin{bmatrix} \ddot{x}_i \\ \ddot{x}_j \end{bmatrix} + \begin{bmatrix} K_{ii} & K_{ij} \\ K_{ji} & K_{jj} \end{bmatrix} \begin{bmatrix} x_i \\ x_j \end{bmatrix} = \begin{bmatrix} f_i \\ 0 \end{bmatrix} \tag{2}$$

where x_i and x_j represent the interface and interior coordinates respectively (Fig. 2). Note that in writing Eq(2) it is assumed that no forces act on the interior coordinates. However, if forces due to actuators and disturbances act on some interior coordinates it is recommended that these coordinates be removed from the j-partition and placed in the i-partition of x_n.

The first k *fixed I/F* normal modes Φ_{jk} and modal frequencies Ω_{kk} are obtained from the solution of the eigenvalue problem

$$-M_{jj}\Phi_{jk}\Omega^2_{kk} + K_{jj}\Phi_{jk} = 0\,; \quad k < j \tag{3}$$

A *constraint* mode is defined as the static deformation shape that results by imposing unit displacement on one coordinate of the i-set while holding the remaining coordinates in the i-set fixed.[8,9] From the definition, the constraint mode set satisfies the matrix equation

$$\begin{bmatrix} K_{ii} & K_{ij} \\ K_{ji} & K_{jj} \end{bmatrix} \begin{bmatrix} I_{ii} \\ \Psi_{ji} \end{bmatrix} = \begin{bmatrix} F_{ii} \\ 0 \end{bmatrix} \tag{4}$$

where I_{ii} is the identity matrix and the columns of F_{ii} represent the forces required to deform the component into the shape of the constraint modes. In the special case of a statically determinate i-set, the constraint modes yield the component rigid body modes and F_{ii} vanishes. It can be shown that the space spanned by the rigid body modes is a subspace within the space spanned by the constraint modes. The matrix Ψ_{ji} is obtained from the bottom partition of Eq(4)

$$\Psi_{ji} = -K^{-1}_{jj}K_{ji} \tag{5}$$

The Craig-Bampton mode set can now be formed by augmenting the constraint modes with the truncated set of fixed I/F normal modes as follows

$$\begin{bmatrix} x_i \\ x_j \end{bmatrix} = \begin{bmatrix} I_{ii} & 0 \\ \Psi_{ji} & \Phi_{jk} \end{bmatrix} \begin{bmatrix} x_i \\ \eta_k \end{bmatrix} \tag{6}$$

It should be noted that the constraint modes are orthogonal to the fixed I/F normal modes with respect to the component stiffness matrix. Finally, Eq(6) can be written in a more compact form as

$$x_n = \Phi^{CB}_{nm}\,\eta_m \tag{7}$$

where $m = i + k$ represents the total number of modes in the set.

MacNeal-Rubin Mode Set[10,11]

The first k *free I/F* normal modes Φ_{nk} and modal frequencies Ω_{kk} are obtained from the solution of the eigenvalue problem

$$-M_{nn}\Phi_{nk}\Omega^2_{kk} + K_{nn}\Phi_{nk} = 0\,; \quad k < n \tag{8}$$

Furthermore, Φ_{nk} can be scaled so that it satisfies the mass orthonormality relation

$$\Phi^T_{nk} M_{nn} \Phi_{nk} = I_{kk} \tag{9}$$

where I_{kk} is the identity matrix. The free I/F normal mode set can be partitioned into rigid and elastic subsets as follows

$$\Phi_{nk} = [\,\Phi_{nr} \quad \Phi_{ne}\,] \ ; \qquad \Omega_{kk} = \begin{bmatrix} 0 & 0 \\ 0 & \Omega_{ee} \end{bmatrix} \tag{10}$$

where $k = r + e$. Now, the component finite element model of Eq(1) can be partitioned as

$$\begin{bmatrix} M_{ii} & M_{il} & M_{ir} \\ M_{li} & M_{ll} & M_{lr} \\ M_{ri} & M_{rl} & M_{rr} \end{bmatrix} \begin{bmatrix} \ddot{x}_i \\ \ddot{x}_l \\ \ddot{x}_r \end{bmatrix} + \begin{bmatrix} K_{ii} & K_{il} & K_{ir} \\ K_{li} & K_{ll} & K_{lr} \\ K_{ri} & K_{rl} & K_{rr} \end{bmatrix} \begin{bmatrix} x_i \\ x_l \\ x_r \end{bmatrix} = \begin{bmatrix} f_i \\ 0 \\ 0 \end{bmatrix} \tag{11}$$

where, as in the Craig-Bampton method, x_i represents the interface coordinates and $x_j = [x_l^T \; x_r^T]^T$ represents the interior coordinates (Fig. 2). Here again it is assumed that no forces act on the interior coordinates. If forces are applied to some interior coordinates, then these coordinates should be removed from the j-set and placed into the i-set. Furthermore, the r-partition of the interior coordinates can be any statically determinate set such that if the component is restrained at x_r, rigid body motion is prevented.

The *residual* modes Ψ_{ni} are linear combinations of the $n - k$ truncated free I/F normal modes. These are obtained from the refined procedure of Craig and Chang[15,16]

$$\Psi_{ni} = [P^T_{nn} G_{nn} P_{nn} - \Phi_{ne} \Omega^{-2}_{ee} \Phi^T_{ne}] F_{ni} \tag{12}$$

where

$$G_{nn} = \begin{bmatrix} \begin{bmatrix} K_{ii} & K_{il} \\ K_{li} & K_{ll} \end{bmatrix}^{-1} & \begin{matrix} 0 \\ 0 \end{matrix} \\ \begin{matrix} 0 & \quad 0 \end{matrix} & 0 \end{bmatrix} \tag{13}$$

$$P_{nn} = I_{nn} - M_{nn} \Phi_{nr} \Phi^T_{nr} \tag{14}$$

$$F_{ni} = \begin{bmatrix} I_{ii} \\ 0 \\ 0 \end{bmatrix} \tag{15}$$

The matrix G_{nn} in Eq(13) is a pseudo-flexibility matrix corresponding to the singular stiffness matrix K_{nn}. The matrix P_{nn} plays the role of a projection matrix such that the columns of $P^T_{nn} G_{nn} P_{nn}$ span the same space as the totality of $n - r$ elastic modes of the component. By subtracting the contribution of the retained normal modes from the elastic flexibility matrix $P^T_{nn} G_{nn} P_{nn}$, one obtains the residual flexibility matrix whose columns are the residual modes. This is a clever way of capturing the contribution of the truncated normal modes without having to compute them in Eq(8).[11,16] Clearly, only the residual modes associated with force-carrying coordinates are of interest. These are stripped from the residual flexibility matrix by post-multiplication with F_{ni}.

The MacNeal-Rubin mode set can now be formed by augmenting the truncated set of free I/F normal modes with the residual modes as follows

$$x_n = [\,\Phi_{nk} \quad \Psi_{ni}\,] \begin{bmatrix} \eta_k \\ \eta_i \end{bmatrix} \tag{16}$$

It should be noted that the residual modes are orthogonal to the free I/F normal modes with respect to both the mass and stiffness matrix of the component. In addition, the MacNeal-Rubin mode set is said to be *statically complete*[17] with respect to all forces in the i-set. That is, the deformation of the component due to static loads acting on the i-set can be written as a linear combination of the modes in the MacNeal-Rubin mode set. Finally, Eq(16) can be written in a more compact form as

$$x_n = \Phi^{MR}_{nm} \, \eta_m \tag{17}$$

where $m = i + k$ represents the total number of modes in the set.

Benfield-Hruda Mode Set[12]

In order to best describe this mode set, consider a multibody system consisting of only two elastic components. These will be referred to as components A and B and subsequent notation will be superscripted accordingly. For simplicity of notation, both components are further assumed to have the same dimension n.

The first m *loaded I/F* normal modes Φ^A_{nm} and modal frequencies Ω^A_{mm} of component A are obtained from the solution to the eigenvalue problem

$$-\mathcal{M}^A_{nn}\Phi^A_{nm}\Omega^{A^2}_{mm} + \mathcal{K}^A_{nn}\Phi^A_{nm} = 0 \; ; \quad m < n \tag{18}$$

The matrices $\mathcal{M}^A_{nn}$ and $\mathcal{K}^A_{nn}$ are given by

$$\mathcal{M}^A_{nn} = \begin{bmatrix} M^A_{ii} & M^A_{ij} \\ M^A_{ji} & M^A_{jj} \end{bmatrix} + \begin{bmatrix} \Psi^{B^T}_{ni} M^B_{nn} \Psi^B_{ni} & 0 \\ 0 & 0 \end{bmatrix} \tag{19}$$

$$\mathcal{K}^A_{nn} = \begin{bmatrix} K^A_{ii} & K^A_{ij} \\ K^A_{ji} & K^A_{jj} \end{bmatrix} + \begin{bmatrix} \Psi^{B^T}_{ni} K^B_{nn} \Psi^B_{ni} & 0 \\ 0 & 0 \end{bmatrix} \tag{20}$$

where, as previously, the i and j partitions of x_n correspond to interface and interior coordinates respectively. Clearly, the first terms on the right side of Eqs(19,20) are the mass and stiffness matrices of component A. The non-zero partitions of the second terms, $\Psi^{B^T}_{ni} M^B_{nn} \Psi^B_{ni}$ and $\Psi^{B^T}_{ni} K^B_{nn} \Psi^B_{ni}$, are referred to as the interface "loading" matrices and represent the mass and stiffness contributions of component B. The matrix Ψ^B_{ni} is formed from the stiffness partitions of component B

$$\Psi^B_{ni} = \begin{bmatrix} I_{ii} \\ -K^{B^{-1}}_{jj} K^B_{ji} \end{bmatrix} \tag{21}$$

in the same way that the constraint modes in the Craig-Bampton mode set were defined. For a statically determinate i-set, the stiffness loading vanishes since the columns of Ψ^B_{ni} span the null space of K^B_{nn}.

The Benfield-Hruda mode set of component A is formed entirely from the truncated set of loaded I/F normal modes

$$x_n = \Phi^{BH}_{nm}\ \eta_m \tag{22}$$

where $\Phi^{BH}_{nm} = \Phi^A_{nm}$ as computed from Eq(18). The corresponding mode set of component B can be formed in similar fashion. The generalization of the approach to more than two components is straightforward.

Before proceeding, a few comments are in order. Loading a component with mass and stiffness contributions from the remaining components is an attempt at capturing the modes of the system that this component is a part of. Such feature yields a much improved system model.[12] However, unlike the Craig-Bampton and MacNeal-Rubin mode sets, information from the remaining components is necessary in forming the Benfield-Hruda mode set. As a consequence, the task of generating the loaded I/F modes can be much more computationally intensive, especially in the case of multibody systems consisting of several components.

RAYLEIGH-RITZ REDUCTION

Having discussed each of the three mode sets, the special notation of the last section is now abandoned. Subscripts indicating vector or matrix dimension will be dropped for convenience of notation. To this effect, the component model of Eq(1) can be written as

$$M\ddot{x} + Kx = Pu \tag{23}$$

where the vector u represents I/F forces due to the attaching components as well as forces due to actuators and disturbances acting on the component. The matrix P represents the spatial distribution of all applied forces. Eq(23) describes the dynamics of the component under the assumptions of small structural deformations and small overall motion. The corresponding output equation can be written in terms of the displacement coordinates and rates as

$$y = H_1 x + H_2 \dot{x} \tag{24}$$

where H_1, H_2 represent the displacement and rate output distribution matrices respectively. These may include sensor outputs as well as other outputs of interest such as component interface displacement and rate.

The component model can now be reduced by letting

$$x = \Phi \eta \tag{25}$$

where the dimension of the modal vector η is much smaller than the dimension of the displacement vector x and the columns of Φ play the role of component Ritz vectors in the classical Rayleigh-Ritz approximation method.[7] Any one of the three truncated mode sets given by Eq(7), Eq(17), and Eq(22) can serve as the Ritz transformation matrix Φ. Furthermore, different components of a multibody system need not be reduced with the same type of mode set. For example, in a system of three components, the first can be reduced using MacNeal-Rubin, the second via Benfield-Hruda, and the third via Craig-Bampton. Alternatively, all three could be reduced via Craig-Bampton. In general, this choice is system dependent.

However, there still exists the question of how many normal modes one should include in the Craig-Bampton, MacNeal-Rubin, and Benfield-Hruda mode sets. Clearly, the answer will most likely depend on many factors inherent to the multibody system in question. As a rule of thumb it is suggested that normal modes with frequencies above two times the system frequency of interest be truncated from any of the three mode sets chosen to represent component flexibility. This claim is shown to be adequate in the example problem of this paper and has proven adequate in numerous other practical problems the authors have studied.

Substituting Eq(25) into Eqs(23,24) and premultiplying Eq(23) by Φ^T yields

$$\Phi^T M \Phi \ddot{\eta} + \Phi^T K \Phi \eta = \Phi^T P u \tag{26}$$

$$y = H_1 \Phi \eta + H_2 \Phi \dot{\eta} \tag{27}$$

These equations represent the reduced order component model. Thus, $(n - m)$ degrees of freedom have been eliminated in going from the n-size model of Eqs(23,24) to the m-size model of Eqs(26,27).

DIAGONALIZATION

Eq(26) will now be brought to diagonal form for reasons that will become clear in the next section. Let

$$\eta = \Psi \xi \tag{28}$$

where the square matrix Ψ satisfies the mass and stiffness orthogonality relations

$$[\Phi\Psi]^T \, M \, [\Phi\Psi] = I \, ; \qquad [\Phi\Psi]^T \, K \, [\Phi\Psi] = \Omega^2 \tag{29}$$

and Ω is the diagonal matrix of frequencies corresponding to the orthogonalized modes. The matrix I is the identity matrix. Substituting Eq(28) into Eqs(26,27), premultiplying Eq(26) by Ψ^T, and adding modal damping one obtains

$$\ddot{\xi} + 2\varsigma\Omega\dot{\xi} + \Omega^2 \xi = [\Phi\Psi]^T P u \tag{30}$$

$$y = H_1 [\Phi\Psi] \xi + H_2 [\Phi\Psi] \dot{\xi} \tag{31}$$

where ς is the diagonal damping matrix. Eq(30) describes the dynamics of the component in diagonal form. Finally, Eqs(30,31) can be written in the more compact form

$$\ddot{\xi} + 2\varsigma\Omega\dot{\xi} + \Omega^2\xi = Bu \tag{32}$$

$$y = C_1\xi + C_2\dot{\xi} \tag{33}$$

where

$$B = [\Phi\Psi]^T P\,; \qquad C_1 = H_1[\Phi\Psi]\,; \qquad C_2 = H_2[\Phi\Psi] \tag{34}$$

Next, the component model of Eqs(32,33) will be reduced further by truncating modes from the orthogonalized set $[\Phi\Psi]$.

BALANCED REDUCTION

The component model of Eqs(32,33) can now be written in first order or state form by letting $\chi = [\xi^T \ \dot{\xi}^T]^T$

$$\dot{\chi} = \mathcal{A}\chi + \mathcal{B}u \tag{35}$$

$$y = \mathcal{C}\chi \tag{36}$$

where

$$\mathcal{A} = \begin{bmatrix} 0 & I \\ -\Omega^2 & -2\varsigma\Omega \end{bmatrix}\,; \qquad \mathcal{B} = \begin{bmatrix} 0 \\ B \end{bmatrix}\,; \qquad \mathcal{C} = [\,C_1 \quad C_2\,] \tag{37}$$

At this point it will be assumed that the states corresponding to component rigid body modes have been partitioned out of Eqs(35,36) such that all eigenvalues of matrix $\mathcal{A}$ have strictly negative real parts. Thus, matrix $\mathcal{A}$ has dimension $2p$ where $p = m - r$ and r represents the number of rigid body modes. Matrices $\mathcal{B}$ and $\mathcal{C}$ are of appropriate dimension.

The reachability and observability gramians of the model are defined in terms of the matrix integrals[34]

$$W = \int_0^\infty e^{\mathcal{A}t}\mathcal{B}\mathcal{B}^T e^{\mathcal{A}^T t}dt\,; \qquad V = \int_0^\infty e^{\mathcal{A}^T t}\mathcal{C}^T\mathcal{C}e^{\mathcal{A}t}dt \tag{38}$$

and are computed from the linear matrix equations

$$\mathcal{A}W + W\mathcal{A}^T + \mathcal{B}\mathcal{B}^T = 0\,; \qquad V\mathcal{A} + \mathcal{A}^T V + \mathcal{C}^T\mathcal{C} = 0 \tag{39}$$

The model is said to be *balanced* if

$$W = V = \Sigma = diag\{\ \sigma_i, \quad i = 1, 2, \ \ldots, \ 2p\} \tag{40}$$

and $\sigma_1 \geq \sigma_2 \geq \sigma_3 \ \ldots \ \geq \sigma_{2p} \geq 0$. Moore[30] showed that any linear, time-invariant, asymptotically stable model can be brought to balanced form via a suitable linear transformation of state. The idea behind balanced model reduction is to bring the model into the balanced form and truncate states in that form. The balanced states to be truncated are identified on the basis of the relative magnitudes of the scalars σ_i. Such rationale comes from input-output considerations based on the notions of controllability and observability.[34] Loosely speaking, the balanced states corresponding to small σ_i's are "least controllable" from the inputs u and "least observable" from the outputs y. Consequently, these states are candidates for truncation. The scalars σ_i are invariant under state transformation and equal to the square roots of the eigenvalues of the gramian product (i.e., $\sigma_i = \sqrt{\lambda_i[WV]}$). Therefore, in the context of model reduction, it is not necessary that the model be balanced in the sense of Eq(40) but only that the gramian product is diagonal (i.e., $WV = \Sigma^2$). Furthermore, an important feature of balanced model reduction is that there exists an ∞-norm frequency error bound[35]

$$\|G^{2p}(j\omega) - G^k(j\omega)\|_\infty \leq 2\sum_{i=k+1}^{2p} \sigma_i\,; \qquad k < 2p \tag{42}$$

where $G^{2p}(s) = C[sI - A]^{-1}B$ is the transfer matrix of the full order model and, similarly, $G^k(s)$ is its k^{th}-order counterpart. For the component model parameters of Eq(37), the transfer matrix can be written as a sum of contributions from each elastic mode

$$G^{2p}(s) = \sum_{i=1}^{p} G_i^{2p}(s) \; ; \qquad G_i^{2p}(s) = \frac{(c_{1i} + c_{2i}s)b_i}{s^2 + 2\varsigma_i\omega_i s + \omega_i^2} \tag{43}$$

where

ς_i is the ii element of the diagonal matrix ς
ω_i is the ii element of the diagonal matrix Ω
b_i is the i^{th} *row* of B
c_{1i} is the i^{th} *column* of C_1
c_{2i} is the i^{th} *column* of C_2

Gregory[32] showed that the modal model of a lightly damped structure with well separated frequencies is approximately balanced. In addition, he obtained closed form expressions for the scalars σ_i in terms of the transfer matrix parameters ς_i, ω_i, b_i, c_{1i}, c_{2i} as follows

$$\sigma_i \approx \frac{\sqrt{b_i b_i^T [c_{1i}^T c_{1i} + \omega_i^2 c_{2i}^T c_{2i}]}}{4\varsigma_i\omega_i^2} \; ; \qquad i = 1, \ldots, p \tag{44}$$

and $\sigma_i \approx \sigma_{p+i}$. Following the rationale of balanced model reduction, component modes with small σ_i are least affected by the applied forces u and contribute least to the outputs y. Consequently, these modes can be truncated from the set $[\Phi\Psi]$. The scalars σ_i indicate modal influence and will therefore be referred to as "modal influence coefficients." The quality of the approximation in Eq(44) depends on how well the following criterion on "close-spaceness" of frequencies is satisfied[32]

$$\frac{max(\varsigma_i, \varsigma_j)max(\omega_i, \omega_j)}{|\omega_i - \omega_j|} \ll 1 \; ; \qquad i \neq j \tag{45}$$

Since most space structures exhibit clusters of closely spaced frequencies, Eq(45) may be violated. In such case, one could ignore Eq(45) and proceed with modal truncation as suggested by Eq(44) thereby retaining only modes with large σ_i. Alternatively, modes that violate Eq(45) can be placed into groups and separate analysis be carried out on each group of closely-spaced modes to determine whether additional modes with small modal influence should be retained. In this case, all modes with large σ_i and some modes with small σ_i may be retained. The approximate error bound of Eq(42) can be used as a guide in determining ***how many*** modes to retain.

Finally, an interesting observation can be made with regard to the approximate balancing formula of Eq(44). When the output equation does not include rates (i.e., $H_2 = C_2 = 0$), Eq(44) reduces to

$$\sigma_i \approx \left(\frac{1}{4\varsigma_i}\right) \|G_i^{2p}(0)\|_F \; ; \qquad i = 1, \ldots, p \tag{46}$$

Furthermore, from Eq(43), the transfer matrix evaluated at zero frequency yields

$$G^{2p}(0) = \sum_{i=1}^{p} G_i^{2p}(0) = \sum_{i=1}^{p} \frac{c_{1i}b_i}{\omega_i^2} = C_1\Omega^{-2}B = H_1[\Phi\Psi]\Omega^{-2}[\Phi\Psi]^T P \tag{47}$$

where one will recognize that the matrix $[\Phi\Psi]\Omega^{-2}[\Phi\Psi]^T$ is the elastic flexibility matrix of the Ritz-reduced component. Eq(46) indicates that the balancing scalar σ_i is proportional to the Frobenious norm of the contribution of mode i to the elastic flexibility matrix. In other words, the balanced modal truncation criterion signifies the modes which participate most in the static response of the component.

EXAMPLE

The proposed two-stage component model reduction methodology is illustrated in Figure 3 and will now be demonstrated with a high order finite element model of the Galileo dual-spin spacecraft. Figure 4(a) shows the three-component topology of the spacecraft. Two of the components are assumed flexible while the third is idealized as rigid. The 243-dof flexible Rotor and the 6-dof rigid Platform are attached to the 57-dof flexible Stator by hinge joints such that the three components articulate relative to each other. The NASTRAN model shown in Figure 4(b) was originally of much larger dimension but was reduced to the aforementioned size via the Rayleigh-Ritz method using a set of appropriately chosen constraint modes as the Ritz transformation.

Two motor actuators located at the Rotor-Stator and Stator-Platform interface provide pointing control to the Platform. The controller accepts Platform attitude measurements from a gyro sensor located on the Platform, calculates the motor torques necessary to accomplish the pointing objective, and commands the motors accordingly. The problem set forth was to develop a system model of much lower order to be used for simulation in view of anticipated control-structure interaction while the system is undergoing large overall motions. In particular, it was deemed that the control loop closed around the Rotor-Stator actuator and Platform gyro would be most critical since the flexible Stator is located in between. Figure 4(a) shows the location of the control input and the two sensor outputs of relevance. The main requirement placed on the low order system model was that the actuator-to-sensor frequency response at all "frozen" configurations be faithfully reproduced in the 0-10 Hz range.

The 243-dof model of the Rotor and 57-dof model of the Stator were passed through the model reduction steps outlined in Figure 3. All three mode sets were formed for both flexible components using truncated fixed, free, and loaded interface modes to twice the system frequency of interest or 20 Hz. This resulted in 74 elastic modes representing the Rotor (i.e., elimination of 163 dof) and 16 elastic modes describing the Stator (i.e., elimination of 35 dof). The orthogonalized mode sets are listed in Table 1. Then, a standard component mode synthesis procedure[16] was employed to assemble the Rotor, Stator, and Platform into a system at one particular configuration. Three system models resulted corresponding to the three mode sets and the actuator-to-sensor frequency response was computed for each. The results were superimposed over the "exact" response obtained from the full order model and are illustrated in Figures 5, 6, and 7. Note that all mode sets performed equally well indicating virtually no error in the 0-10 Hz frequency range of interest. The Craig-Bampton mode set was further reduced via balancing. The 19 Rotor modes and 15 Stator modes with largest modal influence coefficients were retained in the reduced order model. These are marked by an asterisk in Table 1. Once more, the system model was assembled and the input-output frequency response was carried out yielding the result of Figure 8. Surprisingly, no error is apparent in the 0-10 Hz frequency range in spite of eliminating 55 additional modes from the Rotor. This indicates the presence of a large number of low frequency component modes occuring below 10 Hz that do not participate in the response. The reduced and full order system model were assembled in different configurations corresponding to different articulation angle settings and similar results were obtained. The analysis was repeated with the MacNeal-Rubin and Benfield-Hruda mode sets and the actuator-to-sensor frequency response results were nearly identical to those obtained with the Craig-Bampton mode set.

Finally, an interesting experiment was conducted. From Table 1, it was noted that the 19 Craig-Bampton Rotor modes retained by the modal balancing formula were not ordered according to frequency. In fact, the last 6 modes in the set of 74 had large modal influence coefficients. If one was to naively select the first 19 modes to represent the flexibility of the Rotor, the system frequency response result of Figure 9 would be obtained. The large error between the reduced and full order models indicates that the low frequency modes are not always the "most important" and demonstrates the need for intelligent component mode selection.

CONCLUDING REMARKS

A component mode selection and reduction method for modeling flexible multibody systems has been presented. The method combines the Component Mode Synthesis (CMS) approaches of Craig-Bampton,[8,9] MacNeal-Rubin,[10,11] and Benfield-Hruda[12] with the Moore-Gregory[30,32] modal balancing method.

The two-stage modal reduction method works directly on the component finite element model (FEM) and does not require assembly or knowledge of the system FEM. In the first stage, Rayleigh-Ritz reduction via CMS mode sets eliminates the high frequency unimportant and unreliable data from the component FEM. In the second stage, modal balancing further eliminates the modes that are least affected by actuators, disturbances, interface forces, and contribute least to motion at sensor and component interface locations. Thus, modal balancing can be viewed as a second Rayleigh-Ritz reduction where the Ritz vectors are appropriately selected component modes. The proposed method is applicable to both articulating and non-articulating systems and was succesfully used in developing a low order model of the three-body articulating Galileo spacecraft. The truncated mode sets of Craig-Bampton, MacNeal-Rubin and Benfield-Hruda performed equally well in capturing the low frequency system dynamics over all articulated configurations.

ACKNOWLEDGEMENT

The research described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration. The authors are indebted to Dr. Ken Smith for his valuable comments and for many of the special NASTRAN software used in the paper. Finally, many thanks are due to Dr. Bob Laskin and Dr. Mike Lou for making this work possible.

REFERENCES

[1] Man, G. and Laskin, R. (Ed.), *Proceedings of the Workshop on Multibody Simulation*, D-5190 (Internal Document), Jet Propulsion Laboratory, Pasadena, CA, April 15, 1988.

[2] Bodley, C. S., Devers, A. D., Park, A. C., and Frisch, H. P., "A Digital Computer Program for the Dynamic Interaction Simulation of Controls and Structure (DISCOS)," Vols. 1 and 2, NASA TP-1219, May 1978.

[3] Frisch, H. P., "A Vector-Dyadic Development of the Equations of Motion for N-Coupled Flexible Bodies and Point Masses," NASA TN D-8047, Aug. 1975.

[4] Singh, R. P., VanderVoort, R. J., and Likins, P. W., "Dynamics of Flexible Bodies in Tree Topology —A Computer-Oriented Approach," AIAA Journal of Guidance, Control, and Dynamics, Vol. 8, No. 5, Sep.-Oct. 1985, pp. 584-590.

[5] Ho, J. Y. L., "Direct Path Method for Flexible Multibody Spacecraft Dynamics," Journal of Spacecraft and Rockets, Vol. 14, Jan.-Feb. 1977, pp 102-110.

[6] Ho, J. Y. L. and Herber, D. R., "Development of Dynamics and Control Simulation of Large Flexible Space Systems," Journal of Guidance, Control, and Dynamics, Vol. 8, May-June 1985, pp. 347-383.

[7] Meirovitch, L., *Analytical Methods in Vibrations*, The Macmillan Co., New York, 1967.

[8] Hurty, W. C., "Dynamic Analysis of Structural Systems Using Component Modes," AIAA Journal, Vol. 3, No. 4, April 1965, pp. 678-685.

[9] Craig, R. R. Jr., and Bampton, M. C. C., "Coupling of Substructures for Dynamic Analysis," AIAA Journal, Vol. 6, No. 7, July 1968, pp. 1313-1319.

[10] MacNeal, R. H., "A Hybrid Method of Component Mode Synthesis," Computers and Structures, Vol. 1, No. 4, Dec. 1971, pp. 581-601.

[11] Rubin, S., "Improved Component-Mode Representation for Structural Dynamic Analysis," AIAA Journal, Vol. 13, No. 8, August 1975, pp. 995-1006.

[12] Benfield, W. A., and Hruda, R. F., "Vibration Analysis of Structures by Component Mode Substitution," AIAA Journal, Vol. 9, No. 7, July 1971, pp. 1255-1261.

[13] Gladwell, G. M. L., "Branch Mode Analysis of Vibrating Systems," Journal of Sound and Vibration, Vol. I, 1964, pp. 41-59.

[14] Hintz, R. M., "Analytical Methods in Component Modal Synthesis," AIAA Journal, Vol. 13, No. 8, August 1975, pp. 1007-1016.

[15] Craig, R. R. Jr., and Chang, C-J., "On the Use of Attachment Modes in Substructure Coupling for Dynamic Analysis," Proceedings AIAA/ASME 18th Structures, Structural Dynamics and Materials

[15] Craig, R. R. Jr., and Chang, C-J., "On the Use of Attachment Modes in Substructure Coupling for Dynamic Analysis," Proceedings AIAA/ASME 18th Structures, Structural Dynamics and Materials Conference, Vol. B, 1977, pp. 89-99.

[16] Craig, R. R. Jr., *Structural Dynamics: An Introduction to Computer Methods*, J. Wiley and Sons, Inc., New York, 1981.

[17] Hintz, R. M., "Analytical Methods in Component Modal Synthesis," AIAA Journal, vol. 13, No. 8, Aug. 1975, pp. 1007-1016.

[18] Craig, R. R., "A Review of Time-Domain and Frequency-Domain Component Mode Synthesis Method," ASCE/ASME Mechanics Conference, June 1985, Vol. 67, pp. 1-30.

[19] Sunada, W., and Dubowsky, S., "The Application of Finite Element Methods to the Dynamic Analysis of Flexible Spatial and Co-Planar Linkage Systems," Journal of Mechanical Design, Vol. 103, July 1981, pp. 643-651.

[20] Sunada, W., and Dubowsky, S., "On the Dynamic Analysis and Behavior of Industrial Robotic Manipulators With Elastic Members," Trans. of ASME, Journal of Mechanisms, Transmissions, and Automation in Design, Vol. 105, March 1983, pp. 42-51.

[21] Yoo, W. S., and Haug, E. J., "Dynamics of Articulated Structures, Part I: Theory," Journal of Structural Mechanics, Vol. 14, No. 1, 1986, pp. 105-126.

[22] Yoo, W. S., and Haug, E. J., "Dynamics of Articulated Structures, Part II: Computer Implementation and Applications," Journal of Structural Mechanics, Vol. 14, No. 2, 1986, pp. 177-189.

[23] Macala, G. A., "A Modal Reduction Method for use with Nonlinear Simulations of Flexible Multibody Spacecraft," AIAA/AAS Astrodynamics Conference, August 20-22, 1984, Seattle, Washington.

[24] Bamford, R. M., Wada, B. K., and Gayman, W. H., "Equivalent Spring-Mass System for Normal Modes," JPL Tech. Memo. 33-380, Jet Propulsion Laboratory, Pasadena, CA, Feb. 1971.

[25] Hughes, P. C., "Modal Identities for Elastic Bodies with Application to Vehicle Dynamics and Control," Transactions of ASME, Journal of Applied Mechanics, Vol. 47, March 1980, pp. 177-184.

[26] Hughes, P. C., "Space Structure Vibration Modes: How Many Exist? Which Ones Are Important?" IEEE Control Systems Magazine, February 1987, pp. 22-28.

[27] Hablani, H. B., "Modal Identities for Multibody Elastic Spacecraft —An Aid to Selecting Modes for Simulation," AIAA 89-0544, 27th Aerospace Sciences Meeting, Reno, Nevada, January 1989.

[28] Eke, F. O. and Man, G. K., "Model Reduction For Flexible Multibody Systems," JPL D-5190, Vol. II (Internal Document), Proceedings of the Workshop on Multibody Simulation, April 15, 1988, pp. 789-808.

[29] Skelton, R. E., Singh, R., and Ramakrishnan, J., "Component Model Reduction by Component Cost Analysis," AIAA 88-4086-CP, Guidance and Control Conference, Minneapolis, Minnesota, 1988.

[30] Moore, B. C., "Principal Component Analysis in Linear Systems: Controllability, Observability, and Model Reduction," IEEE Transactions on Automatic Control, Vol. AC-26, Feb. 1981, pp. 17-32.

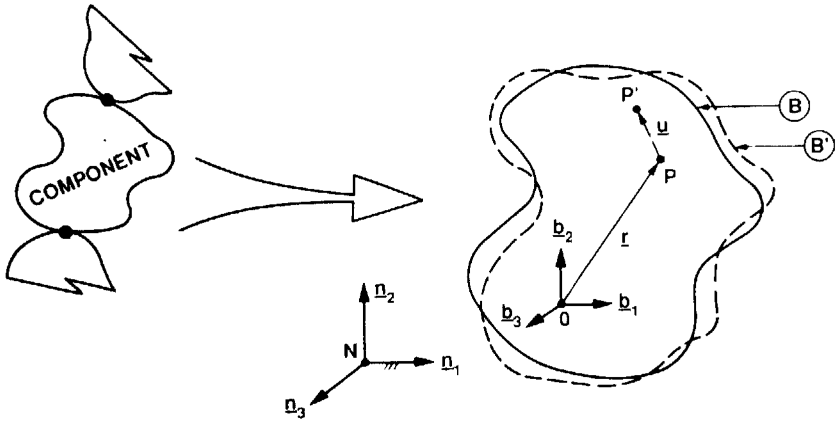
[31] Jonckheere, E. A, "Principle Component Analysis of Flexible Systems —Open-Loop Case," IEEE Transactions on Automatic Control, Vol. AC-29, No. 12, Dec. 1984, pp. 1095-1097.

[32] Gregory, C. Z., "Reduction of Large Flexible Spacecraft Models Using Internal Balancing Theory," AIAA Journal of Guidance and Control, Vol. 7, No. 6, Nov.-Dec. 1984, pp. 725-732.

[33] Blelloch, P. A., Mingori, D. L., and Wei, J. D., "Perturbation Analysis of Internal Balancing for Lightly Damped Mechanical Systems with Gyroscopic and Circulatory Forces," AIAA Journal of Guidance, Control, and Dynamics, Vol. 10, No. 4, July-Aug. 1987, pp. 406-410.

[34] Kailath T., *Linear Systems,* Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1980.

[35] Enns D., "Model Reduction for Control System Design," Ph.D. Dissertation, Dept. of Aeronautics and Astronautics, Stanford Univ., June 1984.



$$\underline{u} = \underline{u}\,(\underline{r}, t) = \sum_{i=1}^{m} \underline{\phi}_i\,(\underline{r})\,\eta_i\,(t)$$

Figure 1. Modeling Flexibility via the "Assumed-Modes" Method

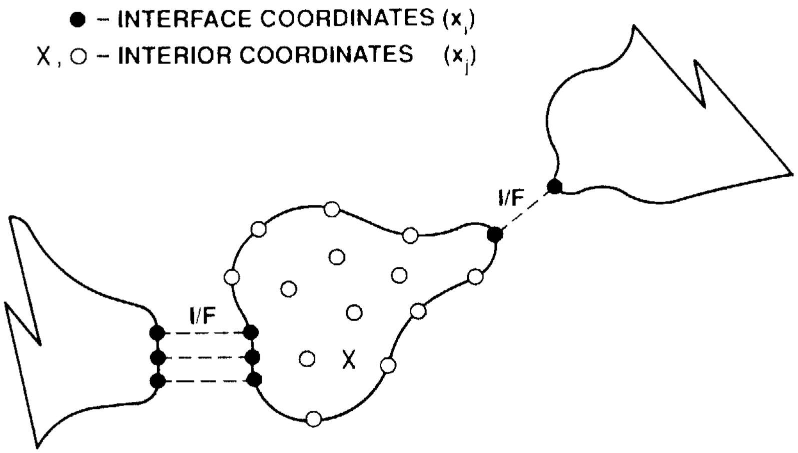
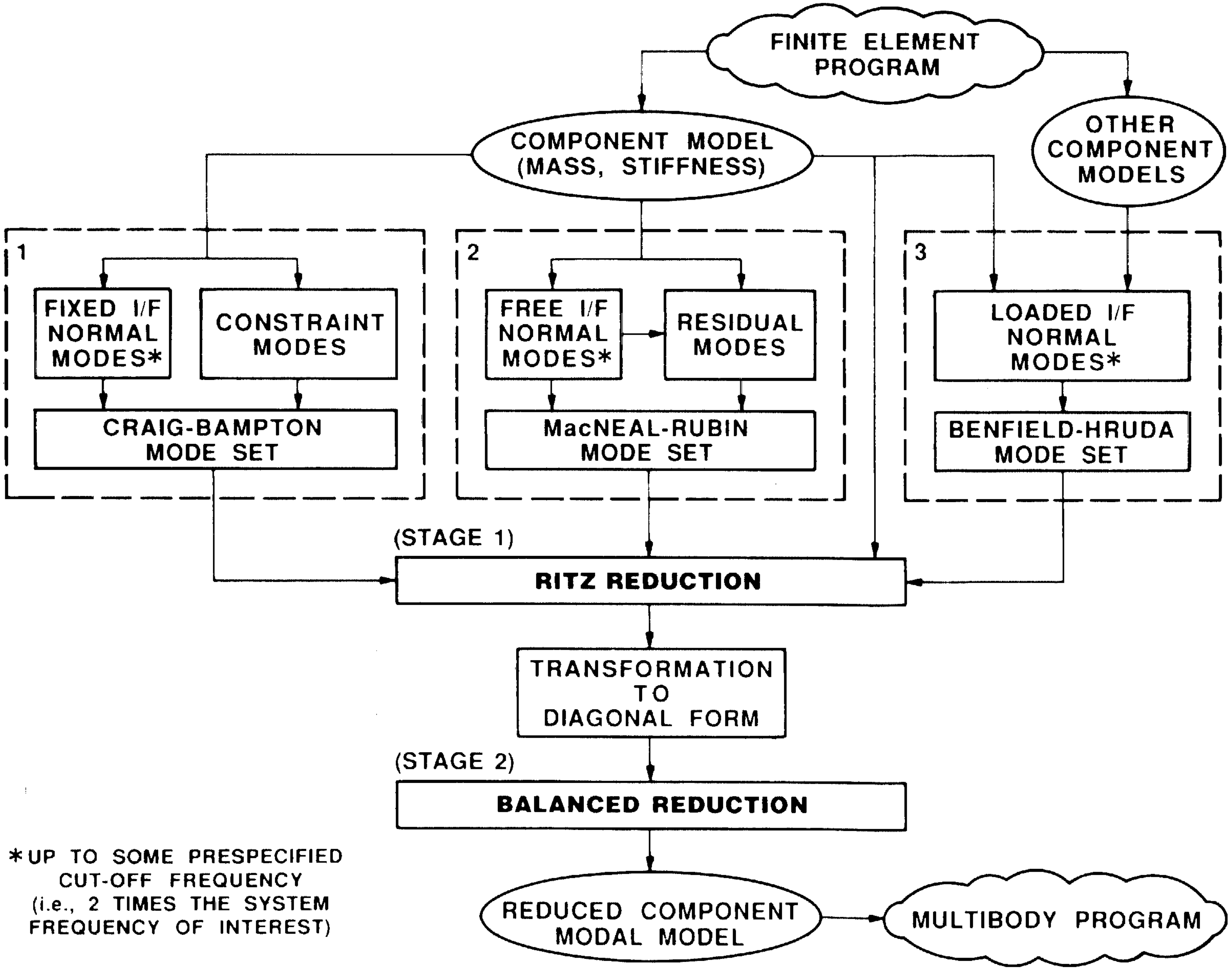

● – INTERFACE COORDINATES (x_i)
X, ○ – INTERIOR COORDINATES (x_j)
I/F
I/F
X

Figure 2. Elastic Component in a Multibody System



*UP TO SOME PRESPECIFIED CUT-OFF FREQUENCY (i.e., 2 TIMES THE SYSTEM FREQUENCY OF INTEREST)

Figure 3. Two-Stage Model Reduction Method

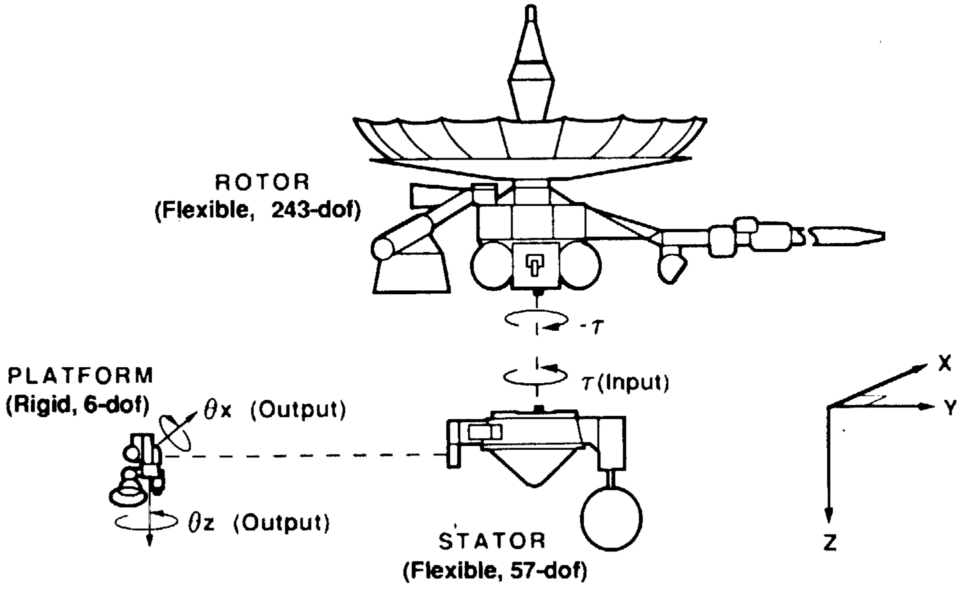


Figure 4(a). Galileo Spacecraft

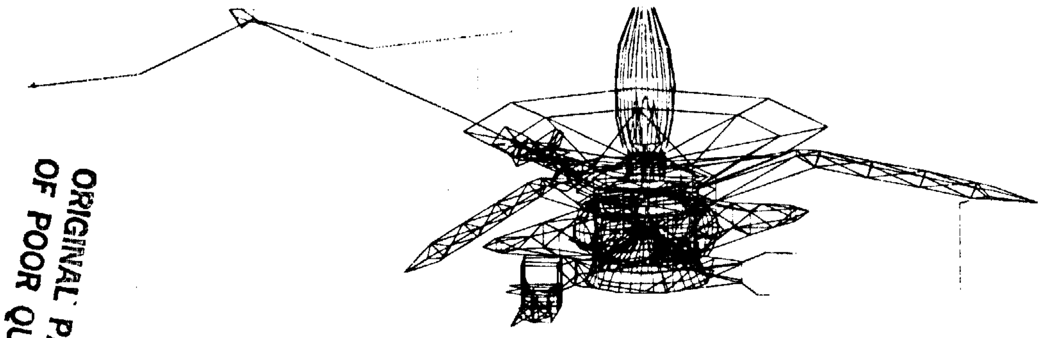


Figure 4(b). Galileo NASTRAN Model

ORIGINAL PAGE IS
OF POOR QUALITY

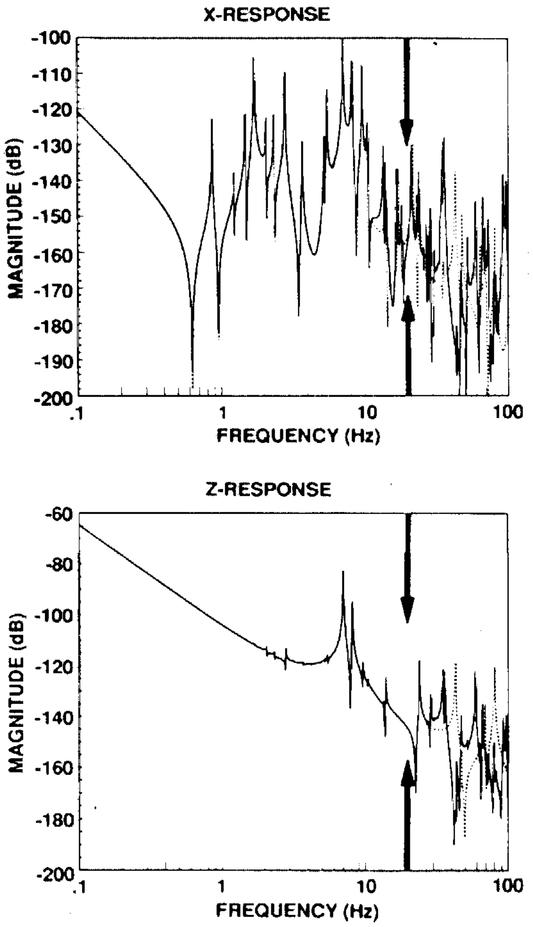
Table 1. Craig-Bampton, MacNeal-Rubin, and Benfield-Hruda Component Modes

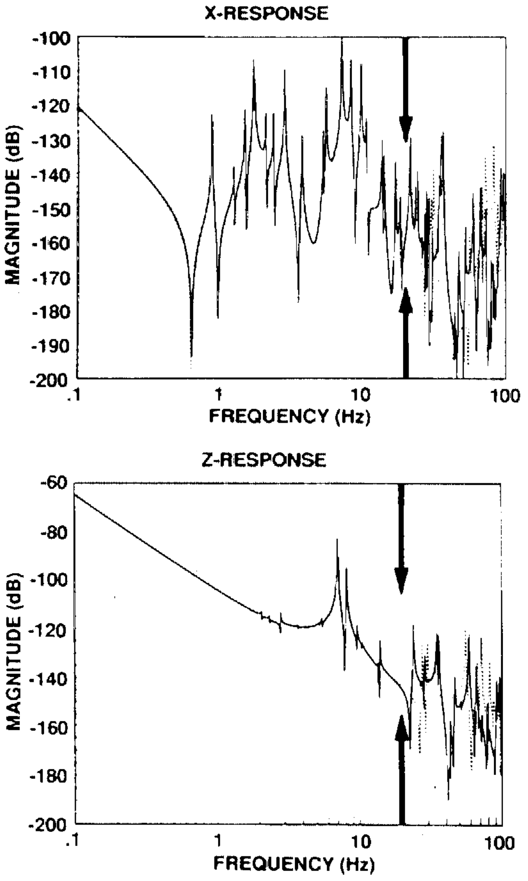
STATOR MODES (Hz)

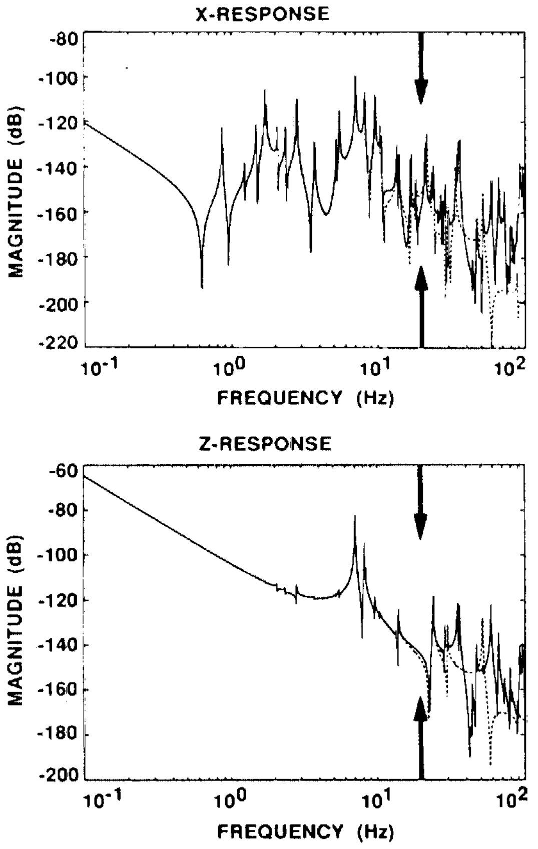
No.	C-B	M-R	B-H
1	7.09*	7.08	7.10
2	9.11*	9.08	9.11
3	10.55*	10.54	10.55
4	14.46*	14.44	14.47
5	23.26	38.99	23.25
6	25.98*	39.41	25.73
7	32.52*	52.46	27.31
8	46.32*	56.15	31.69
9	57.38*	80.92	56.79
10	83.18*	92.69	59.18
11	149.28*	152.43	96.46
12	211.08*	214.72	126.44
13	227.73*	230.84	201.00
14	244.73*	273.37	241.10
15	268.68*	305.07	321.17
16	328.50*	326.48	772.38

ROTOR MODES (Hz)

No.	C-B	M-R	B-H
1	0.14*	0.14	0.14
2	0.86*	0.86	0.86
3	1.22	1.22	1.22
4	1.23*	1.23	1.23
5	1.24*	1.24	1.24
6	1.48*	1.48	1.48
7	1.52	1.52	1.52
8	1.54	1.54	1.54
9	1.60	1.60	1.60
10	1.72*	1.72	1.72
11	2.28*	2.28	2.28
12	2.34	2.34	2.34
13	2.80*	2.80	2.80
14	3.07	3.07	3.07
15	3.23	3.23	3.23
16	3.23	3.23	3.23
17	3.65*	3.65	3.65
18	4.03*	4.03	4.03
19	4.97	4.97	4.97
20	5.02	5.02	5.02
21	5.09*	5.09	5.09
22	5.23*	5.23	5.23
23	5.45	5.45	5.45
24	5.45	5.45	5.45
25	5.58	5.58	5.58
26	5.58	5.58	5.58
27	5.63	5.63	5.63
28	5.63	5.63	5.63
29	5.65	5.65	5.65
30	5.65	5.65	5.65
31	5.65	5.65	5.65
32	5.65	5.65	5.65
33	5.66	5.66	5.66
34	5.78	5.78	5.78
35	5.78	5.78	5.78
36	5.80	5.80	5.80
37	5.81	5.81	5.81
38	5.82	5.82	5.82
39	5.82	5.82	5.82
40	5.83	5.83	5.83
41	5.83	5.83	5.83
42	5.83	5.83	5.83
43	5.83	5.83	5.83
44	5.83	5.83	5.83
45	5.83	5.83	5.83
46	5.83	5.83	5.83
47	5.83	5.83	5.83
48	5.83	5.83	5.83
49	5.83	5.83	5.83
50	5.89	5.89	5.89
51	5.99*	5.99	5.99
52	6.14	6.14	6.14
53	6.22	6.22	6.22
54	6.86	6.86	6.86
55	7.19	7.19	7.19
56	10.17	10.17	10.17
57	10.30	10.30	10.30
58	10.55	10.55	10.55
59	13.53	13.53	13.53
60	17.09	17.09	17.09
61	17.97	17.97	17.97
62	18.06	18.06	18.06
63	18.08	18.08	18.08
64	18.18	18.18	18.18
65	18.52	18.52	18.52
66	18.68	18.68	18.68
67	19.59	19.59	19.59
68	19.59	19.59	19.59
69	57.70*	58.61	20.97
70	58.63*	58.76	23.37
71	60.53*	74.48	23.53
72	66.40*	85.52	59.19
73	75.13*	96.31	59.42
74	100.30*	100.99	72.12







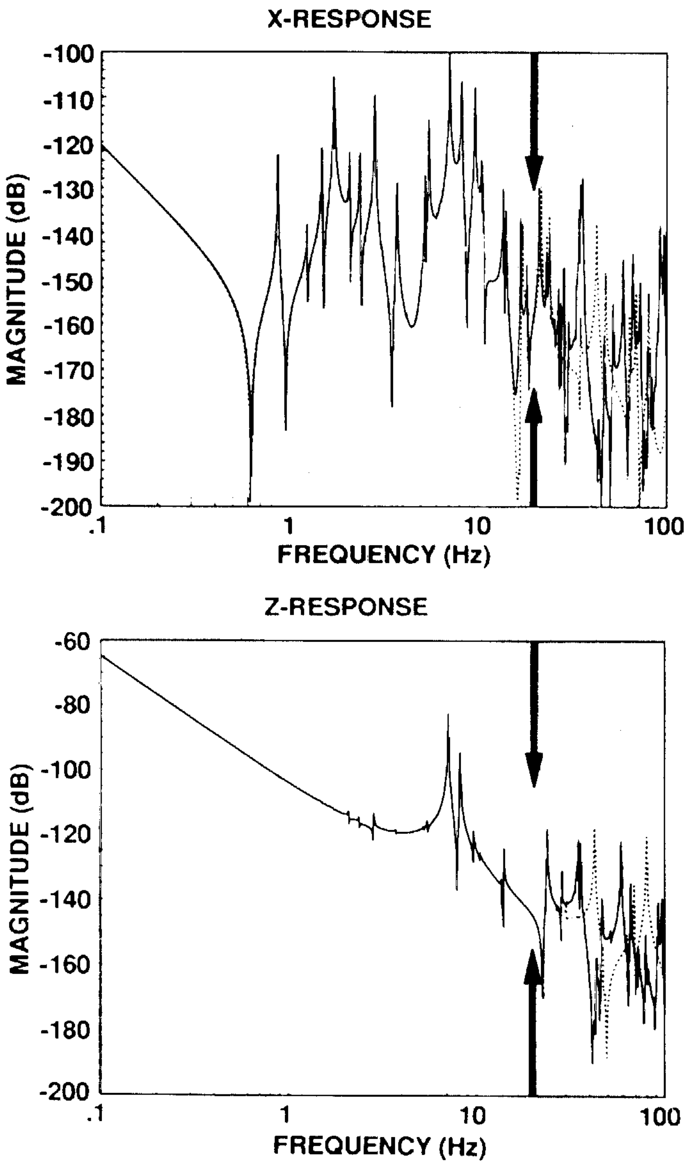
——— FULL ORDER MODEL (ROTOR = 237 MODES, STATOR = 49 MODES)

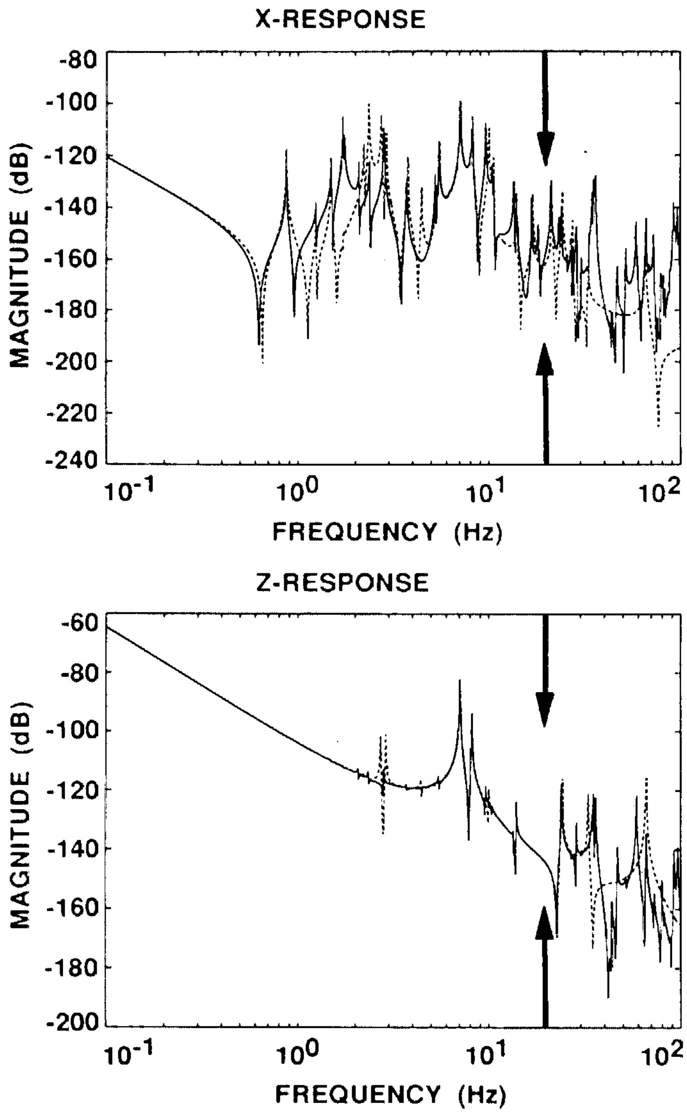
-------- REDUCED ORDER MODEL (ROTOR = 74 MODES, STATOR = 16 MODES)

Figure 5. Craig-Bampton Mode Set (Fixed I/F Component Modes to 20 Hz)

Figure 6. MacNeal-Rubin Mode Set (Free I/F Component Modes to 20 Hz)

Figure 7. Benfield-Hruda Mode Set (Loaded I/F Component Modes to 20 Hz)





——— FULL ORDER MODEL (ROTOR = 237 MODES, STATOR = 49 MODES)

- - - - - REDUCED ORDER MODEL (ROTOR = 19 MODES, STATOR = 15 MODES)

Figure 8. Reduced Craig-Bampton Mode Set via Modal Balancing

Figure 9. Reduced Craig-Bampton Mode Set via Frequency Cutoff

COMPONENT MODEL REDUCTION VIA THE PROJECTION AND ASSEMBLY METHOD

Douglas E. Bernard*

Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive
Pasadena, California 91109

ABSTRACT

The problem of acquiring a simple but sufficiently accurate model of a dynamic system is made more difficult when the dynamic system of interest is a multibody system comprised of several components. A low order system model may be created by reducing the order of the component models and making use of various available multibody dynamics programs to assemble them into a system model. The difficulty is in choosing the reduced order component models to meet system level requirements. The projection and assembly method, proposed originally by Eke, solves this difficulty by forming the full order system model, performing model reduction at the system level using system level requirements, and then projecting the desired modes onto the components for component level model reduction. In this paper, the projection and assembly method is analyzed to show the conditions under which the desired modes are captured exactly–to the numerical precision of the algorithm.

INTRODUCTION

The problem to be solved is that of simulating the dynamics of a multibody system. A multibody system is comprised of two or more bodies or components connected at hinges. In general, the bodies may be rigid or flexible, and the hinges may have from one to six independent degrees of freedom. Often all deformations of each body from its reference condition are in the linear range, although the resulting system dynamics is nonlinear. In this case, nonlinear system models may be constructed using linear dynamic models for each component, but allowing large angle motion between components. This is the approach used in a number of existing multibody software tools.

The problem is that system models constructed in this manner may be too large for use in control system design and simulation trades. Model reduction is needed to bring the model down to manageable size. If the system model is available in linear form, system model reduction can be applied directly. For the class of multibody problems discussed above, only the component models are available in linear form, and existing multibody software can be used if we reduce the component models before assembly into the system model. A multibody system is inherently a geometrically nonlinear system because of the time-varying, large-angle articulation between bodies.

Component model reduction is typically done to some level anyway if the source of the model is a finite element program. This first level of model reduction often uses some simple criterion such as "keep all cantilever modes below 40 Hz." The challenge is to reduce the component model further in some manner that preserves how the component behaves when connected to the complete system; how the component affects system level requirements. The projection and assembly method described in this paper attempts to do this.

Model reduction for linear systems has been addressed by a number of researchers, resulting in a variety of suggested linear system model reduction methods[1234]. Fig. 1 gives a high-level view of how these methods work. Less attention has been paid to the problem of model reduction for components of multibody systems. Component modal synthesis methods[5678] have the capability of producing reduced order component models, but typically do so based on component-level rather than system-level criteria. When only one body in a multibody system is flexible, Macala[9] captures desired system modes exactly by augmenting the flexible body by the mass and inertia of the rigid

*Member, Technical Staff, Guidance and Control Section body. A subset of the free-free modes of this augmented body are then used as the flexible body component modes. Eke and Man[10] extend this capability to systems of more than one flexible body with a method that involves choosing system modes of interest, projecting the mode shapes of these desired modes onto each flexible component, reducing the order of each component accordingly, and assembling the components into a system model. Upon assembly, each of the original desired system modes is recovered exactly (to the numerical precision of the algorithm.) As can be seen in Fig. 2, this approach is conceptually more complicated than that shown in Fig 1., but allows the introduction of system level requirements.

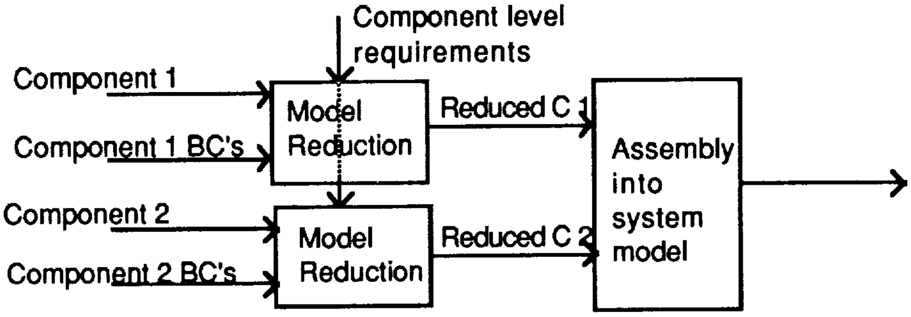


Fig. 1. Conventional Model Reduction

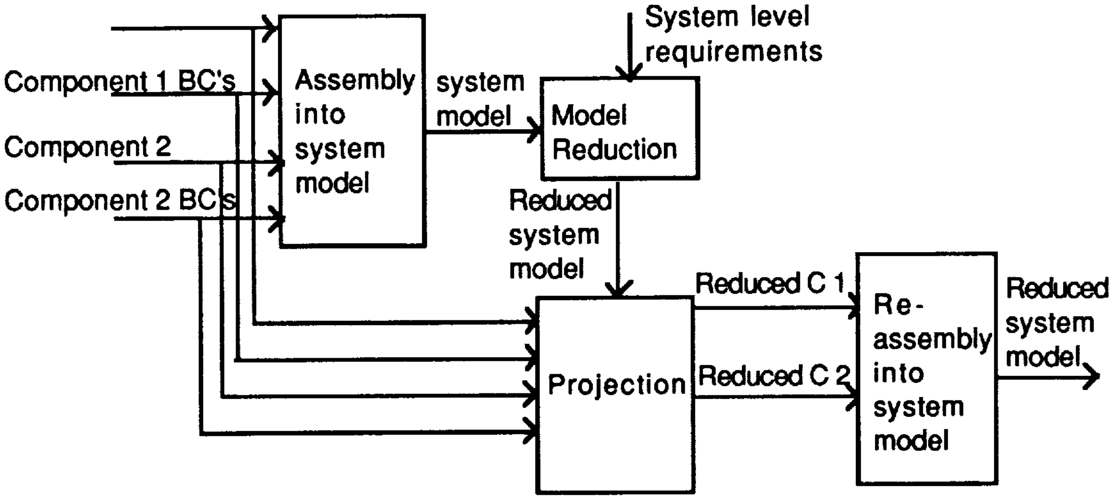


Fig. 2. Projection and Assembly Method

This paper analyzes the method outlined in Ref. 10 to show why the desired modes are returned exactly, presents necessary conditions for the success of the procedure, and proposes an extension to the method to handle situations when these necessary conditions are not met. Simple examples are presented to demonstrate the workings of the algorithm. The name "Projection and Assembly Method" is used to describe this component model reduction method.

DESCRIPTION OF METHOD

The idea of the projection and assembly method is to decide what system modes are important and to choose component models which, when assembled, capture those important system modes. The projection and assembly method is described in detail in Ref. 10. It works as follows:

- Acquire component models
- Synthesize the system model in some configuration of interest
- Apply any system level model reduction desired to choose which system free-free modes to retain
- Project the mode shapes of these retained modes onto each component.
- Choose new component states such that only these projected modes are admissible motions
- Transform the component models into reduced order component models using these new states
- Assemble the reduced order component models into a reduced order system model

The concept is made clearer by considering a simple qualitative example. Consider the planar motion of a system consisting of a rigid hub with two identical beam appendages, one on each side. Ignoring motions along the beam axes, the first five modes of this system are sketched in Fig. 3.

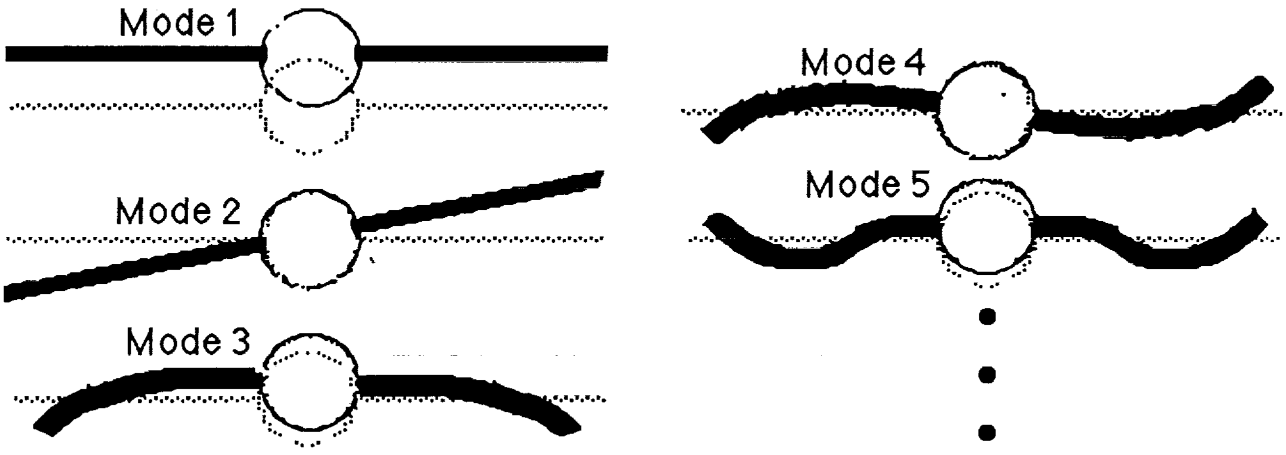


Fig. 3. Qualitative Beam Example

In this example, the lowest three system modes are chosen to be retained in the reduced order model. The projection step is illustrated in Fig. 4; these system modes are projected onto each of the two components. The resulting projections are used as generalized mode shapes for component models.

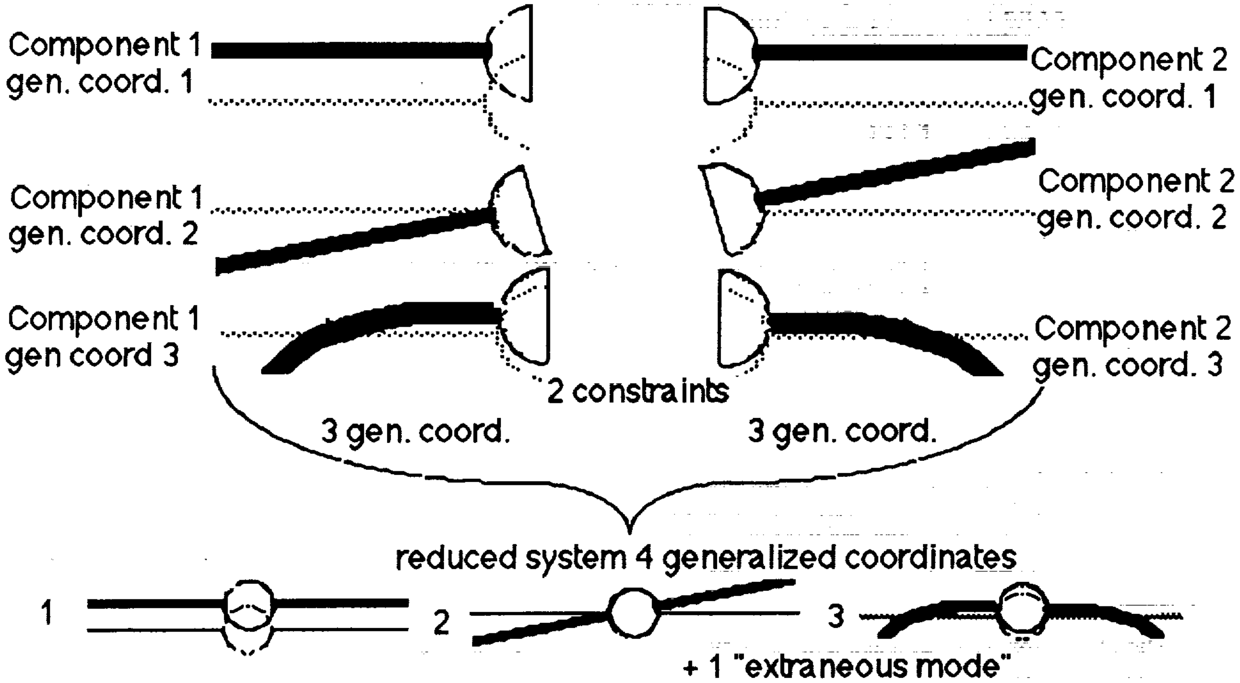


Fig. 4. Example of Projection onto Components

One side effect of the method is apparent by doing a little arithmetic. The projection and assembly method will project three modes onto each body. Assembling the components into a system gives two constraint relations (to match the halves of the rigid body together in angle and offset). When the reduced order component models are assembled, the reduced order system model will have four modes (3+3-2). These four include the three desired modes plus one "extraneous mode."

ANALYSIS

Component Equations of Motion

Assume we have n_b bodies or components. The unconstrained equations of motion of each may be expressed as:

$$M_i \ddot{x}_i + K_i x_i = G_i u, \qquad i = 1, \cdots, n_b \tag{1.}$$

where

i is the body index,

x_i is a set of generalized coordinates describing the motion of body i as a free body in inertial space. This set of coordinates can be anything from geometric coordinates to free-free normal modes to cantilever modes augmented by six rigid body modes for the fixed end,

M_i is the generalized mass matrix for body i,

K_i is the generalized stiffness matrix for body i,

u is the set of control inputs,

G_i is the control distribution matrix for body i, and

n_b is the number of bodies.

System Equations of Motion

A multibody system is created by constraining the components to share certain common motions and by adding flexible connections between bodies. Assume that the constraints can be described in the form:

$$AX = 0 \tag{2.}$$

where

$$A = [\, A_1 \;\; A_2 \cdots A_{n_b}], \qquad X^T = [\, x_1^T x_2^T \cdots \; x_{n_b}^T]$$

Let n_c be the number of constraint equations in Eq. 2. The constraints may be introduced into the equations of motion using a vector, Λ, of Lagrange multipliers. The constrained system is:

$$M_i \ddot{x}_i + K_i x_i = G_i u + A_i^T \Lambda, \qquad i = 1, \cdots, n_b \tag{3.}$$

$$A\dot{X} = 0. \tag{4.}$$

Let P be any full rank matrix mapping a minimal system state, x, into X:

$$X = Px; \quad \text{or} \quad x_i = P_i x, \quad i = 1, \cdots, n_b \tag{5.}$$

The constraint equation becomes:

$$AP\dot{x} = 0. \tag{6.}$$

Since the states x are independent, $AP=0$. Once P is chosen so that Eq. 6 is satisfied, the constraint equation (Eq. 4) is automatically satisfied. Inserting Eq. 5 into Eqs. 3 and pre-multiplying by P_i^T gives:

$$P_i^T M_i P_i \ddot{x} + P_i^T K_i P_i x = P_i^T G_i u + P_i^T A_i^T \Lambda, \qquad i = 1, \cdots, n_b \qquad 7.$$

Summing over i:

$$M\ddot{x} + Kx = Gu \qquad 8.$$

Where

$$M = \sum_{i=1}^{n_b} P_i^T M_i P_i \qquad K = \sum_{i=1}^{n_b} P_i^T K_i P_i \qquad 9.$$

$$G = \sum_{i=1}^{n_b} P_i^T G_i \qquad 10.$$

Equation 8 is the system equation of motion incorporating all constraints. Converting Eq. 8 to modal form:

$$x = \Phi q \qquad 11.$$

$$\ddot{q} + \Omega^2 q = \Phi^T G u. \qquad 12.$$

System Model Reduction

Assume we choose some model reduction method which yields as its output a set of n_R modes, q_R, to be retained with the remaining set of n_Z modes, q_Z, to be zeroed. Then we can partition Φ:

$$x = [\Phi_R \;\; \Phi_Z] \begin{bmatrix} q_R \\ q_Z \end{bmatrix}. \qquad 13.$$

Or, setting $q_Z = 0$, the reduced order system model is:

$$\ddot{q}_R + \Omega_R^2 q_R = \Phi_R^T G u \qquad 14.$$

$$x = \Phi_R q_R \qquad 15.$$

If $\Omega^2 = \mathrm{diag}(\omega_j^2)$ then a homogeneous solution to Eq. 12, and therefore also a solution to the system of equations 3 & 4, is $q = e_j \cos(\omega_j t)$. Each of x_i, $\ddot{x}_i$ and Λ will similarly be described by sinusoids:

$$x_i = P_i \Phi_R e_j \cos(\omega_j t), \qquad \ddot{x}_i = P_i \Phi_R e_j (-\omega_j^2)\cos(\omega_j t), \qquad \Lambda = \Lambda_{oj} \cos(\omega_j t).$$

Inserting the above into Eq. 3 for u=0 gives a relation which will be needed in a later derivation:

$$\left[M_i(-\omega_j^2) + K_i\right] P_i \Phi_R e_j = A_i^T \Lambda_{oj}, \qquad i = 1, \cdots, n_b \qquad 16.$$

Component Model Reduction

None of the above is unique to the projection and assembly method, which uses the above as a starting point. The concept is as follows: Cause each component to have, as an allowable motion, the mode shape of each retained mode projected onto the component. When the system is reassembled from reduced order components, the retained mode will still be an admissible motion of the reduced order system. In the following, it will be shown that in addition to being an admissible motion of the reduced order system, it is a mode of the reduced order system.

Consider the projection of q_R onto component i. Using Eqs. 5 and 15:

$$x_i = P_i \Phi_R q_R. \qquad 17.$$

In general, q_R should be of lower order than x_i. Where before, component i had n_i degrees of freedom, Eq. 15 restricts the motion to n_R degrees of freedom. Let x_{Ri} be a set of component i modes that span the space of component motions allowed by Eq. 15. In Ref. 10, the choice: $x_{Ri} = q_R$ is made, so:

$$x_i = P_i \Phi_R x_{Ri}. \qquad 18.$$

Implicit in this choice is the assumption that the matrix $P_i \Phi_R$ is of full column rank. This assumption is violated in a number of situations. The most obvious case is when one component has fewer degrees of freedom than the number of modes in Φ_R. Other examples arise when the projections of the modes are linearly dependant within the subspace of a particular component. In a later section of this paper, an alternative choice for x_{Ri} is explored for situations where $P_i \Phi_R$ is not of full rank. Writing the component equations of motion (Eq. 3) and constraint relation (Eq. 4) in terms of the x_{Ri}:

$$M_{Ri}\ddot{x}_{Ri} + K_{Ri}x_{Ri} = G_{Ri}u + A_{Ri}^T\Lambda, \qquad i = 1, \cdots, n_b \qquad 19.$$

$$A_R\dot{X}_R = \sum_{i=1}^{n_b} A_{Ri}\dot{x}_{Ri} = 0, \qquad 20.$$

where

$$M_{Ri} = \Phi_R^T P_i^T M_i P_i \Phi_R \qquad K_{Ri} = \Phi_R^T P_i^T K_i P_i \Phi_R \qquad 21.$$

$$G_{Ri} = \Phi_R^T P_i^T G_i \qquad A_{Ri} = A_i P_i \Phi_R \qquad 22.$$

and A_R and X_R are defined in the same manner as A and X. This system of equations in x_{Ri} and Λ may be formulated in terms of a minimal set of states, x_R, with some mapping P_R.

$$X_R = P_R x_R. \qquad 23.$$

With this choice, Eq. 20 becomes:

$$A_R P_R x_R = 0. \qquad 24.$$

Since the x_R are independent:

$$A_R P_R = 0. \tag{25.}$$

In actual practice, P_R has a specific form, but to understand the behavior of the reduced order system, we can consider any P_R which is of full rank and satisfies Eq. 25. If $x_{R1} = x_{R2} = \cdots = x_{Rn_b}$, as will be the case for the desired retained modes, then Eq. 20 becomes:

$$\left(\sum_{i=1}^{n_b} A_i P_i\right)\Phi_R I x_{R1} = 0$$

which is automatically satisfied in view of Eq. 6. This suggests that a partial choice for P_R is the column: $[\,I\,I \cdots I\,]^{\mathrm{T}}$. One full rank P_R which satisfies Eq. 20 may be created by taking the singular value decomposition of a portion of A_R:

$$[\,A_{R2} \cdots A_{Rn_b}] = U_A \Sigma_A V_A^T = U_A [\Sigma_{A1} \;\; 0] \begin{bmatrix} V_{A1}^T \\ V_{A2}^T \end{bmatrix} = U_A \Sigma_{A1} V_{A1}^T \tag{26.}$$

and choosing:

$$P_R^T = \begin{bmatrix} I & I \cdots I \\ 0 & V_{A2}^T \end{bmatrix} \tag{27.}$$

so

$$\begin{aligned} A_R P_R &= \left[\; \left(\sum_{i=1}^{n_b} A_{Ri}\right) \quad [\,A_{R2} \cdots A_{Rn_b}] V_{A2} \right] \\ &= \left[\; \left(\sum_{i=1}^{n_b} A_i P_i\right)\Phi_R \quad U_A \Sigma_{A1} V_{A1}^T V_{A2} \right] = 0 \end{aligned}$$

as desired. Furthermore, P_R is of full rank by construction.

Starting from Eq. 19, the equations of motion in terms of x_R are:

$$\sum_{i=1}^{n_b} P_{Ri}^T M_{Ri} P_{Ri} \ddot{x}_R + \sum_{i=1}^{n_b} P_{Ri}^T K_{Ri} P_{Ri} x_R = \sum_{i=1}^{n_b} P_{Ri}^T G_{Ri} u + P_R^T A_R^T \Lambda. \tag{28.}$$

The form of P_R suggests a partitioning of x_R and P_R into desired and extra states:

$$x_R = \begin{bmatrix} x_D \\ x_E \end{bmatrix}, \qquad P_R = [P_{RD} \;\; P_{RE}]$$

where $P_{RDi} = I$ and $P_{RE} = \begin{bmatrix} 0 \\ V_{A2} \end{bmatrix}$. In partitioned form, Eq. 28 is:

$$\begin{bmatrix} \sum\limits_{i=1}^{n_b} M_{Ri} & \sum\limits_{i=1}^{n_b} M_{Ri}P_{REi} \\ \sum\limits_{i=1}^{n_b} P_{REi}^T M_{Ri} & \sum\limits_{i=1}^{n_b} P_{REi}^T M_{Ri} P_{REi} \end{bmatrix} \begin{bmatrix} \ddot{x}_D \\ \ddot{x}_E \end{bmatrix} + \begin{bmatrix} \sum\limits_{i=1}^{n_b} K_{Ri} & \sum\limits_{i=1}^{n_b} K_{Ri}P_{REi} \\ \sum\limits_{i=1}^{n_b} P_{REi}^T K_{Ri} & \sum\limits_{i=1}^{n_b} P_{REi}^T K_{Ri} P_{REi} \end{bmatrix} \begin{bmatrix} x_D \\ x_E \end{bmatrix} = \begin{bmatrix} \sum\limits_{i=1}^{n_b} G_{Ri} \\ \sum\limits_{i=1}^{n_b} P_{REi}^T G_{Ri} \end{bmatrix} u. \qquad 29.$$

By construction, the system is capable of taking the shape of any of the n_R desired modes. It remains to be shown that the x_D are free-free normal modes of the reduced order system. To show that they are requires only that $x_R = \begin{bmatrix} e_j \\ 0 \end{bmatrix} \cos\omega_j t$ be a solution of Eq. 29 with $u = 0$. Assume $x_R = \begin{bmatrix} e_j \\ 0 \end{bmatrix} \cos\omega_j t$, then: $\ddot{x}_R = (-\omega_j^2)x_R$ and Eq. 29 becomes two equations:

$$\left[\sum_{i=1}^{n_b} M_{Ri}(-\omega_j^2) + \sum_{i=1}^{n_b} K_{Ri} \right] e_j \cos\omega_j t \overset{?}{=} 0 \qquad 30.$$

$$\left[\sum_{i=1}^{n_b} P_{REi}^T M_{Ri}(-\omega_j^2) + \sum_{i=1}^{n_b} P_{REi}^T K_{Ri} \right] e_j \cos\omega_j t \overset{?}{=} 0. \qquad 31.$$

If both left-hand sides in the above equations evaluate to zero, then the desired modes are modes of the reduced order system. Consider ΣM_{Ri} and ΣK_{Ri}:

$$\sum_{i=1}^{n_b} M_{Ri} \underset{(21)}{=} \sum_{i=1}^{n_b} \Phi_R^T P_i^T M P_i \Phi_R = \Phi_R^T \sum_{i=1}^{n_b} P_i^T M P_i \Phi_R \underset{(9)}{=} \Phi_R^T M \Phi_R = I.$$

Similarly,

$$\sum_{i=1}^{n_b} K_{Ri} = \Omega^2.$$

Eq. 30 becomes:

$$((-\omega_j^2)I + \Omega^2)e_j = (\omega_j^2 - \omega_j^2)e_j = 0$$

and so is satisfied. Consider Eq. 31:

$$\left[\sum_{i=1}^{n_b} P_{REi}^T M_{Ri}(-\omega_j^2) + \sum_{i=1}^{n_b} P_{REi}^T K_{Ri}\right] e_j \underset{(21)}{=} \left[\sum_{i=1}^{n_b} P_{REi}^T \Phi_R^T P_i^T \left(M_i(-\omega_j^2) + K_i\right) P_i \Phi_R\right] e_j$$

$$\underset{(16)}{=} \sum_{i=1}^{n_b} P_{REi}^T \Phi_R^T P_i^T A_i^T \Lambda_{oj} \underset{(22)}{=} \sum_{i=1}^{n_b} P_{REi}^T A_{Ri}^T \Lambda_{oj} = [0\ V_{A2}^T] A_R^T \Lambda_{oj} \underset{(26)}{=} 0\Lambda_{oj} = 0.$$

Therefore the desired mode shapes and frequencies satisfy Eq. 29 and thus are normal modes of the reassembled reduced order system.

Component Model Reduction—Extended Method

As mentioned above, the choice: $x_{Ri} = q_R$ depends on the matrix $P_i\Phi_R$ being of full column rank. When this is not the case, the method can be extended to allow model reduction to proceed. Consider the singular value decomposition of $P_i\Phi_R$, suppressing the index i on the products in the SVD, let r be the rank of $P_i\Phi_R$, let n_R be the rank of Φ_R, and let n_i be the rank of P_i (and the number of states in x_i). If $r=n_R>n_i$ (Ref. 10)

$$P_i\Phi_R = U\Sigma V^T = [U_1\ \ U_2]\begin{bmatrix}\Sigma_1\\ 0\end{bmatrix}V^T = U_1\Sigma_1 V^T.$$

If $r=n_i>n_R$ (body i has few DOF)

$$P_i\Phi_R = U\Sigma V^T = U[\Sigma_1\ \ 0]\begin{bmatrix}V_1^T\\ V_2^T\end{bmatrix} = U\Sigma_1 V_1^T.$$

If $r<n_i$, $r<n_R$ (linear dependant projected modes)

$$P_i\Phi_R = U\Sigma V^T = [U_1\ \ U_2]\begin{bmatrix}\Sigma_1 & 0\\ 0 & 0\end{bmatrix}\begin{bmatrix}V_1^T\\ V_2^T\end{bmatrix} = U_1\Sigma_1 V_1^T$$

To ensure that the set x_{Ri} is an independent set spanning the space of component motions, choose $x_{Ri} = \Sigma_1(i)V_1^T(i)q_R$. In the event that $r(i)=n_R$, $V_1^T(i)$ becomes $V(i)$. This choice of x_{Ri} gives for Eq. 18:

$$x_i = U_1(i)x_{Ri} \qquad \text{18A.}$$

In the event that $r(i)=n_i$, $U_1(i)$ becomes $U(i)$. Define $Q_i = U_1(i)$. Eqs. 21–22 now take the form:

$$M_{Ri}=Q_i^T M_i Q_i \qquad K_{Ri}=Q_i^T K_i Q_i \qquad \text{21A.}$$

$$G_{Ri}=Q_i^T G_i \qquad A_{Ri}=A_i Q_i \qquad \text{22A.}$$

Choosing:

$$P_R^T = \begin{bmatrix} V_1(1)\Sigma_1(1) & V_1(2)\Sigma_1(2) \cdots V_1(n_b)\Sigma_1(n_b) \\ 0 & V_{A\,2}^T \end{bmatrix} \qquad \text{27A.}$$

gives as desired, $A_R P_R = 0$. Moreover, P_R is again full rank by construction. Partition P_R as before: $P_R = \left[P_{RD} \;\; P_{RE}\right]$, where $P_{RDi} = \Sigma_1(i)\, V_1^T(i)$ and $P_{RE} = \begin{bmatrix} 0 \\ V_{A2} \end{bmatrix}$. Eq. 29 becomes:

$$\begin{bmatrix} \sum_{i=1}^{n_b} P_{RDi}^T M_{Ri} P_{RDi} & \sum_{i=1}^{n_b} P_{RDi}^T M_{Ri} P_{REi} \\ \sum_{i=1}^{n_b} P_{REi}^T M_{Ri} P_{RDi} & \sum_{i=1}^{n_b} P_{REi}^T M_{Ri} P_{REi} \end{bmatrix} \begin{bmatrix} \ddot{x}_D \\ \ddot{x}_E \end{bmatrix}$$

$$+ \begin{bmatrix} \sum_{i=1}^{n_b} P_{RDi}^T K_{Ri} P_{RDi} & \sum_{i=1}^{n_b} P_{RDi}^T K_{Ri} P_{REi} \\ \sum_{i=1}^{n_b} P_{REi}^T K_{Ri} P_{RDi} & \sum_{i=1}^{n_b} P_{\mathrm{REi}}^T K_{Ri} P_{REi} \end{bmatrix} \begin{bmatrix} x_D \\ x_E \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n_b} P_{RDi}^T G_{Ri} \\ \sum_{i=1}^{n_b} P_{REi}^T G_{Ri} \end{bmatrix} u \qquad 29A.$$

and the proof that the desired modes are normal modes of the reassembled reduced order system proceeds exactly as before, using the above definition of P_{RDi} and Eqs. 21A and 22A.

SIMPLE EXAMPLES

One Dimensional Three Disk Example

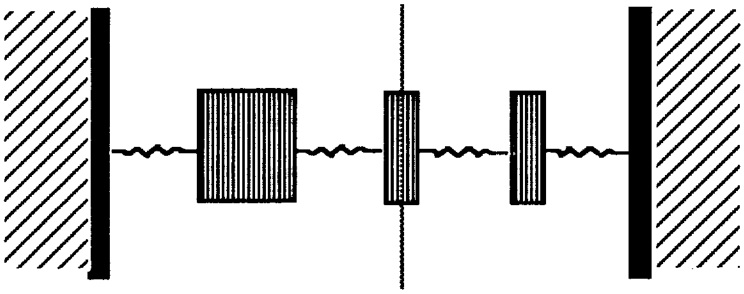


Fig. 5. One Dimensional Three Disk Example

Consider Fig. 5. In this example, there are three disks, with rotational displacements (from left to right) y_1, y_2, and y_3 and inertias $4J$, J, and J connected to ground and each other by torsion rods of equal spring constant, k. We choose to consider this simple system as being composed of two simpler subsystems of components. We divide the middle disk in half and allocate one half to each subsystem. Subsystem 1 contains the large disk and the left half of the middle disk. Take $x_1 = [y_1, y_2]^T$. Subsystem 2 contains the rest of the system. Take $x_2 = [y_2, y_3]^T$. Choosing units to make J and k equal to unity, the mass and stiffness matrices for each component are:

$$M_1 = \begin{bmatrix} 4 & 0 \\ 0 & 0.5 \end{bmatrix}, \; K_1 = \begin{bmatrix} 2 & -1 \\ -1 & 1 \end{bmatrix}, \; M_2 = \begin{bmatrix} 0.5 & 0 \\ 0 & 1 \end{bmatrix}, \; K_2 = \begin{bmatrix} 1 & -1 \\ -1 & 2 \end{bmatrix}.$$

The constraint relation that connects the subsystems is that $x_1(2) = x_2(1)$. Expressed in terms of a constraint matrix, A:

$$A = [\ A_1 \quad A_2\ ], \quad [\ A_1 \quad A_2\ ]\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 0$$

with

$$A_1 = [\ 0 \quad 1\ ], \quad A_2 = [\ -1 \quad 0\ ].$$

One choice of P which reduces this to minimal form is:

$$P = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 0.7071 & 0 \\ 0 & 0.7071 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

The system mass and stiffness matrices are:

$$M = \begin{bmatrix} 4 & 0 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad K = \begin{bmatrix} 2 & 0.7071 & 0 \\ 0.7071 & 1 & -0.7071 \\ 0 & -0.7071 & 2 \end{bmatrix}.$$

The eigenvalue and eigenvector matrices for this system are:

$$\Omega^2 = \begin{bmatrix} 0.2803 & 0 & 0 \\ 0 & 1.1694 & 0 \\ 0 & 0 & 3.0502 \end{bmatrix}; \quad \Phi = \begin{bmatrix} 0.4457 & -0.2153 & 0.0703 \\ -0.5539 & -0.8155 & 1.0140 \\ -0.2277 & -0.6943 & -0.6827 \end{bmatrix}.$$

System model reduction: Assume we wish to capture only the lowest frequency system mode ($\Omega^2 = 0.2803$), then

$$\Phi_R = \begin{bmatrix} 0.4457 \\ -0.5539 \\ -0.2277 \end{bmatrix}.$$

Component model reduction: choose $x_{R1} = x_{R2} = q_R$, so

$$x_1 = P_1 \Phi_R x_{R1} = \begin{bmatrix} -0.4457 \\ -0.3917 \end{bmatrix} x_{R1} \qquad x_2 = P_2 \Phi_R x_{R2} = \begin{bmatrix} -0.3917 \\ -0.2277 \end{bmatrix} x_{R2}$$

and the reduced order component mass and stiffness matrices are:

$$M_{R1} = [0.8714], \quad K_{R1} = [0.2016], \quad M_{R2} = [0.1286], \quad K_{R2} = [0.0787].$$

The reduced order constraint matrix is:

$$A_R = [\ -0.3916 \quad 0.3916\ ].$$

Choose P_R:

$$P_R = \begin{bmatrix} -0.7071 \\ -0.7071 \end{bmatrix}.$$

This gives the reduced order system:

$$[0.5]\ddot{x}_R + [0.1402]x_R = 0.$$

Which has a single eigenvalue at $\Omega^2 = 0.2803$. In this case, no extra modes are created because it happens that ($2n_R$- # constraints) = n_R. This is not true in general. The next example produces extra modes.

One Dimensional Five Disk Example

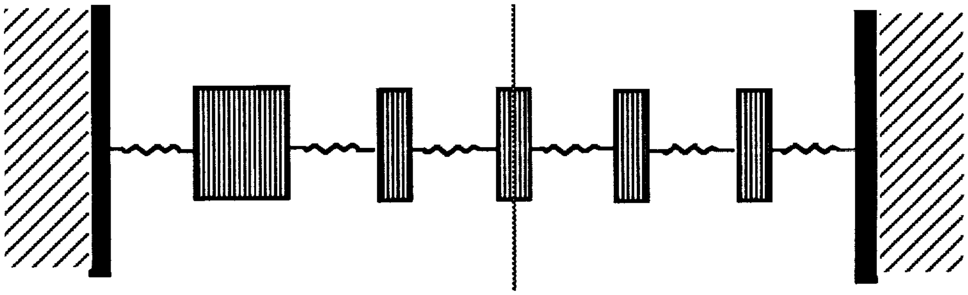


Fig. 6. One Dimensional Five Disk Example

Consider Fig. 6. In this example, there are five disks, with displacements (from left to right) y_1, y_2, y_3, y_4, and y_5 and inertias $4J, J, J, J$, and J connected to ground and each other by torsion rods of equal spring constant, k. We choose to consider this simple system as being composed of two simpler subsystems of components. We divide the middle disk in half and allocate one half to each subsystem. Subsystem 1 contains the large disk through the left half of the middle disk. Take $x_1 = [y_1, y_2, y_3]^T$. Subsystem 2 contains the rest of the system. Take $x_2 = [y_3, y_4, y_5]^T$. Choosing units to make J and k equal to unity, the mass and stiffness matrices for each component are:

$$M_1 = \begin{bmatrix} 4 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0.5 \end{bmatrix}, M_2 = \begin{bmatrix} 0.5 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}; \quad K_1 = \begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 1 \end{bmatrix}, K_2 = \begin{bmatrix} 1 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix}.$$

The constraint relation that connects the subsystems is that $x_1(3) = x_2(1)$. Expressed in terms of a constraint matrix, A:

$$A = [\ A_1 \quad A_2\ ], \quad [\ A_1 \quad A_2\ ]\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 0,$$

with

$$A_1 = [0 \ 0 \ 1], \quad A_2 = [-1 \ 0 \ 0].$$

One choice of P which reduces this to minimal form is:

$$P = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 \\ 0 & -0.7071 & 0.7071 & 0 & 0 \\ 0 & 0.5 & 0.5 & 0 & 0 \\ 0 & 0.5 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

The system mass and stiffness matrices are:

$$M = \begin{bmatrix} 4 & 0 & 0 & 0 & 0 \\ 0 & 0.75 & -0.25 & 0 & 0 \\ 0 & -0.25 & 0.75 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad K = \begin{bmatrix} 2 & -0.7071 & 0.71 & 0 & 0 \\ -0.7071 & 2.2071 & -0.5 & -0.5 & 0 \\ 0.7071 & -0.5 & 0.7929 & -0.5 & 0 \\ 0 & -0.5 & -0.5 & 2 & -1 \\ 0 & 0 & 0 & -1 & 2 \end{bmatrix}.$$

The eigenvalue and eigenvector matrices for this system are:

$$\Omega^2 = \begin{bmatrix} 0.1933 & 0 & 0 & 0 & 0 \\ 0 & 0.5466 & 0 & 0 & 0 \\ 0 & 0 & 1.4696 & 0 & 0 \\ 0 & 0 & 0 & 2.6609 & 0 \\ 0 & 0 & 0 & 0 & 3.6296 \end{bmatrix},$$

$$\Phi = \begin{bmatrix} -0.3489 & 0.3208 & -0.1392 & -0.0704 & -0.0310 \\ 0.1217 & 0.3655 & -0.0438 & 0.7626 & 0.8765 \\ 0.7270 & 0.4501 & -0.8075 & -0.0986 & 0.3272 \\ 0.3386 & 0.5329 & 0.3142 & 0.3895 & -0.5924 \\ 0.1874 & 0.3666 & 0.5924 & -0.5894 & 0.3635 \end{bmatrix}.$$

System model reduction: Assume we wish to capture only the two lowest frequency system modes (Ω^2 = 0.1933, 0.5466), then

$$\Phi_R = \begin{bmatrix} -0.3489 & 0.3208 \\ 0.1217 & 0.3655 \\ 0.7270 & 0.4501 \\ 0.3386 & 0.5329 \\ 0.1874 & 0.3666 \end{bmatrix}.$$

Component model reduction: choose $x_{R1} = x_{R2} = q_R$, so

$$x_1 = P_1 \Phi_R x_{R1} = \begin{bmatrix} 0.3489 & -0.3208 \\ 0.4280 & 0.0598 \\ 0.4244 & 0.4078 \end{bmatrix} x_{R1}, \quad x_2 = P_2 \Phi_R x_{R2} = \begin{bmatrix} 0.4244 & 0.4078 \\ 0.3386 & 0.5329 \\ 0.1874 & 0.3666 \end{bmatrix} x_{R2}.$$

and the reduced order component mass and stiffness matrices are:

$$M_{R1} = \begin{bmatrix} 0.7602 & -0.3357 \\ -0.3357 & 0.4985 \end{bmatrix}, \quad K_{R1} = \begin{bmatrix} 0.1280 & -0.0831 \\ -0.0831 & 0.3689 \end{bmatrix},$$

$$M_{R2} = \begin{bmatrix} 0.2398 & 0.3357 \\ 0.3357 & 0.5015 \end{bmatrix}, \quad K_{R2} = \begin{bmatrix} 0.0653 & 0.0831 \\ 0.0831 & 0.1777 \end{bmatrix}.$$

The reduced order constraint matrix is:

$$A_R = [\, 0.4243 \;\; 0.4078 \;\; -0.4243 \;\; -0.4078 \,].$$

Choose P_R:

$$P_R = \begin{bmatrix} -0.8603 & 0 & 0 \\ 0.2904 & 0.5926 & 0.5696 \\ -0.3021 & 0.7762 & -0.2151 \\ -0.2904 & -0.2151 & 0.7933 \end{bmatrix}.$$

This gives the reduced order system:

$$\begin{bmatrix} 0.8954 & 0.1781 & 0.0875 \\ 0.1781 & 0.2307 & 0.2649 \\ 0.0875 & 0.2649 & 0.3739 \end{bmatrix} \ddot{x}_R + \begin{bmatrix} 0.2029 & 0.0883 & 0.0503 \\ 0.0883 & 0.1494 & 0.1383 \\ 0.0503 & 0.1383 & 0.2062 \end{bmatrix} x_R = 0 \quad ,$$

which has three eigenvalues at Ω^2 = (0.1933, 0.5466, 1.6873). The first two are the desired system modes, while the third does not match any of the original system modes; it is an "extraneous mode."

SUMMARY

In this paper the model reduction method described by Eke and Man in Ref. 10 has been analyzed to demonstrate why the desired modes are returned exactly. An explicit set of necessary conditions involving the rank of the projection matrix has been presented, and an extension to the method has been proposed which removes those conditions. The method was demonstrated using two simple examples.

Future work will address extending the method to handle variable configuration systems such as those with multiple articulation angles, better characterizing the "extraneous" modes which are a by-product of this method, and examining scaling issues which will arise when relative sizes of singular values are used to determine how many independent modes are projected onto a component.

ACKNOWLEDGEMENTS

A number of JPL analysts have contributed to this work. Fidelis Eke receives much of the credit for developing the method. Guy Man, Robert Bamford, and Glenn Macala contributed significantly through technical discussions during the time the method was being applied to model reduction for the Galileo spacecraft.

The research described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

REFERENCES

1) Moore, B., "Principal Component Analysis in Linear Systems: Controllability, Observability, and Model Reduction," IEEE Trans. Automat. Contr.,Vol AC-26, pp 17–31, Feb. 1981.

2) Skelton, R.E., Singh, R., Ramakrishnan, J., "Component Model Reduction by Component Cost Analysis," AIAA Guidance and Control Conference, June, 1988, Paper # 88-4086-CP.

3) Bernard, D.E., "On the Control of Lightly Coupled Large Space Structures," Ph.D. Dissertation, Stanford University, October, 1984.

4) Enns, D. "Model Reduction for Control System Design," Ph.D. Dissertation, Stanford University, June, 1984.

5) Craig, R.R. Jr. and Bampton, M.C.C., "Coupling of Structures for Dynamic Analysis," AIAA Journal, Vol 6, No. 7, pp. 1313–1319, July 1968.

6) Yoo, W.S., and Haug, E.J., "Dynamics of Articulated Structures, Part I, Theory," J. Struct. Mech., Vol. 14, No. 1, pp. 105–126, 1986.

7) MacNeal, R.H., "A Hybrid Method of Component Mode Synthesis," Computers and Structures, Vol. 1, pp. 581–601, 1971.

8) Rubin, S., "An Improved Component-Mode Representation, AIAA Paper 74-386, April 1974.

9) Macala, G.A., "A Model Reduction Method for Use With Nonlinear Simulations for Flexible Multibody Spacecraft," Presented at the AIAA/AAS Astrodynamics Conference, August , 1984.

10) Eke, F.O. and Man, G.K., "Model Reduction in the Simulation of Interconnected Flexible Bodies," AAS Paper 87-445, AAS/AIAA Astrodynamics Specialist Conference, Kalispell, Montana, August 10–13, 1987.

MODEL REDUCTION IN THE PHYSICAL COORDINATE SYSTEM

K. Harold Yae and K.Y. Jeong
Center for Simulation and Design Optimization
Department of Mechanical Engineering
The University of Iowa
Iowa City, Iowa 52242
(319) 335-5683

ABSTRACT

In the dynamics modeling of a flexible structure, finite element analysis employs reduction techniques, such as Guyan's reduction, to remove some of the "insignificant" physical coordinates, thus producing a dynamics model that has smaller mass and stiffness matrices. But this reduction is limited in the sense that it removes certain degrees of freedom at a node point, instead of node points themselves in the model. From the standpoint of linear control design, the resultant model is still too large despite the reduction. Thus, some form of model reduction is frequently used in the control design by approximating a large dynamical system with a fewer number of state variables. However, a problem arises from the placement of sensors and actuators in the reduced model, because a model usually undergoes, before being reduced, some form of coordinate transformations that do not preserve the physical meanings of the states. To correct such a problem, a method is developed that expresses a reduced model in terms of a subset of the original states.

The proposed method starts with a dynamic model that is originated and reduced in finite element analysis. Then the model is converted to the state space form, and reduced again by the internal balancing method. At this point, being in the balanced coordinate system, the states in the reduced model have no apparent resemblance to those of the original model. Through another coordinate transformation that is developed in this paper, however, this reduced model is expressed by a subset of the original states.

INTRODUCTION

In the dynamics modeling of a structure, finite element analysis employs reduction techniques, such as Guyan's reduction, to remove some of the "insignificant" physical coordinates [6, Guyan 1965; 10, Irons 1965], thereby producing a model that has smaller mass and stiffness matrices. But this reduction is limited in the sense that it reduces degrees of freedom at a node point, instead of the number of node points in the model. From the standpoint of linear control design, the resultant model is still too large despite the reduction, because the size of a model depends on degrees of freedom at each node and the number of node points.

In the control literature, there has been extensive research and publication on model reduction methods [5, Genesio and Milanese 1976; 7, Hickin and Sinha 1980], in which the primary objective is the approximation of a large dynamical system by fewer state variables with minimal change on the input-output characteristics. For example, the aggregation method [1, Aoki 1968] reduces a model by "aggregating" the original state vector into a lower dimensional vector, in which the concept of aggregation is a generalization of that of projection and related to that of state vector partitioning. Skelton and Hughes [15, 1980] derived modal cost analysis for linear matrix second order systems that are expressed in the state space form. The decomposition of quadratic cost index into the sum of contributions from each modal coordinate is used to rank the importance of the structure's modes. The internal balancing method [12, Moore 1981; 13, Pernebo and Silverman 1982; 14, Shokoohi, Silverman, and Van Dooren 1983; 4, Gawronski and Natke 1986; 3, Gawronski and Natke 1987] is based on "measures" of controllability and observability, which are defined by the controllability and observability grammians in certain subspaces of the original state space. Then the most controllable and observable part is used as a low-order approximation for the model. Hyland and Bernstein [8, 1985] have derived the first order conditions for quadratically optimal reduced order modeling of linear time invariant systems, in which they show how the complex optimality conditions in [16, Wilson 1970] can be transformed, without loss of generality, into much simpler and more tractable forms. The transformation is facilitated by exploiting the presence of an oblique (i.e., nonorthogonal) projection that was not recognized in [16, Wilson 1970] and that arises as a direct consequence of optimality.

From a close examination of the various reduction methods employed by the two distinctly different communities, it follows that a finite element dynamic model can be further reduced by the reduction methods used in the control community, provided that it is first converted into the state space form by assigning two states—displacement and velocity—to each degree of freedom at the node. However, a problem arises from subsequent structural control design, especially from the placement of sensors and actuators in the reduced model, because a model usually undergoes, before being reduced, some form of coordinate transformation through which a reduced model usually results in a subspace quite different from the original state space. Consequently, it is difficult, sometimes impossible, to recognize any connection between the states of the reduced model and those of the original model.

In the internal balancing method, we discovered that with an additional coordinate transformation it is possible to express the reduced model in terms of a subset of the original states. The method described in this paper proceeds with a finite element model of a structure that was already reduced by Guyan's reduction [6, Guyan 1965]. The model is then converted to the state space form, and is reduced again by the internal balancing method. At this point, being in the balanced coordinate system, the states of the reduced model have no apparent resemblance to those of the original model. But, through another coordinate transformation derived from the states that are deleted during reduction, this reduced model is expressed by a subset of the original states.

The procedure is illustrated through two examples. The first example is hypothetical, simple, yet quite effective for demonstration. The second example starts with a finite element model, and finally arrives at the reduced model that has a fewer number of node points. Throughout the two examples the impulse responses of several states are compared in the time domain.

MODEL REDUCTION BY THE INTERNAL BALANCING METHOD

The structural dynamic model in this paper is assumed to result from finite element analysis and have the following form:

$$\mathbf{M\ddot{q} + D\dot{q} + Kq = f} \tag{1}$$

where M, D and K are the nxn real, symmetric, positive definite matrices reflecting the mass, damping and stiffness properties. The nx1 vector **q** is the displacement vector; that is, each element describes the position of a node. The overdots denote differentiation with respect to time. The nx1 vector **f** represents the external forces applied to the structure. In addition, the system is assumed to be asymptotically stable (hence, the definiteness requirement on M, D and K).

Eq. (1) is first converted into the state space form such that

$$\mathbf{\dot{x} = Ax + Bu} \tag{2}$$

where the state vector **x** and the state matrix A are defined by

$$\mathbf{x} = \begin{bmatrix} \dot{q} \\ q \end{bmatrix} \quad A = \begin{bmatrix} -M^{-1}D & -M^{-1}K \\ I & 0 \end{bmatrix} \tag{3}$$

Here, I denotes the nxn identity matrix, 0 the nxn matrix of zeros, and M^{-1} the inverse of the nonsingular mass matrix M. The matrix B is called the input matrix and has a form determined by the location of the applied forces **f**. The vector **u**, often called the control force, has the form:

$$\mathbf{u} = \begin{bmatrix} \mathbf{f} \\ \mathbf{0} \end{bmatrix} \tag{4}$$

From the conversion, the dimension of **u** has now become 2nx1, and the dimension of A 2nx2n. If **u** is a scalar, i.e., if the input has the same time history at each node, then B becomes a 2nx1 vector that determines the location and the magnitude of an actuator. At this point it is necessary to indicate which states are to be measured or monitored by selecting an output matrix C such that

$$y(t) = Cx(t) \tag{5}$$

where y(t) is a vector consisting of those states that are to be measured. Since B and C are directly related to the locations of measurement and applied force, they influence the degree of controllability and observability of the system.

The system defined by A, the choice of outputs defined by C, and the location of applied forces defined by B, must all be such that the rank of

$$U_c = \begin{bmatrix} B & AB & \cdots & A^{2n-1}B \end{bmatrix} \tag{6}$$

and

$$U_o = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{2n-1} \end{bmatrix} \tag{7}$$

are 2n. That is, the system defined by (A,B,C) must be both controllable and observable. For most structural models in which each part is physically connected with another, the system is controllable and observable for any single applied force and any single state measurement (see, for instance, [9, Inman 1989]).

The concepts of controllability and observability are essential to the balanced model reduction. First, each state is examined on its degree of observability--the amount of contribution by each state to the measurement of the system response, and is also examined on its degree of controllability--the effect of applied force on the system response. The balanced reduction method then suggests that the states that do not affect the response significantly be removed from the model, producing the desired reduced order model. In this way, the method attempts to find a model of the smallest size that best captures the dynamics of the structure.

The controllability and observability grammians [12, Moore 1981; 13, Pernebo and Silverman 1982], which are varying under coordinate transformations, are used to define the "measures" of controllability and observability in a certain state space. Moore [12, 1981] has shown that there exists a coordinate system in which the two grammians are equal and diagonal. The corresponding system representation is called *balanced*. The numerical algorithms of how to obtain the transformation matrix are given both by [12, Moore 1981] and by [11, Laub 1980]. In the remainder of this section the internal balancing method is briefly summarized for completeness.

The *controllability grammian*, denoted by W_c, and the *observability grammian*, denoted by W_o, are defined as:

$$W_c = \int_0^\infty e^{At} BB^T e^{A^T t}\, dt \tag{8}$$

and

$$W_o = \int_0^\infty e^{A^T t} C^T C\, e^{At}\, dt \tag{9}$$

where e^{At} is the matrix exponential function defining the state transition matrix of the system. These grammians provide a measure of how controllable and how observable a structure is with the given input and output configuration. And their values are dependent on the coordinate system in which they are evaluated.

If we denote by P the transformation of the system into the balanced coordinate system, and if we denote by $W_o(P)$ and $W_c(P)$ the grammians defined in the balanced coordinate system, then the balanced system is defined by $\hat{A} = P^{-1}AP$, $\hat{B} = P^{-1}B$, $\hat{C} = CP$, and $\hat{\mathbf{x}} = P^{-1}\mathbf{x}$. In addition, the two grammians are equal:

$$W_c(P) = W_o(P) = \mathrm{diag}\{\sigma_1, \sigma_2 \ldots\ \sigma_{2n}\} \tag{10}$$

where the σ_i denotes the singular values of $W_c(P)$. By arranging the singular values in descending order and permuting the states correspondingly, the states in $\hat{\mathbf{x}}$ are arranged according to their level of controllability and observability; in other words, σ_1 being the largest, $\hat{x}_1$ is the most controllable and most observable state.

The method first partitions the state, input and measurement matrices on the basis of the magnitude of the singular values. For some index 2n-k, σ_{2n-k} would be much smaller than the preceding singular value σ_{2n-k-1}. Thus the vector $\hat{\mathbf{x}}$ can be partitioned as

$$\hat{\mathbf{x}} = \begin{bmatrix} \hat{\mathbf{x}}_r \\ \hat{\mathbf{x}}_d \end{bmatrix} \tag{11}$$

where $\hat{\mathbf{x}}_r$ contains (2n-k) states that are to be retained in the reduced model, and $\hat{\mathbf{x}}_d$ contains k states to be discarded in the model reduction because they correspond to small values of σ_i. These discarded coordinates are least controllable and observable; that is, they have least effect on the response of the system. Accordingly, the balanced system is partitioned as

$$\begin{bmatrix} \hat{\mathbf{x}}_r \\ \hat{\mathbf{x}}_d \end{bmatrix} = \begin{bmatrix} \hat{A}_r & \hat{A}_{12} \\ \hat{A}_{21} & \hat{A}_{22} \end{bmatrix} \begin{bmatrix} \hat{\mathbf{x}}_r \\ \hat{\mathbf{x}}_d \end{bmatrix} + \begin{bmatrix} \hat{B}_r \\ \hat{B}_d \end{bmatrix} \mathbf{u}$$

$$\mathbf{y} = \begin{bmatrix} \hat{C}_r & \hat{C}_d \end{bmatrix} \begin{bmatrix} \hat{\mathbf{x}}_r \\ \hat{\mathbf{x}}_d \end{bmatrix} \tag{14}$$

where $\hat{A}_r$ is a (2n-k)×(2n-k) matrix representing the reduced model in the balanced coordinate system. The reduced model $(\hat{A}_r, \hat{B}_r, \hat{C}_r, \hat{\mathbf{x}}_r)$ of order (2n-k) thus results from the balanced representation by deleting k number of the least controllable and observable states. In this way, the method produces the reduced model that contains the most significant dynamics of the structure with respect to the measurements and the applied forces, as defined by the matrices B and C.

The relative error in this type of model reduction has been defined by [12, Moore 1981]:

$$\text{Relative error} = \frac{\sqrt{\sum_{i=2n-k+1}^{2n} \sigma_i^2}}{\sqrt{\sum_{i=1}^{2n-k} \sigma_i^2}} \tag{12}$$

It provides a quantitative measure of error introduced by the reduced model in calculating the response of the system.

REDUCED MODEL IN PHYSICAL COORDINATES

A problem that has been rarely addressed in the model reduction is the physical interpretation of the reduced model in conjunction with the original model. Apparently the reduced state vector $\hat{\mathbf{x}}_r$ in the balanced representation bears no obvious connection with the physical position vector **q** of Eq. (1). In fact, it may, in theory, result in all the position states being deleted, leaving only velocity states. But for structural control and measurement applications, it is desirable to provide the designer with a clear, physical relationship between the original position vector **q** and the reduced state vector $\hat{\mathbf{x}}_r$.

Such a relationship is attained by using the fact that the balanced states are linear combinations of the original states. Symbolically this is written as:

$$\begin{aligned} \hat{x}_1 &= \sum_{j=1}^{2n} c_{1j}\, x_j \\ &\vdots \\ \hat{x}_{2n-k} &= \sum_{j=1}^{2n} c_{(2n-k)j}\, x_j \\ \hat{x}_{2n-(k-1)} &= \sum_{j=1}^{2n} c_{(2n-k+1)j}\, x_j \rightarrow 0 \\ &\vdots \\ \hat{x}_{2n} &= \sum_{j=1}^{2n} c_{2nj}\, x_j \rightarrow 0 \end{aligned} \tag{15}$$

where c_{ij}'s are the coefficients of linear combinations of $\{x_1, x_2, \ldots, x_{2n}\}$. Here the last k states are set to zero because they represent the least significant states in the balanced system [12, Moore 1981]. That is, the response with the given input and output configuration is least affected by these states. Setting each of these summations equal to zero is equivalent to imposing k constraints on the original 2n states, $\{x_1, x_2, \dots, x_{2n}\}$. Thus, the k states among the original 2n states can be removed, with model reduction error, by the k constraints resulting from the reduction. In other words, one can construct a reduced-order model by selecting (2n-k) states out of the original 2n states. If the (2n-k) selected states from the original system are denoted by $\mathbf{x}_r = [x_{j_1} \ x_{j_2} \ \cdots \ x_{j_{2n-k}}]^T$ and the (2n-k) states of the balanced system by $\hat{\mathbf{x}}_r = [\hat{x}_1 \ \hat{x}_2 \ \cdots \ \hat{x}_{2n-k}]^T$, then the states in $\hat{\mathbf{x}}_r$ are linear combinations of the states in $\mathbf{x}_r$. Thus there exists a new transformation matrix P_r of order $(2n\text{-}k)\times(2n\text{-}k)$ such that $\mathbf{x}_r = P_r \, \hat{\mathbf{x}}_r$.

The above constraints and the resulting transformation allow the designer or analyst to specify which nodes (i.e. which elements of **q**) of the model to be retained in the model reduction.

Now that it is shown that some members of the original states constitute the state vector $\mathbf{x}_r$ of the reduced model, the next question is how many states and which states to select from the original states. The answer to how many states, i.e., the order of the reduced model, depends on the designer's willingness to gain a smaller sized model at the expense of accuracy. The relative error in Eq. (12), defined by the singular values of Eq. (10), indicates a trade-off between error and model size. Once the order is determined, the next task is which states to select from the original states. There is no established methodology in dealing with this problem. However, strictly from the physical considerations of a given structure, the following two observations were made. First, if we recall that a pair of states--displacement and velocity--were assigned to each degree of freedom at the node when the dynamic equation (1) was converted into the state equation (2), then selecting a certain degree of freedom at a certain node is equivalent to selecting the paired states associated with that particular degree of freedom. Therefore, the paired velocity and displacement states must be either selected or deleted together, because they signify one degree of freedom at a node in the actual structure. Another observation is that for the nodes to which actuators and/or sensors are attached, the paired states representing the degree of freedom to whose direction the actuators and/or sensors function must be selected to ensure that the reduced model is under the same input and output condition as the original physical model.

In the following it is shown that the matrix P_r consists of certain rows and columns of the original transformation matrix P, and that there is a systematic way of constructing P_r from P. First, by writing the coordinate transformation, $\mathbf{x} = P \, \hat{\mathbf{x}}$, in matrix elements

$$x_i = \sum_{j=1}^{2n} p_{ij} \, \hat{x}_j \tag{16}$$

next, by the model reduction,

$$\hat{x}_{2n-(k-1)} \rightarrow 0, \dots, \hat{x}_{2n} \rightarrow 0 \tag{17}$$

the original states are expressed as linear combinations of the first (2n-k) balanced states $\{\hat{x}_1, \hat{x}_2, \dots, \hat{x}_{2n-k}\}$. The last k columns of P thereby can be removed from the expression:

$$\begin{bmatrix} x_1 \\ \vdots \\ x_{2n} \end{bmatrix} = \begin{bmatrix} p_{11} & \cdots & p_{1(2n-k)} \\ \vdots & & \vdots \\ p_{2n1} & \cdots & p_{2n(2n-k)} \end{bmatrix} \begin{bmatrix} \hat{x}_1 \\ \vdots \\ \hat{x}_{2n-k} \end{bmatrix} \tag{18}$$

Then by selecting $\{j_1, \ldots, j_{2n-k}\}$ rows that correspond to the rows of the selected original states, the new transformation is established between $\mathbf{x}_r$ and $\hat{\mathbf{x}}_r$

$$\underset{\mathbf{x}_r}{\begin{bmatrix} x_{j_1} \\ \vdots \\ x_{j_{2n-k}} \end{bmatrix}} \underset{=}{=} \underset{P_r}{\begin{bmatrix} p_{j_1 1} & \cdots & p_{j_1 2n-k} \\ \vdots & & \vdots \\ p_{j_{2n-k} 1} & \cdots & p_{j_{2n-k} 2n-k} \end{bmatrix}} \underset{\hat{\mathbf{x}}_r}{\begin{bmatrix} \hat{x}_1 \\ \vdots \\ \hat{x}_{2n-k} \end{bmatrix}} \qquad (19)$$

where p_{ij}'s are the elements of the original transformation matrix P. Finally, the reduced order system (A_r, B_r, C_r, x_r)

$$\dot{\mathbf{x}}_r(t) = A_r\mathbf{x}_r(t) + B_r\mathbf{u}(t)$$

$$y_r(t) = C_r\mathbf{x}_r(t) \qquad (20)$$

is expressed in terms of a subset $\mathbf{x}_r$ of the original state vector $\mathbf{x}$ where $A_r = P_r\,\hat{A}_r\,P_r^{-1}$, $B_r = P_r\,\hat{B}_r$, and $C_r = P_r^{-1}\,\hat{C}_r$.

In summary, the model reduction procedure described in this paper can be illustrated as follows:

$$\begin{tikzcd}
(A, B, C, \mathbf{x}) \arrow[r, Leftrightarrow, "P"] & (\hat{A}, \hat{B}, \hat{C}, \hat{\mathbf{x}}) \arrow[d, Rightarrow, "\text{model reduction}"] & \text{...order } 2n \\
(A_r, B_r, C_r, \mathbf{x}_r) \arrow[r, Leftrightarrow, "P_r"] & (\hat{A}_r, \hat{B}_r, \hat{C}_r, \hat{\mathbf{x}}_r) & \text{...order } 2n-k \\
\text{Original State Space} & \text{Balanced System} &
\end{tikzcd} \qquad (21)$$

where $\mathbf{x}_r$ consists of (2n-k) elements of $\mathbf{x}$, and $\hat{\mathbf{x}}_r$ consists of (2n-k) elements of $\hat{\mathbf{x}}$. In addition, the system $(\hat{A}_r, \hat{B}_r, \hat{C}_r, \hat{\mathbf{x}}_r)$ is the balanced representation of the system (A_r, B_r, C_r, x_r).

The following examples illustrate the proposed model reduction method.

EXAMPLE (1)

The procedure discussed in this paper is demonstrated through the example used by [12, Moore 1981]. The system (A, B, C, x) is given

$$A = \begin{bmatrix} 0 & 0 & 0 & -150 \\ 1 & 0 & 0 & -245 \\ 0 & 1 & 0 & -113 \\ 0 & 0 & 1 & -19 \end{bmatrix} \quad B = \begin{bmatrix} 4 \\ 1 \\ 0 \\ 0 \end{bmatrix} \quad C = [0 \;\; 0 \;\; 0 \;\; 1]$$

with an impulse input $u = \delta(t)$ of different magnitudes applied at the states x_1 and x_2. Then the transformation matrix P is calculated to be

$$P=\begin{bmatrix}29.090 & -4.056 & 0.553 & -0.310\\ 14.784 & 5.449 & -0.557 & 0.426\\ 2.323 & 2.093 & -0.030 & -0.122\\ 0.118 & 0.131 & 0.056 & 0.007\end{bmatrix}$$

Here let us suppose that we decide to delete x_3, so that the reduced model contains the three original states, $\{x_1, x_2, x_4\}$. The transformation P_r is readily obtained by selecting the 1st, 2nd, and 4th rows and removing the 4th column of P,

$$P_r=\begin{bmatrix}29.090 & -4.056 & 0.553\\ 14.784 & 5.449 & -0.557\\ 0.118 & 0.131 & 0.056\end{bmatrix}$$

The system matrices of the reduced model are

$$A_r = P_r\,\hat{A}_r\,P_r^{-1}=\begin{bmatrix}0.090 & -0.290 & -135.898\\ 0.876 & 0.398 & -264.391\\ -0.069 & 0.274 & -16.537\end{bmatrix}$$

$$B_r = P_r\,\hat{B}_r=\begin{bmatrix}3.998\\ 1.003\\ 0.\end{bmatrix}$$

$$C_r = P_r^{-1}\,\hat{C}_r=[0\ \ 0\ \ 1]$$

The reduced model in the physical coordinate is thus

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_4\end{bmatrix}=A_r\begin{bmatrix}x_1\\ x_2\\ x_4\end{bmatrix}+B_r u \qquad y_r=C_r\begin{bmatrix}x_1\\ x_2\\ x_4\end{bmatrix}$$

By setting u = δ(t), the impulse response of each original state is plotted in comparison with the difference between the two impulse responses of the state, one by the full order model and the other by the reduced order model, as shown in Figures 1-3. The difference is obtained by subtracting the response of the state in the reduced model from that of the same state in the original system.

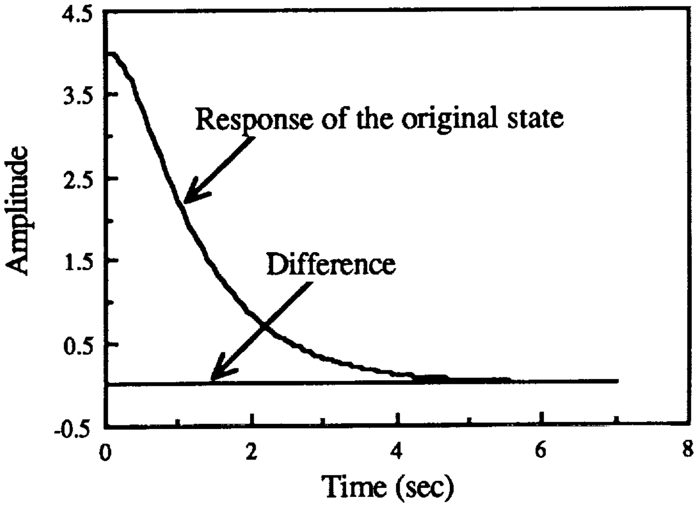


Figure 1 State #1 in Example (1)

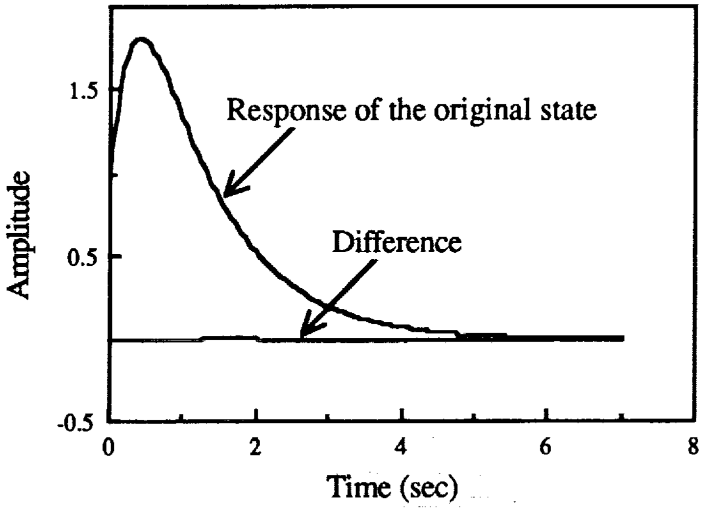


Figure 2 State #2 in Example (1)

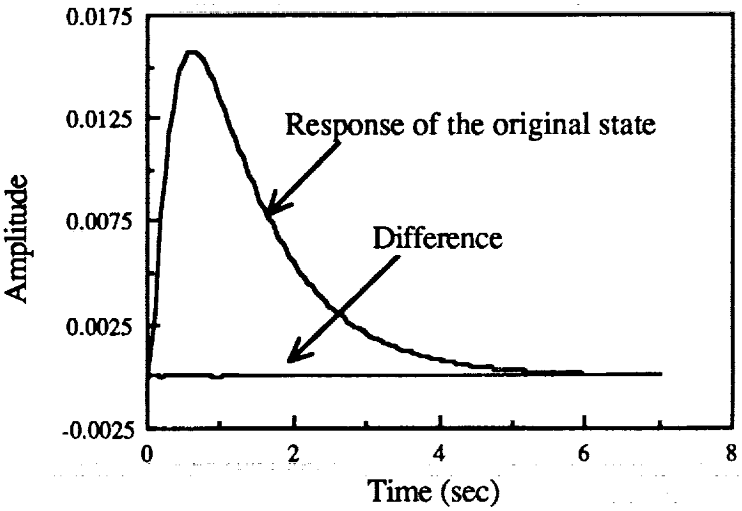


Figure 3 State #4 in Example (1)

In Figures 1-3, the difference is nearly zero, thus indicating that the reduced third-order model is indeed a respectable realization of the original fourth order system.

EXAMPLE (2)

The same procedure is applied to the following finite element model of a cantilever beam. Here the design nature of the proposed method is illustrated by assuming that actuators, machines, or sensors, will be placed at nodes 1, 4, and 5 so that they become important node points to be retained in the final model

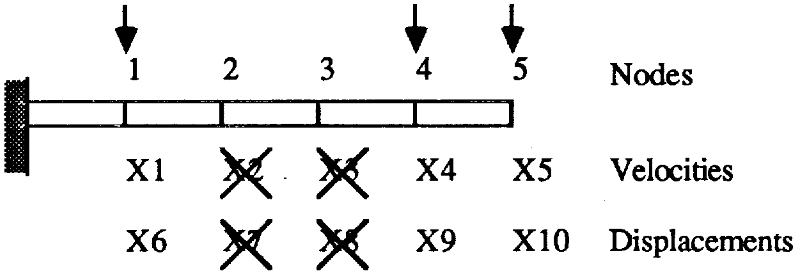


Figure 4 A cantilever beam with 5 nodes

The impulse input is applied through nodes 1, 4, and 5 in this example. Suppose that we decide to delete the states x_2, x_3, x_7, and x_8, so that the reduced model can be expressed by the remaining six states. The diagonal mass matrix is obtained by lumping mass at the node points, and the stiffness matrix by finite element analysis. The damping matrix is made up with a damping coefficient 0.002 for each mode. In Figures 4-7 the responses of the original states at Node 1 and 5 are plotted together with their differences between the responses of the states in the original system and those of the states in the reduced system. The differences are obtained in the same way as in **Example (1)**.

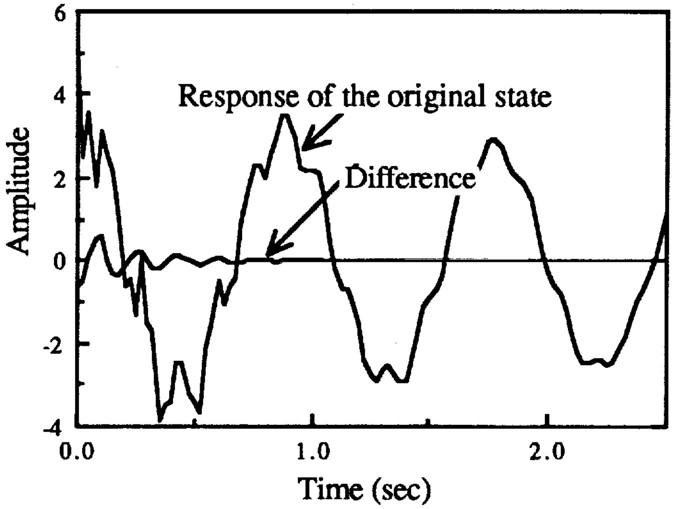


Figure 5 State #5 in Example (2): Velocity at the tip

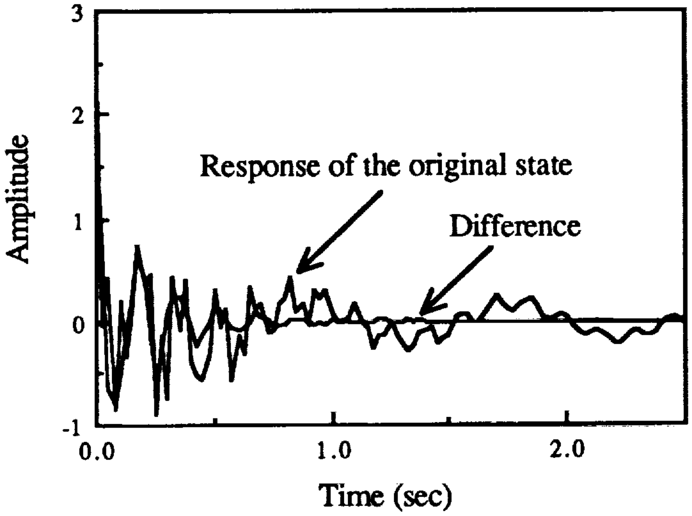


Figure 6 State #1 in Example (2): Velocity at Node #1

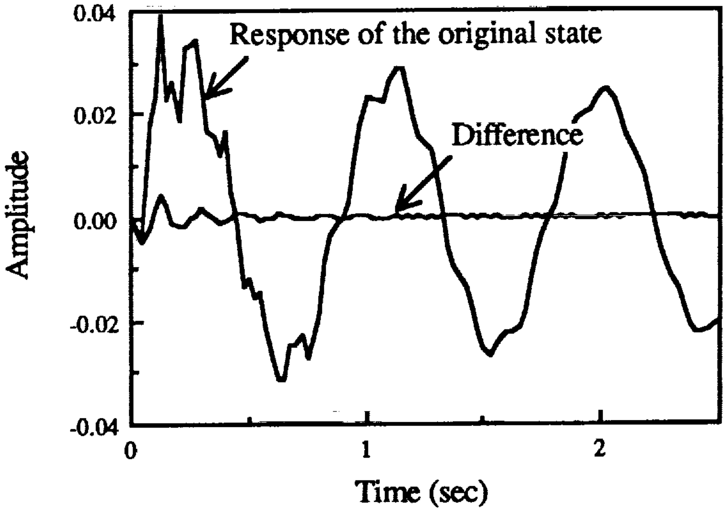


Figure 7 State #6 in Example (2): Displacement at Node #1

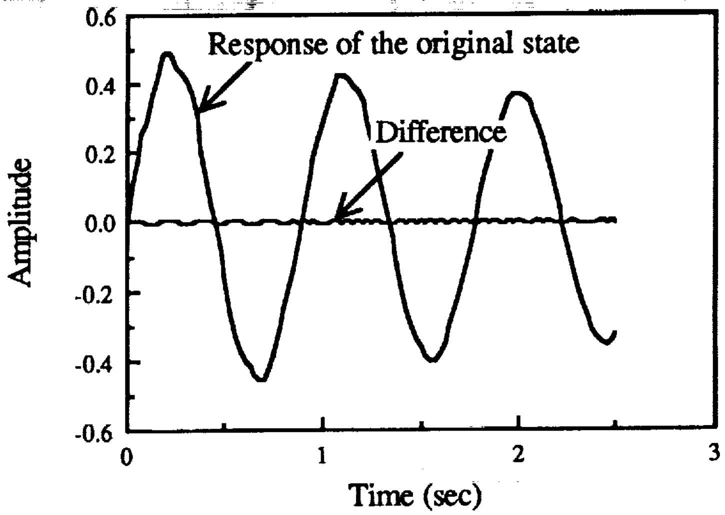


Figure 8 State #10 in Example (2): Displacement at the tip

In Figures 4-7, the difference is almost zero in comparison to the response of the original state. Some nonzero differences are detected in the transient region of the response, which indicates that the reduced model is closer to the full-order model in the steady state response region.

CONCLUSION

A model reduction method that is based on the concept of the internal balancing method is implemented along with another transformation derived from the states that are deleted during the reduction in order that the reduced model may represent the original physical model with fewer states than the original model requires.

The proposed method in this paper takes a finite element model that is reduced by Guyan's reduction, converts it into the state space form, and applies the balanced model reduction. And, through another transformation that is derived from the deleted states in reduction, the model is finally expressed by a subset of the original states. The method thereby provides a clear, physical relationship between the states in the reduced model and those in the original model. The states in the reduced model are selected directly from the original states, thus retaining the same physical meanings as in the original model. This appears to be a new and significant development in the area of model reduction. This method yields not only reduced order state space representations, but also, at the same time, reduced order transfer functions.

The application of this reduction method to a large finite element model generates a reduced model with a fewer number of nodal points, so that the analytical model improvement can be performed on the reduced model instead of a full-scale finite element model, which has been a common practice (for example, [2, Berman and Nagy 1983]). The final reduced model is of a more attractive size for dynamic simulations and subsequent structural control design.

REFERENCES

1. M. Aoki 1968 *IEEE Trans. Automat. Contr.* **AC-13**,.246-253. Control of large-scale dynamics systems by aggregation.
2. A. Berman and E.J. Nagy 1983 *AIAA Journal* **21**, 1168-1173. Improvement of a large analytical model using test data.
3. W. Garwronski and H.G. Natke 1987 *Int. J. Systems Sci.* **18**, 237-249. Balancing linear systems.
4. W. Garwronski and H.G. Natke 1986 *Int. J. Systems Sci.* **17**, 1509-1519. On balancing linear symmetric systems.
5. R. Genesio and M. Milanese 1976 *IEEE Trans. Automat. Contr.* **AC-21**, 118-122. A note on the derivation and use of reduced-order models.
6. R.J. Guyan 1965 *AIAA Journal* **3**, 380. Reduction of stiffness and mass matrices.
7. J. Hickin and N. Sinha 1980 *IEEE Trans. Automat. Contr.* **AC-25**, 1121-1127. Model reduction for linear multivariable systems.
8. D.C. Hyland and D.S. Bernstein 1985 *IEEE Trans. Automat. Contr.* **AC-30**, 1201-1211. The optimal projection equations for model reduction and the relationships among the methods of Wilson, Skelton, and Moore.
9. D.J. Inman 1989 *Vibrations with Control, Measurement and Stability*, Prentice Hall.
10. B. Irons 1965 *AIAA Journal* **3**, 961-962. Structural eigenvalue problem: elimination of unwanted variables.
11. A.J. Laub 1980 *Proc. 1980 JACC* San Francisco, CA, session FA8-E. Computation of balancing transformation
12. B.C. Moore 1981 *IEEE Trans. Automat. Contr.* **AC-26**, 17-32. Principal component analysis for linear systems: controllability, observability, and model reduction.

13. L. Pernebo and L.M. Silverman 1982 *IEEE Trans. Automat. Contr.* **AC-27**, 382-397. Model reduction via balanced state space representation.
14. S. Shokoohi, L.M. Silverman and P.M. Van Dooren 1983 *IEEE Trans. Automat. Contr.* **AC-28**, 810-822. Linear time-variable systems: balancing and model reduction.
15. R.E. Skelton and P.C. Hughes 1980 *ASME Trans. J. Dynamic Sys. Meas. Contr.* **102**, 151-158. Modal cost analysis for linear matrix-second-order systems.
16. D.A. Wilson 1970 *Proc. IEE* **117**, 1161-1165. Optimum solution of model-reduction problem.

MODAL IDENTITIES FOR MULTIBODY ELASTIC SPACECRAFT -- AN AID TO SELECTING MODES FOR SIMULATION

Hari B. Hablani†
Rockwell International, Seal Beach, California

ABSTRACT

This paper answers the question: Which set of modes furnishes a higher fidelity math model of dynamics of a multibody, deformable spacecraft--hinges-free or hinges-locked vehicle modes? Two sets of general, discretized, linear equations of motion of a spacecraft with an arbitrary number of deformable appendages, each articulated directly to the core body, are obtained using the above two families of modes. By a comparison of these equations, ten sets of modal identities are constructed which involve modal momenta coefficients and frequencies associated with both classes of modes. The sums of infinite series that appear in the identities are obtained in terms of mass, and first and second moments of inertia of the appendages, core body, and vehicle by using certain basic identities concerning appendage modes. Applying the above identities to a four-body spacecraft, the hinges-locked vehicle modes are found to yield a higher fidelity model than hinges-free modes, because the latter modes have nonconverging modal coefficients--a characteristic proved and illustrated in the paper.

I. INTRODUCTION

The use of appendage modes for simulating dynamics and control of multibody flexible spacecraft is widespread, in as much as they are eminently suitable for both small angle (linear) and large angle (nonlinear) dynamics. To win this benefit, however, a simulation engineer must retain a sufficient number of these modes for each appendage so that the simulation program has acceptable fidelity. When there are a large number of appendages in a spacecraft, and/or an appendage has a large mass and moment of inertia relative to those of the rigid core body of the spacecraft, the total number of appendage modes for a high accuracy model may become unacceptably great (Reference 1), possibly diminishing the utility of the appendage modes for simulation. Furthermore, control systems for a multibody spacecraft are most easily designed by considering one axis of one body at a time, because different bodies generally serve different purposes and so the control systems' intrinsic features are generally quite different. Having designed them so, to ensure they all perform as desired in the mutual presence and in the presence of flexibility, a compact mathematical model of the entire spacecraft's dynamics is desired so that the control designs can be refined fast and economically about all axes. For this purpose, the linear, small angle models of spacecraft flexible dynamics are just right, and so the engineer could beneficially employ the vehicle modes of the spacecraft. Hughes[2] conceived of two families of vehicle modes for multibody spacecraft: "hinges-free" and "hinges-locked" vehicle modes (although he does not use this terminology). By definition, hinges-free modes are obtained by leaving all hinges free, that is, unlocked and unforced, so that the associated natural vehicle modes may contain motion of the articulated bodies relative to the inboard bodies. Conversely, in the hinges-locked modes, the relative motion of the articulated bodies is, by definition, zero, and some force or torque is applied at the hinges to keep the motion so. In Reference 3, these vehicle modes are formulated, and their zero linear and angular momentum properties, the orthogonality conditions, and the associated modal momenta coefficients are theorized.

A critical question whose answer is sought in this paper is: Between the hinges-free and hinges-locked vehicle modes, which one furnishes a higher fidelity dynamic model, retaining the same number of modes in the simulation? To this end, a multibody spacecraft is considered in this paper that consists of a rigid core body, and N flexible appendages, each articulated directly to the core body. Three sets of discrete motion equations of this spacecraft are obtained from a continuum set by using appendage modes, hinges-free vehicle modes, and hinges-locked vehicle modes. To compare the last two families of modes, modal identities are devised that express the sum of contribution of all infinite number of modes in terms of first and second moments of inertia of the articulated bodies, the core body, and the vehicle, following Hughes[4]. The analysis is amply illustrated, and definitive conclusions are summarized at the end of the paper. Although for concreteness, the paper considers a multibody spacecraft with level-1 articulated bodies (the terminology of Ho[5]), it will be clear that the conclusions drawn apply to a wider range of multibody spacecraft.

II. FORMULATION OF CONTINUUM EQUATIONS OF MOTION

Fig. 1 portrays an N+1-body spacecraft that consists of a three-axis stabilized core rigid body B_0, and deformable bodies $E_1,\ldots,E_N$, each articulated directly to the core body. The motion equations will be developed with respect to the

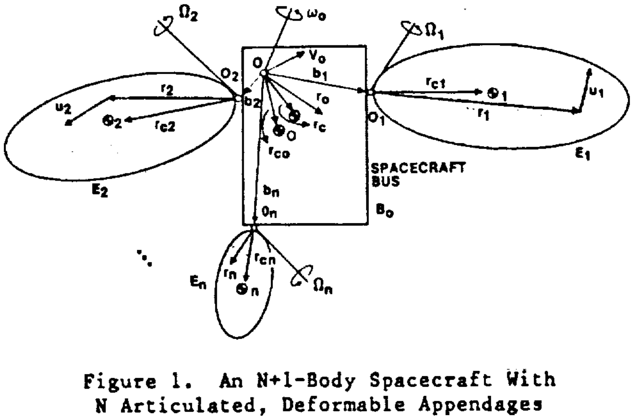


Figure 1. An N+1-Body Spacecraft With N Articulated, Deformable Appendages

†Engineering Specialist, Guidance and Control Group, AIAA Senior Member.

reference point 0 in Fig. 1 which is neither the mass center $\oplus_0$ of the body B_0, nor the mass center $\oplus$ of the entire vehicle V. This generality in the formulation is warranted because the NASTRAN modal data corresponding to such multibody spacecraft are often with respect to an arbitrary reference node 0, and the mass centers are generally nodeless empty points. The mass of each body is denoted m_p $(p=0,1,\ldots,N)$; the mass of all N articulated bodies together, m_e; and the mass of entire spacecraft, m; clearly, $m = m_0 + m_e$. The first moment of mass of B_0 relative to 0 is $\underline{c}_0$, and those for the hinged bodies $(j=1,\ldots,N)$, measured from the respective hinges O_j, are denoted $\underline{c}_j$. Similar to $\underline{c}_p$ $(p=0,1,\ldots,N)$, the vector $\underline{r}_0$ emanates from 0 and $\underline{r}_j$ from O_j. Note that the subscript p covers all bodies, while j covers only the articulated bodies. The vectors $\underline{b}_j$ $(j=1,\ldots,N)$ originating from 0 locate the hinges O_j of the hinged bodies E_j. The first moment of inertia of the entire spacecraft, then, is

$$\underline{c} = \underline{c}_0 + \sum_j \left(m_j \underline{b}_j + \underline{C}_{0j}\underline{c}_j\right) \triangleq \underline{c}_0 + \underline{c}_e;\ \sum_j = \sum_{j=1}^{N} \qquad (1)$$

where the matrix $\underline{C}_{0j}$ transforms the E_j-fixed vector $\underline{c}_j$ to a B_0-fixed vector, and the vectors $\underline{b}_j$ are expressed in the B_0-fixed frame. Next, $\underline{J}_0$ denotes the inertia matrix of the body B_0 about the reference point 0, while $\underline{J}_j$ is the inertia matrix of the hinged body E_j in its own frame about the hinge O_j. The inertia matrix of E_j expressed at the reference point 0 in the B_0-fixed frame is denoted $\underline{J}_j^0$ and

$$\underline{J}_j^0 = \underline{C}_{0j}\,\underline{J}_j\,\underline{C}_{j0} - \left[m_j \underline{b}_j^x\,\underline{b}_j^x + \underline{b}_j^x\,\left(\underline{C}_{0j}\ \underline{c}_j\right)^x + \left(\underline{C}_{0j}\,\underline{c}_j\right)^x \underline{b}_j^x\right] \qquad (2)$$

where $(\bullet)^x$ means the 3x3 skew-symmetrix matrix associated with the vector $(\bullet)$. The inertia matrix $\underline{J}$ of the entire vehicle at the point 0 will then be

$$\underline{J} = \underline{J}_0 + \sum_j \underline{J}_j^0 \triangleq \underline{J}_0 + \underline{J}_e^0 \qquad (3)$$

Anticipating our later needs, the cross inertia matrix $\underline{J}_{0j}$ between the bodies B_0 and E_j expressed in the E_j-fixed frame equals

$$\underline{J}_{0j} \triangleq \underline{J}_j - \left(\underline{C}_{j0}\underline{b}_j\right)^x \underline{c}_j^{\,x} \qquad (4)$$

As for the motion of the spacecraft, its mass center is assumed to perform some orbital motion, not coupled with its attitude motion under consideration. To develop motion equations, the local orbital frame is taken to be an inertial frame. The kinetic quantities of interest are: $\underline{V}_0(t)$, the perturbational velocity of the reference point 0 over the uniform orbital motion at time t; $\omega_0(t)$, the inertial angular velocity of B_0; $\underline{\Omega}_j(t)$, the angular velocity of each articulated body E_j relative to B_0 at the hinge O_j; and $\underline{u}_j(\underline{r}_j,t)$, the deformation of E_j at the location $\underline{r}_j \epsilon E_j$. These quantities are taken to be linear, first order, infinitesimal, so that their products can be ignored in the analysis. The external forces and torques acting on the spacecraft are the force $\underline{f}_0(t)$ and the torque $\underline{g}_0(t)$ acting on B_0 at 0, and the force $\underline{f}_j(t)$ and the torque $\underline{g}_j(t)$ on each E_j at the hinge O_j. The latter pair, $(\underline{f}_j,\underline{g}_j)$, includes a distributed force $\underline{\ell}_j(\underline{r}_j,t)$acting in the domain of the body E_j, and if $\underline{\ell}_j(\underline{r}_j,t)$is the only surface force acting on E_j, then

$$\underline{f}_j(t) = \int_j \underline{\ell}_j(\underline{r}_j,t)\ dA$$
$$\qquad (5)$$
$$\underline{g}_j(t) = \int_j \underline{r}_j^x \underline{\ell}_j(\underline{r}_j,t)\ dA$$

where dA is an elemental area of E_j and $\int_j \triangleq \int_{E_j}$. With the aid of the Dirac delta function and its derivative, the distributed force $\underline{\ell}_j(\underline{r}_j,t)$ also represents a distributed moment. Regarding the control forces and torques, those acting on B_0 are included in the quantities $\underline{f}_0$ and $\underline{g}_0$, whereas, if a control force or torque is produced in the interior domain of E_j without acting against the core body B_0, then that is included in the pair $(\underline{f}_j,\underline{g}_j)$; however, if, for instance, the torque is produced by an electric motor which rests on B_0 at the interface O_j and exerts on E_j, then this is considered separately and denoted $\underline{g}_{0j}(t)$ $(j=1,\ldots,N)$, for it produces a reaction torque $-\underline{g}_{0j}(t)$ which acts on B_0. The total force $\underline{f}(t)$ and torque $\underline{g}(t)$ that act on the vehicle are

$$\underline{f} = \underline{f}_0 + \sum_j \underline{C}_{0j}\underline{f}_j,\ \underline{g} = \underline{g}_0 + \sum_j \left(\underline{C}_{0j}\underline{g}_j + \underline{b}_j^x \underline{C}_{0j}\underline{f}_j\right) \qquad (6)$$

where, of course, $\underline{g}(t)$ does not include the control torque $\underline{g}_{0j}(t)$ at the interface O_j.

The elastic spacecraft under consideration is relatively simple; it is straightforward to develop its linear, continuum motion equations following Hughes[4,6,7]. The equations governing the discrete variables $\underline{V}_0, \underline{\omega}_0, \underline{\Omega}_j$ $(j=1,\ldots,N)$ are:

$$m\dot{\underline{V}}_0 - \underline{c}^x\dot{\underline{\omega}}_0 - \sum_j \underline{C}_{0j}\underline{c}_j^x\dot{\underline{\Omega}}_j + \sum_j \int_j \underline{C}_{0j}\ddot{\underline{u}}_j\ dm = \underline{f}$$

$$\underline{c}^x\dot{\underline{V}}_0 + \underline{J}\,\dot{\underline{\omega}}_0 + \sum_j \underline{C}_{0j}\underline{J}_{0j}\dot{\underline{\Omega}}_j + \sum_j \int_j \left(\underline{b}_j^x\underline{C}_{0j} + \underline{C}_{0j}\underline{r}_j^x\right)\ddot{\underline{u}}_j\ dm = \underline{g} \qquad (7)$$

$$\underline{c}_j^x\underline{C}_{j0}\dot{\underline{V}}_0 + \left(\underline{C}_{0j}\underline{J}_{0j}\right)^T\dot{\underline{\omega}}_0 + \underline{J}_j\dot{\underline{\Omega}}_j + \int_j \underline{r}_j^x\ddot{\underline{u}}_j\ dm = \underline{g}_{0j} + \underline{g}_j\ \ (j = 1,\ldots,N)$$

where an overdot indicates differentiation with respect to time. To write the motion equation governing the deformation $\underline{u}_j(\underline{r}_j,t)$ of the flexible body E_j, denote the related linear stiffness operator by $\underline{L}_j$; the body E_j is allowed to be anisotropic and/or nonhomogeneous, and its mass density is denoted $\sigma_j(\underline{r}_j)$. The continuum motion equation governing the deformation $\underline{u}_j$ is then:

$$\underline{L}_j\underline{u}_j + \sigma_j \{\underline{C}_{j0}\dot{\underline{V}}_0 - (\underline{C}_{j0}\underline{b}_j + \underline{r}_j)^{x}\, \underline{C}_{j0}\dot{\underline{\omega}}_0 - \underline{r}_j^{x}\dot{\underline{\Omega}}_j + \ddot{\underline{u}}_j\} = \underline{f}_j(\underline{r}_j,t) \quad (j=1,\ldots,N) \qquad (8)$$

The continuum motion equations (7) and (8) are discretized in the next section.

III. DISCRETIZATION OF CONTINUUM EQUATIONS OF MOTION

Three families of modes will be employed in this section for discretization: (1) appendage modes, (2) hinges-free vehicle modes, and (3) hinges-locked vehicle modes. The use of appendage modes is standard; they are employed here in order to evaluate the infinite sums that appear in the hinges-free and hinges-locked modal identities in Section IV in terms of mass, and first and second moments of inertia of the appendages, core body, and the vehicle.

DISCRETIZATION BY APPENDAGE MODES

Following Hughes[4], define the modal momenta coefficients $\underline{P}_{j\sigma}$ and $\underline{H}_{j\sigma}$ concerning the appendage (cantilever) modes $\underline{U}^a_{j\sigma}(\underline{r}_j)$ of the articulated body E_j:

$$\underline{P}_{j\sigma} \triangleq \int_j \underline{U}^a_{j\sigma}(\underline{r}_j)\, dm \quad (j=1,\ldots,N;\sigma=1,\ldots,\infty)$$

$$\underline{H}_{j\sigma} \triangleq \int_j \underline{r}_j^{x}\underline{U}^a_{j\sigma}(\underline{r}_j)\, dm \qquad (9)$$

where dm = elemental mass. The coefficient $\underline{P}_{j\sigma}$ is associated with linear momentum and $\underline{H}_{j\sigma}$ with angular momentum of the mode σ at the hinge point O_j. The modal angular momentum coefficient relative to the reference point O (Fig. 1) is defined as

$$\underline{H}^0_{j\sigma} \triangleq \underline{b}_j^{x}\, \underline{C}_{0j}\underline{P}_{j\sigma} + \underline{C}_{0j}\underline{H}_{j\sigma} \qquad (10)$$

Then the continuum equations (7) and (8) discretize to

$$m\,\dot{\underline{V}}_0 - \underline{c}^{x}\dot{\underline{\omega}}_0 - \sum_j \underline{C}_{0j}\underline{c}_j^{x}\dot{\underline{\Omega}}_j + \sum_j \underline{C}_{0j}\sum_\sigma \underline{P}_{j\sigma}\ddot{Q}_{j\sigma} = \underline{f}$$

$$\underline{c}^{x}\dot{\underline{V}}_0 + \underline{J}\,\dot{\underline{\omega}}_0 + \sum_j \underline{C}_{0j}\underline{J}_{0j}\dot{\underline{\Omega}}_j + \sum_j\sum_\sigma \underline{H}^0_{j\sigma}\ddot{Q}_{j\sigma} = \underline{g}$$

$$\underline{c}_j^{x}\underline{C}_{j0}\dot{\underline{V}}_0 + \underline{J}_{0j}^{T}\underline{C}_{j0}\dot{\underline{\omega}}_0 + \underline{J}_j\dot{\underline{\Omega}}_j + \sum_\sigma \underline{H}_{j\sigma}\ddot{Q}_{j\sigma} = \underline{g}_j + \underline{g}_{0j}$$

$$\underline{P}_{j\sigma}^{T}\underline{C}_{j0}\dot{\underline{V}}_0 + \underline{H}^{0^T}_{j\sigma}\dot{\underline{\omega}}_0 + \underline{H}_{j\sigma}^{T}\dot{\underline{\Omega}}_j + \ddot{Q}_{j\sigma} + \Omega^{c^2}_{j\sigma}Q_{j\sigma} = \Upsilon^a_{j\sigma}$$

$$(j=1,\ldots,N;\ \sigma=1,\ldots,\infty) \qquad (11)$$

where the superscript T indicates transpose of the quantity; $Q_{j\sigma}(t)$ is the modal coordinate and $\Omega^c_{j\sigma}$ is the frequency associated with the σ-th appendage mode $\underline{U}^a_{j\sigma}(\underline{r}_j)$ of the body E_j; and $\Upsilon^a_{j\sigma}(t)$ is the modal input to that mode:

$$\Upsilon^a_{j\sigma}(t) = \int_j \underline{U}^{a^T}_{j\sigma}(\underline{r}_j)\underline{f}_j(\underline{r}_j,t)\, dA \qquad (12)$$

Eqs. (11) are a generalization of Eq. (35) of Hughes[4], for the former include articulation motion of the appendages. Much more complex and general equations than Eqs. (11) are available in the vast literature on deformable multibody dynamics; see, for instance, the works of Ho[5], and Singh et al[8] on spacecraft with arbitrary tree topology. Eqs. (11) nevertheless may boast of simplicity which is eminently useful while designing the control systems for articulated bodies. More importantly though, Eqs. (11) are derived here because in Section IV they will aid in developing modal identities. To facilitate this task, Eqs. (11) are abbreviated by using the definitions

$$\underline{C}^{0^T}_A \triangleq -\,[\underline{C}_{01}\underline{c}_1^{x},\ \ldots,\ \underline{C}_{0N}\underline{c}_N^{x}],$$

$$\underline{J}^{T}_{0A} \triangleq [\underline{C}_{01}\underline{J}_{01},\ \ldots,\ \underline{C}_{0N}\underline{J}_{0N}]$$

$$\underline{P}^{T}_j \triangleq [\underline{P}_{j1}\quad \underline{P}_{j2}\ \cdots],$$

$$\underline{P}^{0^T}_A \triangleq [\underline{C}_{01}\underline{P}_1^{T},\ \ldots,\ \underline{C}_{0N}\underline{P}_N^{T}]$$

$$\underline{H}^{0^T}_j \triangleq [\underline{H}^0_{j1}\quad \underline{H}^0_{j2}\ \cdots],$$

$$\underline{H}_A^{\;0^T} \triangleq [\underline{H}^{0^T}_1\ ,\ \ldots,\ \underline{H}^{0^T}_N\ ]$$

$$\underline{H}^{T}_j \triangleq [\underline{H}_{j1}\quad \underline{H}_{j2}\ \cdots],$$

$$\underline{H}_A \triangleq \mathrm{diag}\ [\underline{H}_1\ \cdots\ \underline{H}_N],\quad \underline{J}_A \triangleq \mathrm{diag}\ [\underline{J}_1\ \cdots\ \underline{J}_N]$$

$$\underline{Q}^{T}_j \triangleq [Q_{j1}\quad Q_{j2}\ \cdots],\qquad \underline{Q}^{T}_A \triangleq [\underline{Q}^{T}_1\ \cdots\ \underline{Q}^{T}_N],$$

$$\underline{g}^{T}_A \triangleq [\underline{g}^{T}_1\ \cdots\ \underline{g}^{T}_N],$$

$$\underline{g}^{T}_{0A} \triangleq [\underline{g}^{T}_{01}\ \cdots\ \underline{g}^{T}_{0N}],\qquad \underline{\Omega}^{T}_A \triangleq [\underline{\Omega}^{T}_1\ \cdots\ \underline{\Omega}^{T}_N]$$

$$\underline{\Omega}^{c}_j \triangleq \mathrm{diag}\ [\Omega^{c}_{j1}\ \Omega^{c}_{j2}\cdots],\qquad \underline{\Omega}_c \triangleq \mathrm{diag}\ [\underline{\Omega}^{c}_1\cdots\underline{\Omega}^{c}_N]$$

$$\underline{\Upsilon}^{a^T}_j \triangleq [\Upsilon^a_{j1}\quad \Upsilon^a_{j2}\ \cdots],\qquad \underline{\Upsilon}^{a^T}_A \triangleq = [\underline{\Upsilon}^{a^T}_1\ \cdots\ \underline{\Upsilon}^{a^T}_N] \qquad (13)$$

Eqs. (11) then take this concise form:

$$m\,\dot{\underline{V}}_0 - \underline{c}^{x}\dot{\underline{\omega}}_0 + \underline{C}^{0^T}_A\dot{\underline{\Omega}}_A + \underline{P}^{0^T}_A\ddot{\underline{Q}}_A = \underline{f}$$

$$\underline{c}^{x}\dot{\underline{V}}_0 + \underline{J}\,\dot{\underline{\omega}}_0 + \underline{J}^{T}_{0A}\dot{\underline{\Omega}}_A + \underline{H}^{0^T}_A\ddot{\underline{Q}}_A = \underline{g}$$

$$\underline{C}^{0}_A\dot{\underline{V}}_0 + \underline{J}_{0A}\dot{\underline{\omega}}_0 + \underline{J}_A\dot{\underline{\Omega}}_A + \underline{H}^{T}_A\ddot{\underline{Q}}_A = \underline{g}_A + \underline{g}_{0A}$$

$$\underline{P}^{0}_A\dot{\underline{V}}_0 + \underline{H}^{0}_A\dot{\underline{\omega}}_0 + \underline{H}_A\dot{\underline{\Omega}}_A + \ddot{\underline{Q}}_A + \underline{\Omega}^2_c\,\underline{Q}_A = \underline{\Upsilon}^a_A \qquad (14)$$

It is interesting to compare Eqs. (14) with Eqs. (43) of Hughes[4]. For even more compaction of these equations, the following matrices are introduced:

$$\underline{M}_{VV} \triangleq \begin{bmatrix} m\underline{1} & -\underline{c}^x \\ \underline{c}^x & \underline{J} \end{bmatrix} \quad \underline{M}_{VA} \triangleq [\underline{C}^0_A \quad \underline{J}_{0A}], \; \ddot{\underline{q}}^T_V \triangleq [\underline{V}^T_0 \quad \underline{\omega}^T_0] \\ \underline{\mathcal{P}}^0_A \triangleq [\underline{P}^0_A \quad \underline{H}^0_A], \; \underline{u}^T_V \triangleq [\underline{f}^T \quad \underline{g}^T] \qquad (15)$$

where $\underline{1}$ is a 3x3 identity matrix. Eqs. (14) thereby reduce to the following three matrix equations: one governing six overall degrees of freedom of the spacecraft, $\underline{q}_V(t)$; the second governing n_ax1 vector $\underline{\Omega}_A$ of n_a relative angular velocities of N articulated bodies; and the third governing the ∞x1 vector $\underline{Q}_A$ of modal coordinates of appendage modes of all articulated bodies.

$$\underline{M}_{VV}\ddot{\underline{q}}_V + \underline{M}^T_{VA}\dot{\underline{\Omega}}_A + \underline{\mathcal{P}}^{0^T}_A \ddot{\underline{Q}}_A = \underline{u}_V(t) \qquad (16.a)$$

$$\underline{M}_{VA}\ddot{\underline{q}}_V + \underline{J}_A\dot{\underline{\Omega}}_A + \underline{H}^T_A\ddot{\underline{Q}}_A = \underline{g}_A(t) + \underline{g}_{0A}(t) \qquad (16.b)$$

$$\underline{\mathcal{P}}^0_A\ddot{\underline{q}}_V + \underline{H}_A\dot{\underline{\Omega}}_A + \ddot{\underline{Q}}_A + \underline{\Omega}^2_c\underline{Q}_A = \underline{Y}^a_A(t) \qquad (16.c)$$

Modal identities associated with the modal momenta matrices $\underline{\mathcal{P}}^0_A$ and $\underline{H}_A$ are derived in Section IV.

DISCRETIZATION BY HINGES-FREE VEHICLE MODES

In this technique, the continuum equations (7) and (8) are discretized all at once. For this purpose, the following modal expansion is postulated for the variables in Eqs. (7) and Eq. (8) (Reference 3):

$$\underline{V}_0(t) = \dot{\underline{R}}_0(t) + \Sigma\, \underline{X}_{0\nu}\dot{\eta}_\nu(t),$$
$$\underline{\omega}_0(t) = \dot{\underline{\Theta}}_0(t) + \Sigma\, \underline{\Phi}_{0\nu}\dot{\eta}_\nu(t),$$
$$\underline{\Omega}_j(t) = \dot{\underline{\Theta}}_j(t) + \sum \underline{\Phi}_{j\nu}\dot{\eta}_\nu(t),$$
$$\underline{u}_j(\underline{r}_j,t) = \sum \underline{U}_{j\nu}(\underline{r}_j)\eta_\nu(t), \; \sum = \sum_{\nu=1}^{\infty}, \qquad (j=1,2,\ldots,N) \quad (17)$$

where $\underline{R}_0$, $\underline{\Theta}_0$, and $\underline{\Theta}_j$ are the temporal coordinates for the rigid modes of the spacecraft; the total number of articulation degrees of freedom is n_a, so there are n_a+6 rigid modes in all. Furthermore, $\underline{R}_0$ is the translation of the reference point 0, and Θ_0 is the rotation of the spacecraft, both in rigid modes; similarly, $\underline{\Theta}_j$ is the rotation of the hinged body E_j relative to B_0 at the hinge 0_j (j=1,...,N) in a rigid mode. The quantities $\underline{X}_{0\nu}$, $\underline{\Phi}_{0\nu}$, and $\underline{\Phi}_{j\nu}$ (j=1,...,N; ν=1,...,∞) are ν-th modal coefficients contributing, respectively, to overall discrete motions $\underline{V}_0$, $\underline{\omega}_0$, and $\underline{\Omega}_j$; and $\eta_\nu(t)$ is the associated modal coordinate. The eigenfunction $\underline{U}_{j\nu}(\underline{r}_j)$ is that part of the hinges-free vehicle mode, denoted $\underline{W}_\nu(\underline{r})$, which defines the deformation of body E_j in ν-th mode. Although $\underline{U}_{j\nu}(\underline{r}_j)$ satisfies the condition of zero displacement and zero rotation at the hinge 0_j, that is, $\underline{U}_{j\nu}(0_j) = \underline{0}$ and $\frac{1}{2}\,\underline{\nabla}^x\underline{U}_{j\nu}(0_j) = \underline{0}$, it is not the same as the σ-th appendage mode $\underline{U}^a_{j\sigma}(\underline{r}_j)$ used before, because in the case of $\underline{U}_{j\nu}(\underline{r}_j)$, no torque acts, by definition, at the hinge 0_j to enforce the zero deformation and zero rotation condition, whereas in the case of $\underline{U}^a_{j\sigma}(\underline{r}_j)$, the immobile support of the appendage enforces that condition. Because of the mobile support of the hinge 0_j, the total motion $\underline{W}_{j\nu}(\underline{r}_j)$ of E_j in an inertial frame is

$$\underline{W}_{j\nu}(\underline{r}_j) = \underline{X}_{0\nu} - (\underline{b}_j + \underline{C}_{0j}\underline{r}_j)^x\, \underline{\Phi}_{0\nu} - \underline{C}_{0j}\underline{r}^x_j\underline{\Phi}_{j\nu} \\ + \underline{C}_{0j}\underline{U}_{j\nu}(\underline{r}_j) \qquad (j=1,\ldots,N;\ \nu=1,\ldots,\infty) \qquad (18)$$

where the first two terms in the right side are because of the translation and rotation of the core body in the ν-th mode, and the third term $\underline{r}^x_j\underline{\Phi}_{j\nu}$ is caused by the relative rotation of E_j at the free hinge 0_j. The motion of the core body in ν-th mode is given simply by

$$\underline{W}_{0\nu}(\underline{r}_0) = \underline{X}_{0\nu} - \underline{r}^x_0\,\underline{\Phi}_{0\nu} \qquad (19)$$

Thus a hinges-free vehicle mode $\underline{W}_\nu(\underline{r})$spans entire spacecraft such that

$$\underline{W}_\nu(\underline{r}) = \begin{cases} \underline{W}_{0\nu}(\underline{r}_0), & \text{if } \underline{r} = \underline{r}_0 \\ \underline{W}_{j\nu}(\underline{r}_j), & \text{if } \underline{r} = \underline{b}_j + \underline{C}_{0j}\underline{r}_j \end{cases} \qquad (20)$$
$$(j=1,\ldots,N;\ \nu=1,\ldots,\infty)$$

Following Hughes[2], the 6+n_a rigid modes of a spacecraft with articulated bodies are $\underline{1}$, $-\underline{r}^x$, and $-\underline{r}^x_j$ (j=1,...,n_a). Not surprisingly, the elastic modes $\underline{W}_\nu(\underline{r})$ (ν=1,...,∞) are orthogonal to these rigid modes; that is:

$$\int_V \underline{W}_\nu(\underline{r})\, dm = \underline{0}, \quad \int_V \underline{r}^x\underline{W}_\nu(\underline{r})\, dm = \underline{0},$$
$$\int_j \underline{r}^x_j\underline{W}_{j\nu}(\underline{r}_j)\, dm = \underline{0} \qquad (21.a,b,c)$$

where $\int_V$ means the entire vehicle is the domain of integration. Eqs. (21) can be verified by substituting the expansion (17) in the continuum equations (7) with zero right sides. Indeed, Eqs. (21a,b) state that the linear and angular momentum residing in a ν-th hinges-free vehicle mode are zero, whereas Eq. (21c) expresses a zero momentum-like property of the articulated body E_j. These properties can be stated alternately by defining modal momenta coefficients ($\underline{p}_{j\nu}$,$\underline{h}_{j\nu}$) for each articulated body and ($\underline{p}_\nu$,$\underline{h}^0_\nu$) for all articulated bodies collectively:

$$\underline{p}_{j\nu} \triangleq \int_j \underline{U}_{j\nu}(\underline{r}_j)\, dm \qquad \underline{h}_{j\nu} \triangleq \int_j \underline{r}^x_j\, \underline{U}_{j\nu}(\underline{r}_j)\, dm \qquad (22)$$
$$\underline{p}_\nu \triangleq \sum_j \underline{C}_{0j}\underline{p}_{j\nu} \qquad \underline{h}^0_\nu = \sum_j [\underline{b}^x_j\underline{C}_{0j}\underline{p}_{j\nu} + \underline{C}_{0j}\underline{h}_{j\nu}]$$

where $\underline{h}^0_\nu$ is defined relative to the reference point 0. These may be compared with the definitions (9) and (10). The zero momentum properties (21) then transform to:

$$m\,\underline{\chi}_{0\nu} - \underline{c}^{\times}\underline{\phi}_{0\nu} - \sum_j \underline{C}_{0j}\underline{c}_j^{\times}\underline{\phi}_{j\nu} + \underline{p}_\nu = \underline{0}$$

$$\underline{c}^{\times}\underline{\chi}_{0\nu} + \underline{J}\,\underline{\phi}_{0\nu} + \sum_j \underline{C}_{0j}\underline{J}_{0j}\underline{\phi}_{j\nu} + \underline{h}_\nu^0 = \underline{0}$$

$$\underline{c}_j^{\times}\underline{C}_{j0}\underline{\chi}_{0\nu} + \underline{J}_{0j}^T\underline{C}_{j0}\underline{\phi}_{0\nu} + \underline{J}_j\underline{\phi}_{j\nu} + \underline{h}_{j\nu} = \underline{0}$$

$$(j=1,\ldots N;\ \nu=1,\ldots,\infty) \qquad (23)$$

The eigenvalue problem which governs a hinges-free vehicle mode is given by

$$\underline{L}_j\underline{U}_{j\nu}(\underline{r}_j) = \sigma_j\omega_\nu^2\underline{C}_{j0}\underline{W}_{j\nu}(\underline{r}_j) \quad (j=1,\ldots,N;\ \nu=1,\ldots,\infty) \qquad (24)$$

where ω_ν is the frequency of ν-th mode. The orthogonality conditions are obtained by performing the operation $\sum_j \int_j \underline{U}_{j\mu}^T(\underline{r}_j)\,(\bullet)\,dv$ over the eigenvalue problem (24) and recalling the properties (23). Here, dv is an elemental volume. One then obtains

$$\int_V \underline{W}_\mu^T(\underline{r})\underline{W}_\nu(\underline{r})\,dm = \delta_{\mu\nu} \qquad (25.a)$$

$$\sum_j \left\{ \int_j \underline{U}_{j\mu}^T\underline{U}_{j\nu}\,dm + \underline{h}_{j\mu}^T\underline{\phi}_{j\nu}\right\} + \underline{p}_\mu^T\underline{\chi}_{0\nu} + \underline{h}_\mu^{0^T}\underline{\phi}_{0\nu} = \delta_{\mu\nu} \qquad (25.b)$$

$$\begin{aligned}&\sum_j \Big[\int_j \underline{U}_{j\mu}^T\underline{U}_{j\nu}\,dm + \{\underline{\chi}_{0\mu}^T\underline{C}_{0j}\underline{c}_j^{\times}\underline{\phi}_{j\nu} - \underline{\phi}_{j\mu}^T\underline{c}_j^{\times}\underline{C}_{j0}\underline{\chi}_{0\nu}\} - \\ &(\underline{\phi}_{j\mu}^T\underline{J}_j\underline{\phi}_{j\nu} + \underline{\phi}_{j\mu}^T\underline{C}_{j0}\underline{J}_{j0}\underline{\phi}_{0\nu} + \underline{\phi}_{0\mu}^T\underline{J}_{j0}\underline{C}_{0j}\underline{\phi}_{j\nu})\Big] \\ &- \left[m\,\underline{\chi}_{0\mu}^T\underline{\chi}_{0\nu} + \underline{\phi}_{0\mu}^T\underline{c}^{\times}\underline{\chi}_{0\nu} - \underline{\chi}_{0\mu}^T\underline{c}^{\times}\underline{\phi}_{0\nu} + \underline{\phi}_{0\mu}^T\underline{J}\,\underline{\phi}_{0\nu}\right] = \delta_{\mu\nu}\end{aligned} \qquad (25.c)$$

$$\sum_j \int_j \underline{U}_{j\mu}^T\,\underline{L}_j\underline{U}_{j\nu}(\underline{r}_j)\,dv = \omega_\nu^2\delta_{\mu\nu} \qquad (25.d)$$

where $\delta_{\mu\nu}$ is the Kronecker delta.

Utilizing the modal expansion (17), the zero momentum modal properties (21) and (23), and the orthogonality properties (25), the continuum equations are discretized to these decoupled equations which separately govern the rigid and elastic modes of the spacecraft:

$$m\,\ddot{\underline{R}}_0 - \underline{c}^{\times}\ddot{\underline{\Theta}}_0 - \sum_j \underline{C}_{0j}\underline{c}_j^{\times}\ddot{\underline{\Theta}}_j = \underline{f} \qquad (26.a)$$

$$\underline{c}^{\times}\ddot{\underline{R}}_0 + \underline{J}\,\ddot{\underline{\Theta}}_0 + \sum_j \underline{C}_{0j}\underline{J}_{0j}\ddot{\underline{\Theta}}_j = \underline{g} \qquad (26.b)$$

$$\underline{c}_j^{\times}\underline{C}_{j0}\ddot{\underline{R}}_0 + \underline{J}_{0j}^T\underline{C}_{j0}\ddot{\underline{\Theta}}_0 + \underline{J}_j\ddot{\underline{\Theta}}_j = \underline{g}_{0j} + \underline{g}_j \quad (j=1,\ldots,N) \qquad (26.c)$$

$$\ddot{\eta}_\nu + \omega_\nu^2\eta_\nu = \underline{\chi}_{0\nu}^T\underline{f} + \underline{\phi}_{0\nu}^T\underline{g} + \sum_j \underline{\phi}_{j\nu}^T(\underline{g}_{0j} + \underline{g}_j) + \gamma_\nu(t)$$

$$(\nu=1,2,\ldots,\infty) \qquad (26.d)$$

where $\gamma_\nu(t)$ is the scalar input to ν-th mode considering all articulated bodies collectively:

$$\gamma_\nu(t) \triangleq \sum_j \int_j \underline{U}_{j\nu}^T(\underline{r}_j)\underline{\mathfrak{f}}_j(\underline{r}_j,t)\,dA \qquad (27)$$

These equations are abbreviated by recalling appropriate definitions from (13) and (15) and by the following additional definitions

$$\underline{q}_{VR} \triangleq \begin{bmatrix}\underline{R}_0\\ \underline{\Theta}_0\end{bmatrix},\ \underline{\Theta}_A \triangleq \begin{vmatrix}\underline{\Theta}_1\\ \cdot\\ \cdot\\ \cdot\\ \underline{\Theta}_N\end{vmatrix},\ \underline{\chi}_0 \triangleq \begin{bmatrix}\underline{\chi}_{01}^T\\ \underline{\chi}_{02}^T\\ \cdot\\ \cdot\\ \cdot\end{bmatrix},\ \underline{\phi}_0 \triangleq \begin{bmatrix}\underline{\phi}_{01}^T\\ \underline{\phi}_{02}^T\\ \cdot\\ \cdot\\ \cdot\end{bmatrix},$$

$$\underline{\phi}_A \triangleq \begin{bmatrix}\underline{\phi}_{11}^T & \cdots & \underline{\phi}_{N1}^T\\ \underline{\phi}_{12}^T & \cdots & \underline{\phi}_{N2}^T\\ \vdots & & \vdots\end{bmatrix} \quad \begin{aligned}\underline{\eta}_e^T &\triangleq [\eta_1\ \eta_2\ldots],\\ \underline{\omega} &= \text{diag}\,[\omega_1\ \omega_2\ldots],\\ \underline{\gamma}^T &\triangleq [\gamma_1\ \gamma_2\ldots]\end{aligned} \qquad (28)$$

Here $\underline{q}_{VR}$ is a rigid mode vector, whereas $\underline{q}_V$ in (15) is a vector of overall motion of the spacecraft; the vector $\dot{\underline{\Theta}}_A$ from (28) and $\underline{\Omega}_A$ in (13) differ likewise. Eqs. (26) now condense to this desired compact form:

$$\underline{M}_{VV}\ddot{\underline{q}}_{VR} + \underline{M}_{VA}^T\ddot{\underline{\Theta}}_A = \underline{u}_V$$

$$\underline{M}_{VA}\ddot{\underline{q}}_{VR} + \underline{J}_A\ddot{\underline{\Theta}}_A = \underline{g}_A + \underline{g}_{0A}$$

$$\ddot{\underline{\eta}}_e + \underline{\omega}^2\underline{\eta}_e = \underline{\chi}_0\underline{f} + \underline{\phi}_0\underline{g} + \underline{\phi}_A(\underline{g}_{0A} + \underline{g}_A) + \underline{\gamma} \qquad (29)$$

DISCRETIZATION BY HINGES-LOCKED VEHICLE MODES

Since these modes are defined by forcing the articulation motion $\underline{\Omega}_j$ (j=1,...,N) to be zero, they are obtained by a modal analysis of the first two equations in (7) and Eq. (8) from which the $\underline{\Omega}_j$ (j=1,...,N) terms are ignored. The torque actually required to keep the hinges locked can be evaluated from the third equation in (7) but that is not relevant here. The equations for the modal analysis are therefore:

$$m\dot{\underline{V}}_0 - \underline{c}^{\times}\dot{\underline{\omega}}_0 + \sum \int_j \underline{C}_{0j}\ddot{\underline{u}}_j\,dm = \underline{0}$$

$$\underline{c}^{\times}\dot{\underline{V}}_0 + \underline{J}\,\dot{\underline{\omega}}_0 + \sum \int_j (\underline{b}_j^{\times}\underline{C}_{0j} + \underline{C}_{0j}\underline{r}_j^{\times})\ddot{\underline{u}}_j\,dm = \underline{0}$$

$$\underline{L}_j\underline{u}_j + \sigma_j\,\{\underline{C}_{j0}\dot{\underline{V}}_0 - (\underline{C}_{j0}\underline{b}_j + \underline{r}_j)^{\times}\underline{C}_{j0}\dot{\underline{\omega}}_0 + \ddot{\underline{u}}_j\} = \underline{0} \qquad (30)$$

Eqs. (30), in fact, govern the motion of a free spacecraft with cantilevered appendages, so the sought hinges-locked vehicle modes are the same as the unconstrained modes a la Hughes[4]. The development here parallels that in the previous subsection on hinges-free modes. Accordingly, introduce the following modal expansion:

$$\underline{V}_0 = \dot{\underline{R}}_0 + \sum \underline{\chi}_{0\alpha}^c\dot{\eta}_\alpha^c(t),\ \sum = \sum_{\alpha=1}^{\infty}$$

$$\underline{\omega}_0 = \dot{\underline{\Theta}}_0 + \sum \underline{\phi}_{0\alpha}^c\dot{\eta}_\alpha^c(t),\ \underline{u}_j = \sum \underline{U}_{j\alpha}^c\,(\underline{r}_j)\eta_\alpha^c(t) \qquad (31)$$

where the superscript c reminds us that these modal quantities pertain to hinges-locked modes. The quantities $\underline{\chi}_{0\alpha}^c$ and $\underline{\phi}_{0\alpha}^c$ are the translation and rotation of the core body B_0 in α-th mode. Likewise, the eigenfunction $\underline{U}^c_{j\alpha}(\underline{r}_j)$ is the deformation of E_j in the α-th mode, analogous to $\underline{U}_{j\nu}(\underline{r}_j)$ in the case of ν-th hinges-free mode, except that now a force is exerted at the hinge 0_j to ascertain that $\underline{U}^c_{j\alpha}(0_j) = \underline{0}$ and $\underline{\nabla}^\times\underline{U}^c_{j\alpha}(0_j) = \underline{0}$. The total motion of j-th appendage relative to the B_0-fixed frame in α-th vehicle mode is denoted $\underline{W}^c_{j\alpha}(\underline{r}_j)$ and it equals [cf. Eq. (18)]

$$\underline{W}^c_{j\alpha}(\underline{r}_j) \triangleq \underline{\chi}^c_{0\alpha} - (\underline{b}_j + \underline{C}_{0j}\underline{r}_j)^\times \underline{\phi}^c_{0\alpha} + \underline{C}_{0j}\underline{U}^c_{j\alpha}(\underline{r}_j) \qquad (32)$$

The α-th mode of the core body, $\underline{W}^c_{0\alpha}(\underline{r}_0)$, on the other hand, will be

$$\underline{W}^c_{0\alpha}(\underline{r}_0) \triangleq \underline{\chi}^c_{0\alpha} - \underline{r}_0^\times\underline{\phi}^c_{0\alpha} \qquad (33)$$

Thus, like Eq. (20), the α-th mode $\underline{W}^c_\alpha(\underline{r})$ will be $\underline{W}^c_{0\alpha}(\underline{r}_0)$ or $\underline{W}^c_{j\alpha}(\underline{r}_j)$ depending on the domain under consideration. Orthogonality of these modes with the six rigid modes $\underline{1}$ and $-\underline{r}^\times$, similar to Eqs. (21.a, 21.b), can be proved easily. To express these conditions in terms of hinges-locked modal momenta coefficients, define [cf. Eq. (22)]:

$$\underline{p}^c_{j\alpha} \triangleq \int_j \underline{U}^c_{j\alpha}(\underline{r}_j)\,dm \qquad \underline{h}^c_{j\alpha} \triangleq \int_j \underline{r}_j^\times\underline{U}^c_{j\alpha}(\underline{r}_j)\,dm$$

$$\underline{p}^c_\alpha \triangleq \sum_j \underline{C}_{0j}\underline{p}^c_{j\alpha} \qquad \underline{h}^{0^c}_\alpha \triangleq \sum_j \left(\underline{b}_j^\times\underline{C}_{0j}\underline{p}^c_{j\alpha} + \underline{C}_{0j}\underline{h}^c_{j\alpha}\right) \qquad (34)$$

Then, the above mentioned orthogonality is [cf. Eq. (23)]:

$$m\,\underline{\chi}^c_{0\alpha} - \underline{c}^\times\underline{\phi}^c_{0\alpha} + \underline{p}^c_\alpha = \underline{0}$$

$$\underline{c}^\times\underline{\chi}^c_{0\alpha} + \underline{J}\,\underline{\phi}^c_{0\alpha} + \underline{h}^{0^c}_\alpha = \underline{0} \qquad (35)$$

The eigenvalue problem obeyed by the α-th hinges-locked mode $\underline{W}^c_{j\alpha}(\underline{r}_j)$ is

$$\underline{L}_j\underline{U}^c_{j\alpha}(\underline{r}_j) = \omega^{c^2}_\alpha\,\underline{C}_{j0}\underline{W}^c_{j\alpha}(\underline{r}_j) \qquad (j=1,\ldots,N) \qquad (36)$$

where ω^c_α is the associated hinges-locked frequency. The orthogonality conditions between α-th and β-th modes are

$$\int_V \underline{W}^{c^T}_\alpha\underline{W}^c_\beta\,dm = \sum_j\int_j \underline{U}^{c^T}_{j\alpha}\underline{C}_{j0}\underline{W}^c_{j\beta}(\underline{r}_j)\,dm = \delta_{\alpha\beta}$$

$$\sum_j\int_j \underline{U}^{c^T}_{j\alpha}\underline{U}^c_{j\beta}\,dm - m\underline{\chi}^{c^T}_{0\alpha}\underline{\chi}^c_{0\beta} + \left(\underline{\chi}^{c^T}_{0\alpha}\underline{c}^\times\underline{\phi}^c_{0\beta} - \underline{\phi}^{c^T}_{0\alpha}\underline{c}^\times\underline{\chi}^c_{0\beta}\right)$$

$$- \underline{\phi}^{c^T}_{0\alpha}\underline{J}\,\underline{\phi}^c_{0\beta} = \delta_{\alpha\beta}$$

$$\sum_j\int_j \underline{U}^{c^T}_{j\alpha}\underline{L}_j\underline{U}^c_{j\beta}\,dv = \omega^{c^2}_\alpha\,\delta_{\alpha\beta} \qquad (37)$$

These are a bit more general than Eqs. (62) of Hughes[4].

With the aid of the expansion (31), momental properties (35), and orthogonality properties (37), the original continuum equations (9) are discretized to

$$m\,\ddot{\underline{R}}_0 - \underline{c}^\times\ddot{\underline{\Theta}}_0 - \sum_j \underline{C}_{0j}\underline{c}_j^\times\dot{\underline{\Omega}}_j = \underline{f}(t)$$

$$\underline{c}^\times\ddot{\underline{R}}_0 + \underline{J}\ddot{\underline{\Theta}}_0 + \sum_j \underline{C}_{0j}\underline{J}_{0j}\dot{\underline{\Omega}}_j = \underline{g}(t)$$

$$\underline{c}_j^\times\underline{C}_{j0}\ddot{\underline{R}}_0 + \underline{J}^T_{0j}\underline{C}_{j0}\ddot{\underline{\Theta}}_0 + \underline{J}_j\dot{\underline{\Omega}}_j + \sum_\alpha \underline{h}^{cI}_{j\alpha}\ddot{\eta}^c_\alpha = \underline{g}_{0j} + \underline{g}_j$$

$$(j=1,\ldots,N)$$

$$\sum_j \left(\underline{h}^{cI}_{j\alpha}\right)^T\dot{\underline{\Omega}}_j + \ddot{\eta}^c_\alpha + \omega^{c^2}_\alpha\,\eta^c_\alpha = \underline{\chi}^{c^T}_{\alpha0}\underline{f} + \underline{\phi}^{c^T}_{\alpha0}\underline{g} + \gamma^c_\alpha(t) \qquad (38)$$

Unlike the hinges-free set of discrete equations (28), the last two equations in (38) involve a new coupling term called "inertial modal angular momentum coefficient" $\underline{h}^{cI}_{j\alpha}$ defined as

$$\underline{h}^{cI}_{j\alpha} \triangleq \int_j \underline{r}_j^\times\underline{C}_{j0}\underline{W}^c_{j\alpha}(\underline{r}_j)\,dm =$$

$$\underline{c}_j^\times\underline{C}_{j0}\underline{\chi}^c_{0\alpha} + \left(\underline{J}_j\underline{C}_{j0} - \underline{c}_j^\times\underline{C}_{j0}\underline{b}_j^\times\right)\underline{\phi}^c_{\alpha0} + \int_j \underline{r}_j^\times\underline{U}^c_{j\alpha}\,dm \qquad (39)$$

which is different from $\underline{h}^c_{j\alpha}$ and $\underline{h}^{0^c}_\alpha$ defined in (34). The disturbance input $\gamma^c_\alpha(t)$ to each α-th mode equals [cf. Eq. (29)]

$$\gamma^c_\alpha(t) \triangleq \sum_j\int_j \underline{U}^{c^T}_{j\alpha}(\underline{r}_j)\underline{f}_j(\underline{r}_j,t)\,dA \qquad (40)$$

To compact Eqs. (40), introduce

$$\underline{h}^{cI^T}_j \triangleq \left[\underline{h}^{cI}_{j1}\ \ \underline{h}^{cI}_{j2}\ \ldots\right], \qquad \underline{h}^{cI}_A \triangleq \left[\underline{h}^{cI}_1\ \ \ldots\ \underline{h}^{cI}_N\right] \qquad (41)$$

Then, recalling pertinent definitions from (13), (15), and (28), Eqs. (38) contract to

$$\begin{aligned}
&\underline{M}_{VV}\,\ddot{\underline{q}}_{VR} + \underline{M}^T_{VA}\dot{\underline{\Omega}}_A && = \underline{u}_V(t)\\
&\underline{M}_{VA}\,\ddot{\underline{q}}_{VR} + \underline{J}_A\dot{\underline{\Omega}}_A + \underline{h}^{cI^T}_A\,\ddot{\underline{\eta}}^c_e(t) && = \underline{g}_A(t) + \underline{g}_{0A}(t)\\
&\underline{h}^{cI}_A\dot{\underline{\Omega}}_A + \ddot{\underline{\eta}}^c_e(t) + \underline{\omega}_c^{\,2}\underline{\eta}_e(t) = \underline{\chi}^c_0\underline{f} + \underline{\phi}^c_0\underline{g} + \underline{\gamma}_c && \qquad (42)
\end{aligned}$$

The vector $\underline{\eta}^c_e(t)$ and $\underline{\gamma}_c(t)$, and the matrices $\underline{\omega}_c, \underline{\chi}^c_0, \underline{\phi}^c_0$ are defined like their hinges-free companions in (28).

IV. MODAL IDENTITIES FOR MULTIBODY ELASTIC SPACECRAFT

Our principal concern is to compare hinges-free and hinges-locked modes for their accuracy in representing articulation motion. To accomplish this aim, an equation will be obtained from each of the above three sets of discrete equations which will be solely in terms of the articulation motion $\dot{\underline{\Omega}}_A$ and stimuli. These three equations will then be compared to yield identities.

First, consider the discrete set (16) based on appendage modes. By matrix manipulations, the following equation governing $\dot{\underline{\Omega}}_A$ can be constructed readily:

$$[\underline{1} - \underline{\mathcal{H}}_A^T (\underline{\mathcal{U}}_\infty + \underline{\Omega}_C^2/s^2)^{-1} \underline{\mathcal{H}}_A \underline{\mathcal{J}}^{-1}]\, \underline{\mathcal{J}}\, \dot{\underline{\Omega}}_A =$$

$$\underline{\mathcal{H}}_A^T (\underline{\mathcal{U}}_\infty + \underline{\Omega}_C^2/s^2)^{-1} \underline{\mathcal{P}}_A^0\, \underline{M}_{VV}^{-1}\, \underline{u}_V$$

$$-\underline{\mathcal{H}}_A^T (\underline{\mathcal{U}}_\infty + \underline{\Omega}_C^2/s^2)^{-1} \underline{Y}_A^a - \underline{M}_{VA}\, \underline{M}_{VV}^{-1}\, \underline{u}_V + \underline{g}_H \qquad (43)$$

where s is the Laplace variable; theoretically, $\underline{\mathcal{H}}_A$ is an ∞ x n_a matrix, and $\underline{\mathcal{U}}_\infty$ an ∞ x ∞ symmetric matrix; $\underline{\mathcal{J}}$ is an n_axn_a inertia matrix, and $\underline{g}_H$ is the total hinge torque vector:

$$\underline{\mathcal{H}}_A \triangleq -\underline{\mathcal{P}}_A^0\, \underline{M}_{VV}^{-1}\, \underline{M}_{VA}^T + \underline{H}_A,\ \underline{\mathcal{U}}_\infty \triangleq \underline{1}_\infty - \underline{\mathcal{P}}_A^0 \underline{M}_{vv}^{-1} \underline{\mathcal{P}}_A^{0^T}$$

$$\underline{\mathcal{J}} \triangleq \underline{J}_A - \underline{M}_{VA}\, \underline{M}_{VV}^{-1}\, \underline{M}_{VA}^T,\ \underline{g}_H \triangleq \underline{g}_A + \underline{g}_{0A} \qquad (44)$$

In (44), $\underline{1}_\infty$ is ∞ x ∞ identity matrix. Thus for an equation of $\underline{\mathcal{J}}\, \dot{\underline{\Omega}}_A$ (s), the coefficient of the hinge torque $\underline{g}_H$ (s) is

$$[\underline{1} - \underline{\mathcal{H}}_A^T (\underline{\mathcal{U}}_\infty + \underline{\Omega}_C^2/s^2)^{-1} \underline{\mathcal{H}}_A \underline{\mathcal{J}}^{-1}]^{-1} \qquad (45)$$

Anticipating our later needs, now we shall prove that the lim s→∞ of the matrix [•] in (45) equals

$$\underline{1} - \underline{\mathcal{H}}_A^T\, \underline{\mathcal{U}}_\infty^{-1} \underline{\mathcal{H}}_A\, \underline{\mathcal{J}}^{-1} = \underline{0} \qquad (46)$$

Applying the matrix inversion lemma to $\underline{\mathcal{U}}_\infty$, its inverse is found to be:

$$\underline{\mathcal{U}}_\infty^{-1} = \underline{1}_\infty - \underline{\mathcal{P}}_A^0\, [\underline{\mathcal{P}}_A^{0^T} \underline{\mathcal{P}}_A^0 - \underline{M}_{VV}]^{-1} \underline{\mathcal{P}}_A^{0^T} \qquad (47)$$

On the other hand, owing to the identities (D,E,F)" of Hughes[4]

$$\underline{\mathcal{P}}_A^{0^T} \underline{\mathcal{P}}_A^0 = \begin{bmatrix} m_e \underline{1} & -\underline{c}_e^x \\ \underline{c}_e^x & \underline{J}_e^0 \end{bmatrix} \triangleq \underline{M}_e^0 \qquad (48)$$

Also, by definition of $\underline{M}_{VV}$ in (15), and by virtue of Eq. (1) and Eq. (3)

$$\underline{M}_{VV} = \begin{bmatrix} m_0 \underline{1} & -\underline{c}_0^x \\ \underline{c}_0^x & \underline{J}_0 \end{bmatrix} + \underline{M}_e^0 \triangleq \underline{M}_0 + \underline{M}_e^0 \qquad (49)$$

which reduces $\underline{\mathcal{U}}_\infty^{-1}$ to

$$\underline{\mathcal{U}}_\infty^{-1} = \underline{1}_\infty + \underline{\mathcal{P}}_A^0\, \underline{M}_0^{-1} \underline{\mathcal{P}}_A^{0^T} \qquad (50)$$

A comparison of $\underline{\mathcal{U}}_\infty$ with $\underline{\mathcal{U}}_\infty^{-1}$ amazes. Continuing with the proof nevertheless, call upon the basic identities (D,E,F)" of Hughes[4] to derive the following new identities associated with the articulation degrees of freedom:

$$\underline{H}_A^T\, \underline{H}_A = \underline{J}_A \qquad \underline{H}_A^T\, \underline{P}_A^0 = \underline{C}_A^0 \qquad \underline{H}_A^T\, \underline{H}_A^0 = \underline{J}_{0A}$$

$$\underline{\mathcal{H}}_A^T\, \underline{\mathcal{H}}_A = \underline{\mathcal{J}} - \underline{M}_{VA} \underline{M}_{VV}^{-1} \underline{M}_0\, \underline{M}_{VV}^{-1}\, \underline{M}_{VA}^T,$$

$$\underline{\mathcal{H}}_A^T \underline{\mathcal{P}}_A^0 = \underline{M}_{VA}\, \underline{M}_{VV}^{-1}\, \underline{M}_0 \qquad (I)$$

(New identities derived in this paper will be labeled with Roman numerals as they are cited.) These identities and Eq. (50), in turn, lead to the identity

$$\underline{\mathcal{H}}_A^T\, \underline{\mathcal{U}}_\infty^{-1} \underline{\mathcal{H}}_A = \underline{\mathcal{J}} \qquad (II)$$

which proves Eq. (46)

An equation analogous to Eq. (43) is obtained from the hinges-free discrete set (29). For that, recall the second expansion in (17). Then it can be shown that

$$\underline{\mathcal{J}}\, \dot{\underline{\Omega}}_A = \underline{\mathcal{J}}\, \underline{\phi}_A^T\, (\underline{1}_\infty + \underline{\omega}^2/s^2)^{-1}\, (\underline{X}_0\, \underline{f} + \underline{\phi}_0\, \underline{g} + \underline{Y})$$

$$- \underline{M}_{VA}\, \underline{M}_{VV}^{-1}\, \underline{u}_V(s)$$

$$+ [\underline{1} + \underline{\mathcal{J}}\, \underline{\phi}_A^T\, (\underline{1}_\infty + \underline{\omega}^2/s^2)^{-1}\, \underline{\phi}_A]\, \underline{g}_H(s) \qquad (51)$$

The coefficient of the hinge torque $\underline{g}_H$ (s) in Eq. (51) equals the term (45). They both reduce to $\underline{1}$ for the lim s→0, and for the lim s→∞ they yield, in view of Eq. (46), the identity

$$(\underline{1} + \underline{\mathcal{J}}\, \underline{\phi}_A^T\, \underline{\phi}_A)^{-1} = \underline{0} \qquad (III)$$

which proves a fortiori that, since the inertia matrix $\underline{\mathcal{J}}$ is positive definite and $\underline{\phi}_A^T \underline{\phi}_A$ nonnegative definite, the modal coefficients $\underline{\phi}_{j\nu}$(j=1,...,N; ν=1,...,∞) [Eq. (17)] constitute a nonconverging series.

The hinges-locked discrete set (42) furnishes this equation for $\underline{\Omega}_A$:

$$[\underline{1} - \underline{h}_A^{cI^T} (\underline{1} + \underline{\omega}_c^2/s^2)^{-1} \underline{h}_A^{cI} \underline{\mathcal{J}}^{-1}]\underline{\mathcal{J}}\, \dot{\underline{\Omega}}_A = \underline{g}_H - \underline{M}_{VA}\, \underline{M}_{VV}^{-1}\, \underline{u}_V$$

$$- \underline{h}_A^{cI^T} (\underline{1} + \underline{\omega}_c^2/s^2)^{-1}\, (\underline{X}_0^c\, \underline{f} + \underline{\phi}_0^c\, \underline{g} + \underline{Y}_c) \qquad (52)$$

The equality of the coefficient matrices of $\underline{g}_H$(s) in Eq. (51) and Eq. (52) delivers this identity in the s-domain:

$$[\underline{1} - \underline{h}_A^{cI^T} (\underline{1} + \underline{\omega}_c^2/s^2)^{-1} \underline{h}_A^{cI} \underline{\mathcal{J}}^{-1}]^{-1} = \underline{1} +$$

$$\underline{\mathcal{J}}\, \underline{\phi}_A^T (\underline{1}_\infty + \underline{\omega}^2/s^2)^{-1} \underline{\phi}_A \qquad (IV)$$

For the lim s→0, the left side of (IV) degenerates to $\underline{1}$ as does its right side. On the other hand, taking its limit s→∞ and recognizing the identity (III) produce the identity

$$[\underline{1} - \underline{h}_A^{cI^T} \underline{h}_A^{cI} \underline{\mathcal{J}}^{-1}] = \underline{0} \qquad (V)$$

The identity (IV) can be rearranged such that it reveals poles and zeros of the dynamics. For that, recognize that when $s = \pm j\omega_\nu$ ($\nu=1,\ldots,\infty; j^2 = -1$) the right side of (IV), which is also the coefficient of $\underline{g}_H$(s) in (51), is unbounded, so ± $j\omega_\nu$ are the poles of the spacecraft. Consequently, for unboundedness to occur, the left side of (IV), expressed in terms of individual hinges-locked modes, yields the identity

$$\det\left[\underline{1} - \sum_{\alpha} (1 - \omega_{\alpha}^{c^2}/\omega_{\nu}^{2})^{-1}\, \underline{h}_{\alpha}^{cI}\, \underline{h}_{\alpha}^{cI^T}\, \underline{\mathcal{J}}^{-1}\right] = 0 \quad \text{(VI)}$$

where $\underline{h}_{\alpha}^{cI^T}$ is the α-th row of the matrix $\underline{h}_{A}^{cI}$. Similarly, when $s = \pm j\omega_{\alpha}^{c}(\alpha = 1,\ldots)$, the matrix within [•] on the left side of (IV), which is the coefficient of $\underline{\mathcal{J}}\,\ddot{\underline{\eta}}_{A}$ in (52), is unbounded, which implies that $\pm j\omega_{\alpha}^{c}$ are the zeros of the dynamics. Therefore, to realize unboundedness, the right side of (IV) bears forth

$$\det\left[\underline{1} + \sum_{\mu} (1 - \omega_{\mu}^{2}/\omega_{\alpha}^{c^2})^{-1}\, \underline{\mathcal{J}}\, \underline{\phi}_{\mu}\, \underline{\phi}_{\mu}^{T}\right] = 0 \quad \text{(VII)}$$

where $\underline{\phi}_{\mu}^{T}$ is the μ-th row of the matrix $\underline{\phi}_{A}$ [Eq. (28)]. Knowing the poles and zeros, the identity (IV), keeping in mind its lim, has this alternate form [cf. (Y) of Hughes[4]]: $s \to \infty$

$$\left[\underline{1} - \underline{h}_{A}^{cI^T}\underline{h}_{A}^{cI}\underline{\mathcal{J}}^{-1}\right]^{-1} \prod_{\alpha=1}^{n_f} (s^2+\omega_{\alpha}^{c^2}) \Bigg/ \left[\prod_{\mu=1}^{n_f} (s^2+\omega_{\mu}^{2})\right] =$$

$$\left[1 + \underline{\mathcal{J}}\ \underline{\phi}_{A}^{T}\, \underline{\phi}_{A}\right] \prod_{\alpha=1}^{n_f} (s^2+\omega_{\alpha}^{c^2}) \Bigg/ \left[\prod_{\mu=1}^{n_f} (s^2+\omega_{\mu}^{2})\right] \quad \text{(VIII)}$$

where n_f = total number of retained modes. Because of the identity (V), however, this form seems to be less useful than the form (IV). Following Garg[9], one can examine how far the identities (IV) or (VIII) are satisfied in the s-domain. The identities (VI) and (VII) are useful in several ways; for instance, known hinges-locked parameters can be used to determine hinges-free modal parameters, or vice versa, after Hughes and Garg[10]. Incidentally, the identities (VI) and (VII) are analogous to the identities $(M)_{\theta}$ and (Q) of Hughes[4]. As in Reference 4, under conditions of symmetry, these identities reduce to those concerned with individual articulation degrees of freedom. Owing to symmetry, since different sets of modes will contribute to different articulation degrees of freedom, the set α (α=1,...,∞) may form n_a subsets α_j(j=1,...,n_a) and each α_j will span the range 1,...,∞; the set μ (μ=1,...,∞) fragmentates likewise. The identities (VI) and (VII) then simplify to

$$\sum_{\alpha_{\ell}=1}^{\infty} (1 - \omega_{\alpha_{\ell}}^{c^2}/\omega_{\mu_{\ell}}^{2})^{-1} \left[\underline{h}_{\alpha}^{cI}\underline{h}_{\alpha}^{CI^T}\underline{\mathcal{J}}^{-1}\right]_{\ell,k} = \delta_{\ell k} \quad \text{(IX)}$$

$$\sum_{\mu_{\ell}=1}^{\infty} (\omega_{\mu_{\ell}}^{2}/\omega_{\alpha_{\ell}}^{c^2} - 1)^{-1} \left[\underline{\mathcal{J}}\ \underline{\phi}_{\mu}\underline{\phi}_{\mu}^{T}\right]_{\ell,k} = \delta_{\ell k}(\ell,k=1,\ldots,n_a) \quad \text{(X)}$$

V. ILLUSTRATION OF MODAL IDENTITIES AND DISCUSSION

The identities will now be illustrated for a four-body deformable spacecraft shown in Fig. 2. It has two flexible solar arrays, E_1 and E_2, each having one articulation degree of freedom about y_1- and y_2-axis, respectively, relative to the core body B_0, and a sensor having two rotational degrees of freedom about x_3- and y_3-axis. These four articulation angles are denoted θ_{1y}, θ_{2y}, θ_{3x}, and θ_{3y}, and the spacecraft thus has ten rigid modes. Hinges-free and hinges-locked vehicle modal data for the spacecraft were obtained by using NASTRAN. From a detailed finite element model having 19,434 degrees of freedom and 3,239 nodes, 63 hinges-free and 67 hinges-locked elastic modes below 25 Hz were computed. Since the vehicle is essentially symmetric (the sensor causes a slight asymmetry), both symmetric and antisymmetric vehicle modes arise in transverse bending and in-plane bending of the arrays, and the vehicle modes are categorized accordingly in Table 1 and Table 2. Fig. 3a confirms the prediction from the identity (III) that the hinges-free modal coefficients, in this case $\phi_{1\mu y}$ (μ=1,...,63) for the y_1-solar array, form a nonconverging series. In Fig. 3a, the largest modal coefficients $\phi_{1\mu y}$ for μ=8,11,18,28,... correspond to those vehicle modes which predominately entail torsion of the y_1-array about y_1-axis (Table 1). In contrast, those contributing to θ_{3y}, namely, $\phi_{3\mu y}$(μ=1,...,63), form essentially a convergent series because the sensor is rigid, and symmetric transverse bending of the arrays (Table 1) or local high-frequency deformation of B_0 at the sensor base produce $\phi_{3\mu y}$ (μ=1,...,63). The hinges-locked coupling coefficients $h_{1\alpha y}^{cI}$ for y_1-array and $h_{3\alpha y}^{cI}$ for θ_{3y} rotation for the modes α=1,...,67 are displayed in Fig. 4. Unlike $\phi_{1\mu y}$, $h_{1\alpha y}^{cI}$ forms a converging series.

The identities (III) and (V) are the simplest, for they involve only modal coefficients, no frequencies. The identity (III) is illustrated in Fig. 5. The error indexes e_{kk}^{HF} (k=1, 3) (HF means hinges-free) are the corresponding diagonal elements of the (4x4) matrix $\left[\underline{1} + \underline{\mathcal{J}}\ \underline{\phi}_{A}^{T}\underline{\phi}_{A}\right]^{-1}$. In contrast to their zero ideal value, the asymptotes

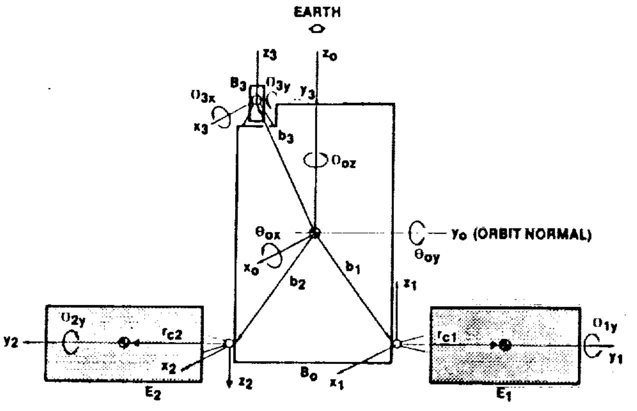


Figure 2. A Four-Body Deformable Spacecraft

ORIGINAL PAGE IS OF POOR QUALITY

Table 1. Hinges-Free Modes

Characteristics of the Mode		Mode No.	Affected Rotational Degrees of Freedom
Transverse bending of arrays and A-frames	Symmetric	1, 5, 9, 13, 15, 21, ...	θ_{oy}, θ_{3y}
	Antisymmetric	2, 6, 10, 14, 16, 22, ...	θ_{oz}
Torsion of arrays and A-frames	Array 1	8, 11, 18, 28, ...	θ_{1y}
	Array 2	7, 12, 19, 29, ...	θ_{2y}
In-plane bending of the A-frames and solar arrays	Symmetric	3, ...	None
	Antisymmetric	4, 17, 20, ...	θ_{ox}, θ_{3x}

Table 2. Hinges-Locked Modes

Characteristics of the Mode		Mode No.	Affected Rotational Degrees of Freedom†
Transverse bending of arrays and A-Frames	Symmetric	1, 2, 6, 8, 12, 16	θ_{oy}
		1, 2, ...	θ_{3y}
	Antisymmetric	3, 9, 13, 17, 20, 27, ...	θ_{oz}
Torsion	Array 1	5, 6, 10, 11, 14, 15, 22, 23, ...	θ_{1y}
In-plane bending of arrays and A-frames	Antisymmetric	7, 18, 21, ...	θ_{ox}
		18, ...	θ_{3y}
Vibrations of the spacecraft		28, 35, 36, 37, 41, 42, 43, 44, 47, 48, 49, 53, ...	θ_{ox}
		28, 36, 37, 41, 42, 43, 44, 45, 47, 48, 49, 53, ...	θ_{oy}
		28, 35, 36, 37, 38, 41, 42, 43, 44, 47, 48, 53, ...	θ_{oz}
†Information about the interaction with θ_{2y} and θ_{3x} not available			

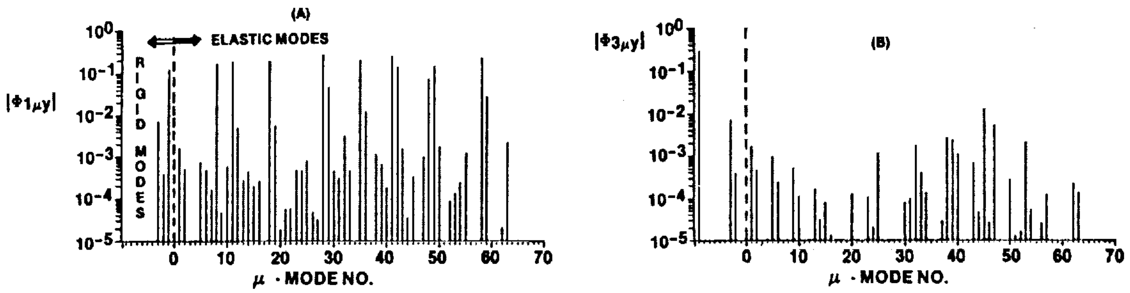


Figure 3. Hinges-Free Modal Coefficients of Articulation Motion of y_1- Solar Array, $|\phi_{1\mu y}|$, and of Sensor B_3, $|\phi_{3\mu y}|$, about y_3-Axis

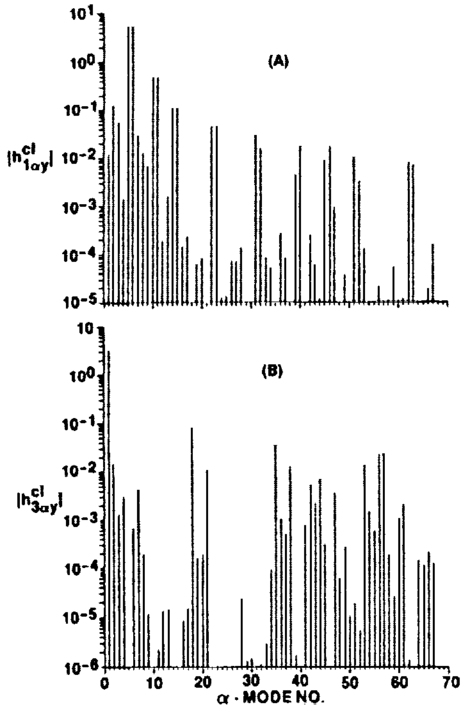


Figure 4. Hinges-Locked Modal Coefficients Associated with the Articulation Motion of y_1-Solar Array, $|h^{cI}_{1\alpha y}|$, and of the Sensor about y_3-Axis, $|h^{cI}_{3\alpha y}|$

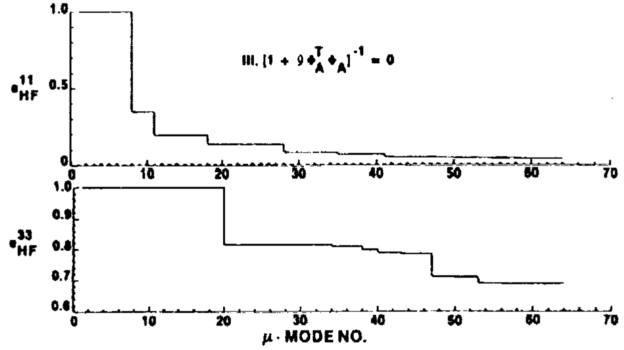


Figure 5. Hinges-Free Identify III: Diminishing of Error Index with Hinges-Free Modes μ

of these indexes are 0.0406 for k=1 (θ_{1y} rotation of the y_1-array) and 0.689 for k=3 (θ_{3x}, the x-rotation of the sensor). The error indexes diminish discretely at appropriate modes as predicted by Table 1. For instance, for θ_{1y}, the error index e^{HF}_{11} diminishes at the torsional modes μ=8,11,18,28,.... The index for the y_2-array motion (k=2) is the same as that for k=1, except that it decreases instead at the adjacent torsional modes μ=7,12,19,29,... (see Table 1). Surprisingly, the asymptote of the error index for θ_{3y} (k=4), not included in Fig. 5, hovers at 0.9976 instead of decreasing to the ideal value zero. Fig. 6 illustrates the hinges-locked identity (V), rearranged as $\underline{h}^{cI^T}_A \underline{h}^{cI}_A \underline{\mathcal{G}}^{-1} = \underline{1}$. For discussing this identity and the ones following, define a "completeness index $\mathcal{C}$ " which approaches unity for an error-free model [Reference 1]. [The completeness index for Fig. 5 is $(1 - e^{HF}_{kk})$.] The growth of the diagonal elements (1,1) and (4,4) of the matrix $\underline{h}^{cI^T}_A \underline{h}^{cI}_A \underline{\mathcal{G}}^{-1}$ are depicted in Fig. 6. Surprisingly, alongside the error index e^{HF}_{11} in Fig. 5a, $\mathcal{C}^{HL}_{11}$(HL means hinges-locked) approaches unity in just two hinges-locked torsional modes, 5 and 6 (Table 2), and its asymptotic value is 1.029. Furthermore, by contrast with the hinges-free completeness index $\mathcal{C}^{HF}_{44}$ equal to 0.0024 (that is, the above mentioned error index e^{HF}_{44} of 0.9976), the hinges-locked completeness index $\mathcal{C}^{HL}_{44}$ = 0.9421 in Fig. 6b is remarkable; in fact, the first hinges-locked mode, a symmetric transverse bending mode of the arrays (Table 2), contributes a mighty share, 0.9412, to $\mathcal{C}^{HL}_{44}$.

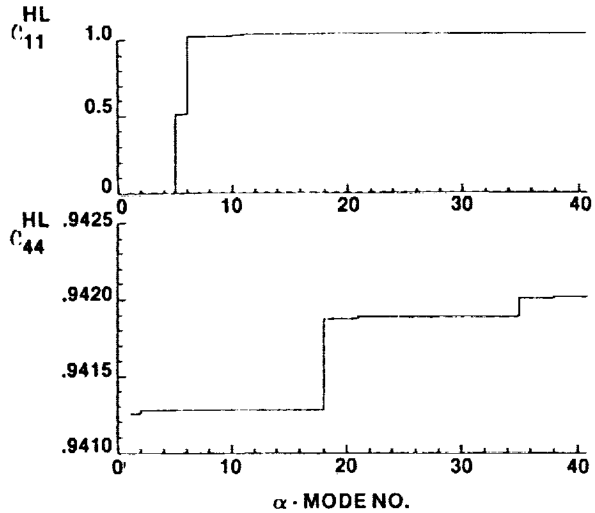


Figure 6. Hinges-Locked Identify V: Growth of Completeness Index with Hinges-Locked Modes α

The identities which involve frequencies as well are now illustrated. First, consider the identity (VII) which is summed over all hinges-free modes (μ=1,...,63) for a specific ω^c_α. When ω_μ and ω^c_α are the same to several decimal places, it is difficult to verify this identity in this form. On the other hand, the identity (VIII) indicates that when ω_μ and ω^c_α are truly the same, the corresponding poles and zeros cancel each other without affecting the articulation dynamics. A physical explanation of this is that when hinges-free and hinges-locked frequencies are truly equal, that particular mode does not contribute to the articulation motion, so such a mode may be deleted from the study. In numerical work, however, it is difficult to establish true equality between two real numbers. Besides, as will be seen shortly, for the example in hand, sometimes even though ω_μ and ω^c_α are the same up to three or four decimal places, the minuscule difference between the two is still important for the verification of an identity. Consequently, the following results are obtained without truncating either modal set. Returning to the identity VII, one finds that when $\alpha > 9$, hinges-locked frequencies ω^c_α are so close to a corresponding hinges-free frequency ω_μ that the determinant, instead of being zero, becomes an arbitrarily

ORIGINAL PAGE IS
OF POOR QUALITY

large number. Among α=1,..., 9, the identity (VII) is best satisfied with α=7 and next best with α=2, for which the determinants are, respectively, -0.00415 and 0.04732 (Table 3). The circumstances which produce these results are revealed by the identity (X). For a given ω_α^c, when all available hinges-free modes are added to calculate the (ℓ,k) element of the left side of (X), it is denoted $\mathcal{C}^{HF}_{\ell k,asy}$ where "asy" means asymptotic value. Fig. 7 shows $\mathcal{C}^{HF}_{\ell\ell,asy}$ for ℓ=1,3, and 4. The ideal value of this index is unity; however, when $\omega_{\mu_\ell} \approx \omega^c_{\alpha_\ell}$ for some u and α, this index assumes an arbitrarily large value, and for plotting purposes, such large numbers are replaced by 2 without altering their signs. In the left side of Fig. 7.a, in the useful range 0 to 1, the maximum value of $\mathcal{C}^{HF}_{11,asy}$ concerning the y_1-array rotation, θ_{1y}, is 0.451 for the hinges-locked mode α=2-- a symmetric transverse bending mode of the arrays (Table 2). On the other hand, the first torsional hinges-locked mode having significant coupling with the rotation θ_{1y} is α=5 (Table 2 and Fig. 4), but $\mathcal{C}^{HF}_{11,asy}$ corresponding to α=5 is 0.16, less than 0.451 for α=2. Although the index $\mathcal{C}^{HF}_{11,asy}$ for α=5 should be, intuitively, greater than that for α=2, this does not happen because the hinges-free frequency ω_1 (0.25724 Hz) is close to ω^c_2 (0.25719 Hz). To determine the contribution of the mode μ=1, the growth of $\mathcal{C}^{HF}_{11}$ with successive addition of μ to the asymptotic value 0.451 for α=2 is shown in the right side of Fig. 7.a. $\mathcal{C}^{HF}_{11}$ is found to escalate discretely at μ=1,8,11,18,28,29,35,41,42,..., which, except for μ=1, involve torsion of the array 1 (Table 1). The contribution from the hinges-free mode μ=1, a symmetric transverse bending mode of the arrays (Table 1) like α=2 hinges-locked mode, is however, extraordinarily large: 93%. Nevertheless, the bending mode μ=1 is not pertinent to the articulation motion θ_{1y}, so $\mathcal{C}^{HF}_{11} = 0.451$ for α=2 cannot be accepted, and, instead, $\mathcal{C}^{HF}_{11} = 0.16$ for α=5, a torsional mode is accepted. Next consider the sensor motion θ_{3x}. The corresponding index, $\mathcal{C}^{HF}_{33,asy}$, shown on the left side of Fig. 7.b, is 1.0059 for α=7 (compare with $\mathcal{C}^{HF}_{11,asy}$, and recall from Table 3 the value 0.00415 of the identity VII for α=7). The growth of $\mathcal{C}^{HF}_{33}$ versus μ for α=7 is displayed on the right side of Fig. 7.b, where it is observed to become unity at once when μ=4. To understand this, note that both μ=4 and α=7 modes involve antisymmetric in-plane bending of the arrays--a motion which induces θ_{3x} (see Table 1 and Table 2), and that $\omega_4 = 0.59541$ and $\omega^c_7 = 0.59538$ Hz. When the rotation θ_{3x} of the rigid sensor is locked, the moment of inertia which must be turned by the antisymmetric in-plane bending, is increased, and that lowers the frequency commensurately. The ratio of the moment of inertia of the sensor and of the core body, both about x_0-axis, is 0.0717. The decrement of 3.0E-5 Hz noted above in the frequency ω_4 is mathematically so precise that $\mathcal{C}^{HF}_{33}$ becomes unity at once when α=7. Moreover, although ω_4 and ω^c_7 are the same up to three decimal places, the two modes cannot be truncated from the study of the verification of the identities VII and X. Next, consider θ_{3y} rotation of the sensor--the rotation coupled with the transverse symmetric bending of the arrays (Table 1 and Table 2). The associated index, $\mathcal{C}^{HF}_{44,asy}$, versus α is shown in Fig. 7c. In the range 0 to 1, the most it becomes is a startling low value: 0.07836 for α=2; for this α, the growth of $\mathcal{C}^{HF}_{44}$ with μ indicates that 99.99% contribution arises from the first symmetric transverse bending mode μ=1.

The verification of the identity (IX) for ℓ=k=1 and 4 is considered in Fig. 8. Since this identity relates to hinges-locked modal parameters, its left side is denoted $\mathcal{C}^{HL}_{\ell k}$. Earlier, the identity (V) and Fig. 4 established that the hinges-locked coupling coefficients form a converging series. Therefore, the determinant identity (VI) and $\mathcal{C}^{HL}_{\ell k,asy}$ in Fig. 8 do not become arbitrarily large numbers once μ ≥ 28. Indeed, only for μ=3,4,21,26,27, is the index $\mathcal{C}^{HL}_{\ell k,asy}$ unbounded, by contrast with the hinges-free index $\mathcal{C}^{HF}_{11,asy}$ in Fig. 7a which is unbounded for all α ≥9. The index $\mathcal{C}^{HL}_{\ell\ell}$ depends on the selected hinges-free frequency ω_μ; for α's having $\omega^c_\alpha > \omega_\mu$, the term $\left(1 - \omega^{c\,2}_\alpha / \omega^2_\mu\right)$ becomes negative and these particular hinges-locked modes diminish the sum. Focusing first on $\mathcal{C}^{HL}_{11,asy}$, surprisingly, it stabilizes early on to 1.05 when μ=7 or 8--the first two hinges-free torsional modes. The ascent of $\mathcal{C}^{HL}_{11}$ to 1.05 for μ=8 with hinges-locked modes α (Fig. 8a) indicates significant contributions from α=5,6,10, and 11--all torsional modes (Table 2); the contribution from higher torsional modes attenuates rapidly because of the fast convergence of $\underline{h}^{cI}_{1\alpha y}$. As for the rotation θ_{3y}, the maximum value of $\mathcal{C}^{HL}_{44,asy}$, displayed in Fig. 8b, in the range 0 to 1 is 0.959 when μ=5--the second hinges-free symmetric transverse bending mode of the arrays (Table 1). The growth pattern of $\mathcal{C}^{HL}_{44}$ versus α for μ=5, also shown in Fig. 8b, states that virtually the entire contribution arises from the first hinges-locked mode (α=1) involving symmetric transverse bending of the arrays.

VI. SUMMING UP

To draw conclusions about the relative merits of hinges-free and hinges-locked vehicle modes, Table 4 summarizes the completeness indexes for the identities (III), (V), (IX), and (X). Evidently, the hinges-locked indexes are far closer to unity than the hinges-free indexes. The superiority of the hinges-locked vehicle modes to the hinges-free modes is established most persuasively by comparing the indexes for the articulation motion θ_{3y} of the sensor: $\mathcal{C}^{HF}_{44,asy}$ are 0.0024 and 0.0784--far remote from unity, whereas $\mathcal{C}^{HL}_{44,asy}$ are 0.9421 and 0.9593--almost unity. It must be understood, nevertheless, that the identities (X) and (IX) (or VII and VI)

Table 3. Identity VII: Variation of the Hinges-Free Determinant With Hinges-Locked Modes; Ideal Value = 0

α	1	2	3	4	5	6	7	8	9
VII	0.97105	0.04732	0.27012	0.79457	0.68305	0.68009	-0.00415	0.19551	-0.32426
Minimum value of the determinant among those for $\alpha = 10, \ldots, 63$, is 74.4, and the maximum value is ∞ when $\omega_\mu = \omega_\alpha$ up to several decimal places									

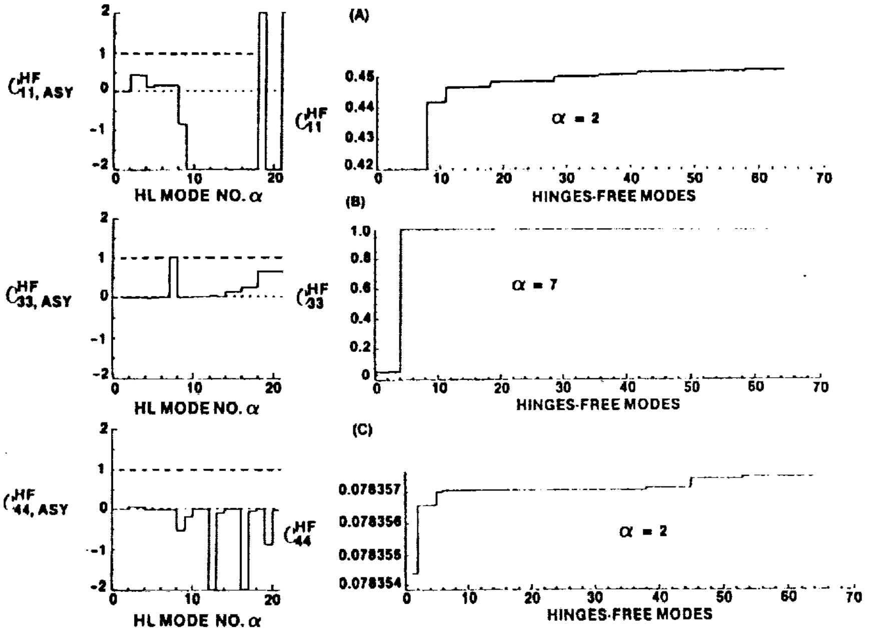


Figure 7. Identity X: Asymptotic Values of the Hinges-Free (HF) Completeness Indexes Versus Hinges-Locked (HL) Mode α, and Growth of this Index Versus Hinges-Free Modes

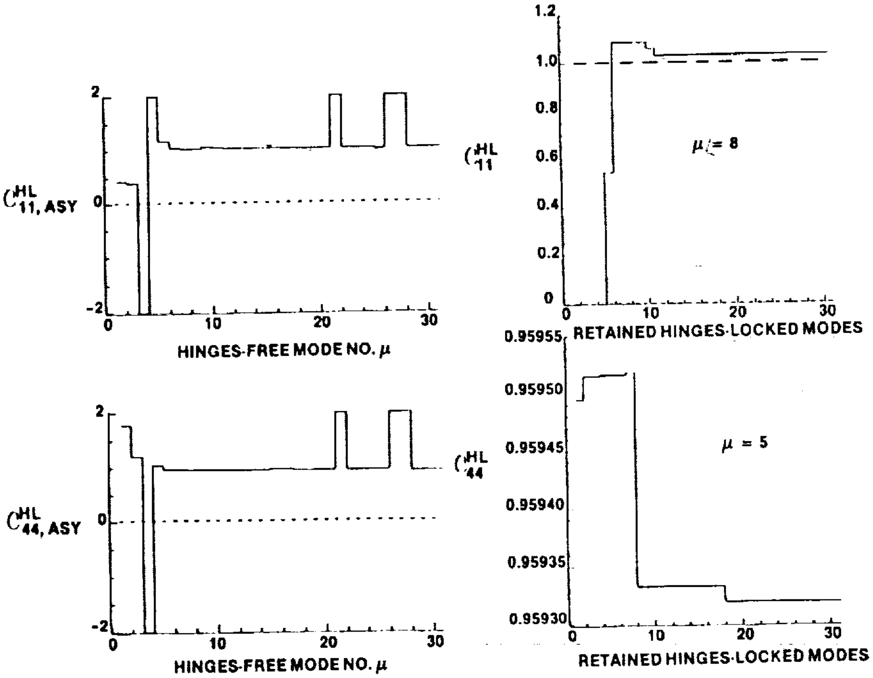


Figure 8. Identity IX: Asymptotic Values of the Hinges-Locked Completeness Indexes Versus Hinges-Free Modes, and Growth of this Index with Hinges-Locked Modes

Table 4. A Summary of the Completeness Indexes for Hinges-Free and Hinges-Locked Vehicle Modes; Ideal Value = 1

Hinges-Free (HF) Indexes	Identity III $1-e^{HF}_{\ell\ell,asy}$	Identity X	Hinges-Locked (HL) Indexes	Identity V	Identity IX	Articulation Motion	Associated Mode of Deformation
$\mathcal{C}^{HF}_{11,asy}$	0.9594	0.160	$\mathcal{C}^{HL}_{11,asy}$	1.029	1.05	θ_{1y}	Torsion
$\mathcal{C}^{HF}_{33,asy}$	0.311	1.0059	$\mathcal{C}^{HL}_{33,asy}$	TBD†	TBD†	θ_{3x}	Antisymmetric in-plane bending
$\mathcal{C}^{HF}_{44,asy}$	0.0024	0.0784	$\mathcal{C}^{HL}_{44,asy}$	0.9421	0.9593	θ_{3y}	Symmetric transverse bending

†to be determined

represent two different situations: in the former, the hinges-free modes are employed to yield a bounded response at a hinges-locked frequency; and in the latter, the hinges-locked modes are used to elicit an unbounded response at a hinges-free frequency. Therefore, a comparison of the indexes from these identities is slightly inappropriate perhaps; yet the conclusion from Table 2 seems inevitable that the hinges-locked vehicle modes yield a much more accurate model for simulation than the hinges-free vehicle modes do. This is caused by the nonconvergence of the hinges-free modal coefficients in contrast with the rapid convergence of the hinges-locked coupling coefficients--the attributes corroborated by the identities. Besides contrasting one family of modes with the other, the identities are clearly useful in sifting through scores of finite-element generated modes to select a few pertinent modes for an articulation degree of freedom in consideration. An important extension of the preceding work is to devise identities which involve modal coefficients and frequencies of only one family of modes, hinges-free or hinges-locked, not both. Hughes[11] has formulated such identities for an elastic body with no articulated members.

ACKNOWLEDGEMENTS

The numerical results in this paper were obtained by Mr. Thomas C. Witham, Avionics Systems Group. The author thanks him with much pleasure for his painstaking and conscientious efforts.

REFERENCES

1. Hablani, H.B., "Constrained and Unconstrained Modes, Some Modeling Aspects of Flexible Spacecraft," Journal of Guidance and Control, Vol. 5, No. 2, March-April, 1982, pp. 164-173.

2. Hughes, P.C., Dynamics of Flexible Spacecraft, UCLA Lecture Notes (1982).

3. Hablani, H.B., "Hinges-Free and Hinges-Locked Modes of a Deformable Multibody Spacecraft--A Continuum Analysis," AIAA 87-0925 CP, Proceedings of the AIAA Dynamics Specialist Conference, Monterey, California, April 1987, Part 2B, pp. 753-768.

4. Hughes, P.C., "Modal Identities for Elastic Bodies with Application to Vehicle Dynamics and Control," Transactions of ASME, Journal of Applied Mechanics, Vol. 47, March 1980, pp. 177-184.

5. Ho, J.Y.L., "Direct Path Method for Flexible Multibody Spacecraft Dynamics," Journal of Spacecraft and Rockets, Vol. 14, No. 2, February 1977, pp. 102-110.

6. Hughes, P.C., Spacecraft Attitude Dynamics, Section 3.6, John Wiley and Sons, New York, 1986, pp. 70-76.

7. Hughes, P.C., "Dynamics of a Chain of Flexible Bodies," The Journal of the Astronautical Sciences, Vol. 27, No. 4, October-December 1979, pp. 359-380.

8. Singh, R.P., VanderVoort, R.J., and Likins, P.W., "Dynamics of Flexible Bodies in Tree Topology--A Computer-Oriented Approach," AIAA Paper No. 84-1024, pp. 327-337.

9. Garg, S.C., "Frequency-Domain Analysis of Flexible Spacecraft," AIAA Journal of Guidance, Control, and Dynamics, Vol. 5, No. 1, January-February 1982, pp. 54-59.

10. Hughes, P.C., and Garg, S.C., Dynamics of Large Flexible Solar Arrays and Applications to Spacecraft Attitude Control System Design, Institute for Aerospace Studies, University of Toronto, UTIAS Report 179, February 1973.

11. Hughes, P.C., "Space Structure Vibration Modes: How many exist? Which ones are important?," Proceedings of the Workshop on Applications of Distributed System Theory to the Control of Large Space Structures, JPL Publication 83-46, July 1983, pp. 31-47.

Issues in CSI Analysis for Large Scale Systems

Paul Blelloch
SDRC

Abstract

Future spacecraft such as the International Space Station result in flexible models with hundreds, or perhaps thousands of modes in a frequency range where the potential for control/structure interaction exists. This provides the analyst with a formidable model reduction problem at both the component and the system level. Approaches to dealing with this problem, including issues associated with using normal modes as a structural representation, applicability of alternate structural representations and algorithms for selecting "important" modal degrees of freedom at both the component and the system level will be discussed. Practical implementation of these techniques on a large scale model of the Space Station will be presented.

C-5

N90-23077

STRUCTURAL MODELING FOR CONTROL DESIGN
(Articulated Multibody Component Representation)

by
E. D. Haugse, R. E. Jones, and W. L. Salus
Boeing Aerospace and Electronics
Seattle, Washington

Abstract

High gain, high frequency flexible responses in gimbaled multibody systems are discussed. Their origin and physical significance are described in terms of detailed mass and stiffness modeling at actuator/sensor interfaces. Guyan Reduction, Generalized Dynamic Reduction, inadequate mass modeling detail, as well as system mode truncation, are shown to suppress the high gain high frequency response and thereby lose system flexibility important for stability and performance predictions. Model validation by modal survey testing is shown to risk similar loss of accuracy. Difficulties caused by high frequency responses in component mode simulations, such as DISCOS, and also linearized system mode simulations, are described, and approaches for handling these difficulties are discussed.

Introduction

The control-structure-interaction problem is concerned with locally applied input forces or torques and localized outputs at actuator/sensor interfaces. These localized inputs and outputs are usually modeled as occurring at single points or sections of structural members. This creates difficulties in regard to dynamic modeling, and careful flexibility and mass modeling at local input/output locations is required to accurately predict dynamic response. Local flexibility is often important because of mechanical details associated with actuators, sensors, and their mounting hardware. The frequencies of vibration at which the dynamic responses occur, characterizing the local flexibilities, depend on both the local flexibility and the mass modeling at the actuator/sensor interfaces. The mass and inertia at the interfaces are difficult to quantify, particularly for rotational degrees of freedom, and are often not done explicitly.

Typically the structural engineer will deliver Craig-Bampton component mode models, fixed at the actuator/sensor interfaces of a servo-mechanism, to the controls engineer. The controls engineer will then merge the component models, freeing the degrees of freedom associated with the control force or torque, and perform a system level simulation. This often results in transfer functions with high gain responses occurring at unexpected high frequencies. These high gain high frequency (HGHF) responses are due to vibration modes associated with small local interface mass and inertia connected to relatively stiff primary structure by flexible elements representing servo mechanisms and fastener schemes. If HGHF responses do not occur the modeling may be suspect for control-structure-interaction simulations.

This paper will address important aspects of flexibility and mass/inertia modeling of structure for the controls problem as they influence solvability of the equations of motion, accuracy of control-structure-interaction analyses, and testing procedures.

Local Flexibility and Mass Modeling

Multibody structures are connected by servo mechanisms that control the relative angle or displacement between one flexible body and another. The servo mounting hardware and its internal parts may cause significant loss of stiffness across the controlled joint. This is illustrated by the hardware schematic in Figure 1 in which a motor is shown attached to a bracket which is in turn attached to body A. The attachment scheme may use fasteners which add to the flexibility of the bracket.

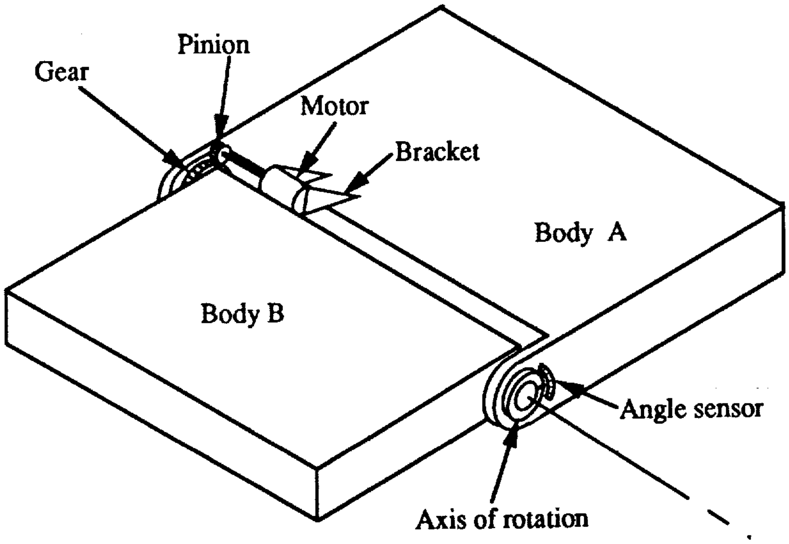
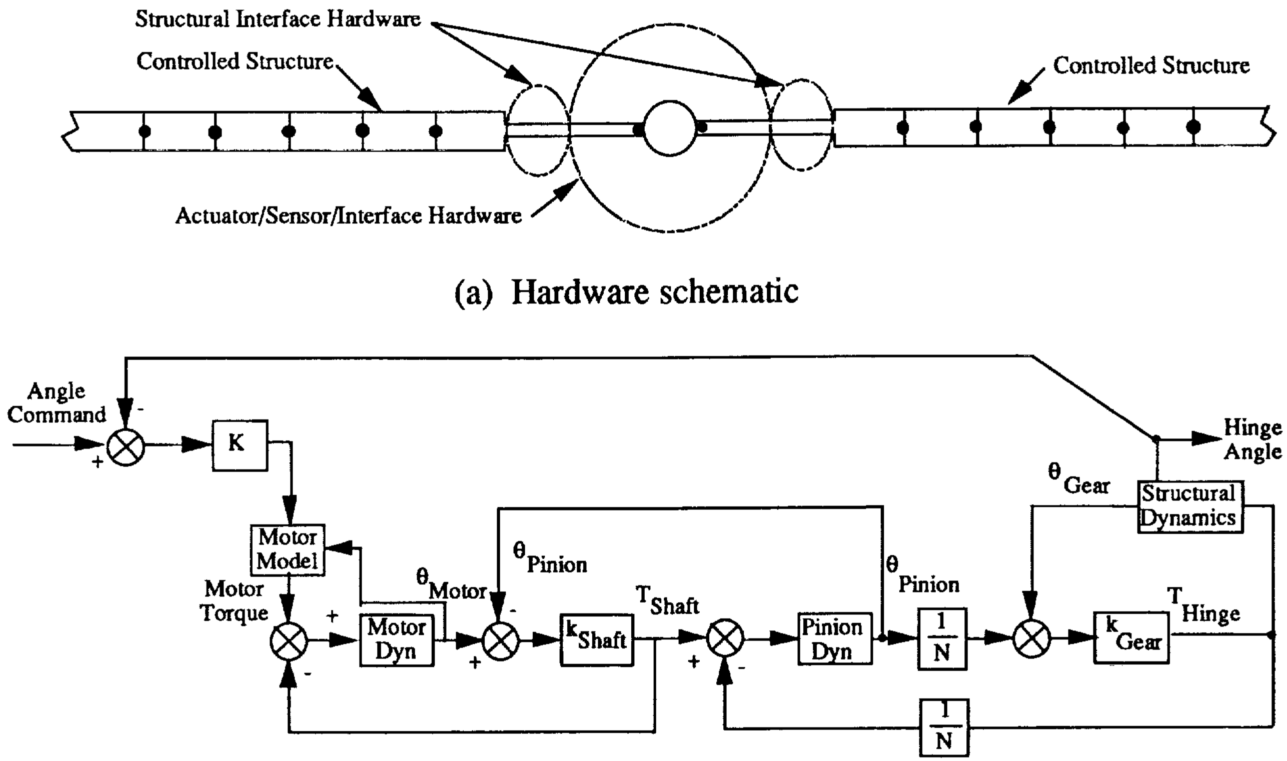


Figure 1. Hardware Schematic

Body B is connected to A through the gear contact forces, the pinion shaft, and some type of spline detail not shown on the figure. These are also sources of flexibility. This type of hardware can only be modeled accurately by a cooperative effort of structures and controls engineers. Some of the flexibility and mass will be modeled by the controls engineer, and some by the structures engineer. Therefore, modeling responsibilities need to be well defined to avoid exclusion or duplication of flexibility and mass/inertia. The structural engineer should understand the system block diagrams that the controls engineer will use in analysis. Similarly, the controls engineer needs to understand the types of structural modeling approximations which would reduce the accuracy of the analysis. To illustrate these points a simplified block diagram is shown in Figure 2.

In this example the controls engineer is responsible for modeling the motor shaft, gears, and the motor itself. The structures engineer is responsible for everything else including the motor bracket and fasteners. Body A should be supported at the motor rotor for modal analysis, and body B should be supported at the gear, thereby including gear spline flexibility effects. The block diagram shows that the driving torque at the hinge is determined from the difference between the rotations in two instances: the motor and the pinion; and the pinion and the gear. Each relative rotation is assigned a flexibility. This suggests that accuracy in computing local contributions to rotation is important. The block diagram illustrates the complexity of actuator-to-structure modeling and the degree to which the structures and controls engineers need to cooperatively build the dynamics simulation.



(a) Hardware schematic

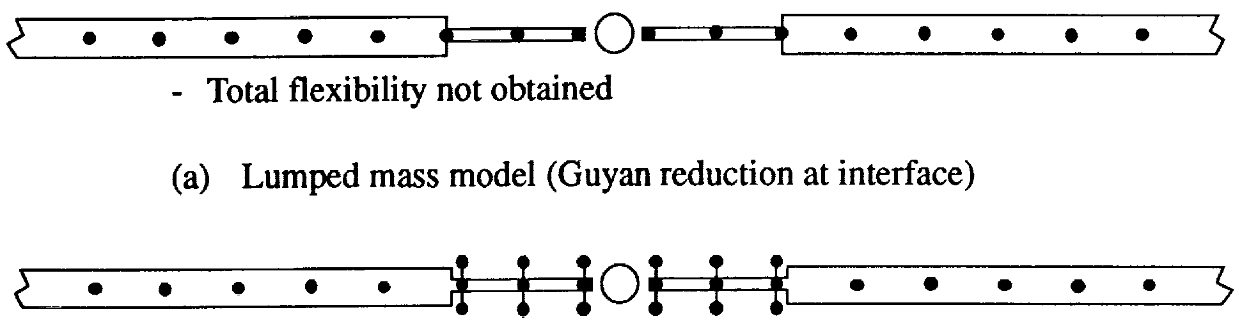
(b) Simplified control - structure - interaction block diagram

Figure 2. Control - Structure Modeling

Local flexibility modeling may not suffice to provide the accuracy needed at actuator/sensor interfaces. Mass and inertia modeling are required to enable this flexibility in modal dynamics models. Since many actuators apply torque and sensors resolve angular motions, it is necessary to treat both rotational flexibilities and inertias very carefully, an area that many structural dynamicists do not pursue in detail when developing models. For example, adding any local inertia will enable the local flexibility, but may not properly define its frequency spectrum.

The vibration modes representing a large portion of the local flexibility often occur at very high frequencies. The high frequencies result from a small inertia supported by local flexibility. Figure 3 attempts to illustrate this.

In Figure 3a the model uses only lumped masses, while the model in Figure 3b includes inertias as well (at least at the hinged interface). The finite elements are intended to be small, aiming at an accurate modeling of local flexibility. Therefore, the masses are very small, and in Figure 3b the inertias are very small also. In the first case the modal analysis requires elimination of the rotational freedom at the hinge. This is done by Guyan reduction, and as a result the effect of the local flexibility is lost. In the second case the local inertias enable the local flexibility, all of which is present in the modal analysis. The small inertias, undergoing rotational modal motions supported by the attached elastic elements, will have very high frequencies. The actual values of these modal frequencies will depend on both the inertia values and the flexibility values of the model at the actuator interface. It may be worth noting that if the lumped mass model is solved for vibration modes without Guyan reduction, numerical error (round-off) may enable the local flexibility, producing extremely high eigenvalues. The accuracy of this numerical process is questionable, especially for controls applications, because it may affect the placement of low frequency transfer function zeros.



- Total flexibility not obtained

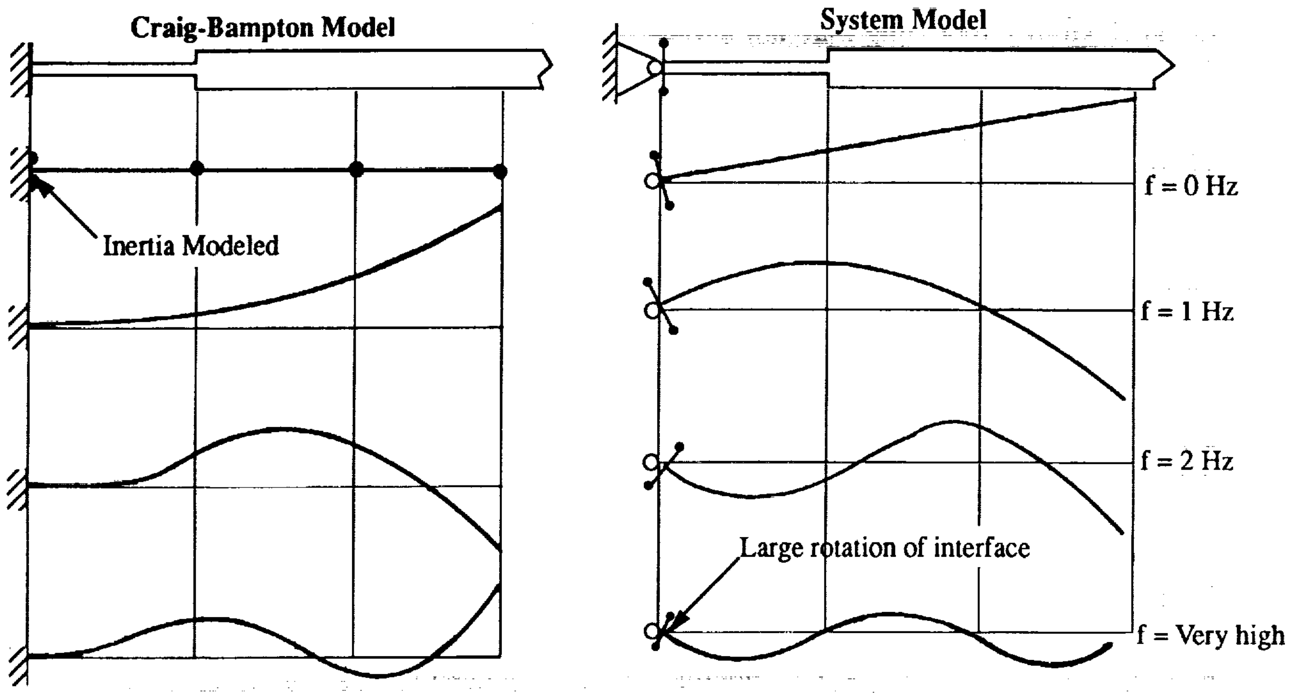
(a) Lumped mass model (Guyan reduction at interface)

- Total flexibility obtained

(b) Model with inertias

Figure 3. Comparison of Models With and Without Inertia

It is reasonable to question how, starting with a Craig-Bampton model having frequencies to, for example, 30 Hz, it is possible to obtain a merged (gimbal-free) model with a highest frequency of perhaps 1000 Hz. The following brief discussion attempts to make this physically plausible. At the component level the Craig-Bampton model contains interface matrices and deformation shapes that fully capture the interface flexibility. When the Craig-Bampton model is used for system level analysis its hinge rotation is made free, and one extra mode, the rigid rotation, is added to the modal set. There are still three flexible modes, however. This is shown in Figure 4.



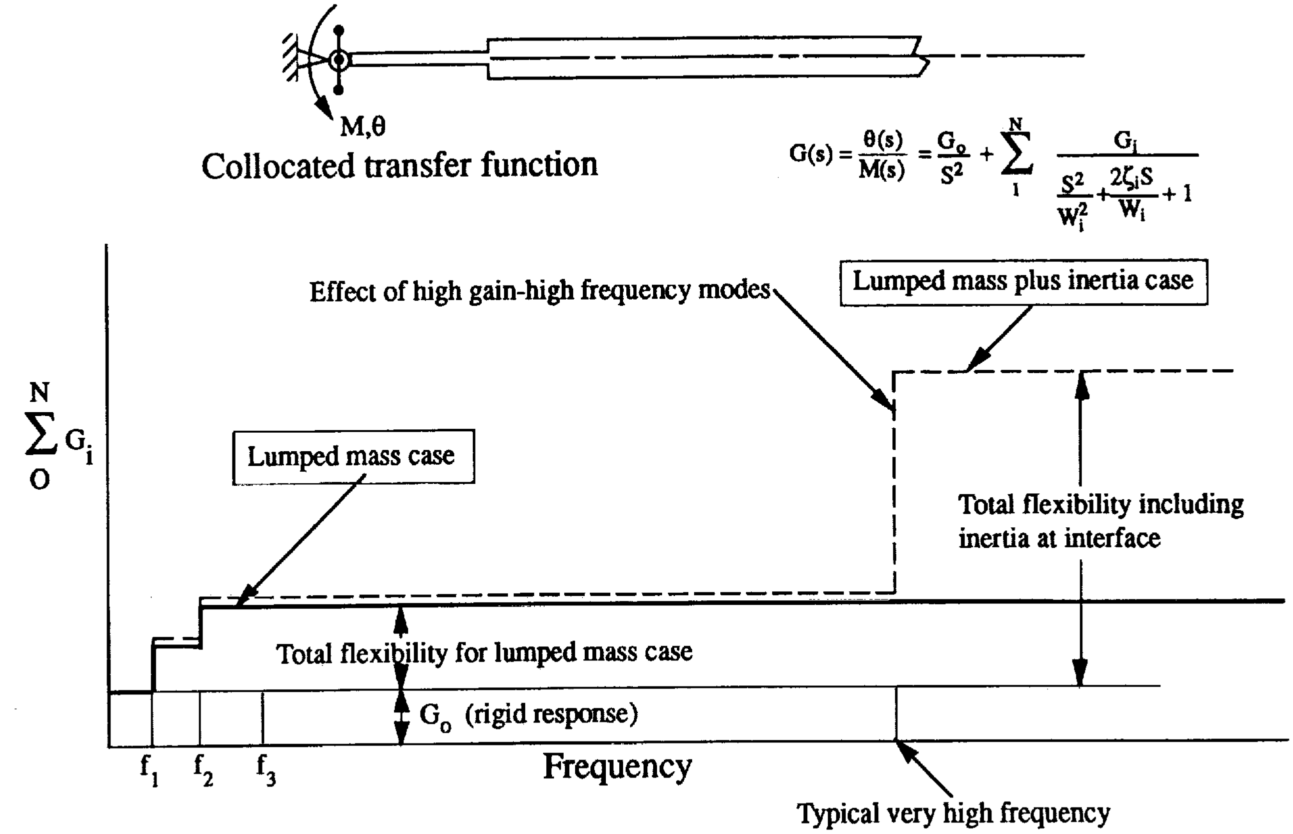
- Total flexibility is present - Flexibility captured by high frequency mode

Figure 4. Cantilever and Pinned Modes of Beam with End Inertia

The rigid and first two flexible modes are inertially dominated by the lumped masses and are orthogonal with respect to these masses with only a very small influence from the inertia at the left end. The highest mode has no "spatial room" to be orthogonal to the first three modes on the basis of the lumped masses, because the "spatial room" has been fully used by the first three modes (three modes : three masses). Therefore, the highest mode must have small displacements of the lumped masses, using them as merely a fine adjustment, and consist mainly of the rotation of the inertia at the left end. This is illustrated by the figure, from which it is clear why the highest modal frequency is so large: its modal mass is derived almost entirely from the local inertia alone, and is very small.

Effect of HGHF Response on Transfer Functions

If the local flexibility at an actuator/sensor interface is large relative to the rest of the structure, and the mass/inertia placed at that interface is very small, the collocated transfer function calculated at the interface will have not only high frequency, but also high gain at high frequency. A large portion of the total flexibility seen directly by the actuator is represented by a high frequency mode of vibration. Change in the mass/inertia affects the frequency of the high gain response but has little effect on the gain itself. A useful plot to illustrate this is one that shows the running sum of modal gains versus frequency, and is shown in Figure 5.



M,θ
Collocated transfer function
$G(s) = \frac{\theta(s)}{M(s)} = \frac{G_o}{S^2} + \sum_{1}^{N} \frac{G_i}{\frac{S^2}{W_i^2} + \frac{2\zeta_i S}{W_i} + 1}$
$\sum_{O}^{N} G_i$
Effect of high gain-high frequency modes
Lumped mass plus inertia case
Lumped mass case
Total flexibility including inertia at interface
Total flexibility for lumped mass case
G_0 (rigid response)
f_1
f_2
f_3
Frequency
Typical very high frequency

Figure 5. Gain Summation Plot for Pinned-Free Beam

This figure illustrates the effects on transfer functions of local inertia in the structural model. Such a plot typically shows very large gain at high frequency. Omitting the local inertia (with Guyan reduction), or equivalently, truncating the high frequency modes, or equivalently, performing the modal analysis by Generalized Dynamic Reduction, eliminates this gain, resulting in inaccurate control response predictions at all frequencies. This type of plot is a simple check that should always be made for structural model transfer functions to evaluate the presence and importance of HGHF responses associated with a particular hardware application and transfer function. If HGHF modes do occur, their accuracy and effect on control response must be studied. If they do not, either the structural design is very efficient and well adapted to the controls application, or the structural model is deficient for control-structure-interaction simulations (ie. local flexibility may be missing). If HGHF modes are present their effects are important in low frequency as well as high frequency dynamics predictions. This has different consequences for component mode and system mode formulations, as will be discussed later.

The Bode plot in Figure 6 illustrates typical low frequency effects of the presence of HGHF responses for a single-input-single-output (SISO) collocated transfer function.

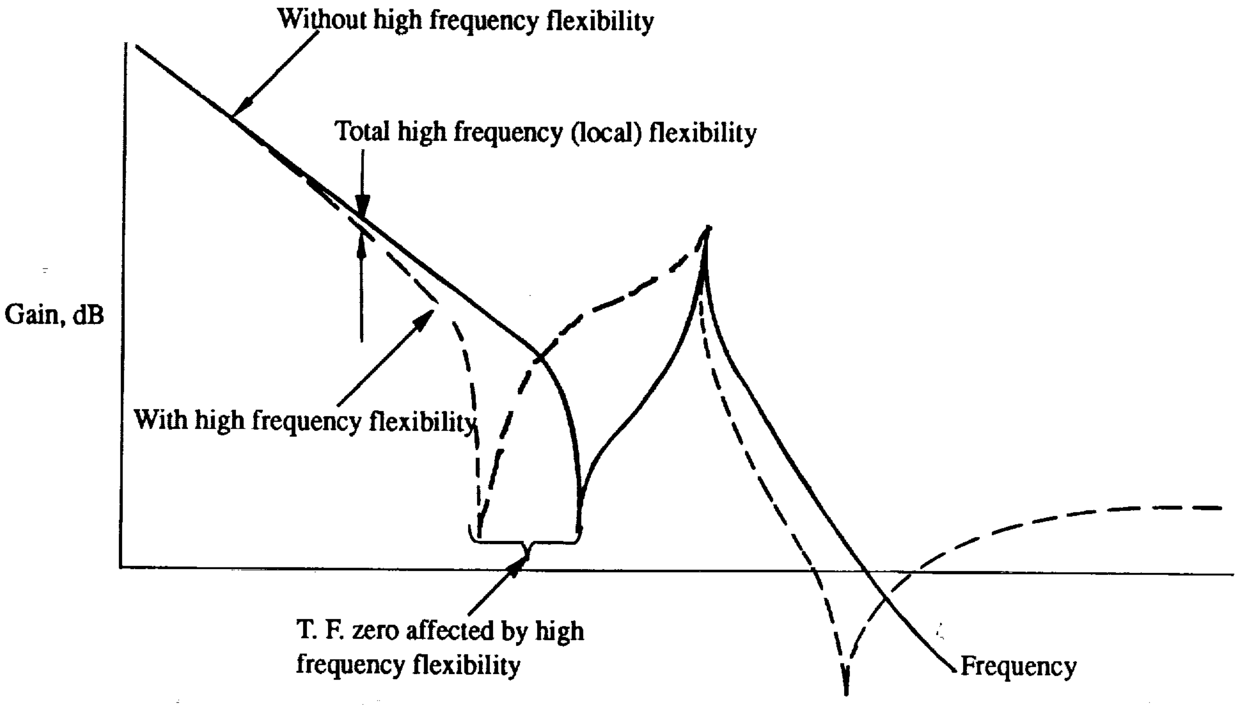


Figure 6. Effect of High Frequency Flexibility on Bode Gain

The effects derive from the placement of the low frequency transfer function zero. Bode plots including high frequency flexibility show that all of the transfer function zeros, particularly the lowest ones, are moved to lower frequencies. In addition, a transfer function zero is produced above the highest retained mode. The low frequency gain is reduced, resulting in loss of agility. The gain is increased between the lowest zero and pole, a region where stability may be in question because of compensator rolloff and phase. Above the first pole, gain is slightly reduced, improving flexible mode stability. Finally, the high frequency flexibility eliminates the structural rolloff customarily seen in truncated modal models. This figure and brief discussion suggest that reliable control-structure-interaction studies require careful attention to local modeling and some means of retaining HGHF responses. Various solution procedure options that effectively eliminate HGHF response should be avoided. These include modal truncation, Guyan reduction, Generalized Dynamic Reduction, and integration schemes that filter out high frequency responses.

Extension of the consequences discussed here for a very simple case to the complicated multi-input-multi-output (MIMO) problem appears to require numerical evaluation for each special case. It appears, however, that important control response consequences are as likely to occur in MIMO as in SISO systems.

Numerical Examples

Examples are given below to help fix the ideas described above. These examples have counterparts that have occurred in practice.

Example 1: The following example is a simplification of a generic cantilevered structure connected to an appendage by a servo mechanism. The actuator/sensor is assumed to be collocated and is represented by a soft spring and inertia. The cantilevered structured is modeled by a beam that has a fundamental frequency of 5.0 Hz when fixed at one end. The appendage is modeled by a beam with a fixed end fundamental frequency of 1.0 Hz. Both beams are finite element models. For simplicity, actuator/sensor hardware is modeled by a single spring and inertia on the cantilevered structure side only. Figure 7 illustrates the model.

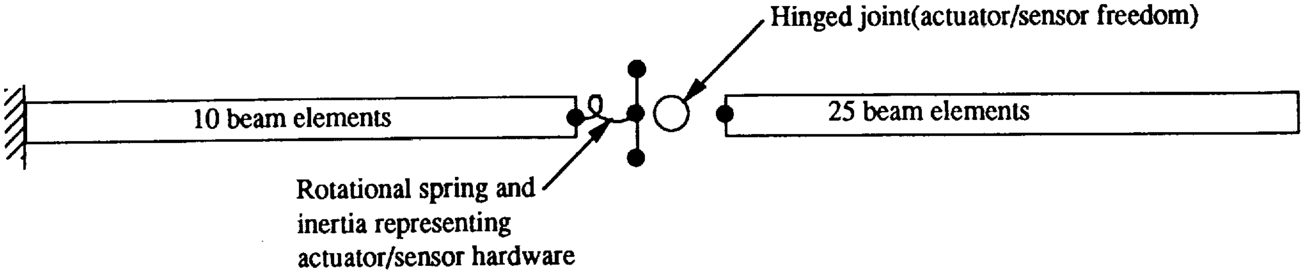


Figure 7. Flexible Beams Connected By Controller

The spring and inertia produce a HGHF mode. The soft interface spring has been chosen such that the gain of the HGHF mode is five times the sum of the gains of the other modes (a factor of five is not uncommon, in practice, for models that have been validated by traditional methods). The local inertia has been chosen to be arbitrarily small.

Figure 8 is a plot of the running sum of the modal gains for the collocated transfer function at the actuator/sensor interface, with and without the HGHF mode.

Figure 9 is a Bode plot of the transfer function with and without the HGHF mode.

The HGHF response tends to swamp the effects of the other modes, causing the Bode gain to remain high as the frequency increases. The low frequency transfer function zeros occur at lower frequencies due to the HGHF response. An important feature is the gain increase just above the lowest zero. This can cause low frequency stability problems.

Example 2: This is a qualitative discussion that refers to the previous example. The text thus far has referred to collocated transfer functions only. The HGHF response problem also applies to non-collocated transfer functions. If the sensor had been chosen to be at the free end of the appendage in the previous example, then care would be required to model local flexibility and inertia at that location. The transfer function would now be affected by at least two HGHF modes. In addition, the previous example assumed actuator flexibility to occur on only the cantilevered structure side. In actuality, there is local flexibility, due to the actuator, on the appendage side also. If all three sources of flexibility were modeled by single springs, there may be up to three HGHF modes each representing a simplification of a portion of the local hardware flexibility.

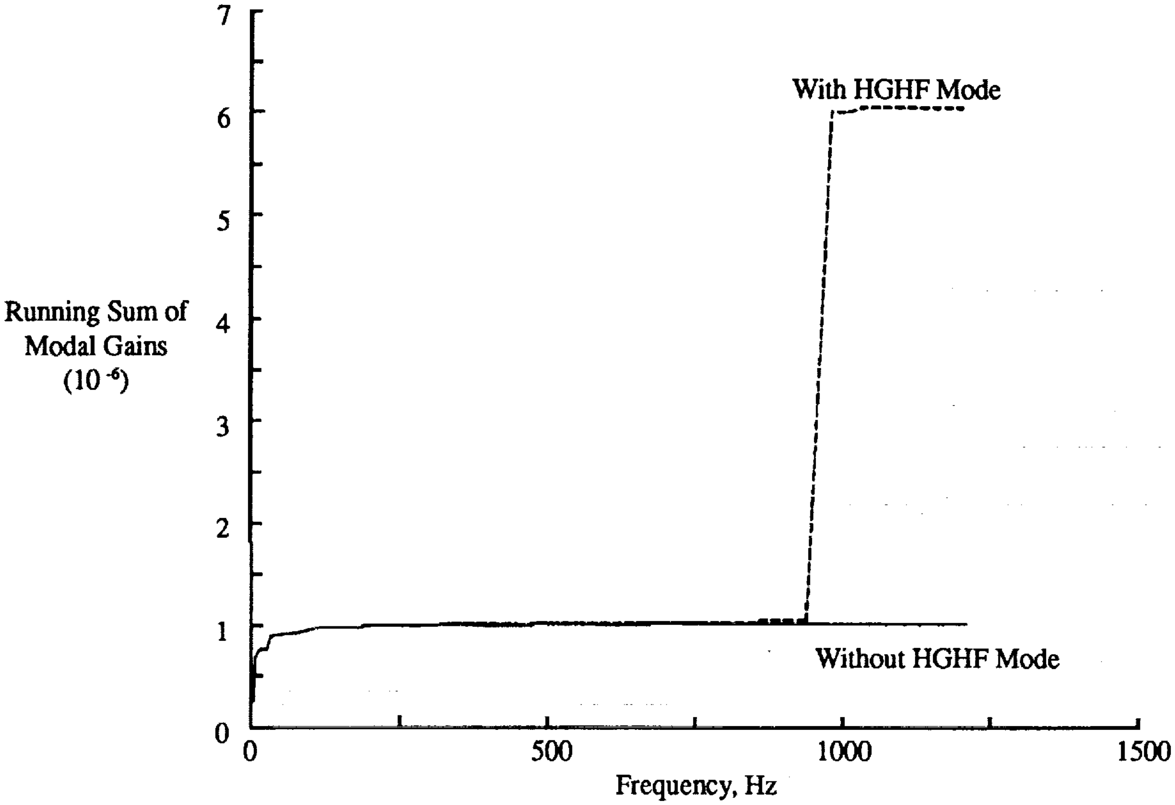


Figure 8. Gain Summation Plot for Cantilever Beam Hinged to Free Beam

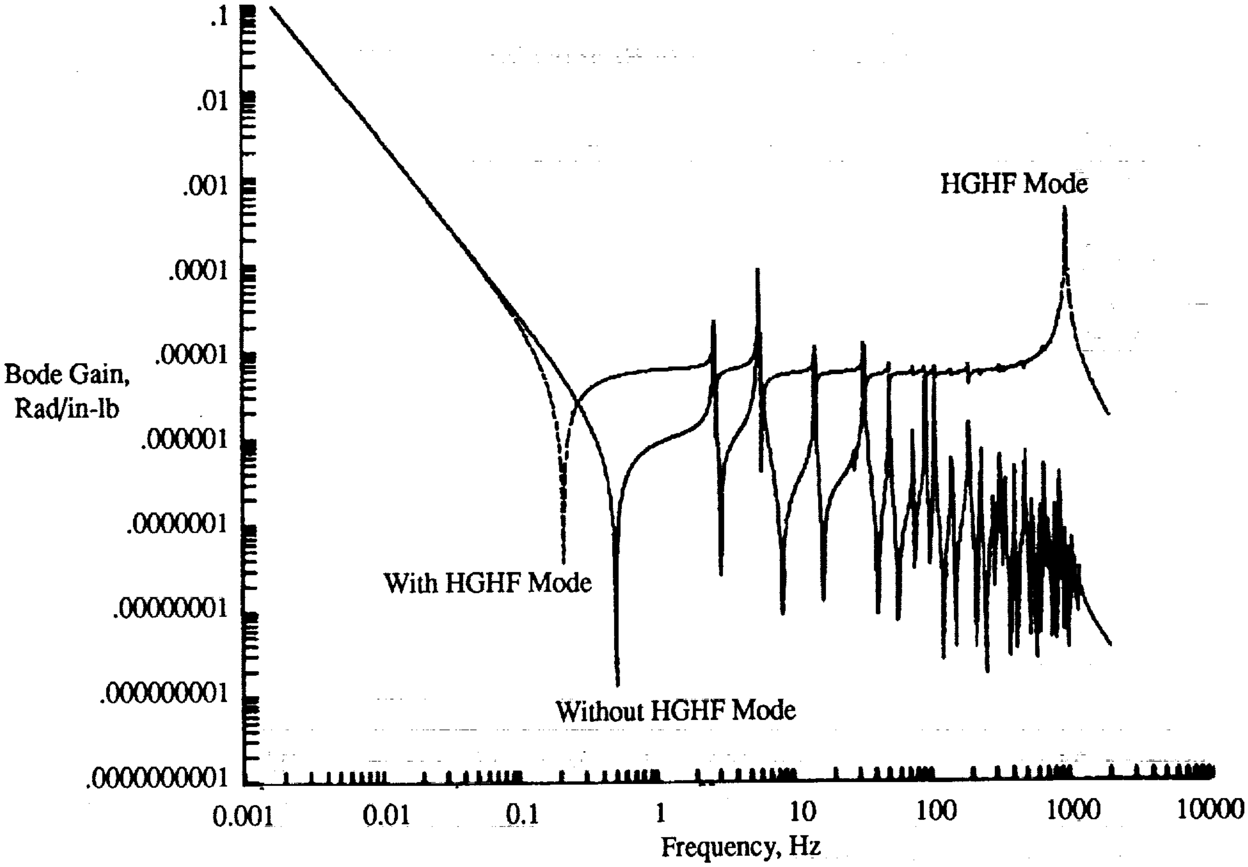


Figure 9. Transfer Function for Cantilever Beam Hinged to Free Beam

Example 3: The previous two examples discussed local flexibilities that were modeled by single rotational springs and inertias. Single spring approximations may be the result of condensing a much more complicated interface flexibility model to a single degree of freedom spring. Figure 10 shows a more refined interface model that produces similar response to that discussed above, but adds important features to the problem.

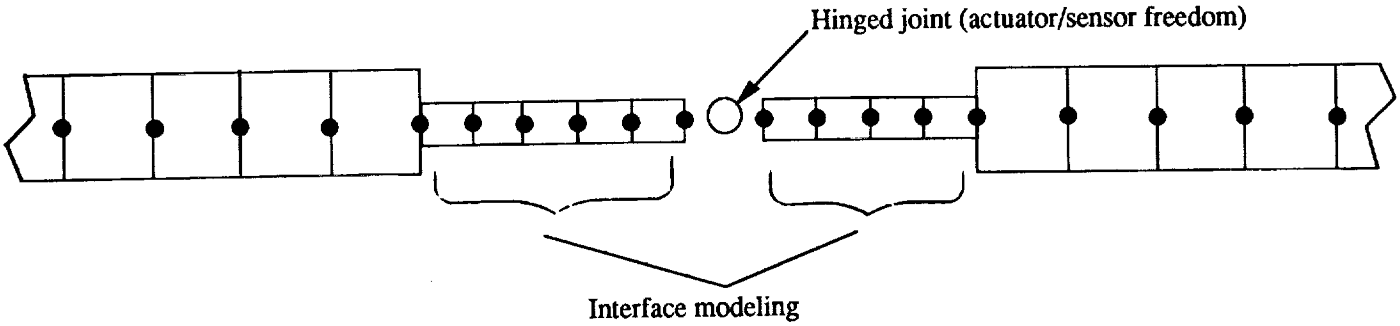


Figure 10. Hinged Beam With Locally Refined Modeling

The modeling includes a series of short (relative to the appendages that are connected by the actuator/sensor interface hardware) flexible beams. In the case of a single rotational spring an inertia had to be added to dynamically capture the local flexibility. Here, provided the short beam elements closest to the interface are not much more flexible than the other beams modeling the interface hardware, modes associated with the interface beams and lumped masses may capture a large portion of the local interface flexibility. Adding inertias will enable the rest. The Figure 11 running sum gain plot shows the effect of interface flexibility and mass/inertia modeling on the transfer function gains.

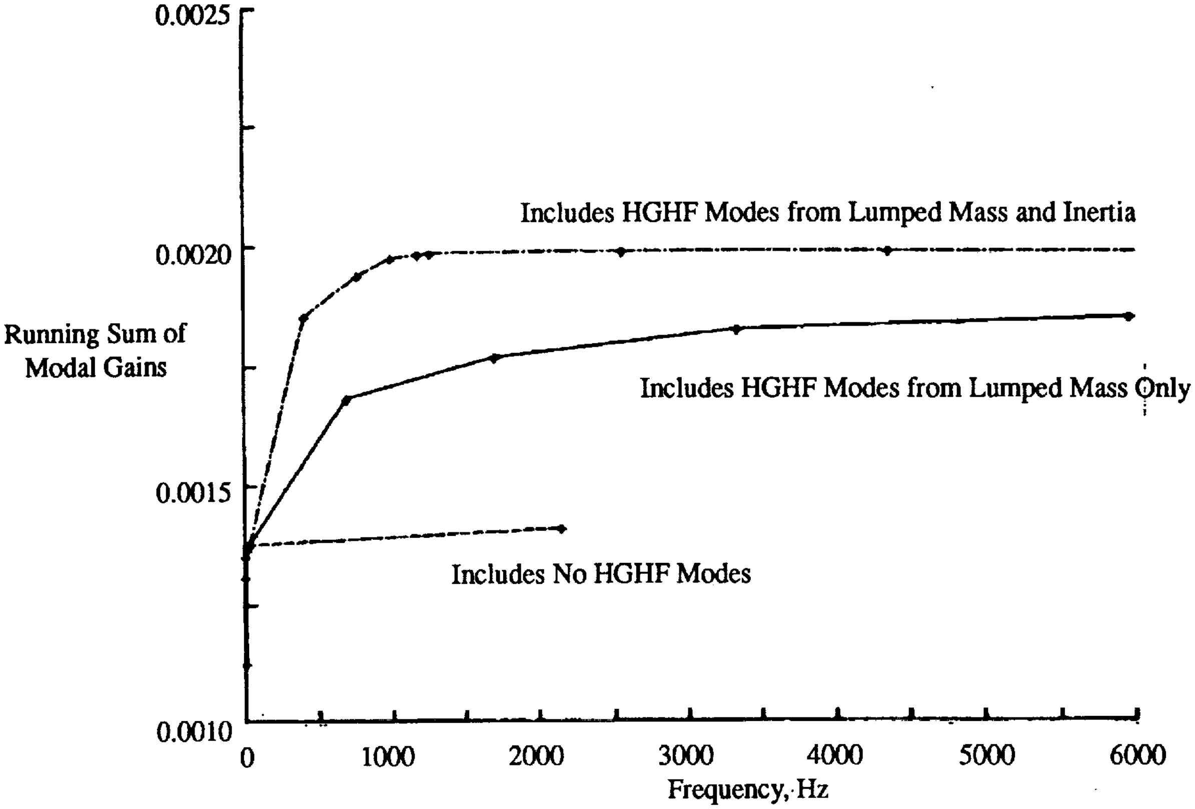


Figure 11. Gain Summation Plot for Beams Hinged at Flexible Interface

Each of the HGHF modes shown on the plot (by the marked points) are associated with local bending modes of the beam model representing the actuator/sensor interface hardware.

Testing to Validate Transfer Function Poles and Zeros

Testing to validate control-structure-interaction models is difficult because for the controls application it is really the system transfer functions, not only the low frequency modal data, that are needed. Unfortunately, tests to validate transfer function poles and zeros are subject to several severe limitations. First, although the best test is a system level transfer function test, the hardware to perform this test is generally not available until late in a program. Thus the data may confirm a model but it is not timely for design purposes. Second, a fixed interface modal survey test may partially confirm a model but it is unlikely that the available fixed interface for testing is truly the actuator/sensor interface. In addition, the torquer may not be available for early testing. If it is available it most likely cannot be physically separated into fixed interface parts as may be desirable in the math model formulation. Even if it could be separated, high frequency data are not practically obtainable in modal survey tests. Despite these real concerns over testing practicality and hardware availability, it is worthwhile to outline an overall testing plan that, if implemented, would address the model validation issue for controls. The testing would begin with fixed interface modal surveys of structural components to determine the overall flexibility of the component tested, up to and including an available fixed interface. This is illustrated by Figure 12.

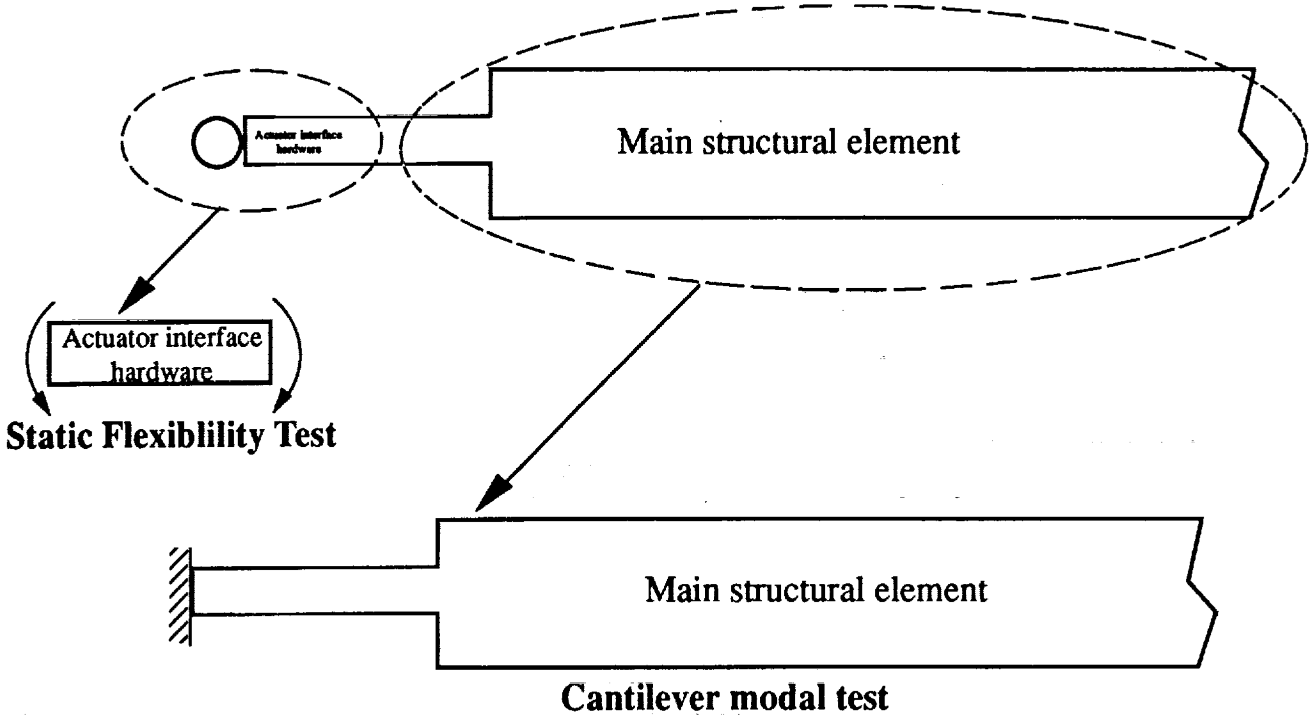


Figure 12. Structural Component Static and Modal Testing Scheme

In order to test the actuator interface hardware it is necessary to supplement component modal testing with static flexibility tests to characterize local flexibility. This will provide data to accurately define flexibility of interface hardware. From these data Craig-Bampton models can be validated and local flexibilities characterized. Component mode dynamic analysis or system modal analysis can proceed from this basis with a reasonable level of confidence. The models should show HGHF responses if careful modeling of both local flexibility and inertia is done to match test data and mass distribution in the interface region.

Finally, system level modal testing is also necessary. This provides low frequency validation of the merging of all of the component models. It approximately describes the low frequency mode shapes and the placement of the low frequency poles. The model resulting from component modal, local static flexibility, and system modal testing may be able to predict system transfer functions in the low frequency range. However, this prediction is a sensitive one. In particular, the system transfer function zeros have not been validated directly and are unlikely to be accurately located. Therefore system level transfer function testing, primarily to validate and adjust zero placement, is a desirable final step in validating structural models for the controls application. Since such testing is difficult and costly, and, as discussed above, may be impractical in some cases, analysis should be done to assess the sensitivity of control response to uncertainty in the low frequency transfer function gain. This analysis is a combined structures and controls endeavor. For sensitive cases, transfer function testing appears an essential final testing step.

HGHF Response Effect on Time Domain Simulations

Provided flexibility and mass have been modeled correctly, system level simulations will almost always include HGHF responses. For component mode simulations this will require integration with very small time steps. This is undesirable, and may even be completely impractical for many problems. A way to avoid this problem by model changes is to add mass and inertia to the hinge interfaces. Whether or not this can produce accurate control response predictions is problem specific. In any case, this appears an undesirable approach. Other approaches necessitate modification of the dynamic analysis methodology.

System mode simulations can escape this problem. These simulations can group the high frequency responses and treat them statically. The total flexibility will be obtained, and the integration can use large time steps. This approach has been very successful in practice.

Component mode methodologies, as presently formulated, do not have this capability. To follow such an approach in component mode analysis it would be necessary to use free-free component modes rather than cantilever or Craig-Bampton modes, and to include residual flexibility of all components as static responses.

Summary

High gain high frequency (HGHF) responses, in dynamic simulations, are the result of small local interface masses and inertias connected to relatively stiff primary structure by flexible elements representing servo mechanisms and their structural attachment schemes. Locally applied forces and torques at control system actuators result in the static response of HGHF modes in addition to dynamic response of low frequency modes.

HGHF responses affect transfer functions by moving all zeros to lower frequencies, particularly those occurring in the low frequency spectrum, and increasing system gain. Excluding local flexibility and local mass/inertia, or equivalently, reducing the modal set via modal truncation or Generalized Dynamic Reduction, suppresses HGHF responses, and can cause inaccurate control-structure-interaction predictions.

Structures and controls engineers need to maintain a close working relationship. An understanding of each others technical tools is necessary to assure accurate modeling of the control-structure-interaction problem. Modeling responsibilities need to be well defined to avoid exclusion or duplication of flexibility, and mass/inertia.

Component modal testing should be supplemented by static flexibility testing of local actuator/sensor hardware since it is unlikely that the available fixed interface, for the component modal test, is actually the actuator/sensor interface. System modal testing should be performed to validate the low frequency poles. If analysis shows the control response to be sensitive to uncertainty in the low frequency transfer function gain, it is desirable to supplement system modal testing with transfer function testing.

If local flexibility and mass/inertia have been modeled correctly, system level simulations will almost always include HGHF responses. System mode simulations can group the HGHF modes statically, thereby retaining the total flexibility and allowing large time steps in time domain analysis. Component mode simulations do not currently have this capability and face difficulties in application to analysis of structures with detailed modeling of actuator/sensor interfaces.

Acknowledgements

The authors wish to express their appreciation to Sherman Bigelow, Stephen Church, Dean Clingman, and Donald Skoumal of Boeing Aerospace and Electronics, for their support in this work.

References

[1] R. R. Craig Jr. and M. C. C. Bampton, "Coupling of Substructures for Dynamic Analysis", AIAA Journal, Vol. 6, No. 7, July 1968.

[2] W. C. Hurty , "Dynamic Analysis of Structural Systems Using Component Modes", AIAA Journal, Vol. 3, No. 4, April 1965.

[3] R. R. Craig Jr., "Structural Dynamics An Introduction to Computer Methods", John Wiley and Sons, 1981.

[4] R. D. Cook, "Concepts and Applications of Finite Element Analysis", John Wiley and Sons, 1974, 1981.

[5] K. J. Bathe and E. L. Wilson, "Numerical Methods In Finite Element Methods", Prentice Hall, 1976.

[6] "Generalized Dynamic Reduction", MSC/NASTRAN Handbook for Dynamic Analysis - Version 63, Macneal-Schwendler Corporation, Jan. 1983.

N90-23078

SIGNIFICANCE OF NORMS AND COMPLETENESS IN VARIATIONAL BASED METHODS

Joel A. Storch

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA

Abstract

By means of a simple structural problem, we bring into focus an important requirement often overlooked in practice on the basis functions used in Rayleigh-Ritz-Galerkin type methods. The problem of the static deformation of a uniformly loaded beam is solved variationally by expanding the beam displacement in a Fourier Cosine series. The potential energy functional is rendered stationary subject to the geometric boundary conditions. It is demonstrated that the variational approach does not converge to the true solution. The object of the paper is to resolve this paradox, and in so doing, indicate the practical implications of norms and completeness in an appropriate inner product space.

Introduction

Virtually all flexible multibody codes in use today are based upon some variational principle of mechanics. The most common of these being Hamilton's Principle and its discretized version-Lagrange's Equations. Regardless of the particular label attached to the technique (e.g., assumed modes method, Ritz-Galerkin), the problem reduces to rendering stationary a certain definite integral with respect to a sufficiently regular family of functions subject to certain "geometric" boundary conditions. In practice the basis functions are generated using a general purpose Finite Element program and may be subject to further manipulation such as modal synthesis before being incorporated into the multibody program. It is assumed by many analysts that the basis functions used are to a certain extent arbitrary. It is argued that so long as they are members of an infinite family of orthogonal functions and satisfy the geometric boundary conditions, convergence of the dynamic response is guaranteed.

It is well-known in the mathematical theory of variational methods that the basis functions must be complete in an appropriate inner product space. Our problem demonstrates the importance of this somewhat subtle criterion. It is extremely significant that application of the variational principle using an inappropriate set of basis functions is convergent, but to an erroneous result. This would be difficult if not impossible to identify in a typical spacecraft application

Problem Statement and Exact Solution

A cantilever beam constant bending stiffness *EI* and unit length is acted on by a uniform distributed loading. If the load per unit length is *q*, and we write *p* for the ratio *q/EI*, then the static deflection W(x) is governed by

$$\frac{d^4W}{dx^4} = p, \quad W(0) = W'(0) = W''(1) = W'''(1) = 0 \tag{1}$$

With constant *p* the solution is a polynomial of degree four:

$$W(x) = p\left(\frac{x^4}{24} - \frac{x^3}{6} + \frac{x^2}{4}\right) \tag{2}$$

For future reference we develop this solution in a Fourier cosine series, extending it as an even function to (- 1,0):

$$W(x) = p\left[\frac{1}{20} - \sum_{n=1}^{\infty} \frac{2\cos n\pi x}{n^4\pi^4} + \sum_{n=1}^{\infty} \frac{(-1)^n \cos n\pi x}{3n^2\pi^2}\right] \tag{3}$$

Variational Solution

A variational principle equivalent to the boundary value problem (1) is the principle of minimum total potential energy: Find the function W(x) satisfying W(0) = W'(0) = 0 (and sufficiently regular) that minimizes the total potential energy *EI P(W)*, where

$$P(W) = \int_0^1 \left[\frac{1}{2}(W'')^2 - pW\right] dx \tag{4}$$

We apply the Rayleigh-Ritz method to the energy functional in (4). The set of basis functions will be {1, cos πx, cos 2πx,...}, which is complete over the interval $0 \le x \le 1$. The variational solution is then a cosine expansion

$$W = a_o + \sum_{1}^{\infty} a_n \cos n\pi x \tag{5}$$

which automatically satisfies the geometric boundary condition W'(0) = 0 (as well as the natural boundary condition W'''(1) = 0). Substituting the assumed expansion (5) into the integral (4), the orthogonality of the cosines gives the value J for the total potential energy:

$$J = \frac{1}{4}\sum_{n=1}^{\infty} n^4\pi^4 a_n^2 - pa_0 \tag{6}$$

We calculate the coefficients a_0, a_1,... by rendering J stationary subject to the remaining geometric boundary condition: $W(0) = 0$, or $\Sigma a_n = 0$.

Following the standard method we introduce a Lagrange multiplier λ and build the constraint into the auxiliary function

$$L = \frac{1}{4} \sum_{n=1}^{\infty} n^4 \pi^4 a_n^2 - pa_0 + \lambda \left(a_o + \sum_{n=1}^{\infty} a_n \right)$$

The equation $\partial L/\partial a_o = 0$ yields $\lambda = p$. Then $\partial L/\partial a_n = 0$ gives

$$a_n = \frac{-2p}{n^4 \pi^4} \quad (n=1,2,3,\ldots) \tag{7}$$

The geometric constraint then yields

$$a_o = \frac{2p}{\pi^4} \sum_{n=1}^{\infty} \frac{1}{n^4} = p/45 \tag{8}$$

When the coefficients (7), (8) are inserted into (5) we obtain the variational solution

$$W^*(x) = p \left[\frac{1}{45} - \sum_{n=1}^{\infty} \frac{2 \cos n\pi x}{n^4 \pi^4} \right] \tag{9}$$

This can also be summed exactly, and it is again a fourth degree polynomial - but a different one!

$$W^*(x) = p \left[\frac{x^4}{24} - \frac{x^3}{6} + \frac{x^2}{6} \right] \tag{10}$$

Comparing with (2), the difference W-W* is $px^2/12$. Only geometric boundary conditions were enforced in computing W*. One expects that the natural boundary conditions will automatically be satisfied. Note however that $(W^*)'' = - p/6$ at $x = 1$, and not zero. A plot of W (exact) and W* (Ritz) is given in Figure 1.

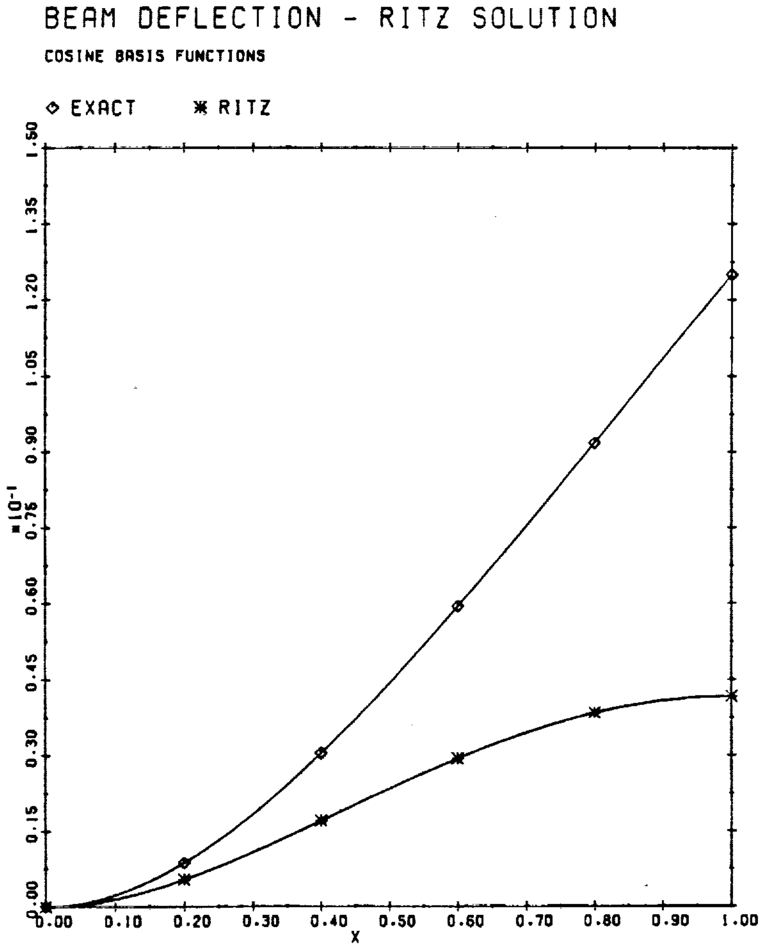


Figure 1

An Observation

The paper started by expanding the exact solution W into the cosine series (3). That series has coefficients of order $1/n^2$, and it converges to W. But computing the bending energy in these two forms of the solution, polynomial and Fourier, leads to a disturbing result:

For the polynomial,

$$\int_0^1 (W'')^2\,dx = p^2 \int_0^1 \left(\frac{1}{2}x^2 - x + \frac{1}{2}\right)^2 dx \qquad \text{is } \textit{finite}.$$

For the series,

$$\int_0^1 (W'')^2 dx = p^2 \sum_1^\infty \frac{n^4\pi^4}{2}\left[\frac{-2}{n^4\pi^4} + \frac{(-1)^n}{3n^2\pi^2}\right]^2 \qquad \text{is } \textit{infinite}.$$

The orthogonality of the cosines produces this sum of positive terms, roughly $n^4 a_n^2$, and the sum does not converge.

Resolution of the Paradox

When the geometric boundary conditions are linear and homogeneous, it is standard practice to choose the basis functions such that each individually satisfies them. Since our basis functions of cosines do not all vanish at x=0, one might be tempted to point to this fact as the root of the problem. However, our enforcement of the geometric boundary condition through a Lagrange multiplier is perfectly legitimate. There is nothing in the theory that forces us to satisfy the geometric boundary conditions by each basis function; only the resultant linear combination must satisfy them. (See the section "Numerical Results" for substantiation of this statement.) The key to resolving the paradox lies in a loose statement preceding eq. (5), regarding the "completeness" of the cosines. The word "complete" in itself has no meaning. We have to identify the space of admissible functions, as well as its norm (the measure of distance in that space), before it can be claimed that a set of trial functions is complete - in other words, before we can say that the combinations of the trial functions can approximate with arbitrary accuracy any admissible function. To discuss accuracy we need a norm.

The most common measure of distance is the L^2 norm - the square root of $\int f^2 dx$. The function f need not be continuous; step functions present absolutely no difficulty. The space contains functions much worse than that, although a delta function has infinite length and is not allowed. In the L^2 norm the set $\{1, \cos \pi x, \cos 2\pi x, \ldots\}$ is *complete* (on the interval $0 \le x \le 1$) and this is a cornerstone of Fourier analysis. The cosines are also complete in the L^p spaces, with norm $(\int |f|^p dx)^{1/p}$ and $p \ge 1$, but L^2 is special: it is associated with an inner product. That makes it a Hilbert space, in which $(f, g) = \int fg\, dx$ matches the norm: (f,f) agrees with $\|f\|^2$. The L^2 space admits angles, and orthogonality, and all the geometry of ordinary Euclidean space. But it is not the only Hilbert space, nor is it necessarily the right one.

Our fourth-order problem comes with its own norm and inner product and space of admissible functions. We look there for the resolution of the paradox; we have to work with the right space. The *norm* comes directly from the bending energy:

$$\|W\|^2 = \int_0^1 (W''(x))^2 dx$$

The inner product is determined by the norm:

$$(W,V) = \int_0^1 W''(x)V''(x)dx$$

The admissible functions are also determined: Their norm must be finite and they must satisfy the essential boundary conditions. Thus W is admissible if

$$\int_0^1 (W'')^2 dx < \infty \quad \text{and } W(0)=0 \quad \text{and } W'(0)=0$$

Such a function W comes from integrating twice a function in L^2:

$$W(x) = \int_0^x \int_0^y f(t)dt\ dy$$

Notice that W is not required to satisfy the natural conditions $W''(1)=0$ and $W'''(1)=0$. We could not make that requirement and still have a complete space. Functions that satisfy these extra conditions can come arbitrarily close (measured by the norm) to functions that don't. The process of completion wipes out the natural conditions as a requirement on admissible W. In Reference 1 the second author referred to the admissible space as $H_E{}^2$ – the Hilbert space of functions that have two derivatives (in $L^2 = H^0$) and that satisfy the essential boundary conditions. Remark 3 will justify more fully the choice of bending energy – the second-degree term in the total potential energy – as the norm.

That finishes the functional analysis. It was needed in order to ask the right question: *Are the cosines complete in the space of admissible functions?* We suspect that the answer must be no.

Apart from boundary conditions, we are asking whether combinations of the cosines (including cos 0=1) can come arbitrarily close to W. We know they can do so in the ordinary L^2 norm, and the Fourier expansion (3) does it explicitly. The question is whether the cosines can come arbitrarily close in the 'second-derivative norm'. Equivalently, the second derivatives of our set of cosines must come close, in the ordinary sense to W". But the second derivatives are

$$(0,\ -\pi^2 \cos \pi x,\ -4\pi^2 \cos 2\pi x, \ldots)$$

You see the problem. We are missing the constant term! We cannot approximate $W''=2$ with the functions we have left. In other words, we cannot come close to $W=x^2$ with our original set of trial functions. The cosines were not complete, but if this additional trial function x^2 is included, the set is complete. That would add the constant function to the list of second derivatives. So it was no accident that the discrepancy between W and W* was a multiple of x^2.

Before drawing a final conclusion about W* we add four observations.

1. The new function x^2 not only completes the set, it is orthogonal to the original cosines. The inner product is

$$\int_0^1 -n^2\pi^2 \cos n\pi x \cdot 2\ dx = 0$$

2. The original cosines were not in H_E^2 – they were not really admissible – because they violated the essential condition W(0)=0. One way to correct that would have been to construct the trial functions more carefully; they could have been

$$\{1 - \cos \pi x,\ 1 - \cos 2\pi x,\ 1 - \cos 3\pi x, \ldots\} \quad (\text{and also } x^2!)$$

The alternative of keeping the extra trial function 1, and imposing W(0)=0 through a Lagrange multiplier, is equally correct. The example verifies that the choice can be based on computational convenience. In other problems it might be less easy to adjust the trial functions to satisfy the essential conditions.

3. The special feature of the norm $\|W\|^2 = \int_0^1 (W'')^2\,dx$ is that the distance from any admissible w to the exact solution W satisfies

$$\frac{1}{2} \| w - W\|^2 = P(w) - P(W) \tag{11}$$

It follows that minimizing the potential energy P over all trial functions w automatically minimizes the distance to W. If the trial space actually contains W, the distance is zero and we have the global minimum of P. In the typical case, when the Rayleigh-Ritz method keeps w in a finite-dimensional space, the best w is the projection of the exact W onto the trial space. The w that minimizes P also has minimum error – but to establish (11) we must measure the error $\|w - W\|$ in the correct 'energy norm'.

The verification splits off a term of integration by parts:

$$\|w - W\|^2 = \int_0^1 [(w'')^2 + 2W''(W'' - w'') - (W'')^2]\,dx$$

$$\int_0^1 W''(W'' - w'')\,dx = \int_0^1 \frac{d^4W}{dx^4}(W - w)\,dx + [W''(W' - w') - W'''(W - w)]_0^1$$

the boundary terms vanish because of the essential conditions on W and w at x=0, and the natural conditions on W at x=1. Writing p for d^4W/dx^4, and substituting into the first line, we recognize its right hand side as 2P(w) - 2P(W). The energy norm imposes itself. In the 'energy inner product', there is an orthogonal projection of W onto the trial space. That is why Rayleigh-Ritz, and finite elements, do so well.

4. A final small worry. If a set is not complete, as the original cosines were not, the terms they give should satisfy Bessel's inequality:

$$\|\Sigma a_n \cos n\pi x\|^2 \leq \|W\|^2$$

This means that the component of W inside the trial space should not be larger than W itself. But the inequality was tested in the short section prior to this one, and it failed. The left side was $+\infty$ and the integral of $(W'')^2$ on the right side was finite. After some thought one realizes that the a_n may be the cosine coefficients of W, but they are not the coefficients in the right inner product! Bessel's inequality with the series (3) is satisfied in the L^2 norm. In the energy norm, we must use the coefficients (7), (8) obtained from the Rayleigh-Ritz method.

Summary

It follows from the above discussion that the Rayleigh-Ritz method has converged upon the best approximation (as measured in the energy norm) in the space spanned by the cosines. We can write the approximation (9) in the alternate form

$$W^* = \sum_{n=1}^{\infty} \frac{2p}{n^4\pi^4}(1 - \cos n\pi x)$$

The above is not the cosine expansion of W (eq. 3). The two are different because the x^2 term was forgotten. Without that term we have Bessel's inequality $\|W^*\|^2 \leq \|W\|^2$. With x^2 included to complete the set of trial functions, the variational solution will become the exact one:

$$W = \frac{p}{12}x^2 + \sum_{1}^{\infty} \frac{2p}{n^4\pi^4}(1 - \cos n\pi x)$$

We can verify that p/12 is the Fourier coefficient in the correct inner product:

$$\frac{(W,\ x^2)}{(x^2,\ x^2)} = \frac{\int W'' \cdot 2dx}{\int 2 \cdot 2dx} = \frac{p}{2}\int_0^1 \left(\frac{x^2}{2} - x + \frac{1}{2}\right) dx = \frac{p}{12}$$

This example serves to illustrate an elusive pitfall in practice. It is possible to see rapid numerical convergence and conclude that a good approximation to the exact solution has been obtained. In reality, if one's set of basis functions are not complete (in the appropriate inner product space) the approximate solution may be highly inaccurate. Finite element experts will note that the pitfall could have been avoided by applying the *patch test*. That requires the special solution $W=x^2$, with constant strain, to be reproduced by the trial functions - after imposing conditions on an element boundary consistent with this particular W. The cosines would not have reproduced x^2. Thus the patch test would have failed, correctly indicating that the set was incomplete.

The reader will have noticed from the beginning that all combinations of the cosines satisfy the extra condition $W'(1)=0$. If that had been the essential condition at the right-hand end (see Reference 2, p. 174), with $W'''(1)=0$ as natural condition, the cosines would have been adequate.

Note: A completely parallel example can be constructed for the second-order equation $-u''=1$, with essential condition $u(0)=0$ and natural condition $u'(1)=0$. The corresponding set of trial functions, complete in L^2 but incomplete in H_E^1, is $\{\sin n\pi x\}$.

Numerical Results

Here we compare the cosine expansion to a different expansion that the theory guarantees to succeed. The latter comes from the eigenfunctions of a uniform vibrating free-free beam:

$$\frac{d^4\phi}{dx^4} = \beta^4\phi \quad \text{with} \quad \phi''(0) = \phi'''(0) = \phi''(1) = \phi'''(1) = 0 \tag{12}$$

Note that a more natural choice is a clamped-free beam, satisfying $\phi(0)=\phi'(0)=0$ at the left endpoint (and more like $1 - \cos n\pi x$). We wanted to see how the Lagrange multipliers would enforce these geometric conditions, when they are not imposed on each eigenfunction.

The results are striking. For the cosine expansion (through $\cos N\pi x$) we tabulate the deflections at $x=1$ and their errors (see Table I). The exact value is $W^*(1)=p/24$. W^* will be supplemented by the $px^2/12$ correction term, which is twice as large! Together they reach the correct value $W(1)=p/8$.

What is significant is the $1/N^3$ convergence rate. Doubling N reduces the error in the last column by a factor of 8.

Contrast that with the results using the free-free eigenfunctions

$$\phi_1 = 1 \quad \text{with} \quad \beta_1 = 0$$

$$\phi_2 = x - \frac{1}{2} \quad \text{with} \quad \beta_2 = 0$$

$$\begin{aligned}\phi_n = {} & (\sin \beta_n - \sinh \beta_n)(\cos \beta_n x + \cosh \beta_n x) \\ & - (\cos \beta_n - \cosh \beta_n)(\sin \beta_n x + \sinh \beta_n x) \\ & \text{with } \cos \beta_n \cosh \beta_n = 1\end{aligned}$$

The frequencies β_n are asymptotic to πn + constant. The expansions can almost be carried out by hand (with the help of orthogonality), but imposing $W(0)=W'(0)=0$ by Lagrange multipliers needs a simple code. The deflections are again tabulated at the free end $x=1$, and you will notice the change in convergence rate (see Table II). The error decreases like $1/5N$. At $N=800$ we are far above the error achieved previously at $N=10$.

Table I.

N	$W*_N(1)/p$	Error
10	0.04165996	0.00000671
20	0.04166582	0.00000085
40	0.04166656	0.00000011
80	0.04166665	0.00000002
Exact	0.04166667	

Table II.

N	$W_N(1)/p$	Error
50	0.120908	0.004092
100	0.122966	0.002034
200	0.123988	0.001012
400	0.124496	0.000504
800	0.124750	0.000250
Exact	0.125000	

Finite Elements

This last section looks at the effect of enforcing natural boundary conditions (as well as essential conditions) in the finite element method. The trial functions will be piecewise polynomials, and for the bending problem the natural choice is piecewise cubics. There are two parameters at each node - displacement and slope. Because they are continuous between elements, the trial functions w are in the class C^1 - with one continuous derivative, and jump discontinuities in the second derivative. This guarantees that w will be conforming; its bending energy (and therefore its norm!) is finite. We have only to think about the boundary conditions.

The essential conditions fix $w_0=0$ and $w'_0=0$ at the left end. With N intervals of length h=1/N, and two degrees of freedom per node, that leaves a 2N-dimensional trial space - provided no conditions are applied at the right end. If we do impose the natural boundary conditions, they yield two relations between w_{N-1}, w'_{N-1}, w_N, w'_N. Therefore the trial space is reduced to dimension 2N -2. The question is whether this is wise - to compel w to meet conditions that we know to be satisfied by the true solution W.

A functional analyst would say it is foolish. The smaller trial space (it is a subspace of the larger one) cannot give a better approximation to W. The error must be greater when degrees of freedom are removed. But that is the error in the energy norm, where the identity (11) holds and Rayleigh-Ritz picks the best finite element approximation as the projection. We might still hope that pointwise, and particularly near x=1, there is something to be gained by imposing the natural conditions.

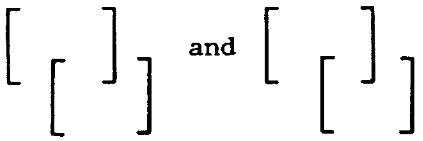
This example is so simple that the calculations can be done with pencil and paper. Furthermore, it has the special property of *superconvergence*. The finite element approximation is *exact at the nodes*. In the full trial space, with no constraints at x=1, there is agreement with the true W at every meshpoint. In the smaller trial space, the natural boundary conditions require the cubic to be a linear function within the final interval. (Then it satisfies $w''=w'''=0$ throughout the interval; that is the price for imposing those conditions at x=1.) In this case it is still exact at all other nodes! The discrepancy between the two finite element solutions is small, and very local, but the winner is clear.

Even near $x=1$, it is better when the natural conditions are left alone. By satisfying them, we spoil the accuracy.

It is a pleasure to verify superconvergence in this case. The element stiffness matrix and element force vector are

$$\frac{1}{h^3}\begin{bmatrix} 12 & 6h & -12 & 6h \\ 6h & 4h^2 & -6h & 2h^2 \\ -12 & -6h & 12 & -6h \\ 6h & 2h^2 & -6h & 4h^2 \end{bmatrix} \quad \text{and} \quad \frac{ph}{12}\begin{bmatrix} 6 \\ h \\ 6 \\ -h \end{bmatrix}$$

Those refer to the local parameters w_{j-1}, w'_{j-1}, w_j, w'_j. The assembly combines the overlapping parts:



Therefore the global stiffness equations for $j<N$ (found from $\partial P/\partial w_j=\partial P/\partial w'_j=0$) come from the middle rows of that assembly:

$$\begin{aligned} -12w_{j-1} - 6hw'_{j-1} + 24w_j - 12w_{j+1} + 6hw'_{j+1} &= ph^4 \quad , \\ 6hw_{j-1} + 2h^2w'_{j-1} + 8h^2w'_j - 6hw_{j+1} + 2h^2w'_{j+1} &= 0 \end{aligned} \tag{13}$$

Those equations are exactly satisfied by the nodal values W_j and W'_j of the true solution W in equation (2).

At the end $x=1$, imposing $w''_N=0$ and $w'''_N=0$ leads to

$$w'_N = w'_{N-1} \quad \text{and} \quad w_N = w_{N-1} + hw'_{N-1} \tag{14}$$

The cubic w is forced to be linear in the last interval. On the other hand, if these natural conditions are not imposed, the only difference from (13) is that no terms are assembled for the interval beyond the boundary:

$$\begin{aligned} -12w_{N-1} - 6hw'_{N-1} + 12w_N - 6hw'_N &= ph^4/2 \\ 6hw_{N-1} + 2h^2w'_{N-1} - 6hw_N + 4h^2w'_N &= -ph^5/12 \end{aligned} \tag{15}$$

The latter are satisfied by the true W but (14) is not - even though W itself does satisfy the natural conditions.

Remark. Even if the right side of $W''''=p$ is not constant in the original differential equation, it is still true that the finite element approximation has $w=W$ and $w'=W'$ at the nodes. This was observed already by Pin Tong[3]. Those conditions determine one particular interpolating w - and to show that this special w is the finite element solution, we verify that its error w - W is orthogonal to all trial functions. Then w is the correct projection of W, and superconvergence is verified.

Proof of orthogonality: In each interval the trial functions are cubics C(x), and the contribution to the inner product is an integral over that interval:

$$\int (w-W)''C''dx = [(w - W)'C'' - (w - W)C'''] + \int (w - W)C''''dx \tag{16}$$

this is zero because w-W and (w-W)' are zero at the endpoints, and C"" vanishes identically.

The argument still applies when we impose the natural boundary conditions, and force w to be linear in the last interval. Over all other intervals (16) is zero as before. Over the last interval it is zero because C"=0. Therefore the cubic which matches W and W' at every node except x=1 is the finite element solution in this case. It would have been more accurate to match at *every* node, by not forcing w"(1)=w'''(1)=0.

In this example, and surely in more realistic and more complicated applications of the displacement method, *it is better to leave the natural boundary conditions to nature*.

References

1. G. Strang and G. Fix, An Analysis of the Finite Element Method, Prentice-Hall, Englewood Cliffs, NJ, 1973.

2. G. Strang, Introduction to Applied Mathematics, Wellesley-Cambridge Press, Box 175, Wellesley, MA, 1986.

3. Pin Tong, "Exact Solution of Certain Problems by the Finite Element Method", AIAA J. 7, 178-180 (1969).

N90-23079

The Power and Efficiency of Advanced Software and Parallel Processing

Ramen Singh, Dynacs Engineering, Inc.
Lawrence W. Taylor, Jr., NASA Langley Research Center

Abstract

Real-time simulation of flexible and articulating systems is difficult because of the computational burden of the time varying calculations. The mobile servicing system of the NASA Space Station Freedom will handle heavy payloads by local arm manipulations and by translating along the spline of the Station. Because such motion can be very disruptive to the attitude of the Space Station, it is crucial to have real-time simulation available.

To enable such a simulation to be of high fidelity and to be able to be hosted on a modest computer, special care must be made in formulating the structural dynamics. frontal solution algorithms save considerable time in performing these calculations. In addition, it is necessary to take advantage of parallel processing, and in particular, certain powerful processors available at modest cost. It is crucial that both the algorithm and the parallel processing be compatible to take full advantage of both. an approach is offered which will result in high fidelity, real-time simulation for flexible, articulating systems such as the space Station remote servicing system.

DYNAMIC ANALYSIS OF FLEXIBLE MECHANICAL SYSTEMS USING LATDYN

by

Shih-Chin Wu
Che-Wei Chang
COMTEK
Grafton,VA 23662

and

Jerrold M. Housner
NASA Langley Research Center
Hampton,VA 23665

presented
at

3rd Annual Conference on Aerospace Computational Control
August 28-30, 1989 Oxnard, California

1. INTRODUCTION

Recently proposed space structures have grown increasingly large and complex. These may be delivered to orbit by the space shuttle and then deployed/assembled on orbit. To reduce weight, efficient designs of such systems tend to lead to flexible, low-frequency, and often joint-dominated structures. Interaction between rigid body motion and structural deformation will likely occur. For efficient operation of system requiring component articulation, it is desirable to maneuver components as rapidly as possible. Operational speed is limited by excessive dynamic deformation if vibrations are not suppressed. In order to suppress excessive vibration response, active controls may be utilized. The control design is usually based on linear methods, however the articulation is governed by nonlinear equations, moreover, design methods use reduced structural models. To access these design performance as well as stabilities, analytical simulations are usually performed.

Simulation codes for multibody systems such as DADS[1], DISCOS[2], and TREETOPS[3] use assumed mode approach to describe the structural deformations of components. This approach requires users to pick a reference frame for the component, discretize the component into finite elements, select component boundary conditions upon which the modes will be generated, solve the eigenvalue problem for deformation modes and select a modal set for the application at hand. Modal selection is often the most crucial part in the procedure. Since deformations of the component are defined as linear combination of the selected modes, the component can only deform in the space spanned by the selected modes. Therefore, results of the modal approach will be misleading if any modes are significantly excited were not selected. To predict which modes will get excited can be a difficult challenge in a flexible multibody system, since the system configurations are changing with time. Especially is thus true for many proposed future spacecrafts which have complicated geometry and joint hinges.

To provide an alternative approach which circumvented some of these difficulties, the LATDYN computer code was developed for research purposes. The LATDYN program is finite-element-based. The user model the component with finite elements, instead of using truncated modes which have to be generated outside the multibody analysis codes. In order to separate the rigid body motion and small deformations in the finite element approach, a coordinate system is chosen to represent the large displacement and rotation of the element. Deformations of the element are then defined with respect to the rigid body configuration of the element. At the element-level, mass matrices are calculated. The component mass matrix is obtained by assembling each elemental mass matrices as is typically done in conventional small motion/deformation finite element methods.

To form the system mass matrix, most multibody simulation codes impose nonlinear kinematic constraints on components that connect to the same joint. Instead of using constraint equations, the LATDYN program builds the hinge degree-of-freedom into the system equations of motion to connect components that share a common joint in a manner patterned after connectivity relations in conventional, small motion, finite elements. The mass of the interconnecting joint between the bodies represents a significant portion of the total mass and the orientation of the joint's hinge lines play an important role in determining structural behavior. It is thus reasonable to construct the finite element program with the joints as a part of element connectivity.

2. KINEMATICS OF BEAM ELEMENT

The kinematics developed here is applicable to arbitrarily large displacement and rotational motion of a beam with small deformations. Consider the beam element with finite element nodes 1 and 2 at initial (undeformed) and current (deformed) configurations in an inertial X-Y-Z frame, as shown in Fig. 1. In order to specify the configuration of the beam element, it is necessary to define a set of generalized coordinates that uniquely define the global displacement of every point in the deformed element. For each node of the element, an x_i-y_i-z_i (i=1,2) nodal reference frame having its x axis tangent to the neutral axis of the beam and y, z axes coincide with the principle axes of the beam cross section, is chosen to locate and orient the node in the inertia frame. Vectors $\mathbf{r}_i$ (i=1,2) from the origin of the initial x_i-y_i-z_i nodal reference frame to the origin of the current x_i-y_i-z_i nodal reference frame define the global displacement of nodes 1 and 2. Transformation matrices $\mathbf{T}_i$ from nodal reference frames to the global frame define orientations of the nodal reference frame. Deformation of the beam and displacement of any point in the beam now can be determined using $\mathbf{r}_i$ and $\mathbf{T}_i$.

Before deformations of the beam can be defined, rigid body motion of the beam has to be separated from the large displacement of the beam. It is chosen to specify rigid body motion of the element by use of a convected coordinate system X_c-Y_c-Z_c whose origin is located at node 1. Initially, orientations of the convected coordinate system coincides with the nodal reference frame of both nodes 1 and 2. As the element moves with large displacement and small deformation, the orientation of the convected coordinate frame, hence the rigid body motion of the beam, is determined by defining the X_c axis of the X_c-Y_c-Z_c frame always lie along the line connecting nodes 1 and 2, and the Y_c axis to lie in the plane formed by the y axis of x_1-y_1-z_1 frame and the X_c axis of X_c-Y_c-Z_c frame. With these definitions, the convected coordinate system is uniquely determined.

Deformations of the beam element are defined with respect to its rigid body configuration as

$$\mathbf{D}_i = \mathbf{T}_c^T \mathbf{T}_i \qquad (1)$$

where $\mathbf{T}_c$ denotes the transformation matrix from the rigid body configuration (or the convected coordinate frame) to the global frame. Where $\mathbf{D}_i$ is the difference in orientations between $\mathbf{T}_c$ and $\mathbf{T}_i$ at any time step, due to flexural deformation, namely, a transformation from $\mathbf{T}_c$ to $\mathbf{T}_i$. Note that, $\mathbf{D}_i$ can also be regarded as the transformation of $\mathbf{T}_i$ that rotates about a vector from the undeformed states to the current states. Assume that the rotation angles between $\mathbf{T}_i$ and $\mathbf{T}_c$ are small, then the components of this vector are the three rotation angles measured with respect to the three axes of $\mathbf{T}_c$ [4]. Therefore, $\mathbf{D}_i$ can be simply represented by

$$\mathbf{D}_i = \begin{bmatrix} 1 & -\phi_{zi} & \phi_{yi} \\ \phi_{zi} & 1 & -\phi_{xi} \\ -\phi_{yi} & \phi_{xi} & 1 \end{bmatrix}$$

where ϕ_{xi}, ϕ_{yi}, and ϕ_{zi}, are rotation angles of $\mathbf{T}_i$ about x, y, and z axis of $\mathbf{T}_c$, respectively. Physically, they correspond to flexural deformations of the beam element at nodes 1 and 2.

The rotation angles may be readily extracted from $\mathbf{D}_i$ as follows:

$$\phi_{xi} = (0,\ 0,\ 1)\mathbf{D}_i(0,\ 1,\ 0)^T \tag{2}$$

$$\phi_{yi} = (1,\ 0,\ 0)\mathbf{D}_i(0,\ 0,\ 1)^T \tag{3}$$

$$\phi_{zi} = (0,\ 1,\ 0)\mathbf{D}_i(1,\ 0,\ 0)^T \tag{4}$$

Substitution of Eq. 1 into Eqs.2-4 yields

$$\phi_{xi} = \mathbf{t}_{c3}{}^T\mathbf{t}_{i2} \tag{5}$$

$$\phi_{yi} = \mathbf{t}_{c1}{}^T\mathbf{t}_{i3} \tag{6}$$

$$\phi_{zi} = \mathbf{t}_{c2}{}^T\mathbf{t}_{i1} \tag{7}$$

where $\mathbf{t}_{cj}$ and $\mathbf{t}_{ij}$ are jth column of $\mathbf{T}_c$ and $\mathbf{T}_i$, respectively. Since the x axis of $\mathbf{T}_c$ lies on the line connecting nodes 1 and 2, the direction cosines of the vector from nodes 1 to node 2, which is the first column of $\mathbf{T}_c$, can be written as

$$\mathbf{t}_{c1} = (\mathbf{r}_2 - \mathbf{r}_1 + \mathbf{r}_0)/L \tag{8}$$

where $\mathbf{r}_0$ is a vector from node 1 to node 2 in the initial configuration. Since the Y_c axis of $\mathbf{T}_c$ lies on the same plane formed by $\mathbf{t}_{c1}$ and $\mathbf{t}_{i2}$, the Z_c axis of $\mathbf{T}_c$, hence the third column of $\mathbf{T}_c$, is the cross product of $\mathbf{t}_{c1}$ and $\mathbf{t}_{i2}$, namely,

$$\mathbf{t}_{c3} = \tilde{\mathbf{t}}_{c1}\mathbf{t}_{i2} \tag{9}$$

The Y_c axis can be easily obtained by taking the cross product of $\mathbf{t}_{c3}$ and $\mathbf{t}_{c1}$,

$$\mathbf{t}_{c2} = \tilde{\mathbf{t}}_{c3}\mathbf{t}_{c1} \tag{10}$$

where $\tilde{\mathbf{a}}$ is the skew-symmetric matrix

$$\tilde{\mathbf{a}} = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix}$$

with a_x, a_y, a_z being the vector components of vector **a**. Note that $\tilde{\mathbf{a}}^T = -\tilde{\mathbf{a}}$ and that $\tilde{\mathbf{a}}\mathbf{b} = -\tilde{\mathbf{b}}\mathbf{a}$, which agrees with the vector product property $\mathbf{a}\mathbf{x}\mathbf{b} = -\mathbf{b}\mathbf{x}\mathbf{a}$.

Let $\rho = [x_C, y_C, z_C]$ be the position vector of an arbitrary point P in the beam element, defined with respect to $\mathbf{T}_C$. Define that

$$\xi = x_C/L$$

$$\eta = y_C/L$$

$$\zeta = z_C/L$$

where L is the length of the beam element. The displacement of a point P on a beam element due to flexural deformation may be expressed as

$$\mathbf{u} = \mathbf{N}\Phi \qquad (11)$$

where **N** is a 3x6 matrix of shape functions similar to that used in the standard finite element method, namely

$$\mathbf{N} = \left[\begin{array}{cccccc} 0 & (1-4\xi+3\xi^2)L\zeta & (-1+4\xi-3\xi^2)L\eta & 0 & (-2\xi+3\xi^2)L\zeta & (2\xi-3\xi^2) \\ -(1-\xi)L\zeta & 0 & (\xi-2\xi^2+\xi^3)L & -L\xi\zeta & 0 & (-\xi^2+\xi^3) \\ -(1-\xi)L\eta & (-\xi+2\xi^2-\xi^3)L & 0 & -L\xi\eta & (\xi^2-\xi^3)L & 0 \end{array}\right.$$

with ξ, η, and ζ being x_C/L, y_C/L, and z_C/L respectively, and Φ, the composite vector of rotation angles of nodes 1 and 2,

$$\Phi = [\ \phi x1,\ \phi y1,\ \phi z1,\ \phi x2,\ \phi y2,\ \phi z2\ ]^T$$

The total displacement of point P as shown in Fig. 1 is, in vector form,

$$\vec{\mathbf{r}}^P = \vec{\mathbf{r}}_1 + \vec{\rho} + \vec{\mathbf{u}} - \vec{\rho}_0$$

where ρ is the vector $[x_C, y_C, z_C]^T$ and in algebraic form,

$$\mathbf{r}^P = \mathbf{r}_1 + \mathbf{T}_C\rho + \mathbf{T}_C\mathbf{u} - \mathbf{T}_{Co}\rho_o \qquad (12)$$

where $\mathbf{T}_{Co}$ is the initial transformation matrix of $\mathbf{T}_C$ and ρ_o is the initial position vector of point P in $\mathbf{T}_C$. Note that, axial deformation is implicitly included in the second term of the right-hand-side of Eq. 12.

3. SUPER-BEAM ELEMENT

In some multibody formulation, the joint connection between elements is imposed through constraint equations. Here, instead of introducing additional constraints, extra degree-of-freedom are added to the original element generalized coordinates to form a super beam-element consisting of joint and beam.

Consider a beam element with rigid joint bodies at both ends, as shown in Fig. 2. For simplicity, assume that no other element is attached to both joint bodies, and both joints have a one degree-of-freedom hinge. For each joint body, an X_i-Y_i-Z_i $(i=1,2)$ body reference frame is chosen to locate and orient the joint body in the inertia frame. Vectors $\mathbf{R}_i$ $(i=1,2)$ from the origin of the initial body reference frame to the origin of the current body reference frame define the global displacement of joint bodies 1 and 2. Transformation matrices Γ_i from joint body reference frames to the global frame define orientations of the joint body. Vectors that locate joint attachment points in joint bodies are denoted $\mathbf{s}_i$ $(i=1,2)$, defined with respect to joint body reference frames. Therefore, nodal displacements $\mathbf{r}_i$ of the beam element can be represented by the joint body displacements $\mathbf{R}_i$ as

$$\mathbf{r}_i = \mathbf{R}_i + \Gamma_i \mathbf{s}_i - \Gamma_{io} \mathbf{s}_i \tag{13}$$

To determine the relation between $\mathbf{T}_i$ and Γ_i, consider coordinate systems X_i'-Y_i'-Z_i' and x_i'-y_i'-z_i' that are located at a joint attachment point one fixed to the joint body and one fixed to the nodal frame. Initially, let X_i'-Y_i'-Z_i' and x_i'-y_i'-z_i' coincide with the z axes parallel to the hinge axis. Then, the difference in orientations of both systems at any time is the relative rotation about the axis of hinge. The transformation matrix from x_i'-y_i'-z_i' frame to X_i'-Y_i'-Z_i' frame is

$$\Theta_i = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 \\ \sin\theta_i & \cos\theta_i & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

where θ_i is the relative rotation angle. Denote Γ_i' and $\mathbf{T}_i'$ be the transformation matrices of X_i'-Y_i'-Z_i' and x_i'-y_i'-z_i' frames with respect to X_i-Y_i-Z_i and x_i-y_i-z_i frames, respectively. Then, the transformation matrix from joint body frame to the nodal reference frame can be obtained by sequential transformations as

$$\mathbf{T}_i = \Gamma_i \Gamma_i' \mathbf{\Theta}_i \mathbf{T}_i'^{T} \tag{14}$$

Substituting Eqs. 13 and 14 into Eq. 12, the displacement of an arbitrary point in the beam can be represented in terms of displacements and orientations of joint bodies at the ends of beam element, and relative rotations of joint degrees-of-freedom, i.e.,

$$\mathbf{r}^P = \mathbf{G}(\mathbf{R}_1, \Gamma_1, \theta_1, \mathbf{R}_2, \Gamma_2, \theta_2) \tag{15}$$

4. VARIATIONAL EQUATIONS OF MOTION OF A SUPER-BEAM ELEMENT

The variational equations of motion of a beam element at time t, for a virtual displacement field that is consistent with the constraints is written as,

$$-\int_{\Omega}\mu\delta\mathbf{r}^{P^T}\ddot{\mathbf{r}}^P d\Omega+\int_{\Omega}\delta\mathbf{r}^{P^T}\mathbf{f}^P d\Omega+\int_{\sigma}\delta\mathbf{r}^{P^T}\mathbf{h}^P d\sigma=\int_{\Omega}\delta\varepsilon^{P^T}\tau^P d\Omega \tag{16}$$

where $\delta\mathbf{r}^P$ is a virtual displacement of point P that is consistent with constraints, $\ddot{\mathbf{r}}^P$ is the acceleration of point P, $\mathbf{f}^P$ is body force density at point P , $\mathbf{h}^P$ is the surface traction at point P, $\delta\varepsilon^P$ is a kinematically compatible strain variation vector, τ^P is the associated stress vector at point P, and Ω and σ are the volume and surface of the beam before it is deformed.

By taking the variation of Eq. 15, the virtual displacement of point P is obtained as

$$\delta\mathbf{r}^P= \mathbf{g}_{R_1}\delta\mathbf{R}_1+\mathbf{g}_{\Gamma_1}\delta\pi_1+\mathbf{g}_{\theta_1}\delta\theta_1+\mathbf{g}_{R_2}\delta\mathbf{R}_2+\mathbf{g}_{\Gamma_2}\delta\pi_2+\mathbf{g}_{\theta_2}\delta\theta_2 \tag{17}$$

where $\delta\pi_i$ is virtual rotation of Γ_i. Therefore, the virtual displacement of a point P in the beam element is represented by the virtual displacements and rotations of the joint bodies, and virtual rotations of relative joint degrees-of-freedom.

The acceleration vector of a typical point can be obtained by taking two time derivatives of Eq. 15, which gives

$$\begin{aligned}\ddot{\mathbf{r}}^P= {}&\mathbf{g}_{R_1}\ddot{\mathbf{R}}_1+\mathbf{g}_{\Gamma_1}\dot{\omega}_1+\mathbf{g}_{\theta_1}\ddot{\theta}_1+\mathbf{g}_{R_2}\ddot{\mathbf{R}}_2+\mathbf{g}_{\Gamma_2}\dot{\omega}_2+\mathbf{g}_{\theta_2}\ddot{\theta}_2\\ &+\dot{\mathbf{g}}_{R_1}\dot{\mathbf{R}}_1+\dot{\mathbf{g}}_{\Gamma_1}\omega_1+\dot{\mathbf{g}}_{\theta_1}\dot{\theta}_1+\dot{\mathbf{g}}_{R_2}\dot{\mathbf{R}}_2+\dot{\mathbf{g}}_{\Gamma_2}\omega_2+\dot{\mathbf{g}}_{\theta_2}\dot{\theta}_2\end{aligned} \tag{18}$$

where ω_i and $\dot{\omega}_i$ are angular velocity and angular acceleration of the joint body i.

Substituting Eqs. 17 and 18 into Eq. 16, the first term is

$$-\int_{\Omega}\mu\delta\mathbf{r}^{P^T}\ddot{\mathbf{r}}^P d\Omega$$

$$= -\left[\delta\mathbf{R}_1^{\ T},\delta\pi_1^{\ T},\delta\theta_1,\delta\mathbf{R}_2^{\ T},\delta\pi_2^{\ T},\delta\theta_2\right]\left\{\mathbf{M}\begin{bmatrix}\ddot{\mathbf{R}}_1\\ \dot{\omega}_1\\ \ddot{\theta}_1\\ \ddot{\mathbf{R}}_2\\ \dot{\omega}_2\\ \ddot{\theta}_2\end{bmatrix} + \mathbf{S}(\dot{\mathbf{R}}_1,\omega_1,\dot{\theta}_1,\dot{\mathbf{R}}_2,\omega_2,\dot{\theta}_2)\right\} \tag{19}$$

where **M** is the generalized mass matrix,

$$\mathbf{M} = \int_\Omega \mu \begin{bmatrix} \mathbf{g}_{R_1}{}^T\mathbf{g}_{R_1} & \mathbf{g}_{R_1}{}^T\mathbf{g}_{\Gamma_1} & \mathbf{g}_{R_1}{}^T\mathbf{g}_{\theta_1} & \mathbf{g}_{R_1}{}^T\mathbf{g}_{R_2} & \mathbf{g}_{R_1}{}^T\mathbf{g}_{\Gamma_2} & \mathbf{g}_{R_1}{}^T\mathbf{g}_{\theta_2} \\ & \mathbf{g}_{\Gamma_1}{}^T\mathbf{g}_{\Gamma_1} & \mathbf{g}_{\Gamma_1}{}^T\mathbf{g}_{\theta_1} & \mathbf{g}_{\Gamma_1}{}^T\mathbf{g}_{R_2} & \mathbf{g}_{\Gamma_1}{}^T\mathbf{g}_{\Gamma_2} & \mathbf{g}_{\Gamma_1}{}^T\mathbf{g}_{\theta_2} \\ & & \mathbf{g}_{\theta_1}{}^T\mathbf{g}_{\theta_1} & \mathbf{g}_{\theta_1}{}^T\mathbf{g}_{R_2} & \mathbf{g}_{\theta_1}{}^T\mathbf{g}_{\Gamma_2} & \mathbf{g}_{\theta_1}{}^T\mathbf{g}_{\theta_2} \\ & & & \mathbf{g}_{R_2}{}^T\mathbf{g}_{R_2} & \mathbf{g}_{R_2}{}^T\mathbf{g}_{\Gamma_2} & \mathbf{g}_{R_2}{}^T\mathbf{g}_{\theta_2} \\ & & & & \mathbf{g}_{\Gamma_2}{}^T\mathbf{g}_{\Gamma_2} & \mathbf{g}_{\Gamma_2}{}^T\mathbf{g}_{\theta_2} \\ & \textbf{Symmetric} & & & & \mathbf{g}_{\theta_2}{}^T\mathbf{g}_{\theta_2} \end{bmatrix} d\Omega$$

and S is quadratic in velocity,

$$\mathbf{S} = \int_\Omega \mu \begin{bmatrix} \mathbf{g}_{R_1}{}^T(\dot{\mathbf{g}}_{R_1}\dot{\mathbf{R}}_1+\dot{\mathbf{g}}_{\Gamma_1}\omega_1+\dot{\mathbf{g}}_{\theta_1}\dot{\theta}_1+\dot{\mathbf{g}}_{R_2}\dot{\mathbf{R}}_2+\dot{\mathbf{g}}_{\Gamma_2}\omega_2+\dot{\mathbf{g}}_{\theta_2}\dot{\theta}_2) \\ \mathbf{g}_{\Gamma_1}{}^T(\dot{\mathbf{g}}_{R_1}\dot{\mathbf{R}}_1+\dot{\mathbf{g}}_{\Gamma_1}\omega_1+\dot{\mathbf{g}}_{\theta_1}\dot{\theta}_1+\dot{\mathbf{g}}_{R_2}\dot{\mathbf{R}}_2+\dot{\mathbf{g}}_{\Gamma_2}\omega_2+\dot{\mathbf{g}}_{\theta_2}\dot{\theta}_2) \\ \mathbf{g}_{\theta_1}{}^T(\dot{\mathbf{g}}_{R_1}\dot{\mathbf{R}}_1+\dot{\mathbf{g}}_{\Gamma_1}\omega_1+\dot{\mathbf{g}}_{\theta_1}\dot{\theta}_1+\dot{\mathbf{g}}_{R_2}\dot{\mathbf{R}}_2+\dot{\mathbf{g}}_{\Gamma_2}\omega_2+\dot{\mathbf{g}}_{\theta_2}\dot{\theta}_2) \\ \mathbf{g}_{R_2}{}^T(\dot{\mathbf{g}}_{R_1}\dot{\mathbf{R}}_1+\dot{\mathbf{g}}_{\Gamma_1}\omega_1+\dot{\mathbf{g}}_{\theta_1}\dot{\theta}_1+\dot{\mathbf{g}}_{R_2}\dot{\mathbf{R}}_2+\dot{\mathbf{g}}_{\Gamma_2}\omega_2+\dot{\mathbf{g}}_{\theta_2}\dot{\theta}_2) \\ \mathbf{g}_{\Gamma_2}{}^T(\dot{\mathbf{g}}_{R_1}\dot{\mathbf{R}}_1+\dot{\mathbf{g}}_{\Gamma_1}\omega_1+\dot{\mathbf{g}}_{\theta_1}\dot{\theta}_1+\dot{\mathbf{g}}_{R_2}\dot{\mathbf{R}}_2+\dot{\mathbf{g}}_{\Gamma_2}\omega_2+\dot{\mathbf{g}}_{\theta_2}\dot{\theta}_2) \\ \mathbf{g}_{\theta_2}{}^T(\dot{\mathbf{g}}_{R_1}\dot{\mathbf{R}}_1+\dot{\mathbf{g}}_{\Gamma_1}\omega_1+\dot{\mathbf{g}}_{\theta_1}\dot{\theta}_1+\dot{\mathbf{g}}_{R_2}\dot{\mathbf{R}}_2+\dot{\mathbf{g}}_{\Gamma_2}\omega_2+\dot{\mathbf{g}}_{\theta_2}\dot{\theta}_2) \end{bmatrix} d\Omega$$

Similarly, the second and third term in Eq. 16 become

$$\int_\Omega \delta\mathbf{r}^{P^T}\mathbf{f}^P d\Omega + \int_\sigma \delta\mathbf{r}^{P^T}\mathbf{h}^P d\sigma$$

$$= \left[\delta\mathbf{R}_1{}^T, \delta\pi_1{}^T, \delta\theta_1, \delta\mathbf{R}_2{}^T, \delta\pi_2{}^T, \delta\theta_2\right]\mathbf{Q} \tag{20}$$

where Q is the external generalized force vector,

$$\mathbf{Q} = \int_\Omega \begin{bmatrix} \mathbf{g}_{R_1}{}^T\mathbf{f}^P \\ \mathbf{g}_{\Gamma_1}{}^T\mathbf{f}^P \\ \mathbf{g}_{\theta_1}{}^T\mathbf{f}^P \\ \mathbf{g}_{R_2}{}^T\mathbf{f}^P \\ \mathbf{g}_{\Gamma_2}{}^T\mathbf{f}^P \\ \mathbf{g}_{\theta_2}{}^T\mathbf{f}^P \end{bmatrix} d\Omega + \int_\sigma \begin{bmatrix} \mathbf{g}_{R_1}{}^T\mathbf{h}^P \\ \mathbf{g}_{\Gamma_1}{}^T\mathbf{h}^P \\ \mathbf{g}_{\theta_1}{}^T\mathbf{h}^P \\ \mathbf{g}_{R_2}{}^T\mathbf{h}^P \\ \mathbf{g}_{\Gamma_2}{}^T\mathbf{h}^P \\ \mathbf{g}_{\theta_2}{}^T\mathbf{h}^P \end{bmatrix} d\sigma$$

For a Bernoulli beam, the right hand side of Eq. 16, or the virtual work done by the internal force, may be expressed as

$$\int_\Omega \delta\varepsilon^{P^T}\boldsymbol{\tau}^P d\Omega$$

$$=\int_{0}^{L}\left\{EAu_1\delta u_1+EI_y u_2''\delta u_2''+EI_z u_3''\delta u_3''+GJ\phi_{x_2}\delta\phi_{x_2}\right\}dx_c$$

$$=\left[\delta\mathbf{R}_1{}^T,\delta\pi_1{}^T,\delta\theta_1,\delta\mathbf{R}_2{}^T,\delta\pi_2{}^T,\delta\theta_2\right]\mathbf{U} \qquad (21)$$

where u_1 is the axial deformation,

$$u_1=\sqrt{(\mathbf{r}_2-\mathbf{r}_1+\mathbf{r}_o)^T(\mathbf{r}_2-\mathbf{r}_1+\mathbf{r}_o)}\ -\ \sqrt{\mathbf{r}_o{}^T\mathbf{r}_o}$$

u_2 and u_3 are bending displacements in the y and z direction of the neutral axis, which can be obtained from Eq. 11, and **U** is the generalized internal force vector.

Substituting Eqs. 19-21 into Eq. 16, the variational equation of motion of a super-beam element can be written as

$$\left[\delta\mathbf{R}_1{}^T,\delta\pi_1{}^T,\delta\theta_1,\delta\mathbf{R}_2{}^T,\delta\pi_2{}^T,\delta\theta_2\right]\left\{\mathbf{M}\begin{bmatrix}\ddot{\mathbf{R}}_1\\ \dot{\omega}_1\\ \ddot{\theta}_1\\ \ddot{\mathbf{R}}_2\\ \dot{\omega}_2\\ \ddot{\theta}_2\end{bmatrix}+\mathbf{S}(\dot{\mathbf{R}}_1,\omega_1,\dot{\theta}_1,\dot{\mathbf{R}}_2,\omega_2,\dot{\theta}_2)+\mathbf{U}-\mathbf{Q}\right\}=0$$

for all virtual displacements $\delta\mathbf{R}_1$ and $\delta\mathbf{R}_2$, virtual rotations $\delta\pi_1$ and $\delta\pi_2$, and virtual hinge rotations $\delta\theta_1$ and $\delta\theta_2$ that are consistent with the constraints.

5. APPLICATIONS

5.1.1 Analysis of A Deployable Space Structure

The deployable space structure shown in Fig. 3 is a 20 meter long, triangular cross section, joint dominated truss structure, referred to as the Mini-Mast. The structure is used at NASA Langley as a ground test article for the development of research techniques in structural dynamic characterization of large space structures and control of flexible structures. A total of 18 bays, each 1.12 meter long, make up the 20 meter length of the beam above the deployer mechanism. Figure 4 illustrates two deployed bays of the beam design in more detail. One bay of the truss beam consists of three longerons, three diagonal members and a batten triangular truss whose cross-section fits inside a 1.4 m diameter circle. The longerons and diagonal members are connected to batten triangles at each corner (three corner bodies are built into each corner of the batten triangle) by revolute joints. A sketch of a corner body is shown in Fig. 5, primarily to indicate geometric complexity.

The system is deployed/retracted with two bays at a time. During deploying/retraction, the vertices of two batten triangles are held fixed in orientation while a third batten triangle, located between the two fixed ones, rotates about the longitudinal axis. Upward/downward forces are then applied to deploy/retract the system. Revolute joints in corner bodies at each apex of the triangular cross-section of each bay and a nearly over-center hinge in each diagonal member allow the beam truss to deploy/fold into a repeatable beam/stack, as shown in Fig. 4. At the final stage of deploying, mid-hinge of the diagonal member is locked up to ensure that the system becomes a structure. It is reopened as retraction starts. Since each two deploying bays are symmetric to the middle batten triangle, it is sufficient to analyze only one bay of the system.

The objective of the analysis is to determine loadings on flexible members during deployment of one bay of the truss beam. Because of the symmetric geometry of the system, corner bodies are constrained to move on a 1.4 m diameter cylinder during deployment. The longerons and diagonal members are deformed, due to kinematic constraints imposed at the joints, during deployment. Therefore, orientations of revolute joints of the longeron and diagonal member play an important role in design of the truss beam. A set of properly designed revolutes will decrease deformations, hence decreasing the force required to deploy the truss beam. The system is designed such that it is not deformed in its fully retracted position and this then serves as a good starting point to analyze the system response during deployment.

5.1.2 LATDYN Model Description

The model can be simplified by taking the advantage of symmetric geometry of triangular cross section of the system. By constraining the upper triangle to only move along and rotate about the longitudinal axis of the truss, the LATDYN model of one bay of Mini-Mast reduces to 3 flexible longerons, 3 flexible diagonal members, and 2 batten triangles that are modelled as rigid bodies, as shown in Fig. 6. The lower batten triangle, batten triangle A in Fig. 6, is grounded. The upper batten triangle, batten triangle B in Fig. 6, is driven up and down. The batten triangles are connected to the longerons and diagonal members with revolute joints at each corner, respectively. The geometry and material properties of the longeron and diagonal members are listed in Table 1. Initial configuration of the model is chosen with the system in its fully packaged position, as shown in Fig. 6. Locations and orientations of each revolute joints of longeron and diagonal member are listed in Tables 2 and 3, respectively.

5.1.3 Results

The system is deployed by driving the upper triangle in the longitudial direction without constraining its rotation about the longitudial axis. The driving constraint is

$$Z=\frac{L}{T}\left[t-\frac{T}{2\pi}\sin\left(\frac{2\pi t}{T}\right)\right], \quad t<T$$

$$Z=L, \qquad\qquad t\geq T$$

where L is length of the longeron, T is total deploying time, and Z is the height of the upper triangle. The deployment moves the upper triangle a distance L in the z direction in T seconds. In the simulations which follow, T is taken as 1.0 second.

Figures 7-8 show the variations of the bending and twisting moments of the longeron at its midpoint, with the z displacement of the upper triangle. Figures 9-10 show the bending moments of diagonal member A and B while figures 11-13 display the time history of the axial forces of longeron and diagonal member, showing that all truss members in the system are actually in compression after deployed. This tends to increase bending stiffness of the truss beam. Figure 14 shows the LATDYN results for the bending moments of the longeron at the end joining the upper triangle. Also shown are the predictions of the Astro Co.[8]- the original Mini-Mast designer and producer, and the best result authors can achieve using the assumed mode approach. In Fig. 14, both LATDYN and Astro results predict a minimum bending moment in the longeron when it rotates about 50 degrees which is reasonable from the geometry of the structure. Results from the assumed mode approach indicate that the deformation modes used in the analysis are not complete, which is not apparent without another analysis test results.

5.2.1 Flapping Motion of Rotor Blade

A number of problems arise which make it necessary to study the effects of flexibility on blade motion. For example, the affect of flexible motion on the performance, stresses occur in the deformed blade, and interactions between the rotational speed and the natural frequencies of the flexible blade. An aditional complicating fator is that due to the stiffening effect of centrifugal force, natural frequencies of the blade increase as blade rotation speed increases.

A simplified model of an articulated blade with no flap hinge offset or spring restraint is shown in Fig. 15. Initially, the blade is straight and tilted 0.157 radian. With no initial hinge velocity, the blade rotates at a constant speed. Since the centrifugal force always acts radially outward in a plane normal to the rotation axis, it acts as a spring force opposing the blade flap motion, hence initicating the flap motion and deformation.

Simulation of the same flapping blade, using an assumed mode approach, producing a diverging solution as is reported in Reference 5. This is because the geometric stiffening effect is not properly accounted for[6][7].

5.2.2 LATDYN Model and Results

In the simulation of the flapping blade using LATDYN, Rotation speed of the blade is kept constant in each simulation and gradually increased in succeeding simulations, starting with 1 rad/sec and going up to 9 rad/sec. Frequencies are calculated from the transient response of the simulation using a Fast Fourier Tranform.

Figures 16 and 17 show the bending moment of the blade at the midpoint when it rotates at 3 and 6 rad/sec. The results clearly indicate that the natural frequency of the first flapping bending mode increases as the rotation speed increases, due to the centrifugal stiffening effect. Figure 18 displays natural frequencies of the first bending modes for different rotation speeds, compared to the solutions derived by Southwell [9]. Good agreement between the LATDYN results and the Southwell solution is shown: Also shown (dotted lines) in Fig. 18 are different harmonics of the rotor speed. As shown, the natural frequency of the first mode intersects with the third harmonic around 8 rad/sec, fourth harmonic around 4 rad/sec ,fifth harmonic around 3 rad/sec, and so on for higher harmonics. A resonance may then occur when the blade rotates around these speeds. Figure 19 shows the bending moment of the blade when it rotates at 8 rad/sec, which show that the magnitude of the bending blade keeps increasing with time. The frequency of the blade is about three times the rotational speed. The magnitude of the response in Fig. 19 may not increase indefinitely, but may represent a beating phenomenon with the period of the beat depending on the closeness of 8 rad/sec to the intersection point.

CONCLUSIONS

A three dimensional, finite element based simulation tool for flexible multibody systems is presented. Hinge degrees-of-freedom is built into equations of motion to reduce geometric constraints. The approach avoids the difficulty in selecting deformation modes for flexible components by using assumed mode method. The tool is applied to simulate a practical space structure deployment problem. Results of examples demonstrate the capability of the code and approach.

REFERENCES

1. Yoo, W. S., and Haug, E. J.,"Dynamics of Articulated Structures, Part I: Theory and Part II: Computer Implementation and Applications", Journal of Structure Mechanics, Vol. 14, 1986, pp.105-126 and pp.177-189.

2. Bodley, C. S., Devers, A. D., Park, A. C., and Frisch, H. P., "A Digital Computer Program for the Dynamic Interaction Simulation of Controls and Structure (DISCOS)", Vols. 1 & 2, NASA Technical Paper 1219, May 1978.

3.Singh, R. P., VanderVoort, R. J., and Likins, P. W., "Dynamics of Flexible Bodies in Tree Topology - A Computer Oriented Approach", Paper No. AIAA-84-1024, AIAA/ASME/ASCE 25th Structural Dynamics and Materials Conf.,Palm Springs,Ca, May 14-18, 1984.

4. Goldstein, H., Classical Mechanics, 2nd edition,Addison-Wesley Publishing Co., 1980.

5. Wu, S. C., A Substructure Method for Dynamic Simulation of Flexible Mechanical Systems with Geometric Nonlinearities, Ph.D Dissertation, University of Iowa, Iowa City, IA, 1987.

6. Kane, T. R., Ryan, R. R., and Banerjee, A. K.,"Dynamics od A Beam Attached to a Moving Base", AAS/AIAA Astrodynamics Specialist Conference, Vail, Colorado, Aug. 12-15, 1985.

7. McGowan, P. E. and Housner, J. M., "Nonlinear Dynamic Analysis of Deploying Flexible Space Booms", NASA TM-87617, September, 1985.

8. Adams, L. R.,"Design, Development and Fabrication of A Deployable/Retractable Truss Beam Model for Large Space Structures Application", NASA CR-178287, June 1987.

9. Bramwell, A. R. S., Helicopter Dynamics, John Wiley & sons, New York, 1976.

Table 1 Geometry and Material Properties

	Inner Diameter	Outer Diameter
Longeron	0.01491 m	0.02019 m
Diag. Member	0.01115 m	0.01511 m
Modulus of Elasticity = 1.17E+11 N/m^2		
Shear Modulus of Elasticity = 4.0E+9 N/m^2		
Density = 1.616E+3 kg/m^3		

Table 2 Joints Location

Joint Name	X	Y	Z
Longeron A1	-0.60598	-0.34987	0.02379
Longeron B1	-0.23372	0.65954	0.02379
Diagonal A1	-0.58031	-0.34987	0.02379
Diagonal B2	-0.49966	-0.50693	0.02379
Diagonal A1B2	-0.14861	0.26657	0.02379

Table 3 Joints Orientations

Joint Name	X-component	Y-component	Z-component
Longeron A1	0.91838	0.07838	0.38786
Longeron B1	0.64757	-0.65591	-0.38786
Diagonal A1	0.98896	0.09973	-0.10956
Diagonal B2	0.88133	-0.39542	-0.25868
Diagonal A1B2	-0.88076	0.47339	0.01163

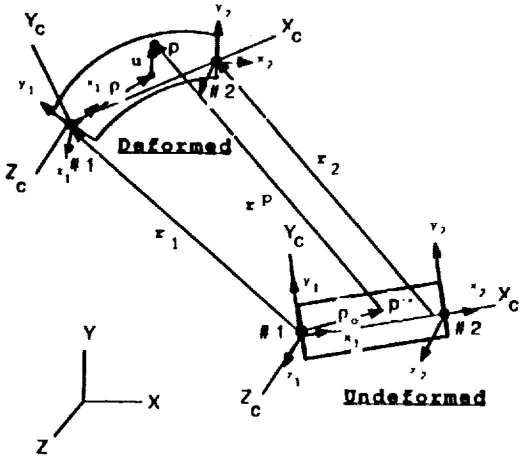


Figure 1 Kinematics of Beam Element

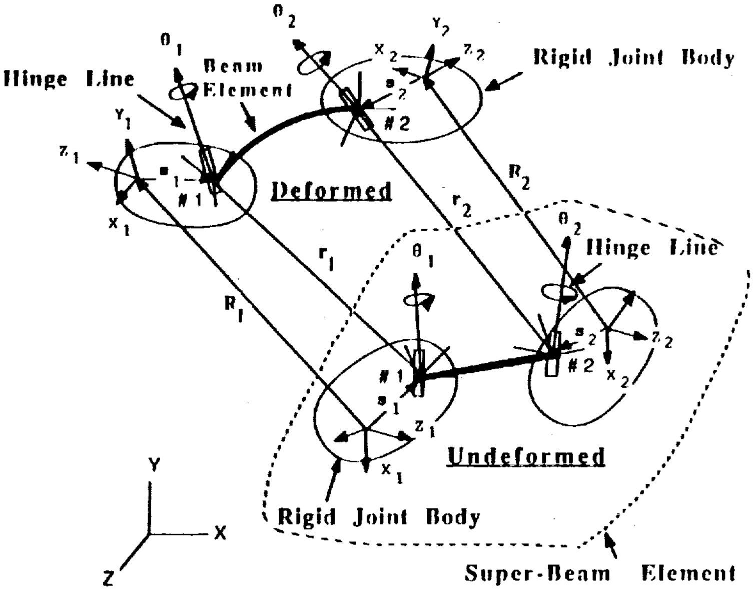


Figure 2 Kinematics of Super-Beam Element

ORIGINAL PAGE IS
OF POOR QUALITY

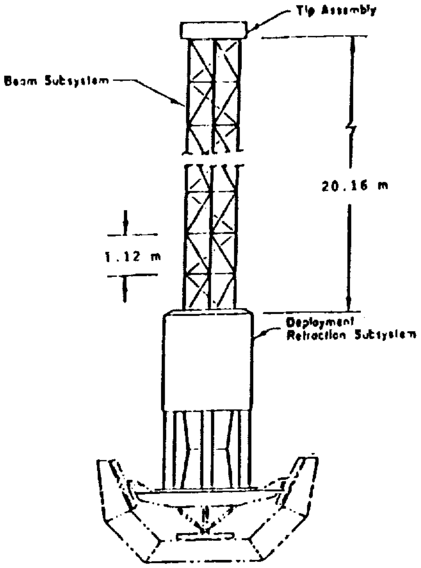


Figure 3 A Deployable Space Structure

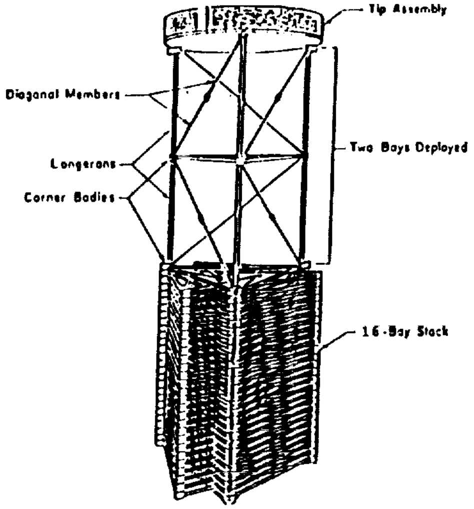


Figure 4 Two Bays Deployed

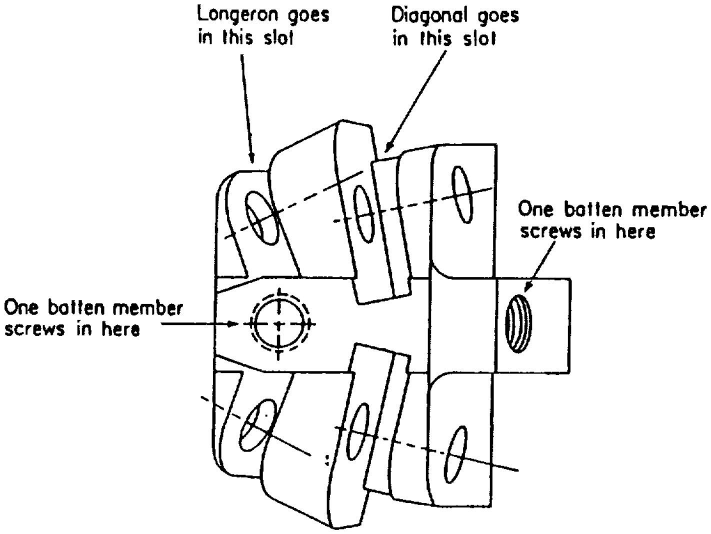

Longeron goes in this slot
Diagonal goes in this slot
One batten member screws in here
One batten member screws in here

Figure 5 A Corner Body

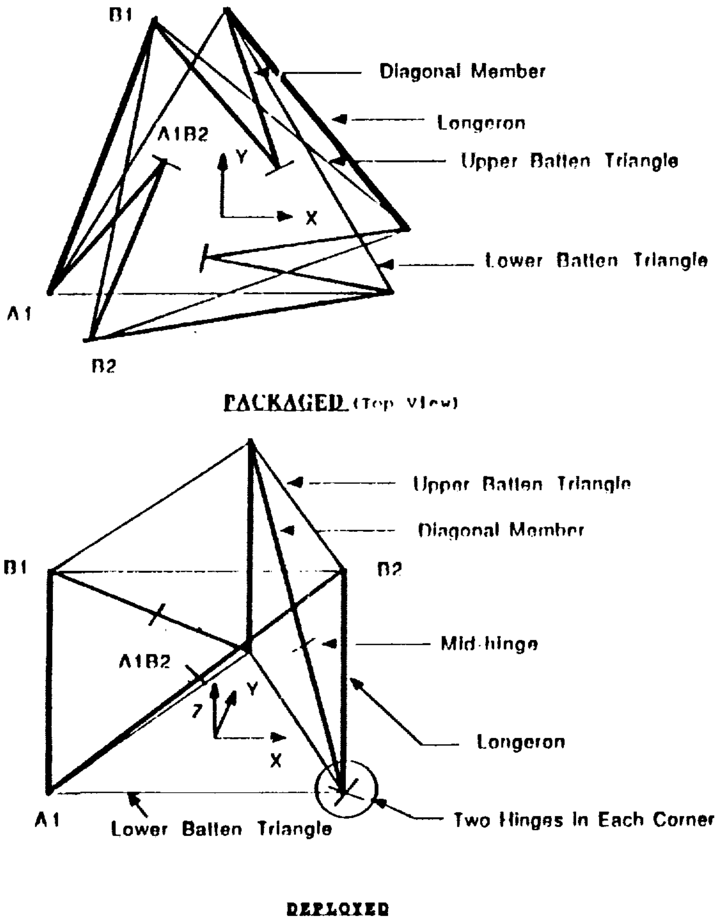


Figure 6 One Bay of Minimast

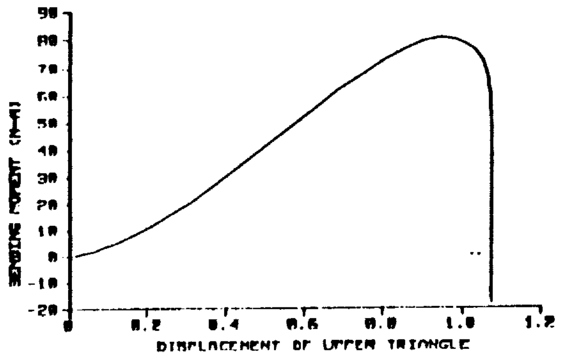


Figure 9 Bending Moment of Diagonal A at Midpoint

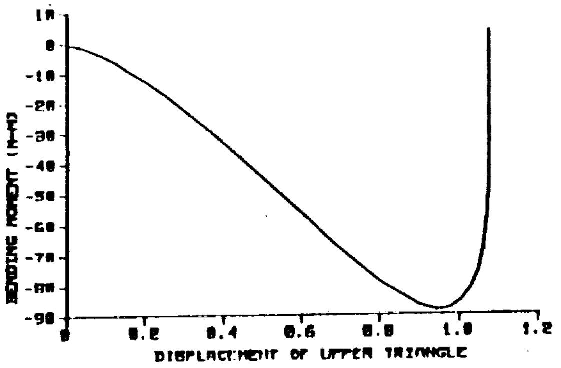


Figure 10 Bending Moment of Diagonal B at Midpoint

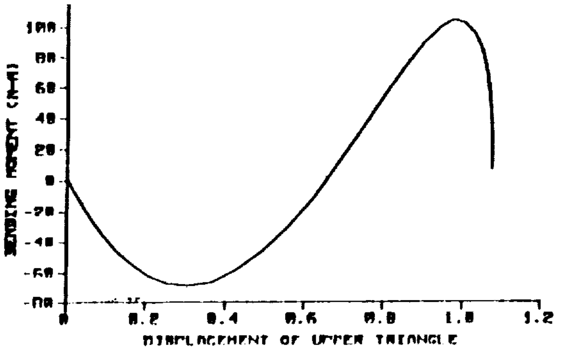


Figure 7 Bending Moment of the Longeron at Midpoint

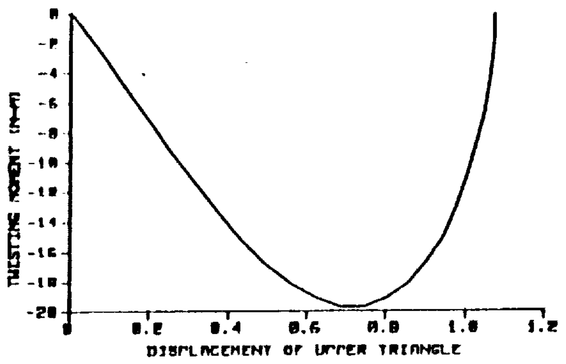


Figure 8 Twisting Moment of the Longeron

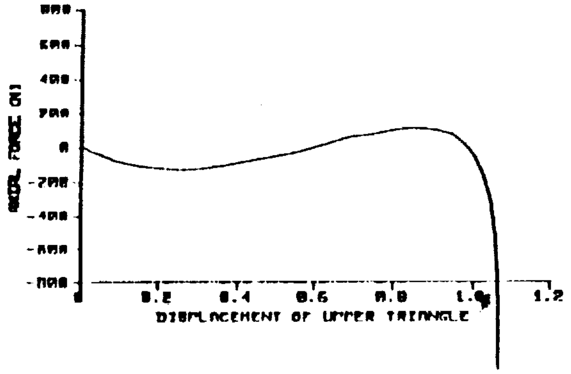


Figure 11 Axial Force of the Longeron

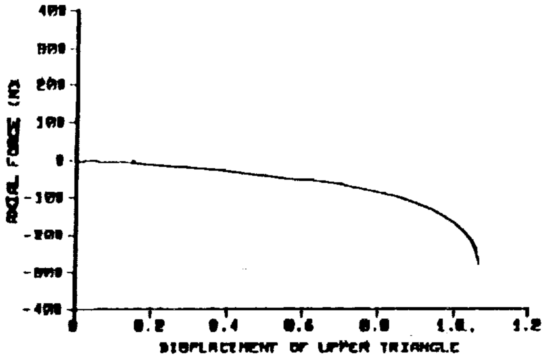


Figure 12 Axial Force of the Diagonal A

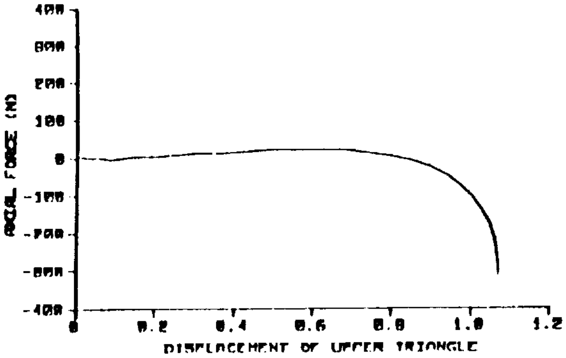


Figure 13 Axial Force of the Diagonal B

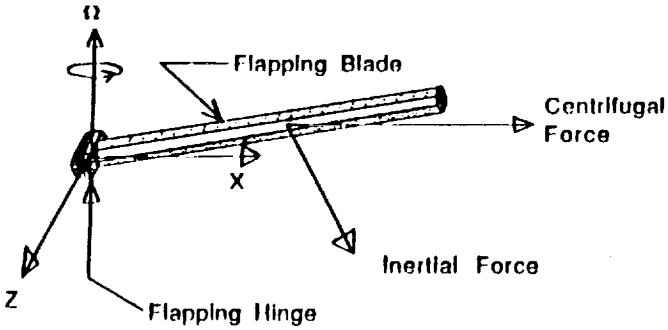


Figure 15 A Flapping Blade

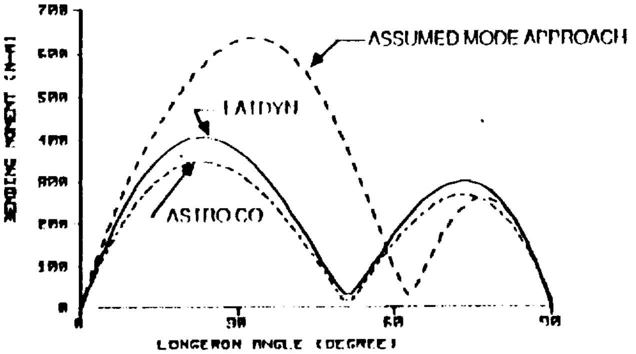


Figure 14 Bending Moment of the Longeron at Upper end

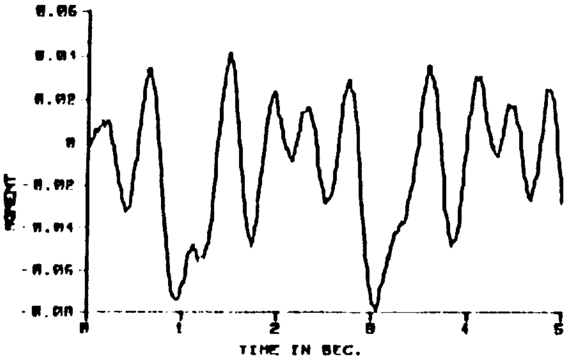


Figure 16 Flapping Bending Moment (Ω=30 rad/sec)

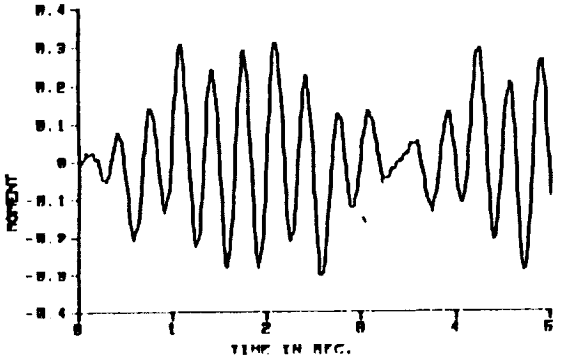


Figure 17 Flapping Bending Moment (Ω=60 rad/sec)

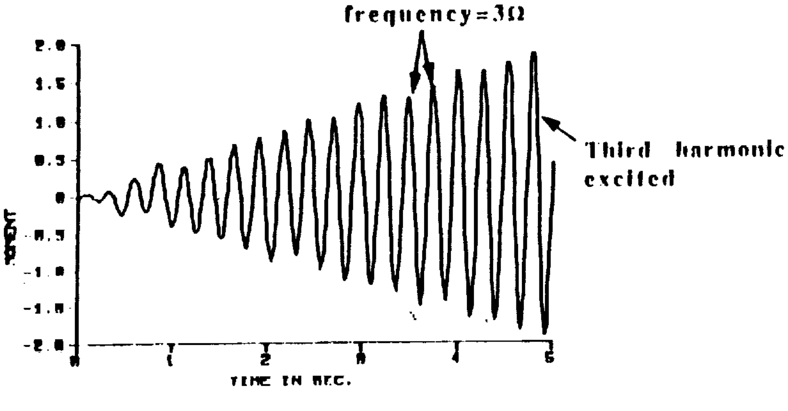


Figure 19 Resonance of Flapping Blade (Ω=8.0 rad/sec)

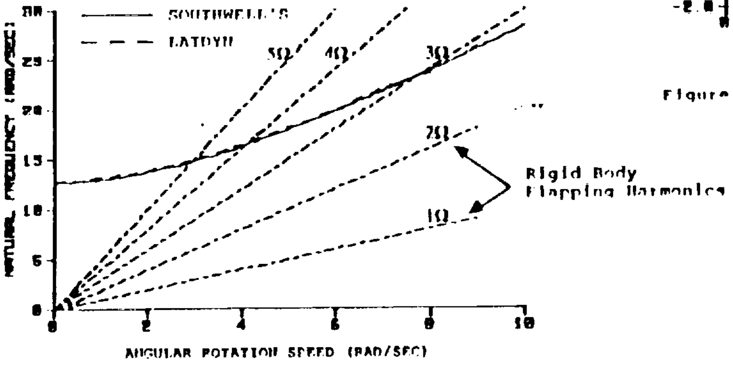


Figure 18 Coalescence of Flexible and Rigid Body Harmonics

Control Design and Simulation of Systems Modeled Using ADAMS

Vikram N. Sohoni, Ph.D
Mechanical Dynamics, Inc.
Ann Arbor, MI

ABSTRACT

This paper presents a technique for control design and simulation using the ADAMS software and a control design software package. For design of control systems ADAMS generates a minimum realization linear time invariant (LTI), state space representation of multi-body models. This LTI representation can be produced in formats for input to several commercial control design packages. The user can exercise various design strategies in the control design software to arrive at a suitable compensator. The resulting closed loop model can then be simulated using ADAMS. This procedure is illustrated with two examples.

1. INTRODUCTION

Due to decreasing cost and increasing reliability of computing hardware, there is increasing interest in control of complex multi-body systems. Before any such systems can be implemented in hardware, it has to be designed using computer software. ADAMS[1] is a commercial software package for modeling and large displacement simulation of non-linear multi-body dynamical systems. It has been successfully used for modeling multi-body systems such as on and off road ground vehicles, aerospace mechanisms and structures and general machinery amongst others. Control design software packages such as Matrixx[2], Pro-Matlab[3] and others have implementations of commonly used control design methodologies. Most popular control design methods require a linear time invariant (LTI) representation of the multi-body system. This LTI representation for a multi-body system can be obtained automatically from ADAMS, in a format suitable for direct input to control design packages.

The paper begins with an introduction to the ADAMS software, followed by a brief review of one of the commonly used state space control methods. The methodology being introduced in this paper is then presented. Two examples are used to illustrate this methodology.

2. OVERVIEW OF ADAMS

ADAMS is a multi-body dynamics software package[4]. As input ADAMS requires a system description, consisting of the mass and inertia data of all bodies in the systems and connections between various bodies. Others data such as environmental effects and simulation parameters have also to be specified. Once a user specifies the analysis mode for the system, ADAMS automatically constructs the governing equations for the system and solves these equations. Bodies can be connected by kinematic or force connections. Kinematic connections represent joints between bodies. These connections are typically represented by algebraic equations. Force connections represent compliant elements in the multi-body system.

2.2 Kinematic connections - Joint Types : Some of the joints allowed by ADAMS are

(1) Revolute - allows for relative rotation about 1 axis.
(2) Translational - allows for relative translation along 1 axis.
(3) Spherical - allows relative rotation about 3 axes.
(4) Universal - allows for relative rotation about 2 axes.
(4) Cylindrical - allows relative rotation about 1 axis and translation about the same axis.
(5) Several types of gear joints are permitted.

(6) Joint primitives - allow for joints to be constructed that represent motions that cannot be represented by one of the joint types stated above.

(7) User can also introduce non-standard kinematic connections through user supplied subroutines.

2.3 Force Connections: Standard forces in ADAMS include

(1) Translational forces - these apply forces along an axis

(2) Rotational forces - apply a torque about an axis

(3) Bushings - apply 3 forces and 3 torques between the two bodies that are connected by this element.

(4) Fields - allow for a user supplied stiffness matrix to be applied between two bodies connected by this element.

(5) Beams - apply a beam element stiffness matrix between the two bodies being connected by this element.

Force elements can have linear as well as non-linear characteristics. Non-standard forces can be introduced via user-supplied subroutines.

2.4 Analysis Modes

ADAMS allows the following analysis modes

(1) Static and Quasi-static : For a dynamical systems this mode computes the equilibrium position of the systems.

(2) Kinematic: This mode only applies for systems with no dynamical degrees of freedom. In this mode displacement, velocity, accelerations and forces for the multi-body system are computed in response to motion inputs.

(3) Transient dynamics: The time response of a dynamical system is computed in this analysis mode.

(4) Eigensolution: The generalized eigenvalue problem for the ADAMS model is solved for eigenvalues and eigenvectors of the model.

(5) State Matrices Computation: LTI representation of the ADAMS model is computed and output in the form of (**A**, **B**, **C** and **D**) matrices.

2.5 Governing Equations

Given the elements that make up the multi-body system, ADAMS automatically generates the non-linear governing equations for the model. In the ADAMS modeling methodology these governing equations are written in a surplus set of coordinates[4], resulting in a system of implicit first order differential-algebraic equations given as,

$$\mathbf{g}\left(\mathbf{Y},\ \dot{\mathbf{Y}},\ t\right) = 0 \qquad (1)$$

where

$\mathbf{Y}$ = vector of solution variables.

t = independent variable time.

These equations can be linearized about a suitable operating point $\mathbf{Y}^* = \left(\mathbf{Y}_0,\ \dot{\mathbf{Y}}_0,\ t_0\right)$ to give

$$\left(\frac{\partial \mathbf{g}}{\partial \mathbf{Y}}\right)\Bigg|_{\mathbf{Y}^*} \Delta\mathbf{Y} + \left(\frac{\partial \mathbf{g}}{\partial \dot{\mathbf{Y}}}\right)\Bigg|_{\mathbf{Y}^*} \Delta\dot{\mathbf{Y}} = 0 \qquad (2)$$

where

Δ indicates small perturbations in the respective variables about their nominal values at $\mathbf{Y}^*$

subscript $\mathbf{Y}^*$ indicates that the matrix is being evaluated about the reference position.

Defining inputs **u**, to and outputs **y** from the plant, it is possible to express the plant model as a LTI form given as,

$$\begin{aligned}\dot{\mathbf{x}} &= \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}\\ \mathbf{y} &= \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u}\end{aligned} \qquad (3)$$

where

x = state vector for the plant

y = vector of outputs from the plant

u = vector of inputs to the plant

A, **B**, **C** and **D** = Constant state matrices

Equation (3) is in a minimum realization form. A condensation procedure is used for reducing equation (2) which is in a surplus coordinates sets to a the minimum realization form of equation (3) [5, 6]. Some of the variables in Δ**Y** are retained in **x**, while others (dependant variables) are condensed out algebraically.

3. CONTROL DESIGN METHODS

Several classical (root locus, Bode plots...) and modern methods (pole placement, LQG, LQG/LTR, H∞,...) are available for control design[7,8]. Most commercial control design packages have procedures that implement these design methods. The commonly used LQR design methodology states the optimal control problem as

Determine gains **k** for a state feedback of the form **u**=-**kx** to

Minimize the functional $J = \frac{1}{2}\int_0^\infty (x' \, Qx + u' \, Ru) \, dt$

Subject to the state equations

$$\dot{x} = Ax + Bu$$
$$y = Cx + Du$$

where

Q and **R** are square weighting matrices

By applying different weighting coefficients in the **Q** and **R** matrices the control designer can exercize different control design options. The solution to the steady state optimal control problem is obtained by solving the algebraic Riccati equation[7].

The essential element of this methodology from the perspective of the control design for multi-body dynamical systems is the need for a LTI representation of the plant model.

4. PROPOSED METHODOLOGY

The methodology being proposed in this paper begins with the an open loop model of the multi-body system created using the ADAMS language elements.

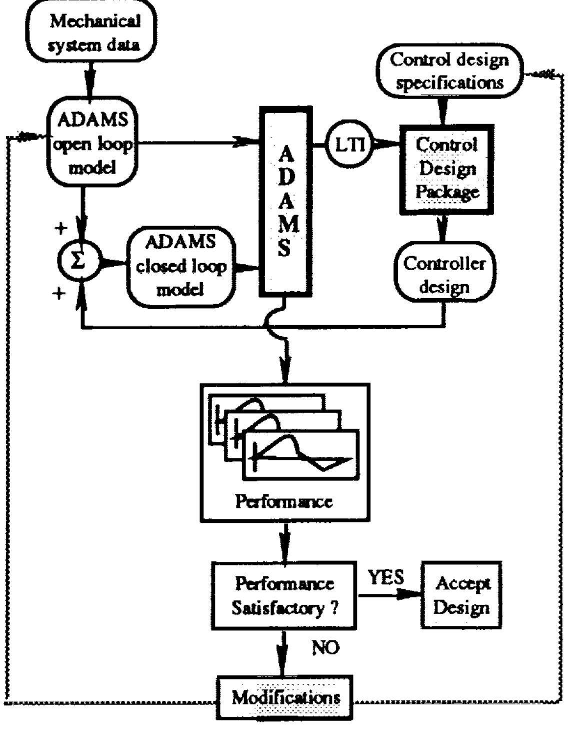


Figure 1. Schematic of Proposed Methodology

This non-linear model can be placed in a nominal position Y^* and linearized to obtain its LTI representation as in equation (3). Presently this LTI representation can be obtained from ADAMS in the Matrixx and Pro-Matlab formats. Translation into any other data format can be easily performed. Once the plant model has been read into the control design package, the control design problem can be defined and a compensator designed using any one of the control design methods stated earlier. This compensator is then implemented using ADAMS language elements. Combining the open loop ADAMS model and the ADAMS representation of the compensator gives the closed loop model. A non-linear time domain simulation of the closed-loop ADAMS model is performed to evaluate its performance. If the closed-loop model performances as expected, the compensator design is accepted. On the other hand if the closed loop system does not perform as desired, the open loop model may have to be modified or the control design problem specifications adjusted. More often than not adjusting the control design specification is sufficient to achieve the desired closed loop performance. This design and simulation process is repeated till a satisfactory compensator design is obtained.

5. EXAMPLES

In this section two examples for control design for multi-body systems will be demonstrated using the methodology presented in earlier sections.

5.1 Balancer 1 Model

The fist examples considered is a mechanical balancer shown in figure 2. This device consists of a frame that is pivoted in ground. At its reference position this frame is in the vertical position. The frame includes of a cross member, on which is located a sliding mass. By applying actuation forces between the slider and the frame it is possible to move the slider on the cross member in a controlled manner.

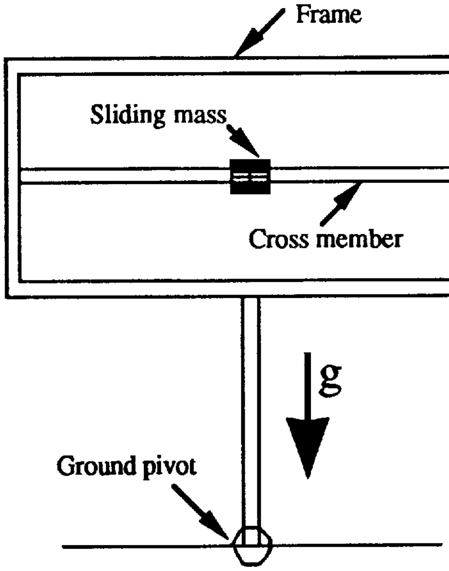


Figure 2 Initial Configuration of Balancer 1 model

From the open loop poles for this system shown in figure 3 it can be seen this system is unstable because of a pole in the right half plane. The control design objective for this system is to stabilize the frame at a position 2.5 degrees from the vertical.

The control inputs to the plant are actuation forces applied on the sliding mass. Outputs from the plant are the angle and velocity of inclination of the frame from the vertical and the position and velocity of the slider on the cross member. Since there is a constant external disturbance, i.e. gravity, integral of the inclination of the frame is also used as an output. This permits a PID control law to be devised for this model. These input/output descriptions are implemented using ADAMS data elements.

The open loop ADAMS model was placed in the vertical position and linearized. The state matrices for the model were read into Matrix$_X$ as being the plant model description. The LQR control design methodology was used to design a static compensator. Different weighting coefficient were tried in the the Q and R matrices till an acceptable compensator design was obtained. When the final compensator design was incorporated into the open loop model, the resulting closed loop model has a pole pattern shown by circles in figure 3. As is evident the compensated model is stable about its nominal position.

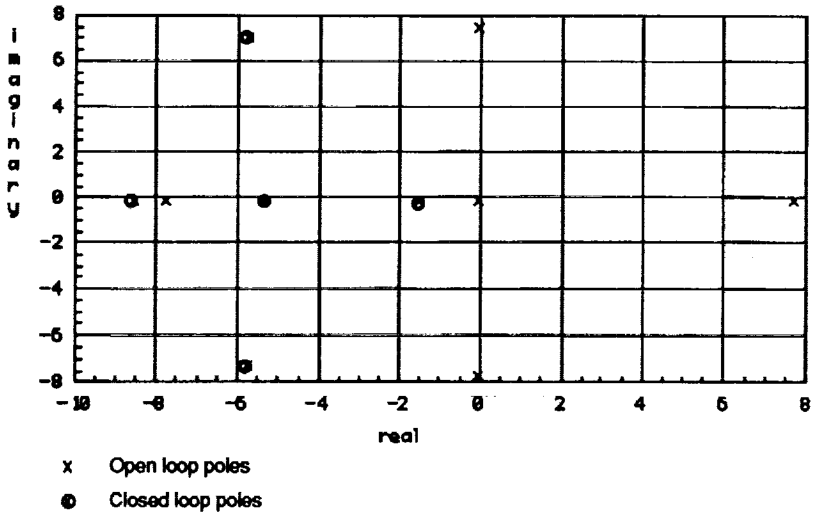


Figure 3 Open and Closed Loop Poles for Balancer 1 Model

To assess the time response of the closed loop systems a command is initiated at 0.5 seconds. This command to orient the frame at 2.5 degrees from the vertical, is ramped in over a period of 2 seconds from initiation. As is shown in figure 4 the frame tilts more than the desired angle of inclination, before stabilizing at the desired position. In the final position the orientation of the frame is as shown in figure 5.

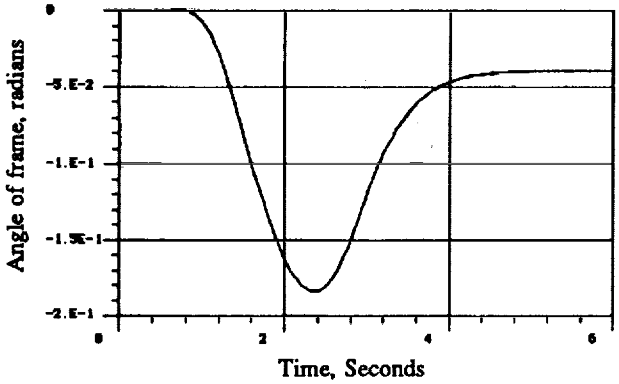


Figure 4 Time Response of frame to command

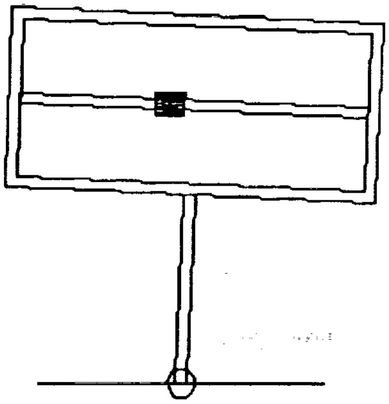


Figure 5. Final Position of balancer frame at time=6 seconds

5.2 Balancer 2 Model

The second model considered is similar to the previous model. This balancer model, as shown in figure 6, differs from the previous example in the respect that the frame cross member axis passes through the pivot axis. In the previous example these two were offset by some distance. The frame pivots in a base and a slider can slide along the axis of a cross member. The slider in this case is a solenoid. By applying appropriate voltage to this solenoid, the slider can be positioned on the cross member. There is one sensor on the base, that is able to measure the angular rotation of the frame with respect to the base. A reduced order filter is used to estimate the angular orientation of the frame. This is a single input single output system. The ADAMS model is linearized about its nominal position, ie, the frame is horizontal and the sliding mass is in the center of its travel. The root locus design methodology in Matrix$_x$ was used to design a feedback control for this model. As shown in figure 7, this systems has 7 poles. Due to a pole in the right half plane, the open loop systems is unstable. As the feedback gain magnitude is increased from 0 (open loop) to -1.0 the poles move as indicated on figure 7. For purpose of this example a gain of -1.0 was chosen. This gain was used to create the compensated model and the model simulated in the time domain in

ADAMS. For simulation purposes a perturbation force is applied to the frame at time 1.0 second. As can be seen, from figure 8, the frame returns to its original position after about 2.5 seconds have elapsed.

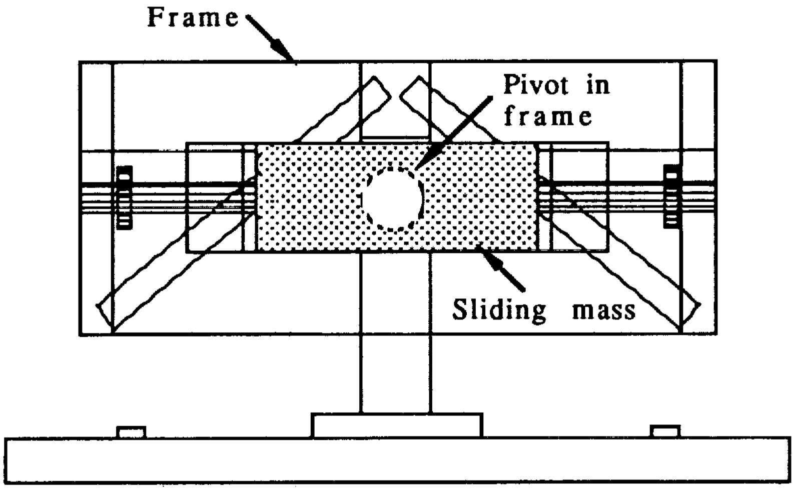


Figure 6 Balancer 2 Model

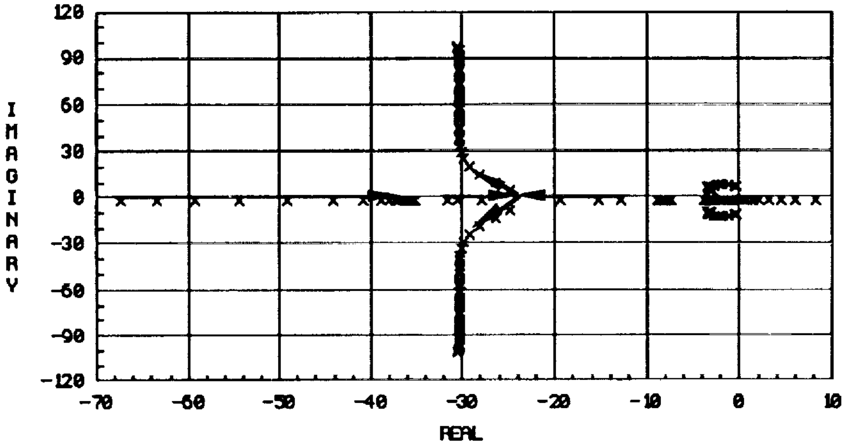
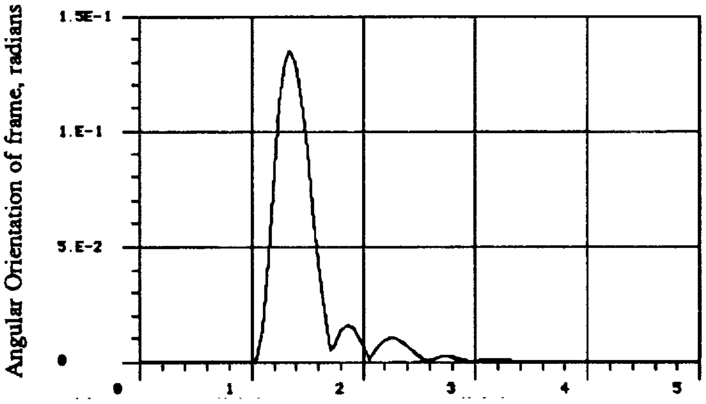


Figure 7 Root locus for balancer 2 model



Time, seconds

Figure 8 Time Response of Frame to a perturbation force

6. SUMMARY

This paper presented a general methodology for combining the powerful physical modeling features of the ADAMS multi-body dynamics software with the advanced control design methodologies in commercial controls software. Application of this methodology for control design of 2 multi-body dynamical systems was demonstrated. In general this technique can be applied to wide variety of complex multi-body dynamical system.

7. ACKNOWLEDGEMENTS

The author would like to thank his colleagues Mr. James McConville and Dr. Robert Ryan, at Mechanical Dynamics, for the assistance provided in preparation of this paper.

8. REFERENCES

1. ADAMS User's Manual, Mechanical Dynamics, Inc. Ann Arbor, MI., October 1989.

2. Matrixx 7.0 User's Guide, Integrated Systems, Inc., Santa Clara, CA., October 1988.

3. Moler, C., J. Little and S. Bangert, 'Pro-Matlab User's Guide', The MathWorks, Inc. Sherborn, MA., October 1987.

4. Chace, M.A.,' Methods and Experience in Computer Aided Design of Large-Displacement Mechanical Systems', *Computer Aided Analysis and Optimization of Mechanical Systems,* Haug,E.J(ed), NATO ASI Series F, Vol. 9, Springer-Verlag, Heidelberg, 1984.

5. Sohoni, V.N and Whitesell, J.,' Automatic Linearization of Constrained Dynamical Models', *ASME J. of Mechanisms, Transmissions and Automation in Design,* vol. 108, No.3, pp. 300-304, September. 1986.

6. Sohoni, V.N and J.R. Winkelman, 'Control Systems Analysis Using ADAMS and MATRIXx', Proceedings of International ADAMS Users Conference, Ann Arbor, MI, Sept. 12-13, 1988, Mechanical Dynamics, Inc.

7. Friedland,B., *Control Systems Design: An Introduction to State-Space Methods,* McGraw-Hill Book Company, 1986.

8. Athans,M.,' Computer-Aided Multivariable Control Systems Design', Lecture notes, MIT, June 1988.

MULTIBODY DYNAMICS MODEL BUILDING USING GRAPHICAL INTERFACES

Glenn A. Macala
Member of Technical Staff
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

ABSTRACT

In recent years, the extremely laborious task of manually deriving equations of motion for the simulation of multibody spacecraft dynamics has largely been eliminated. Instead, the dynamicist now works with commonly available general purpose dynamics simulation programs which generate the equations of motion either explicitly or implicitly via computer codes. The user interface to these programs has predominantly been via input data files, each with its own required format and peculiarities, causing errors and frustrations during program setup. This paper describes recent progress in a more natural method of data input for dynamics programs: the graphical interface.

INTRODUCTION

User interfaces to the variety of commonly used engineering software typically consist of input data files, each with its own required format and peculiarities. This situation inevitably causes errors and frustrations during program setup. Experience has shown that the development of graphical interfaces, full-screen, protected-field editors, and input error detection and correction schemes for commonly used programs not only alleviates most of these problems, but also increases the engineer's productivity. Building these interfaces is not a trivial matter, however, and more often than not the available resources are spent in creating program capabilities rather than easy to use interfaces to the programs.

At the Jet Propulsion Laboratory, a project known as CASCADE (Computer Aided System Control And Dynamics Environment) has been charged with the task of producing graphical interfaces for a set of programs covering the design, analysis, and simulation of control systems for spacecraft. The CASCADE system has made significant progress in the area of interfaces for multibody dynamics. Prototype interfaces now exist for three programs under the general category of dynamics: a mass property calculator, an equations of motion generator for rigid multibody systems, and an equations of motion generator for elastic multibody systems. An additional fallout of this work has been the development of a PC based program to depict three-dimensional animations of rigid multibody spacecraft. The following sections describe the development of these interfaces, in chronological fashion, and will provide an introduction to the existing prototype interfaces and animation program.

MASS PROPERTY FACILITY

Complex spacecraft typically consist of hundreds of pieces of structure. The mass properties of the spacecraft are determined by weights and balances engineers via combining the individual mass properties of these elements according to their assembled locations and orientations on the spacecraft. Dynamics and control engineers need to know mass properties for major portions of the spacecraft in order to perform multibody dynamics simulations. Therefore, comprehensive mass properties lists of spacecraft subassemblies are generated for them by the weights and balances engineers. Invariably, however, there is some combination of subassemblies that is not on a list and is subsequently found to be needed.

A locally developed program that computes composite mass properties for a collection of spacecraft subassemblies, commonly referred to as bodies, had been used for years to calculate the mass properties for combinations of subassemblies missing from the master list. This program used the parallel axis theorem to calculate the mass properties of collections of bodies whose individual mass properties were known. The original user interface to this program was a "question and answer" type but had subsequently been changed to an input data file type. This program was a natural for development of a graphical interface. It was simple, yet had elements common to the more complex dynamics programs in use. Therefore, the first interface was developed using this mass property calculator and was named the MPF - Mass Property Facility.

The first development decision to be made was a hardware issue. What computers and terminals would comprise the system and where would the software be based? The engineers who would be using the program had IBM PC's with Hercules graphics cards. However, the majority of the dynamics software was hosted on a VAX 11-780 and could not be ported to the PC's because of sheer size. Therefore, the decision was made to use the PC's as terminal emulators and to host all of the interface software on the VAX. The PC based terminal emulator was able to emulate a VT100 alpha-numeric terminal and a Tektronix 4010 graphics terminal and was able to switch between emulations via a software command (receipt of certain byte sequences). Therefore, detailed line drawings and cursor position reporting could be done in the Tektronix mode. Full-screen editor style data entry could be done in the VT100 mode.

The resulting features of the MPF can best be conveyed by the following commentary on how an engineer would use the program to calculate the composite mass properties of a set of bodies. The engineer starts the terminal emulator on his PC, connects to the VAX, and logs on to his account. He gives the command to run the MPF. Figure 1 depicts the opening graphics menu of the MPF. The area which takes up most of the left half of the screen contains labeled boxes which will activate a host of functions from file manipulation to operations on the bodies of the system. The upper right portion of the screen contains small pictures of common objects: a cylinder, a rectangle, and a frustrum. A scale is also located there to indicate relative size in pixels. The lower right corner of the screen will contain a 5/12 scaled down version of the current full-screen graphics page on which the system of bodies is drawn. A two body system is shown in this figure.

The engineer selects bodies from the upper right portion of the screen and places them on the full-screen graphics page. The bodies are to be used as sketches of the bodies in the system under study. Their sizes, shapes, and positions on the graphic screen have no effect on the subsequent mass property calculations. They are used by the engineer as mnemonic aids to represent the bodies of the system under study.

The engineer proceeds by using the PC's arrow keys to move the on-screen cursor until it points to one of the bodies in the upper right portion of the screen. He then pushes the space bar to select the body. He is asked the screen dimension of the body and may accept a default by pushing the "Enter" key. The screen clears to the graphics page and a cursor appears. The arrow keys are used to move the cursor until the desired position for the body is located. Pushing the space bar causes the body to be drawn at that screen location. He may continue to place bodies of this type around the screen, or return to the main menu by pushing the "R" key. At the main menu, he may similarly select other types of bodies to be placed in the system.

After having selected and placed the bodies of the system on the graphics page, mass properties for each body must be specified. The engineer selects the "Values" function from the main menu, either by using the cursor keys to "point and shoot" as with the bodies, or alternatively, by pushing the "V" key. Note that each function on the main menu has one capital letter. If the PC key corresponding to this letter is pushed, that function will be activated.

Once "Values" has been selected, the full-screen graphics page consisting of the previously placed bodies is presented to the engineer. He selects a body, using the cursor method, and a full-screen menu appears for that body as shown in Figure 2. This menu is actually a full-screen, protected-field editor that lets the engineer type in a name for the body, its mass, mass center location, and inertia tensor. These fields are shown in reverse video and the cursor cannot move outside of them. Therefore, the engineer need not worry about where he needs to type data. The bottom line of the menu contains some help information, but on-line help is available by pushing the F10 function key. The VAX's help facility is then activated and the appropriate help text and subjects for this menu are available for perusal by the user. When he is done reading the help text, pushing the "Enter" key will return the full-screen menu to the display.

When the engineer has completed filling in the data for this menu, he pushes the Alt-F1 function key combination, and the data can be saved. After similarly entering data for each of the other bodies in the system, he is ready to calculate the resulting system mass properties. He does this by returning to the main menu and selecting "System". An asterisk appears in the box next to "System" indicating that the items enclosed by the larger box, "Move", "Delete", and "Values", now will effect the entire system rather than individual bodies of the system. He then selects "Values" and the program proceeds to read all of the data that has been entered for all bodies in the system. If an error is detected during the reading of the data, say the engineer forgot to enter the mass of a body called Antenna, the program would display the following message on the screen: "An error has been detected. The mass for the body named Antenna must be a floating point number. Push the Enter key and the cursor will be positioned at the field where the error occurred. You may then correct the error, save the data, and try again." When the engineer pushes the "Enter" key, the full-screen field-protected mass property editor appears for the Antenna body. The cursor appears in the "Mass" field and the engineer would see that he forgot to enter the mass. He would enter the mass, save the data, and return to the main menu where he would again select "Values". If no errors were detected this time, a file would be automatically written in the manner required by the original mass property program.

The MPF then runs the original mass property program and feeds it this file for its required input. The old program calculates the mass properties of the system and then writes the answer to another file. The MPF then reads this file and displays it to the engineer using the full-screen mass property menu format.

Note that the engineer may save his work at any time by selecting "Save picture file" from the main menu and specifying a name for the file. He may copy previously created mass property data into currently existing bodies by selecting "Get mass property file" from the main menu. A full-screen file selection menu appears, as shown in Figure 3, and the user selects a file by using the cursor keys to highlight the file name of interest and then pushing the "Enter" key. The main menu returns and the user then selects the "Deposit" function. The graphics screen appears and the user points at the body he wishes to deposit the data into, using the cursor, and then presses the space bar. If he returns to the main menu and selects "Values", he can then inspect the data just deposited into this body via the full-screen mass property editor.

He may delete bodies from the system by selecting "Delete" from the main menu. He may "pretty up" the graphics page by moving the bodies of the system around using "Move". All of these functions work similarly to the methods described earlier: the graphics page appears, the user selects the body he is manipulating using the cursor, and so forth.

SYMBOLIC DYNAMICS FACILITY

Once the MPF had been created and exercised, improvements were made and bugs were removed. It was then time to move on to the next interface in the series: the SDF - Symbolic Dynamics Facility. The basis for this interface was the SD/Exact [1] program. This program can compute the symbolic equations of motion for a system of hinge-connected rigid bodies. This is typically how the dynamics and control engineer represents articulated portions of a spacecraft for simulation purposes. For example, an antenna may be attached to the main bus of a spacecraft by a motor/gimbal combination that allows the antenna to be moved relative to the bus. This antenna may be required to point at a receiver on the Earth. It would be of interest to the dynamics and control engineer to simulate this two-body system to confirm that it met its pointing requirements.

The SD/Exact program requires that a file be created for input to the program. This file contains a description of each body in the system to be simulated. The description for each body consists of specifying its mass properties and also how it connects to other bodies in the system. Note that the MPF allows each body's mass properties to be specified. Accordingly, we had only to augment the MPF interface with a scheme to record body connection information in order to create the SDF interface.

Figure 4 presents the opening menu of the SDF. Note the similarity to the menu of the MPF (Figure 1). The most notable difference is in the upper right hand corner of the menu. Now in addition to bodies, there are "hinge" or "joint" icons. These icons allow the user to select 1, 2, or 3 degree-of-freedom rotational hinges or a 1 degree-of-freedom translational hinge to connect between bodies of the system. These were chosen because they are the hinge types supported by the SD/Exact program.

The engineer proceeds to set up the system of bodies as he did with the MPF. Once he has finished placing the bodies of the system on the full-screen graphics page and specifying their mass properties via the "Values" function, he concentrates on placing the hinges. This proceeds in a manner analogous to selecting and placing the bodies. For example, if the engineer wished to allow 1 degree of rotational freedom between two bodies of his system, he would select the 1 degree-of-freedom joint and place it between these bodies on the full-screen graphics page. He would then return to the main menu and select the "Connect" function. The graphics screen would re-appear and he would select the hinge he just placed using the cursor. He would then be asked to select the first body the hinge connects. He would use the cursor to do this. He would then be asked to select the second body the hinge connects. Again, he would "point and shoot" with the cursor. The SDF would then record this connection information.

One last major step must be taken before the system is completely specified. Just as "Values" is used to enter mass properties for bodies in the system, it is also used to enter information about the hinges in the system. Figure 5 shows the full-screen hinge menu that appears when "Values" is used on a 1 degree-of-freedom rotational hinge. This menu works like the mass properties menu: protected fields allow the user to enter data only in certain fields. As with a body, the user fills in a name for the hinge and the location of the hinge. Now, however, he must specify the direction cosines of the hinge: along what spatial vector it turns. Also note that the names of the bodies connected by this hinge are displayed at the top of the menu. This is the result of using the "Connect" function.

Once all hinges have been specified via "Values", the system is completely specified and the user can select "Generate equations" from the main menu. The SDF then reads all the data and looks for errors. If an error is found, as with the MPF, the user is informed and automatically placed in the menu of interest, mass properties or hinge properties, to correct the error. Once all errors have been corrected, the SDF automatically writes the required input file for SD/Exact and runs SD/Exact. SD/Exact generates equations of motion for the system, in terms of FORTRAN subroutines, and stops. The SDF then takes these subroutines and appends them to a bare-bones shell of an ACSL [2] simulation program it creates based on the problem. At this point, the user may exit the SDF and use this simulation program as he normally would have had he written it himself by hand.

FLEXIBLE DYNAMICS FACILITY

After the SDF had been completed, it was time to move on to the last interface in the dynamics series: the FDF - Flexible Dynamics Facility. The basis for this interface was the DISCOS [3] program. This program can compute the equations of motion for a system of hinge-connected rigid and elastic bodies and is also a self-contained simulation program. Typically, the dynamics and control engineer uses DISCOS as a final, full-fidelity check on his control system design. For example, the Earth pointing antenna example mentioned in the SDF section was modeled there as a rigid antenna connected to a rigid bus. The antenna may have actually been elastic in nature but, for simplicity, may have been modeled as a rigid body during the design phase. Now the engineer wants to use DISCOS to simulate it as elastic to make sure that his design assumptions were valid and that his design still meets requirements.

The DISCOS program requires that a file be created for input to the program. This file contains a description of each body in the system to be simulated. The description for each rigid body consists of specifying its mass properties and also how it connects to other bodies in the system. In addition, each elastic body in the system must have its elasticity properties described in terms of its eigenvalues and eigenvectors. Note that the SDF provides for the specification of mass properties and connection properties. Accordingly, we had only to augment the SDF interface with a scheme to record a body's elasticity information in order to create the FDF interface.

Figure 6 presents the opening menu of the FDF. Note the similarity to the menu of the SDF (Figure 4). The most notable change is in the upper right hand corner of the menu. The 4 types of hinges of the SDF have now been replaced by a general 6 degree-of-freedom hinge: 3 translations and 3 rotations.

The engineer proceeds to set up the system of bodies and hinges as he did with the SDF. Once he has finished placing the bodies and hinges of the system on the full-screen graphics page, he then uses "Values" to specify their mass properties and hinge properties and "Connect" to specify hinge connections between bodies. If a body is rigid, this procedure is identical to that for the SDF with minor variations in the content of the hinge menus to accommodate the DISCOS type of hinge. If a body in the system is elastic, however, and the engineer uses "Values" to select that body, the menu of Figure 7 is displayed.

For DISCOS, an elastic body consists of a set of joints where lumped masses and their self-inertias reside. The locations and mass properties of joints are available from finite element analysis of the elastic body. The menu shown in Figure 7 requires the engineer to place the "joints" or "grid points" on the elastic body in question. Once this has been done, in the usual "point and shoot" manner, the engineer may specify the mass properties of each joint by selecting the "Mass properties" function on this menu. He is then able to select the joint of interest with the cursor and fill in the usual mass properties menu. The modes (eigenvalues) and their damping are specified by selecting the "Modes" function from the menu. Again, a full-screen editor type menu opens up to allow the engineer to type in the required information. The mode shapes of each joint are specified by using the "Mode shapes" function from this menu. Once again, the engineer "points and shoots" to select a joint. A full-screen menu appears and can then be filled in with mode shape information.

Once the engineer has placed all joints for each flexible body, and has entered all elastic data, the system is completely specified and the user can select "Setup Discos file" from the main menu. The FDF then reads all of the data and looks for errors. If an error is found, as with the SDF and MPF, the user is informed and automatically placed in the menu of interest to correct the error. Once all errors have been corrected, the FDF automatically writes the required input file for DISCOS. At this point, the user may exit the FDF and execute the DISCOS program as he normally would have had he written the input data file himself by hand.

WRITEANI

A PC based program, WRITEANI, has been a by-product of the graphical interface work done on the SDF. The program itself was written for amusement by the author on his home computer, but was subsequently integrated with the SDF. WRITEANI reads an "animation" file created by the SDF and then automatically writes a FORTRAN animation program which can be run on a PC equipped with a Hercules graphics card. The resulting animation program then reads the file containing the results (hinge positions at each time step) of an SDF rigid multibody simulation run on the VAX. One 3-D picture of the multibody system is drawn for each simulation time step. Hidden-line removal is performed on each body, but not on overlapping bodies. Perspective is also provided. Figure 8 shows one frame of such a spacecraft animation.

Although the animation program was originally written for amusement purposes, there have been several occasions when engineering insight has been obtained by observing an animation of the results of a simulation. The program has also been found useful for explaining complex dynamic effects to interested parties.

SUMMARY

The summaries of the various dynamics interfaces presented in this paper have only touched on their basic operation. These interfaces present the engineer with a logical, consistent format for visually assembling dynamic systems and for specifying their properties along with on-line help and extensive error checking. Although some of these dynamics programs are very complex in nature and require lengthy study and use in order to completely master their capabilities, it has been estimated that setup time for these programs has been reduced by an order of magnitude through the use of these interfaces. These interfaces are considered to be working prototypes. It would be desirable to port them to workstation class computers in the future for improvements in graphics drawing speed and also for incorporating windowing techniques into the interfaces.

ACKNOWLEDGMENT

The research described in this paper was performed by the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

REFERENCES

1. Rosenthal, D. E., and Sherman, M. A., "High Performance Multibody Simulations via Symbolic Equation Manipulation and Kane's Method," Journal of the Astronautical Sciences, Vol. 34, No. 3, July-September 1986, pp. 223-239.

2. Mitchell and Gauthier, "Advanced Continuous Simulation Language (ACSL) Reference Manual," 1987, Mitchell and Gauthier Associates, Concord, Mass.

3. Bodley, C. A., et al., "A Digital Computer Program for the Dynamic Interaction simulation of Controls and Structure (DISCOS)," NASA Technical Paper 1219, 1978.

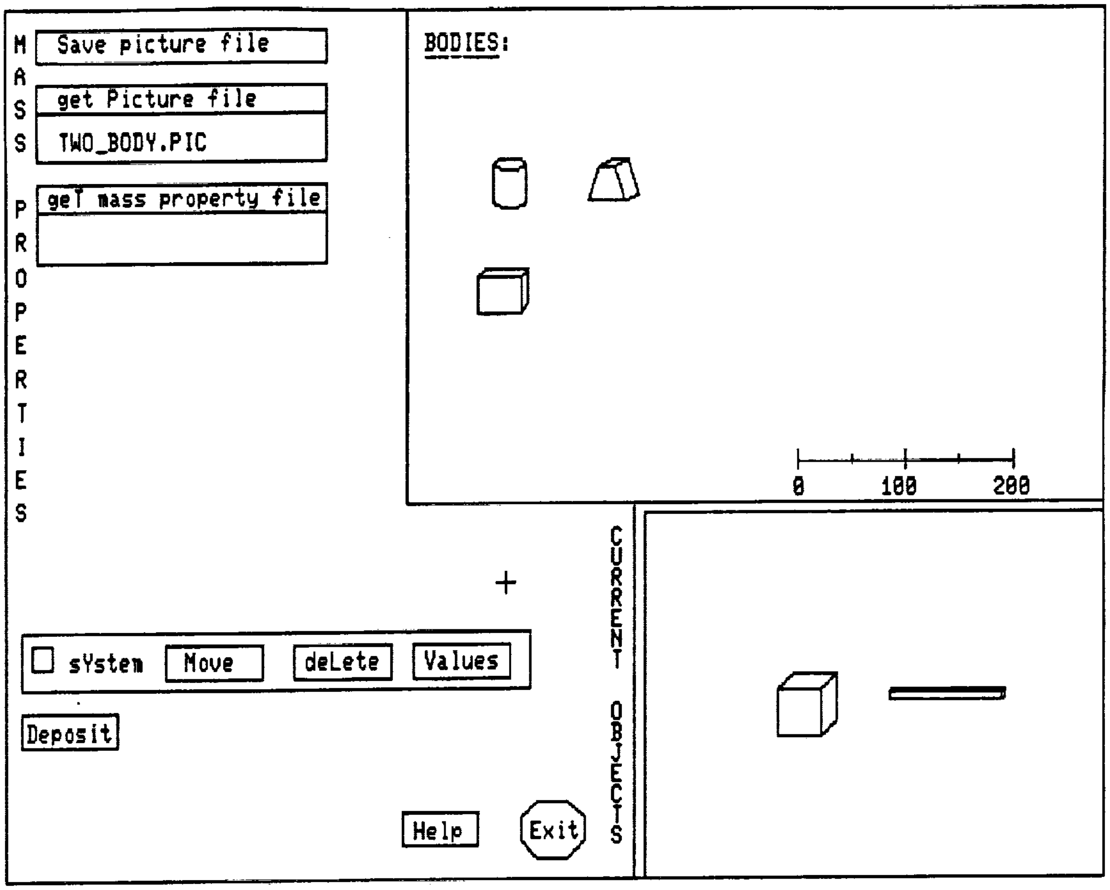
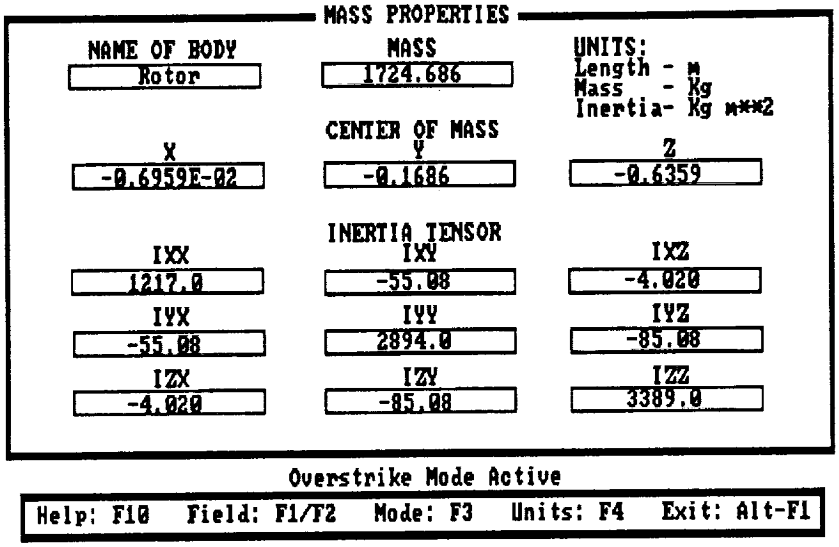
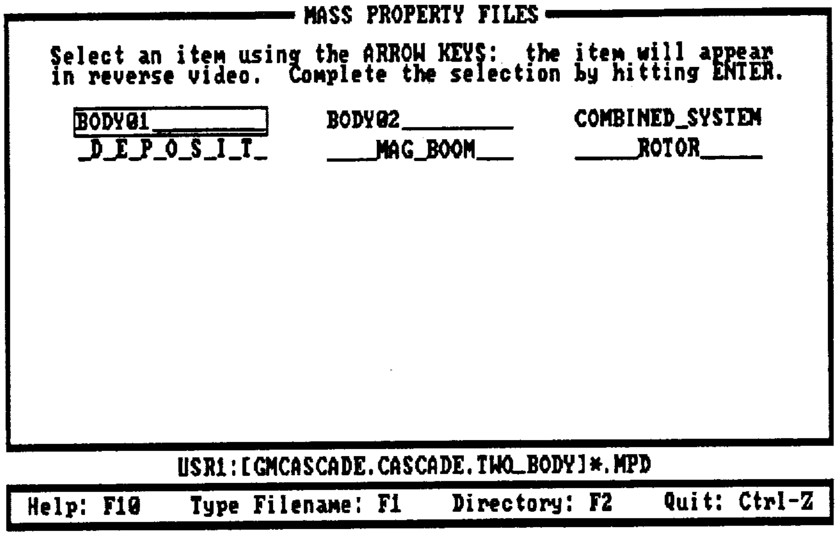


Figure 1. MPF Main Menu



```
                          MASS PROPERTIES
   NAME OF BODY             MASS              UNITS:
   [    Rotor    ]     [  1724.686  ]         Length - m
                                              Mass   - Kg
                                              Inertia- Kg m**2
                        CENTER OF MASS
        X                    Y                    Z
   [ -0.6959E-02 ]     [  -0.1686   ]      [  -0.6359   ]

                        INERTIA TENSOR
       IXX                  IXY                  IXZ
   [   1217.0    ]     [   -55.08   ]      [  -4.020    ]
       IYX                  IYY                  IYZ
   [   -55.08    ]     [   2894.0   ]      [  -85.08    ]
       IZX                  IZY                  IZZ
   [   -4.020    ]     [   -85.08   ]      [   3389.0   ]

                      Overstrike Mode Active
 Help: F10   Field: F1/F2   Mode: F3   Units: F4   Exit: Alt-F1
```

Figure 2. Full-Screen, Protected-Field Mass Properties Editor



```
                         MASS PROPERTY FILES
   Select an item using the ARROW KEYS: the item will appear
   in reverse video. Complete the selection by hitting ENTER.

   BODY01________      BODY02________      COMBINED_SYSTEM
   _D_E_P_O_S_I_T_     ____MAG_BOOM___     _____ROTOR_____

                 USR1:[GMCASCADE.CASCADE.TWO_BODY]*.MPD
 Help: F10   Type Filename: F1   Directory: F2   Quit: Ctrl-Z
```

Figure 3. Full-Screen File Selection Menu

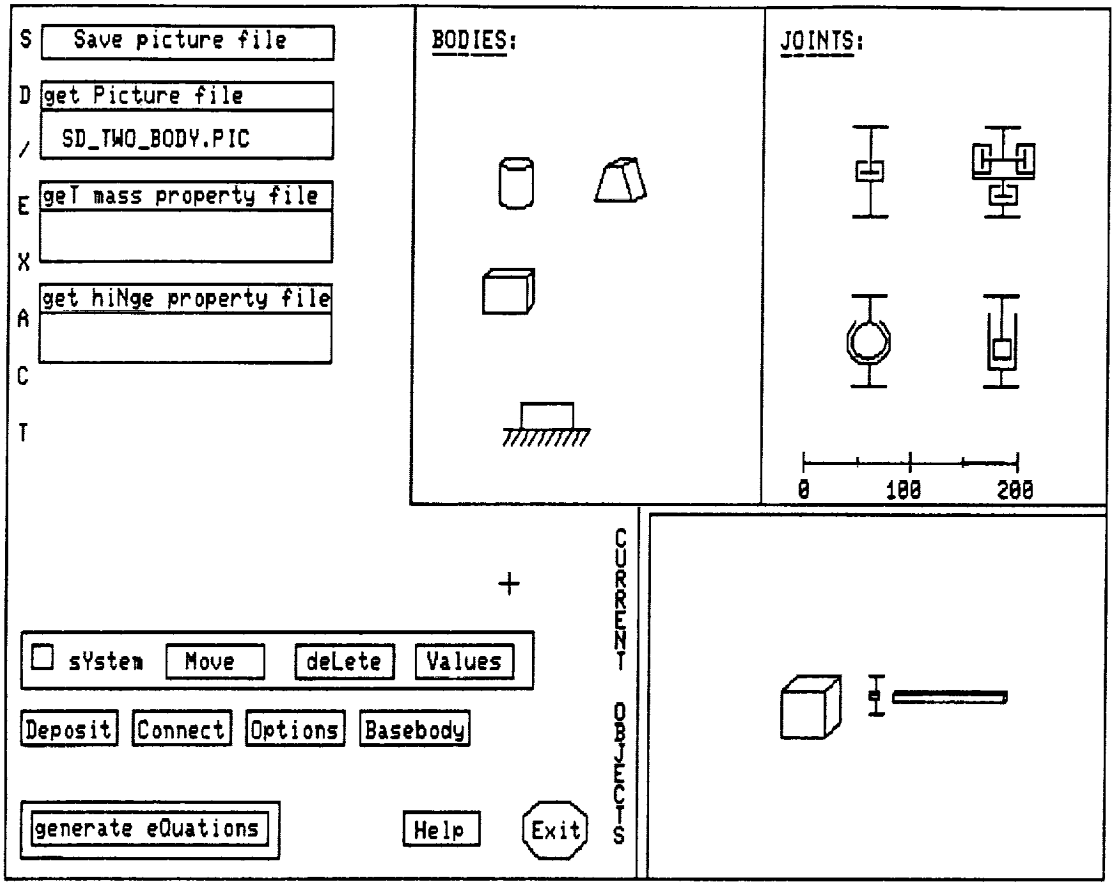
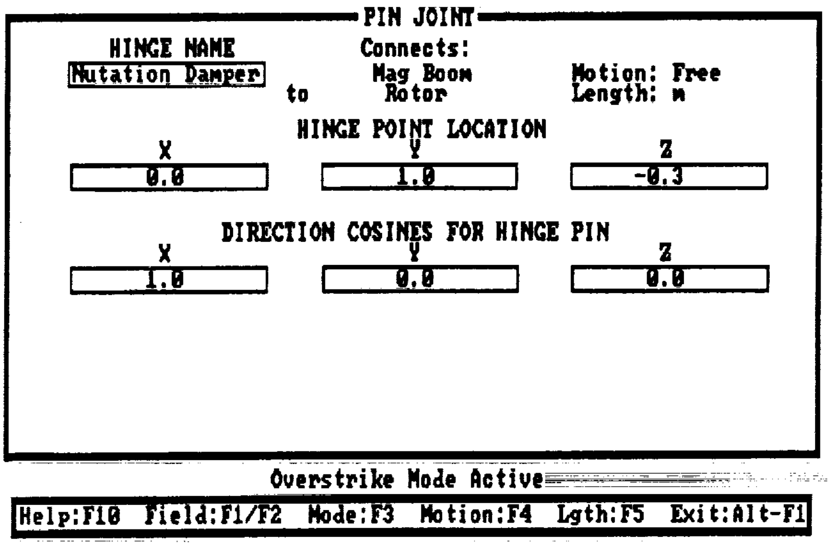


Figure 4. SDF Main Menu



PIN JOINT

HINGE NAME: Nutation Damper

Connects: Mag Boom to Rotor

Motion: Free
Length: m

HINGE POINT LOCATION

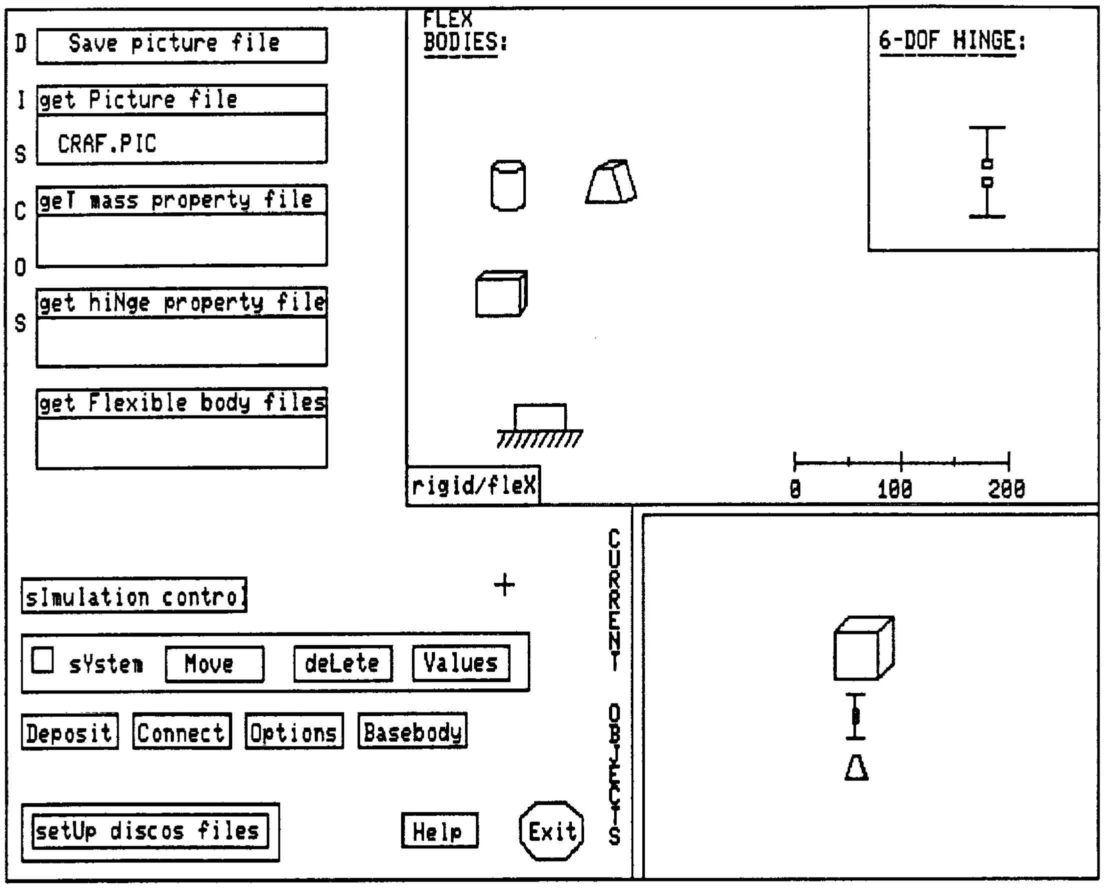
X	Y	Z
0.0	1.0	-0.3

DIRECTION COSINES FOR HINGE PIN

X	Y	Z
1.0	0.0	0.0

Overstrike Mode Active

Figure 5. Full-Screen, Protected-Field Hinge Property Editor



get Picture file: CRAF.PIC

Figure 6. FDF Main Menu

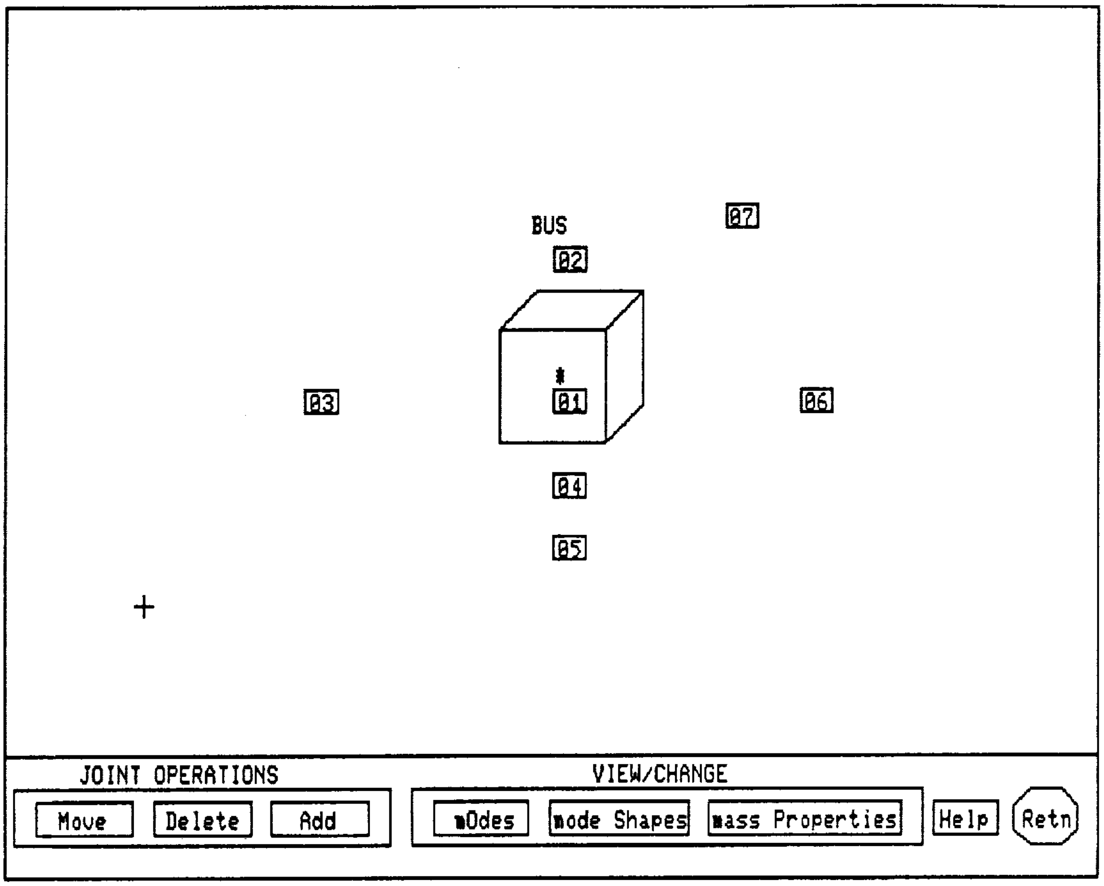


Figure 7. FDF Values Menu for Flexible Body

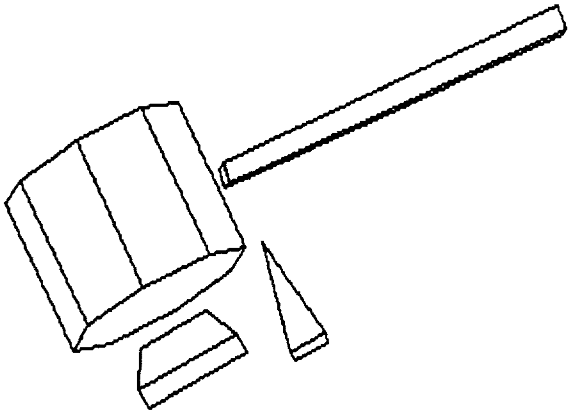


Figure 8. One Frame from Animation Display

N90-23083

CONTROLLING FLEXIBLE STRUCTURES WITH SECOND ORDER ACTUATOR DYNAMICS

Daniel J. Inman
Professor

Jeffrey W. Umland
Research Assistant

John Bellos
Research Assistant

Department of Mechanical and Aerospace Engineering
University at Buffalo
Buffalo, NY 14260

ABSTRACT

This paper examines the control of flexible structures for those systems with actuators that are modeled by second order dynamics. Two modeling approaches are investigated. First a stability and performance analysis is performed using a low order finite dimensional model of the structure. Secondly, a continuum model of the flexible structure to be controlled, coupled with lumped parameter second order dynamic models of the actuators performing the control is used. This model is appropriate in the modeling of the control of a flexible panel by proof-mass actuators as well as other beam, plate and shell like structural members. The model is verified with experimental measurements.

INTRODUCTION

This paper presents a study of active vibration suppression in flexible structure using actuators with second order dynamics. First, a low order model of the structure is used to investigate stability properties. It is then shown that the common practice of maintaining stability by using relative velocity feedback (i. e. the difference between the structure's velocity and the actuator mass velocity) does not necessarily lead to the best closed loop performance.

In addition to a finite dimensional analysis of the effects of actuator dynamics on active vibration suppression, an infinite dimensional model is suggested. This model proposes adapting an approach presented by previous authors on combined dynamical systems. This approach is adapted here to include a control law acting between a (finite dimensional) second order actuator and a structure defined by a second order in time partial differential equation. The specific case of a beam modeled as an Euler-Bernoulli equation with both internal (Kelvin-Voight) and external (air) damping included and actuator is presented. Modal equations are presented in terms of the Green's function without actuator dynamics. The case of velocity feedback of the structure at the point of attachment on the beam is derived. The open loop equation with point dynamics are then verified experimentally. A short list of acronyms used in this paper follows in the appendix.

SDOF/PMA

The linear proof mass actuator (PMA) is a solenoid-like device(3). Current flowing in a coil of wire attached to the frame of the PMA, combined with permanent magnets fixed to the proof mass produces an electromagnetic force that accelerates the proof mass. Furthermore, this electromagnetic force produces a reaction force on the PMA frame that can be used for control law actuation. By regulating the current supplied to the coil one can control the force produced by the actuator. A linear variable differential transformer (LVDT) is used to measure η, the proof mass's position relative to the PMA frame. This signal can be differentiated to give $\dot{\eta}$, the proof mass's velocity relative to the frame. An accelerometer attached to the PMA gives the structural acceleration, which is integrated to give its velocity, $\dot{x}_s$. These three measurements provide three local feedback paths that can be used for output feedback control. The combination of the proof mass and these feedback paths can be modeled as a second order system as shown in fig. 1.

In fig. 1 the structure to be controlled is represented by a damped single degree of freedom system, (M_s, K_s, C_s), the actuator is also modelled by a single degree of freedom system with an accompanying force generator. The spring stiffness, k_a, represents the servo stiffness of the actuator. The damper, c_a, represents the friction inside the actuator. A force generator, f_g, is used to show the use of either velocity feedback path. The dead mass, m_d, associated with the actuator is fixed to M_s. The equations of motion for this system are

$$\begin{bmatrix} M_c & m_p \\ m_p & m_p \end{bmatrix} \ddot{X} + \begin{bmatrix} C_s & 0 \\ 0 & c_a \end{bmatrix} \dot{X} + \begin{bmatrix} K_s & 0 \\ 0 & k_a \end{bmatrix} X = \begin{Bmatrix} 0 \\ f_g \end{Bmatrix} \tag{1}$$

$$X = [\, x_s \;\; \eta \,]^T, \qquad M_c = M_s + m_d$$

In order for any control formulation to be useful it must be stable. Therefore, the effects on stability of each of the feedback paths must be examined.

The relative position feedback gain, k_a, must be positive$^{(4)}$ for the stiffness matrix to be positive definite. If k_a were zero the actuator would have an uncontrolled rigid body mode. Therefore, a first requirement for stability is that k_a be positive.

With relative velocity feedback, i.e. $f_g = - c_g \dot{\eta}$, the damping matrix becomes

$$C = \begin{bmatrix} C_s & 0 \\ 0 & c_a + c_g \end{bmatrix} \tag{2}$$

Which is symmetric and positive definite for all values of $c_g > 0$, making this a stable control law. Yet, the term $c_a + c_g$ in the damping matrix casts this type of control into the same formulation as a traditional vibration absorber problem$^{(1,10)}$.

If the control force is made proportional to the structure's velocity, i.e. $f_g = -g\dot{x}_s$, the damping matrix becomes

$$C = \begin{bmatrix} C_s & 0 \\ g & c_a \end{bmatrix} \tag{3}$$

Which is asymmetric, such that the system

$$M\ddot{X} + C\dot{X} + KX = 0 \tag{4}$$

becomes potentially unstable as g increases. Therefore, the question arises: why use $f_g=-g\dot{x}_s$ if this type of control can become unstable? If one were designing a colocated velocity control law that ignored the actuator dynamics, this is exactly what one would do(9). Our purpose here is to determine whether using non-colocated velocity feedback will provide performance benefits over the use of only colocated, while including the actuator dynamics in the analysis.

Finally, a combination of both relative velocity feedback and direct velocity feedback is considered rather than each path individually. That is $f_g = - g\dot{x}_s - c_g\dot{\eta}$ which produces no new stability problems, yet provides for the interaction of both feedback paths. In other words, the use of the colocated feedback can be used to stabilize the non-colocated feedback. Therefore, the problem becomes that of finding an optimal choice for k_a, c_g, and g. This problem will be formulated as a linear quadratic regulator and solved as a parameter optimization.

The cost function is chosen to be the infinite time integral of the square of the structure's position,

$$J=\int_0^\infty x_s^2 dt = \int_0^\infty q^T Q q dt \tag{5}$$

$$q= [\, x_s\ \eta\ \dot{x}_s\ \dot{\eta}]^T$$

$$Q_{ij} = \begin{cases} 1 & i=j=1 \\ 0 & \text{otherwise} \end{cases}$$

The equations of motion are written in state space form

$$\dot{\mathbf{q}}=\mathbf{A}\mathbf{q} \tag{6}$$

$$A=\begin{bmatrix} 0_{2x2} & I_{2x2} \\ -M^{-1}K & -M^{-1}D \end{bmatrix} \qquad M=\begin{bmatrix} M_s+m_p & m_p \\ m_p & m_p \end{bmatrix}$$

$$D=\begin{bmatrix} C_s & 0 \\ g & c_a+c_g \end{bmatrix} \qquad K=\begin{bmatrix} K_s & 0 \\ 0 & k_p \end{bmatrix}$$

The cost function can be rewritten as

$$J=\mathbf{q}^T(0)\mathbf{P}\mathbf{q}(0) \tag{7}$$

where P represents the solution of a Lyapunov equation

$$A^TP + PA = -Q \tag{8}$$

Note, that in comparison to a typical LQR optimal control problem, that is

$$J=\int_0^\infty \mathbf{q}^T\mathbf{Q}\mathbf{q} + \mathbf{u}^T\mathbf{R}\mathbf{u}\ dt \tag{9}$$

there is no penalty on the control here, i. e. R=0. In fact there is no control, **u**, in this problem, only feedback gains which are treated as parameters. Hence, while this problem resembles an optimal control problem it is really a parameter optimization.

The cost function J is a function of both the initial conditions and P, where P is a matrix whose elements are a function of the feedback gains. To remove the dependence of J on $\mathbf{q}(0)$ the initial conditions are treated as a random vector, and the expected value of J is taken as

$$E\{J\} = E\{\mathbf{q}^T(0)P\mathbf{q}(0)\} = tr(PZ^\circ) \tag{10}$$

$$Z^\circ = \mathbf{q}(0)\mathbf{q}^T(0) \tag{11}$$

Z° is a normalized second order moment matrix of the initial conditions. For this example it is assumed that the structure and actuator are at rest with the structure given an initial deflection and the proof mass at its equilibrium position relative to the structure. Therefore Z° is

$$Z^\circ = \begin{cases} 1 & i=j=1 \\ 0 & \text{otherwise} \end{cases}$$

The final expression for the cost function is

$$J = P_{11} \tag{12}$$

A necessary condition for optimality is that

$$\left.\frac{dJ}{d\mathbf{F}}\right|_{\mathbf{F}*} = 0 = \left.\frac{dP_{11}}{d\mathbf{F}}\right|_{\mathbf{F}*} \tag{13}$$

where **F** represents the set of feedback gains. The matrix P was computed algebraically, with the aid of MACSYMA, and the required derivatives calculated analytically. With these analytical expressions the optimal feedback gains were determined numerically. A special case of this formulation is for zero structural velocity feedback, g=0, the equations simplify substantially and the results of Juang[(1)] are obtained.

Example

Consider attaching a PMA to the tip of an undamped cantilevered beam, with the desire to attenuate the first bending mode of vibration of this beam. The material properties of the beam are given in Table 1[(6)]. The mass properties of the PMA are given in Table 2.

The first natural frequency for the beam plus dead mass calculated from a finite element analysis is 12.66 rad/sec (2.02 Hz). The equivalent bending spring stiffness is calculated to be 158.2 N/m, and the structural first modal mass plus actuator dead mass is 0.987 kg. Table 3 gives the optimal parameter settings for two conditions: 1) zero structural velocity feedback (i. e., a vibration absorber), and 2) with structural velocity feedback and the relative velocity feedback gain held to 17.5 N-s/m.

Fig. 2 shows the structural response of both systems to the same initial condition. Note that, the settling time, and maximum overshoot of system 2 is superior to that of system 1. Fig. 3 shows the response of the actuator mass for this simulation. The major disadvantage of this control law can be seen in this figure, which is the actuator of system 2 requires a greater stroke length.

There are several lessons that can be learned from controlling a single degree of freedom structure with a PMA that can be applied to the vibrational control of both multiple degree of freedom and distributed parameter structures. Control laws that ignore actuator dynamics may result in closed loop instability. The use of only safe or nondestabilizing feedback paths may not yield the best performance. Furthermore, using only relative position and velocity feedback results in a control law that is no different than that of a traditional vibration absorber. This type of design tends to require low feedback gains, such that the motion of the proof mass is unimpeded. Finally, better performance is achieved with structural velocity feedback combined with relative velocity feedback. In fact, a high structural feedback gain can only be tolerated in the presence of a high relative velocity feedback gain.

MDOF/MPMA

Fig. 4 shows two proof mass actuators attached to four degree of freedom structural model. Assume that the actuators are not interconnected. That is, any measurements made by PMA one are not shared with PMA two and vice versa. In this case we have the same feedback paths as in the SDOF/PMA, case just more of them.

The relative position feedback gain, k_{pi}, must be positive for each actuator i. This requirement eliminates the uncontrolled rigid body mode of the actuator. Furthermore, $k_{pi} > 0$ is necessary so that the stiffness matrix for this system is positive definite. Similarly, the combined relative velocity gain for each actuator, $(c_a + c_g)_i$ must be positive. The intuitive proof for these requirements is that the addition of a damped single degree of freedom to a stable damped multiple degree of freedom structure results in a stable system.

If structural velocity feedback is used, the stability of the system can be examined using the well known Routh-Hurwitz criteria. For high order systems this test becomes difficult to implement. An alternative is to examine the system damping matrix. For multiple degree of freedom systems that can be described by the system of equations, Eq. (4), stability is guaranteed if the symmetric portion of the damping matrix is positive definite, provided that the mass and stiffness matrices are symmetric positive definite. The damping matrix C is made asymmetric by the introduction of the feedback gains g_i in some of the off diagonal terms. Any asymmetric matrix can be written as the sum of a symmetric and a skew symmetric matrix.

$$C = C_{sym} + G, \qquad C_{sym} = C_{sym}^T \qquad G = -G^T$$

Therefore, the stability of the MDOF/ MPMA system is guaranteed if the matrix C_{sym} is positive definite(7). This test can be applied to the system of the previous section. The matrix C_{sym} for this system is

$$C_{sym} = \begin{bmatrix} C_s & \frac{g}{2} \\ \frac{g}{2} & c_a + c_g \end{bmatrix} \tag{14}$$

and will be positive definite if the following relations are satisfied

$$C_s > 0 \tag{15}$$

$$4\, C_s\, (c_a + c_g) > g^2 \tag{16}$$

Determining whether high order matrix with a small number of feedback gains is positive definite can become just as tedious as the full Routh-Hurwitz test. Therefore, it is useful to simplify this result by examining the definiteness of the principal minors of C_{sym}. The matrix C_{sym} can be written as the sum of a positive definite matrix C_c and a sparse matrix of zeros and 2x2 symmetric blocks containing g_i placed along the diagonal according to the actuator placement. When this sparse block diagonal matrix is added to C_c to form C_{sym} the only minors of C_c that are affected are the 2x2 blocks added at the position of the actuator coordinates. Therefore, only those minors changed by the addition of feedback need be checked. This test leads to the following result for each actuator(8).

$$4(C_{si} + C_{si+1})(c_a + c_g)_i > g_i^2 \tag{17}$$

It should be recalled that these results if satisfied ensure stability, but if violated do not imply instability. Therefore, we feel that this stability criteria is of a conservative nature.

DPS/PMA

In this section we examine a beam modelled as an Euler-Bernoulli beam with both air damping and Kelvin-Voight internal strain rate damping. A proof mass actuator is attached to the beam at some point, fig. 5. The analysis here follows that of Bergman(2). The equations of motion for this system are:

$$\rho u_{tt} + (c_1 + c_2 I\frac{\partial^4}{\partial x^4})\, u_t + EI\, \frac{\partial^4}{\partial x^4} u = \sum_{i=1}^{r} \{ [F_i(t) - f_{gi}(t) - m_{di} u_{tt}(h_i,t)]\ \delta(x-h_i) \} \tag{18}$$

$$m_{pi}\ddot{\eta}_i + m_{pi}u_{tt}(h_i,t) + c_{pi}\dot{\eta}_i + k_{pi}\eta_i = f_{gi}(t) \tag{19}$$

$$F_i(t) = -m_{pi}\ddot{\eta}_i - m_{pi}u_{tt}(h_i,t) + f_{gi}(t) \tag{20}$$

$$f_{gi}(t) = -\ c_{gi}\ u_t(h_i,t) \tag{21}$$

r = the number of actuators

Where c_{gi} is the structural velocity feedback gain. Note that in this case only feedback of the local velocity is considered.

These equations can then be nondimensionalized according to the following definitions

$$\begin{array}{llll}
\bar{u} = \frac{u}{l}, & \bar{\eta} = \frac{\eta}{l}, & \xi = \frac{x}{l}, & \delta_i = \frac{h_i}{l}, \\
\tau = \Omega\, t, & \Omega^2 = \frac{EI}{\rho l^4}, & \overline{\delta(\xi - \delta_i)} = l\delta(x-h_i), & \\
\varepsilon_1 = \frac{c_1}{\rho\Omega}, & \varepsilon_2 = \frac{c_2 I}{\rho\Omega l^4}, & \overline{F_i(t)} = \frac{l^2}{EI} F_i(t), & \\
\mu_{di} = \frac{m_{di}}{\rho l}, & \mu_{pi} = \frac{m_{pi}}{\rho l}, & \overline{f_i(t)} = \frac{l^2}{EI} f_i(t), & \\
K_i = \frac{k_{pi} l^3}{EI}, & \alpha_{oi}{}^4 = \frac{K_i}{\mu_{pi}}, & C_{gi} = \frac{l^3\Omega c_{gi}}{EI} &
\end{array}$$

The nondimensional equations of motion are, where the overbars have been dropped

$$u_{\tau\tau} + (\varepsilon_1 + \varepsilon_2\frac{\partial^4}{\partial \xi^4})\, u_\tau + \frac{\partial^4}{\partial \xi^4} u = \sum_{i=1}^{r} \{ [- \mu_{pi}\ddot{\eta}_i - \mu_{pi}u_{\tau\tau}(\delta_i,\tau) - \mu_{di}u_{\tau\tau}(\delta_i,\tau)]\ \delta(\xi-\delta_i) \} \tag{22}$$

$$\ddot{\eta}_i + u_{\tau\tau}(\delta_i,t) + \varepsilon_i\dot{\eta}_i + \alpha_{oi}{}^4\eta_i = -\frac{1}{\mu_{pi}}\, C_{gi}\ u_\tau(\delta_i,\tau) \tag{23}$$

The solution to this problem is assumed to be of the form

$$\overline{u(\xi,\tau)} = \sum_{n=1}^{\infty} a_n(\tau)X_n(\xi) \tag{24}$$

$$\overline{\eta(\tau)} = \sum_{n=1}^{\infty} a_n(\tau)A_{in}X_n(\delta_i) \tag{25}$$

where

$$A_{in} = \frac{\alpha_n{}^4}{\alpha_{oi}{}^4 - \alpha_n{}^4}, \qquad \tilde{A}_{in} = A_{in} + 1 = \frac{\alpha_{oi}{}^4}{\alpha_{oi}{}^4 - \alpha_n{}^4} \tag{26}$$

and $X_n(\xi)$ are the eigenfunctions of the undamped system. $\alpha_{oi}{}^4$ represents the square of the actuator's undamped natural frequency, and α_n is the eigenvalue corresponding to $X_n(\xi)$.
For the special case of zero actuator dead mass, the temporal coefficients satisfy

$$\ddot{a}_n(\tau) + (\varepsilon_1 + \varepsilon_2\, \alpha_n{}^4)\, \dot{a}_n(\tau) + \alpha_n{}^4\, a_n(\tau)$$

$$+ \sum_{m=1}^{\infty} \left\{ C_{gi}\, (1 - \frac{1}{\mu_{pi}A_nA_m}) + \varepsilon\, \frac{\alpha_n{}^4\alpha_m{}^4}{\alpha_o{}^8} - \varepsilon_1 - \varepsilon_2\alpha_n{}^4 \right\} \mu_{pi}A_{in}A_{im}X_n(\xi)X_m(\xi)\, \dot{a}_n(\tau) = 0$$

(27)

The eigenvalues $\alpha_n{}^4$ are calculated from

$$\sum_{i=1}^{r} \left\{ \left[\delta_{ij} - \mu_{pi}\, \frac{\alpha_{oi}{}^4\alpha_n{}^4}{\alpha_{oi}{}^4 - \alpha_n{}^4}\, G(\delta_i,\delta_j;\alpha_n) \right] X_n(\delta_i) \right\} = 0 \tag{28}$$

where G is the Green's function for a clamped - free Euler-Bernoulli beam. These equations are then useful for stability analysis, control gain formulation, and experimental verification.

EXPERIMENTAL RESULTS

A combined lumped parameter zero gain distributed parameter experiment was performed to verify Eq. (28) for the special case of $\mu_{di} = 0$. A clamped-free Euler-Bernoulli beam, whose material properties are given in table one, was appended with two identical passive spring-mass systems. One was placed in the middle of the beam, x = l/2, and one at the free end, x = l. The mass was measured to be 49.2×10^{-3} kg. The spring stiffness was 858 N/m.

In a test, an accelerometer was placed at the free end of the beam to measure the response of the structure. An impact hammer was used to excite the beam with hammer hits made at various points along the length the beam. The first nine natural frequencies were estimated from the experimental data using a Nyquist plot circle fitting technique. These same natural frequencies were calculated by numerical solution of Eq. (28) for the first nine modes. These results are given in table four.

Apparently, the estimated frequencies agree very well with the theoretically predicted ones. In fact, the standard deviation of the estimated frequencies to the theoretical is generally small. The error between the estimates and predictions is on the order of 1%.

SUMMARY

Actuators that can be modeled as lumped second order systems were examined for use in vibration control of a distributed parameter system. A finite dimensional model provided insight that was extended to the distributed parameter case. It was seen that ignoring the actuator dynamics can lead to an unstable control law formulation. Secondly, the feedback paths available for output feedback control were identified and examined in terms of closed loop stability. This resulted in closed loop stability conditions for computing stable control gains. Finally, an example was given where it was seen that the use of noncolocated feedback gave better performance than solely colocated feedback.

An infinite dimensional formulation of a cantilevered beam with actuator dynamics was presented and experimentally verified. This model included both structural damping, in the form of viscous and Kelvin-Voight damping, and the actuator dynamics. It remains to complete the computational studies of the infinite dimensional model and its approximations. Initial experimental verification showed good agreement between theoretically predicted and experimentally estimated natural frequencies.

ACKNOWLEDGEMENT

This work was supported by NASA Grant number NAG-1-993 monitored by Dr. Ernest Armstrong of the Langley Research Center.

APPENDIX: ACRONYMS

DPS - Distributed Parameter System
LVDT - Linear Variable Differential Transformer
MDOF - Multiple Degree of Freedom
MPMA - Multiple Proof Mass Actuator
PMA - Proof Mass Actuator
SDOF - Single Degree of Freedom

REFERENCES

(1) Juang, J. N., "Optimal Passive Vibration Absorber for a Truss Beam," AIAA Journal of Guidance, Control and Dynamics, Vol. 7, No. 6, Nov-Dec. 1984, pp 733-739.

(2) Bergman, L. A., and Nicholson, J. W., "Forced Vibration of a Damped Combined Linear System," ASME Journal of Vibration, Acoustics, Stress, and Reliability in Design, Vol. 107, pp. 275-281, 1985.

(3) Zimmerman, D. C., Horner, G. C. and Inman, D. J., "Microprocessor Controlled Force Actuator," AIAA Journal of Guidance, Control, and Dynamics, Vol. 11, No. 3, May-June 1985, pp. 230-236.

(4) Zimmerman, D. C., and Inman, D. J., "On the Nature of the Interaction Between Structures and Proof-Mass Actuators," AIAA Journal of Guidance, Control and Dynamics, to appear.

(5) Bellos, J., Lumped, Distributed, and Combined Dynamic Structures with Non-Proportional Damping, Ph D dissertation, University at Buffalo, 1989.

(6) Cudney, H. H. and Inman, D. J., "Experimental Verification of Damping Mechanisms in a Composite Beam," International Journal of Analytical and Experimental Modal Analysis, to appear.

(7) Inman, D. J., Vibration with Control, Measurement and Stability, Prentice Hall, Englewood Cliffs, New Jersey, 1989.

(8) Inman, D.J., "Control Structure Interactions: Effects of Actuator Dynamics," Mechanics and Control of Large Space Structures, Chapter 19, ed. J. Junkins, AIAA Progress in Aeronautics And Astronautics Series, 1990.

(9) Balas, M. J. "Direct Velocity Feedback Control of Large Space Structures," AIAA Journal of Guidance and Control, Vol. 2, No. 3, May-June 1979, pp. 252-253.

(10) Den Hartog, J. P., Mechanical Vibrations, 4th ed., McGraw-Hill Book Co., New York, 1956.

Table 1. Beam parameters

Property	Units	Symbol	Value
Length	m	l	1.0
Moment of Inertia	m^4	I	1.64×10^{-9}
Area, cross section	m^2	A	0.597×10^{-3}
Young's Modulus	GPa	E	26.8
Viscous damping	$N\,s\,m^{-1}$	c_1	1.75
Kelvin-Voight damping	$kN\,s\,m^{-1}$	c_2	20.5
linear Density	$kg\,m^{-1}$	ρ	1.02

Table 2. Actuator mass parameters

Property	Unit	Symbol	Value
Proof mass	kg	m_p	0.225
Dead mass	kg	m_d	0.730

Table 3. Example gain settings

Gain	System 1	System 2
K_a $(N\,m^{-1})$	23.9	211
g $(N\,s\,m^{-1})$	0	77.5
ca + cg $(N\,s\,m^{-1})$	1.0	17.5
Cost J	0.183	0.0710

Table 4. Measured and predicted frequencies

Mode	Analytical Freq. (Hz)	Experimental Freq. (Hz)	Std dev.
1	3.49	3.48	0.127
2	17.7	17.8	0.233
3	22.4	22.6	0.0479
4	29.3	28.5	0.916
5	68.3	67.8	0.517
6	133	132	0.995
7	219	217	1.30
8	327	324	3.24
9	457	452	4.40

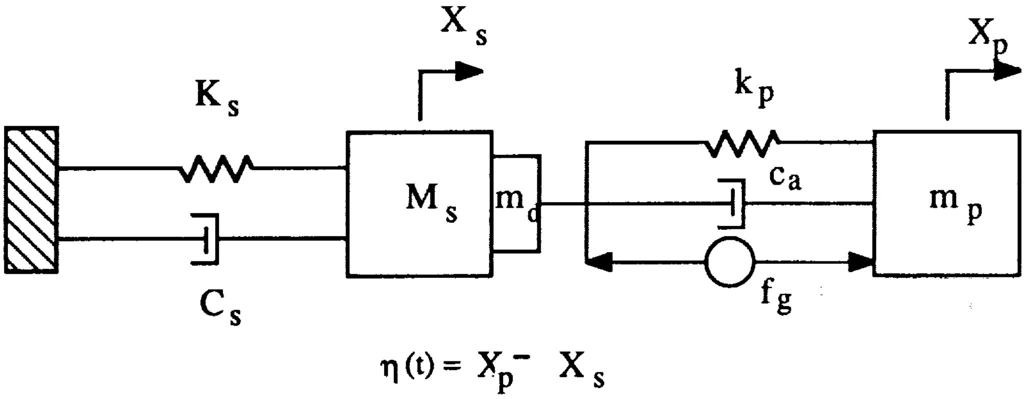


Figure 1. Single degree of freedom stucture with proof mass actuator (SDOF/PMA)

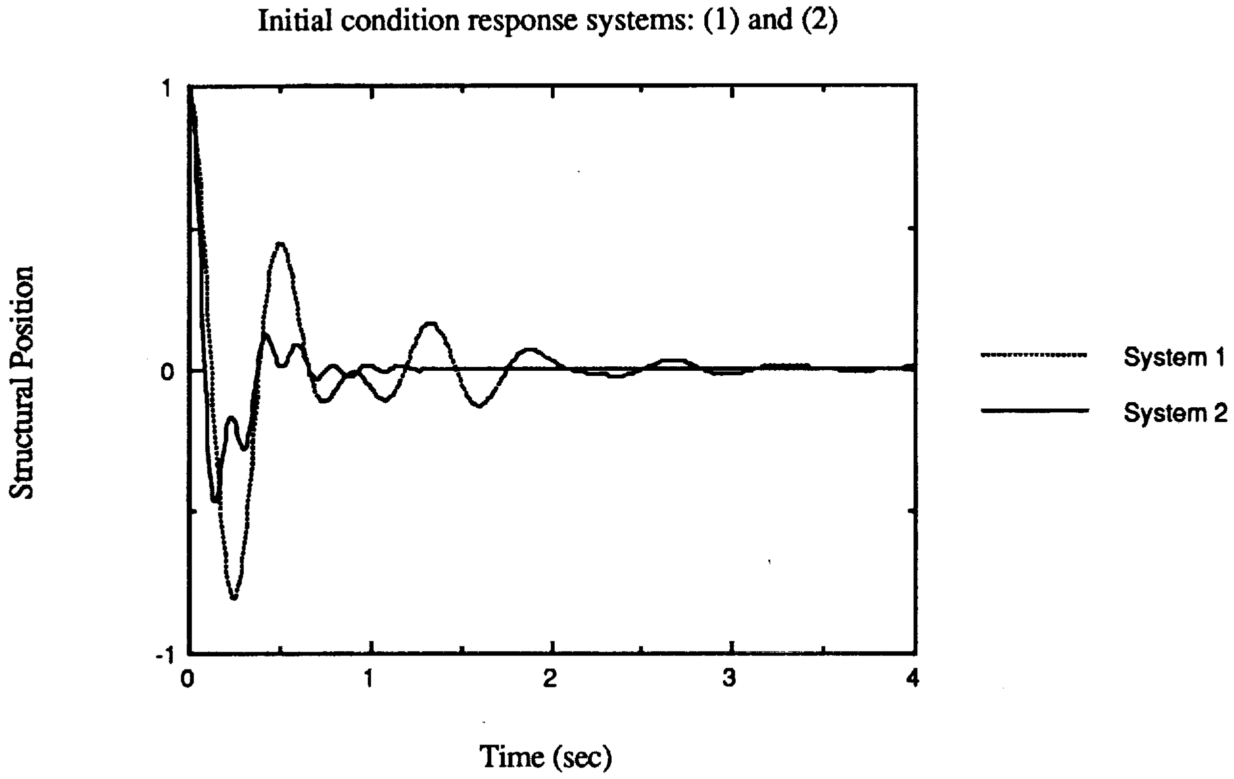


Figure 2. Structural position response: system (1) vibration absorber
system (2) active control with velocity feedback

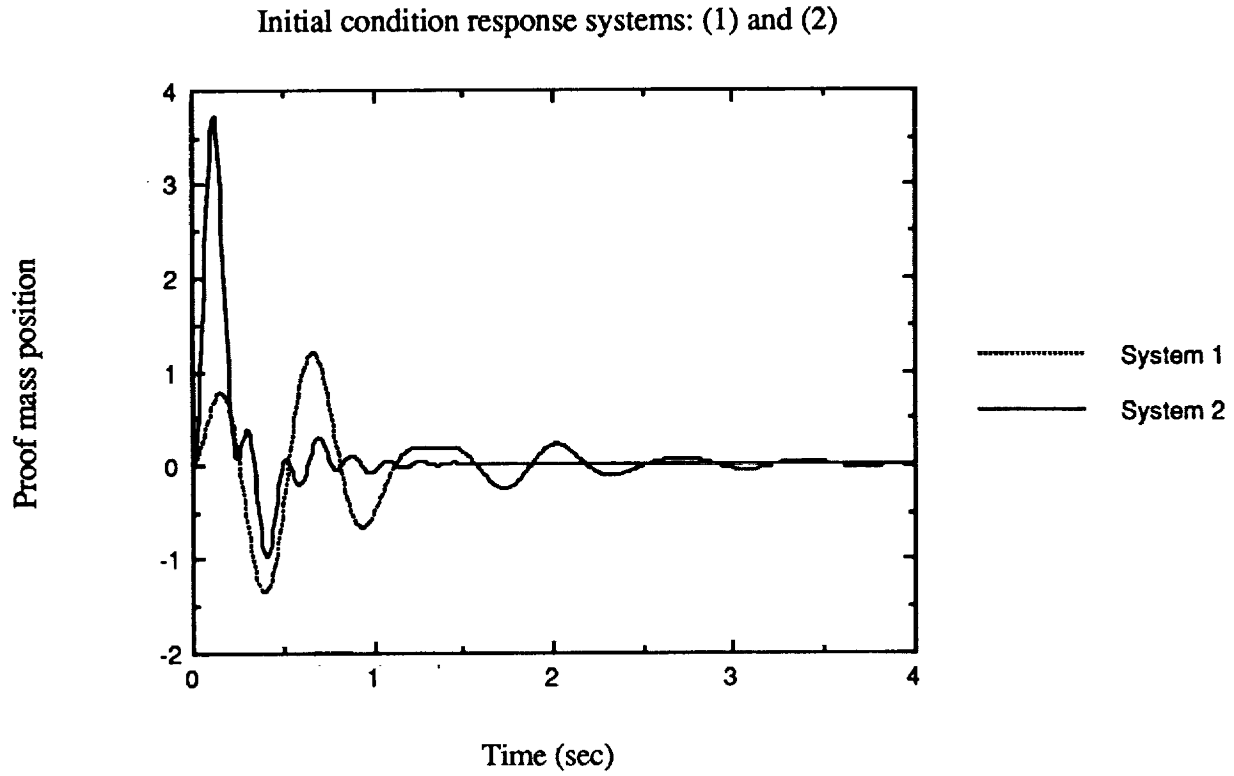


Figure 3. Actuator mass response: system (1) vibration absorber
system (2) active control with velocity feedback

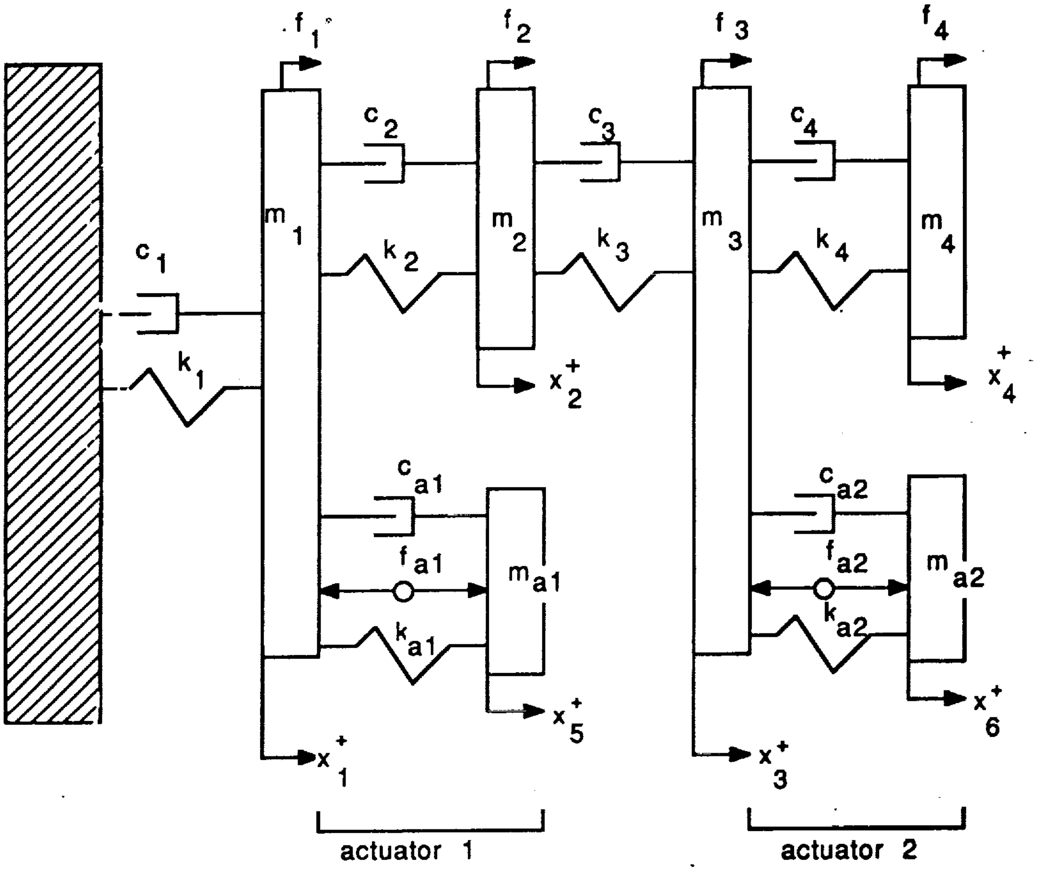


Figure 4. Multiple degree of freedom structure with multiple actuators (MDOF/MPMA)

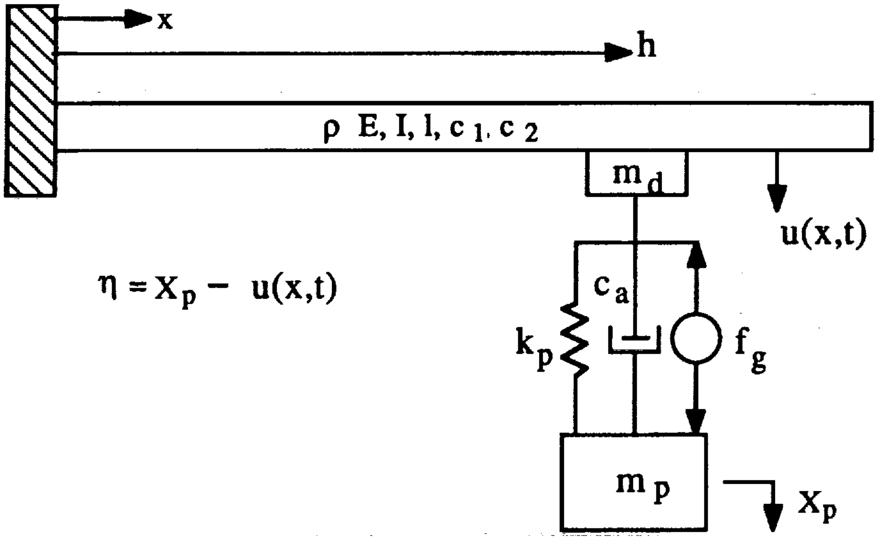


Figure 5. Euler-Bernoulli beam with proof mass actuator (DPS/PMA)

Approximation in LQG Control of a Thermoelastic Rod

J.S. Gibson [*]
Mechanical, Aerospace and Nuclear Engineering
University of California, Los Angeles 90024

I.G. Rosen [†]
Department of Mathematics
University of Southern California, Los Angeles 90089

G. Tao
Electrical Engineering–Systems
University of Southern California, Los Angeles 90089

Abstract

Control and estimator gains are computed for linear-quadratic-Gaussian (LQG) optimal control of the axial vibrations of a thermoelastic rod. The computations are based on a modal approximation of the partial differential equations representing the rod, and convergence of the approximations to the control and estimator gains is the main issue.

1 Introduction

The axial vibrations of a uniform rod are represented by a one-dimensional wave equation with constant coefficients, and thermoelastic damping in the rod is represented by a one-dimensional heat equation coupled to the wave equation. The solutions to the wave and heat equations are, respectively, the axial displacement and temperature fields in the rod. [See 1, 2].

The length of the rod in this paper is normalized to 1. For active control, a single force is distributed parallel to the rod, uniformly over the portion $s_0 \le s \le s_1$ of the rod. The equations of motion of the plant are then

$$\rho w_{tt} = (\lambda + 2\mu)w_{ss} - \alpha(3\lambda + 2\mu)\theta_s + bu + b\eta_1, \qquad t > 0, \quad 0 < s < 1, \tag{1.1}$$

$$\rho c\theta_t = k\theta_{ss} - \theta_0\alpha(3\lambda + 2\mu)w_{ts}, \qquad t > 0, \quad 0 < s < 1, \tag{1.2}$$

where

$$b(s) = \begin{cases} 1, & s_0 \le s \le s_1, \\ 0, & \text{otherwise.} \end{cases} \tag{1.3}$$

[*] This author was supported by AFOSR Grant 870373.

[†] This author was supported by AFOSR Grant 870356.

In these equations, $w(t) = w(t,s)$ is the axial displacement, $\theta(t) = \theta(t,s)$ is the temperature distribution and $u(t)$ is the control force. We assume that the actuator force has the form $u + \eta_1$ where $u(t)$ is the known control function and η_1 is zero-mean Guassian white noise with intensity q_1. The constants $\rho, \alpha, \lambda, \mu, \theta_0, c$ and k are physical constants with values to be given later.

We assume that we have a sensor that measures the displacement at the left end of the rod segment over which the actuator force is distributed. This measurement is then

$$y(t) = w(t, s_0) + \eta_0 \tag{1.4}$$

where η_0 is zero-mean Guassian white noise with intensity 1.

In this paper, we use the boundary conditions

$$\begin{aligned} w(t,0) = w(t,1) = 0, \\ \theta_s(t,0) = \theta_s(t,1) = 0, \end{aligned} \tag{1.5}$$

which mean that the rod is clamped and insulated at both ends. Because of the insulated, or Neumann, bondary conditions for the heat equation, zero is an eigenvalue of the open-loop system and the corresponding eigenvector represents a constant temperature distribution. The span of this eigenvector is an uncontrollable and unobservable subspace, and the orthogonal complement of this subspace contains only states for which the average temperature along the rod is zero. Whatever the initial conditions and the control function, the average temperature in the rod therefore is a constant function of time. We will denote this average temperature by $\bar{\theta}$.

2 The Control Problem

We set $\tilde{\theta} = \theta - \bar{\theta}$ and define the state vector

$$x(t) = (w(t), w_t(t), \tilde{\theta}(t)). \tag{2.1}$$

We take the state space to be the Hilbert space

$$E = H_0^1(0,1) \times L_2(0,1) \times L_2(0,1) \tag{2.2}$$

where $H_0^1(0,1)$ is the first-order Sobolev space containing functions that vanish at both ends of the interval. The system in (1.1) - (1.4) then has the form

$$\dot{x} = Ax + Bu + B\eta_1, \tag{2.3}$$

$$y = Cx + \eta_0, \tag{2.4}$$

where A generates a strongly continuous semigroup of contraction operators on E (see [3]). We note that $|x(t)|_E^2$ is the sum of twice the mechanical energy in the rod and the integral over the rod of $\tilde{\theta}^2$ at time t.

The LQG optimal control problem in this paper is to find u to minimize

$$J = \lim_{\bar{t}\to\infty} \mathcal{E}\{\frac{1}{\bar{t}} \int_0^{\bar{t}} [\langle Qx(t), x(t)\rangle_E^2 + u^2(t)]dt\} \tag{2.5}$$

where

$$Qx = (w, w_t, 0). \tag{2.6}$$

The operator Q is chosen so that mechanical energy is penalized but temperature variations from $\bar{\theta}$ are not penalized. As usual, this problem separates into a deterministic linear-quadratic regulator problem and a state estimation, or filtering, problem. Each of these problems has a unique solution because the open-loop system can be shown to be uniformly exponentially stable.

The optimal control law has the form

$$u(t) = -F\hat{x}(t) \tag{2.7}$$

where the state estimate $\hat{x}$ satisfies

$$\dot{\hat{x}} = A\hat{x} + Bu + \hat{F}(y - C\hat{x}). \tag{2.8}$$

The gain operators F and $\hat{F}$ are given by

$$F = B^*P \tag{2.9}$$

and

$$\hat{F} = \hat{P}C^* \tag{2.10}$$

where P and $\hat{P}$ are the unique nonnegative self-adjoint bounded linear operators on E satisfying the Riccati operator equations [4,5]

$$A^*P + PA - PBB^*P + Q = 0 \tag{2.11}$$

and

$$A\hat{P} + \hat{P}A^* - \hat{P}C^*C\hat{P} + q_1BB^* = 0. \tag{2.12}$$

The gain operators can be represented in terms of elements of E; i.e.,

$$Fx = \langle (f_1, f_2, f_3), x\rangle_E, \qquad (f_1, f_2, f_3) \in E, \tag{2.13}$$

$$\hat{F} = (\hat{f}_1, \hat{f}_2, \hat{f}_3) \in E. \tag{2.14}$$

3 Approximation

We approximate the infinite dimensional system in (2.3) and (2.4) with a sequence of finite dimensional control systems of the form

$$\dot{x}_n = A_nx_n + B_nu + B_n\eta_1, \tag{3.1}$$

$$y_n = C_nx_n + \eta_0. \tag{3.2}$$

In [10], we compared two Galerkin approximations: a finite element scheme in which linear splines were the basis vectors and a modal scheme in which the open-loop eigenvectors of the distributed system were the basis vectors. The modal scheme gave faster convergence for approximations to control gains like those in (2.9) and (2.13). We use the modal scheme here.

It is easy to see that the eigenspaces of the open-loop thermoelastic rod with the boundary conditions in (1.5) are three-dimensional subspaces each spanned by a two-dimensional eigenspace of the undamped wave equation and a one-dimensional eigenspace of the heat equation. The eigenvectors are sine waves for the wave equation and cosine waves for the heat equation. Since the modal approximation amounts to projection onto a sequence of complete orthogonal subspaces, it is easy to show that the approximations to the open-loop semigroup and adjoint semigroup converge strongly, as commonly needed in numerical solution of infinite dimensional Riccati equations for distributed systems [4-7].

For each n, we approximate the solutions to the infinite dimensional Riccati equations (2.11) and (2.12) by solving a pair of finite dimensional Riccati equations involving A_n, B_n and C_n. With the solutions to these matrix Riccati equations, we approximate the control and estimator gains as in [5]. In particular, we compute approximations to the functional gains (f_1, f_2, f_3) and $(\hat{f}_1, \hat{f}_2, \hat{f}_3)$ in (2.13) and (2.14).

4 Numerical Results

We chose the constants in (1.1) and (1.2) for an aluminum rod of length $100in$ (see [8, 9]). We normalized the length to 1, so that the constants take the numerical values

$$\begin{array}{lll} \rho = 9.82 \times 10^{-2} & \lambda = 2.064 \times 10^{-1} & \mu = 1.11 \times 10^{-1} \\ \alpha = 1.29 \times 10^{-3} & c = 5.40 \times 10^{-1} & \\ k = 7.02 \times 10^{-7} & \theta_0 = 68. & \end{array}$$

The actuator force is spread uniformly between $s_0 = .385$ and $s_1 = .486$, and the intensity of η_1 is $q_1 = 1$.

Figures 1-3 show the approximations to the control functional gains (f_1, f_2, f_3) for a range of approximation orders n, and Figures 4-6 show the approximations to the estimator functional gains $(\hat{f}_1, \hat{f}_2, \hat{f}_3)$. (The number of modal subspaces used is n.) Because thermoelastic damping is very light in aluminum, as in most metals, many modes of the rod must be controlled actively. Therefore, many modal subspaces must be used in the approximations before the functional gains converge.

5 References

1. D.E. Carlson, "Linear Thermoelasticity," in *Handbuch der Phisik*, Bd. Vla/2, edited by C. Truesdell, Springer, Berlin, 1972.

2. A.D. Day, *Heat Conduction within Linear Thermoelasticity*, Springer-Verlag, New York, 1985.

3. J.A. Walker, *Dynamical Systems and Evolution Equations*, Plenum, New York, 1980.

4. J.S. Gibson, "The Riccati Integral Equations for Optimal Control Problems on Hilbert Spaces," *SIAM J. Contr. Opt.*, Vol. 17, No. 4, July 1979, pp. 537-565.

5. J.S. Gibson and A. Adamian, "Approximation Theory for LQG Optimal Control of Flexible Structures," ICASE Report No. 88-48, August 1988, NASA Langley Research Center, Hampton, VA.

6. H.T. Banks and K. Ito, "A Unified Framework for Approximation and Inverse Problems for Distributed Parameter Systems," *Control Theory and Advances in Technology*, Vol. 4, 1988, pp 73–90.

7. H.T. Banks and K. Kunisch, "The Linear Regulator Problem for Parabolic Systems," *SIAM J. Contr. Opt.*, Vol. 22, No. 5, September 1984, pp. 684–696.

8. M.N. Ozisik, *Heat Conduction*, Wiley, New York, 1980.

9. E.P. Popov, *Introduction to Mechanics of Solids*, Prentice-Hall, Englewood Cliffs, NJ, 1968.

10. J.S. Gibson, I.G. Rosen and G. Tao, "Approximation in Control of Thermoelastic Systems," *Proceedings of 1989 Americal Control Conference*, Pittsburgh, PA, June 1989.

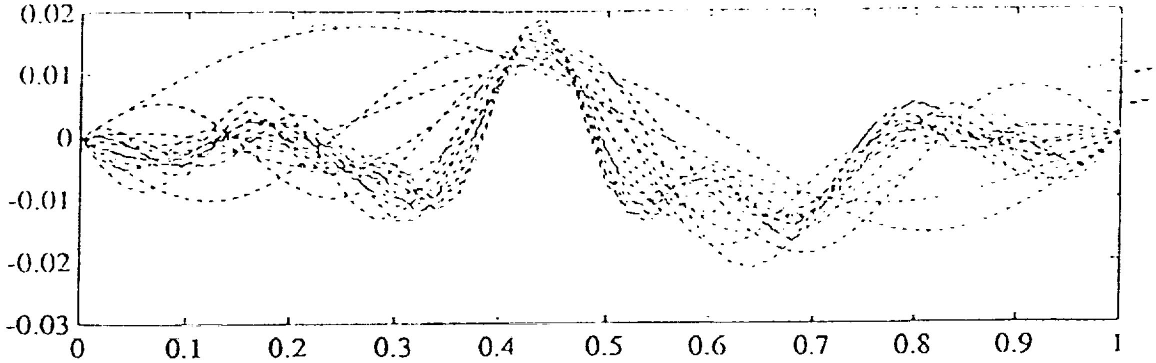


Figure 1(a): f1, the control gain, n = 2 - 17

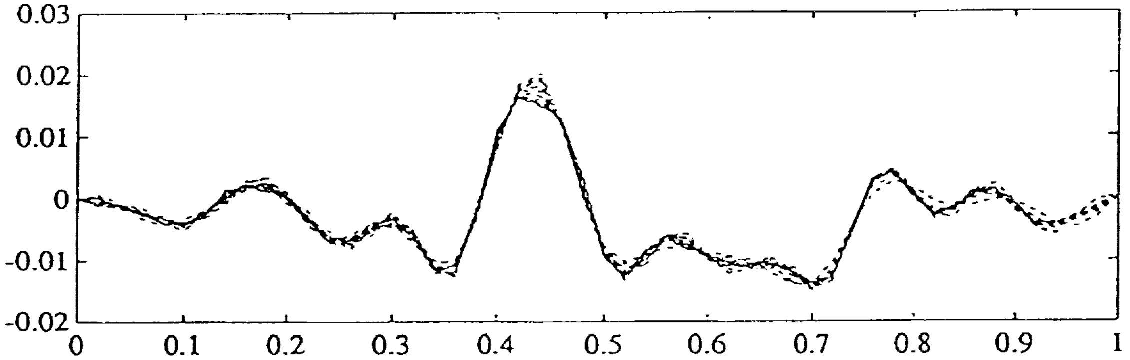


Figure 1(b): f1, the control gain, n = 18 - 33

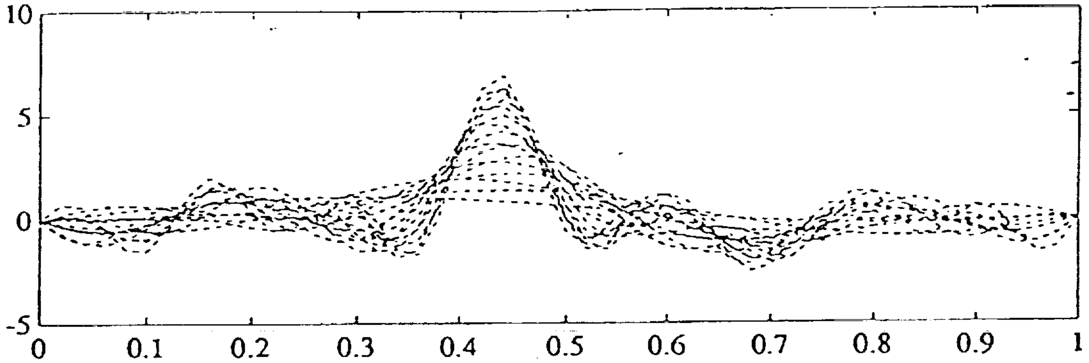


Figure 2(a): f2, the control gain, n = 2 - 17

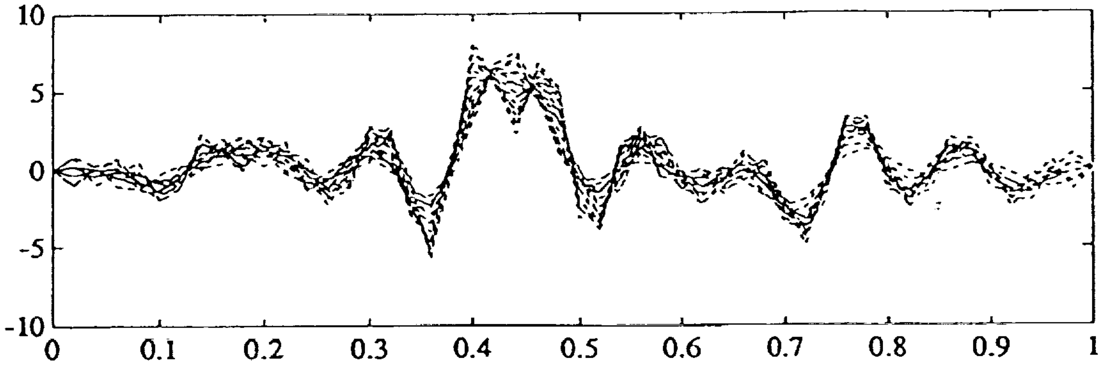


Figure 2(b): f2, the control gain, n = 18 - 33

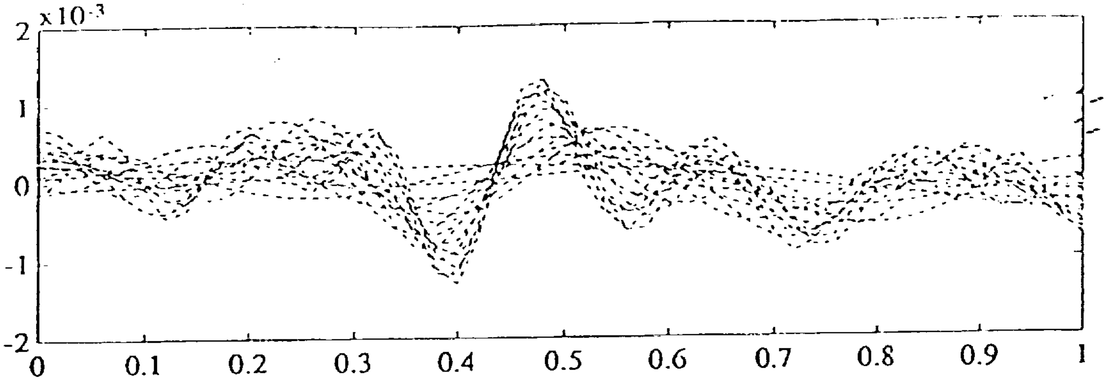


Figure 3(a): f3, the control gain, n = 2 - 17

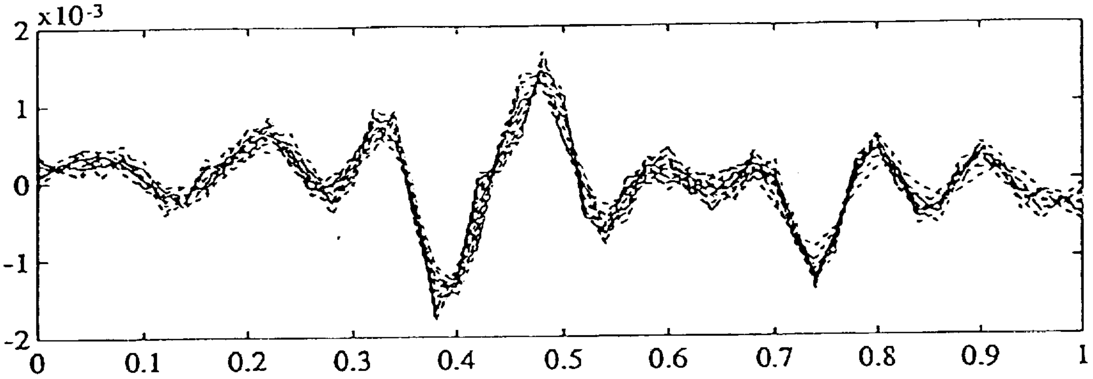


Figure 3(b): f3, the control gain, n = 18 - 33

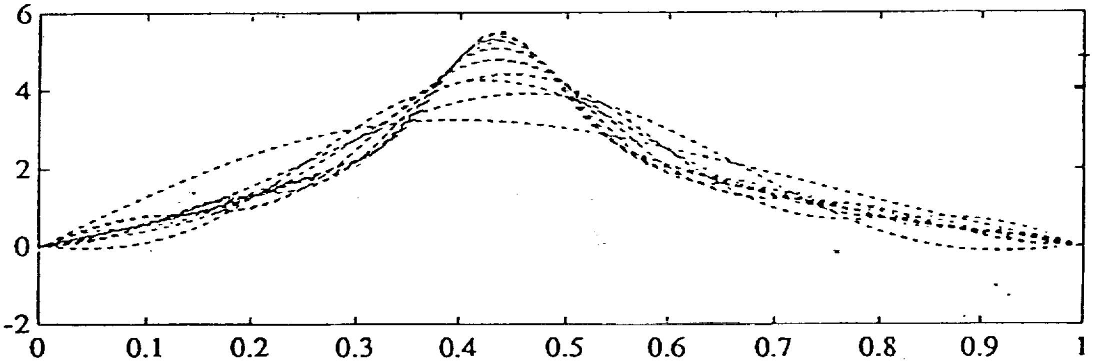


Figure 4(a): f1, the estimator gain, n = 2 - 17

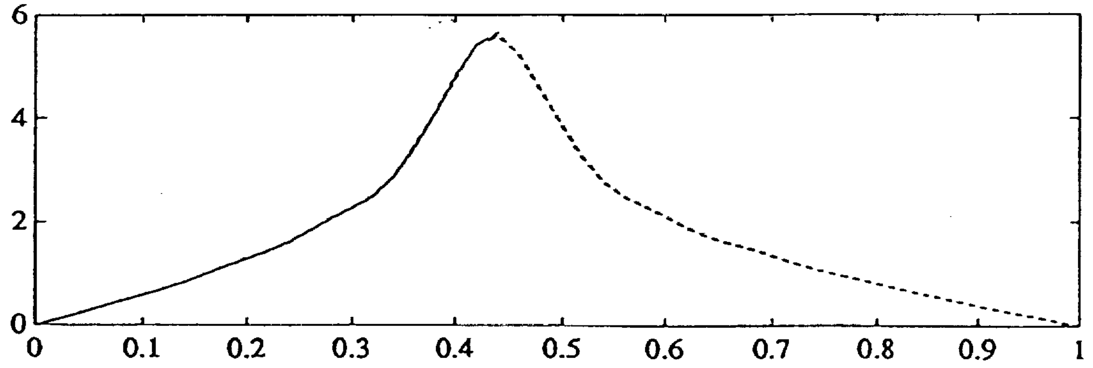


Figure 4(b): f1, the estimator gain, n = 18 - 33

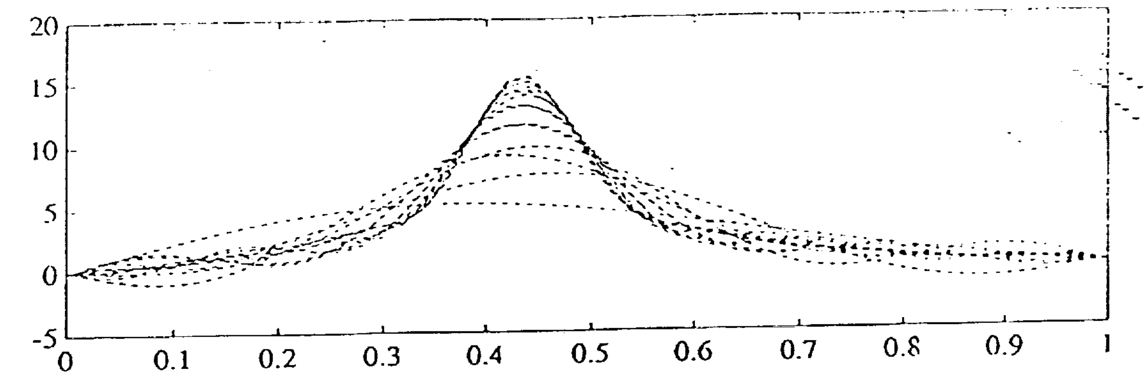


Figure 5(a): f2, the estimator gain, n = 2 - 17

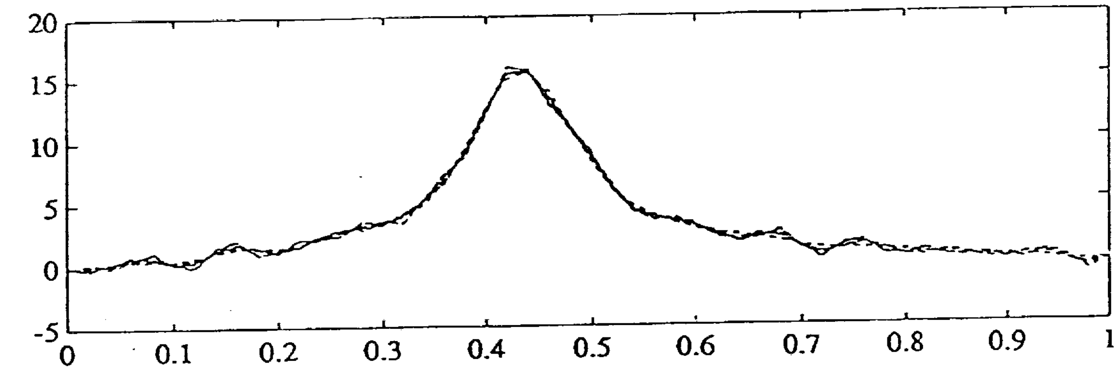


Figure 5(b): f2, the estimator gain, n = 18 - 33

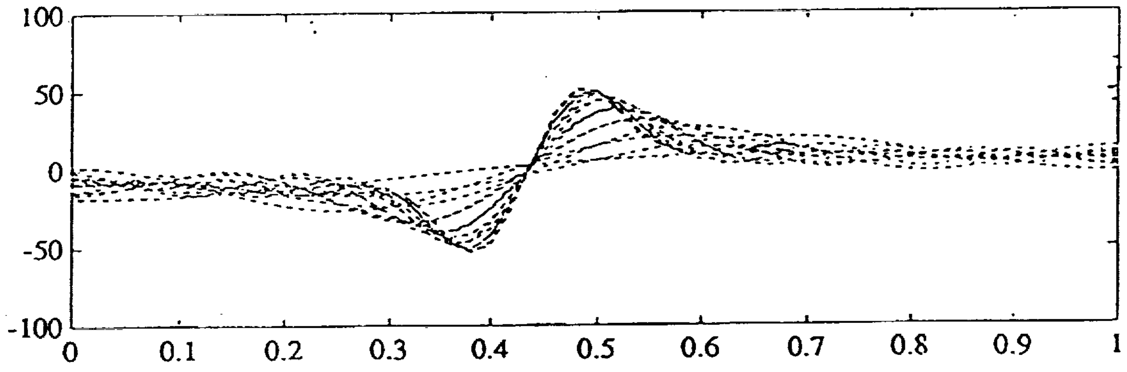


Figure 6(a): f3, the estimator gain, n = 2 - 17

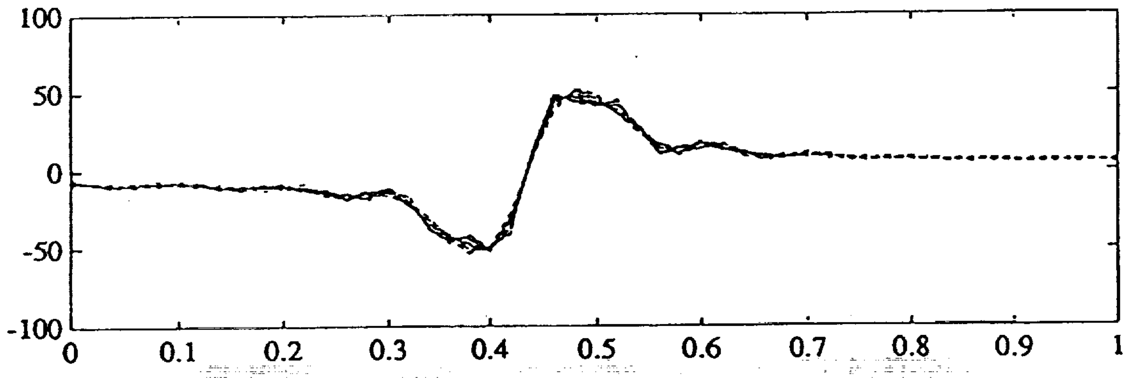


Figure 6(b): f3, the estimator gain, n = 18 - 33

Numerical Algorithms for Computations of Feedback Laws Arising in Control of Flexible Systems

Irena Lasiecka
Applied Math Department
University Virginia
Charlottesville, VA 22901

Abstract

Several continuous models will be examined, which describe flexible structures with boundary or point control/observation. The main goal of the talk is to discus issues related to the computation of feedback laws (particularly stabilizing feedbacks) with sensors and actuators located either on the boundary or at specific point locations of the structure.

One of the main difficulties is due to the great sensitivity of the system (hyperbolic systems with unbounded control actions), with respect to perturbations caused either by uncertainty of the model or by the errors introduced in implementing numerical algorithms. Thus, special care must be taken in the choice of the appropriate numerical schemes which eventually lead to implementable finite dimensional solutions. Finite dimensional algorithms will be constructed on a basis of a priori analysis of the properties of the original, continuous (infinite diversional) systems with the following criteria in mind:

(1) convergence and stability of the algorithms

(2) robustness - reasonable insensitivity with respect to the unknown parameters of the systems.

Examples with mixed finite element methods and spectral methods will be provided.

N90-23086

MODELING AND CONTROL OF FLEXIBLE SPACE STATIONS (SLEW MANEUVERS)

N. U. Ahmed S. S. Lim

Ottawa Carleton Institute of Electrical Engineering
University of Ottawa
Ottawa, Canada
K1N 6N5

ABSTRACT

Large orbiting space structures are expected to experience mechanical vibrations arising from several disturbing forces such as those induced by shuttle takeoff or docking and crew movements. In this paper, we consider the problem of modeling and control of large space structures subject to these and other disturbing forces. The system consists of a (rigid) massive body, which may play the role of experimental modules located at the center of the space station and flexible configurations, consisting of several beams, forming the space structure. A complete dynamic model of the system has been developed using Hamilton's principle. This model consists of radial equations describing the translational motion of the central body, rotational equations describing the attitude motions of the body and several beam equations governing the vibration of the flexible members (platform) including appropriate boundary conditions. In summary the dynamics of the space structure is governed by a complex system of interconnected partial and ordinary differential equations.

Using Lyapunov's approach the asymptotic stability of the space structure is investigated. For asymptotic stability of the rest state (nominal trajectory) we have suggested feedback controls. In our investigation stability of the slewing maneuvers is also considered.

Several numerical results are presented for illustration of the impact of coupling and the effectiveness of the stabilizing controls. This study is expected to provide some insight into the complexity of modeling, analysis and stabilization of actual space structures.

1.INTRODUCTION

Structural vibrations in future large space systems(Fig.1) such as space antennas, space platforms, space stations, or deployed flexible payloads attached to the space shuttle orbiter, and their interaction with the other members of the system have become a major concern in design of reliable systems satisfying stringent stability requirements. During the past few years, considerable attention have been focused on the development of mathematical model and stabilizing controls for such systems[1-13].

The most natural model for flexible space structures is given by a hybrid system of equations which is a combination of ordinary differential equations for rigid parts and partial differential differential equations for flexible members. Hybrid models for some simple structures have been considered in [5-11]. Recently Lim[12] developed a complete dynamic model, which includes the orbital dynamics, attitude dynamics and equations for vibrations of flexible members and all the relevant boundary conditions, for flexible space structures. Here, in this paper, based on the dynamic model we develop a control scheme to suppress the vibration induced by slew maneuvers in space stations.

The paper is organized as follows. In the next section we present the equations of motion for large spacecraft derived in [12]. In section 3, based on Lyapunov's approach, asymptotic stability of slew maneuvers for the system using feedback control is considered. For illustration, in section 4, we present some numerical results demonstrating the effectiveness of the stabilizing controls for vibration suppression associated with slew maneuvers. Finally, the concluding remarks are presented.

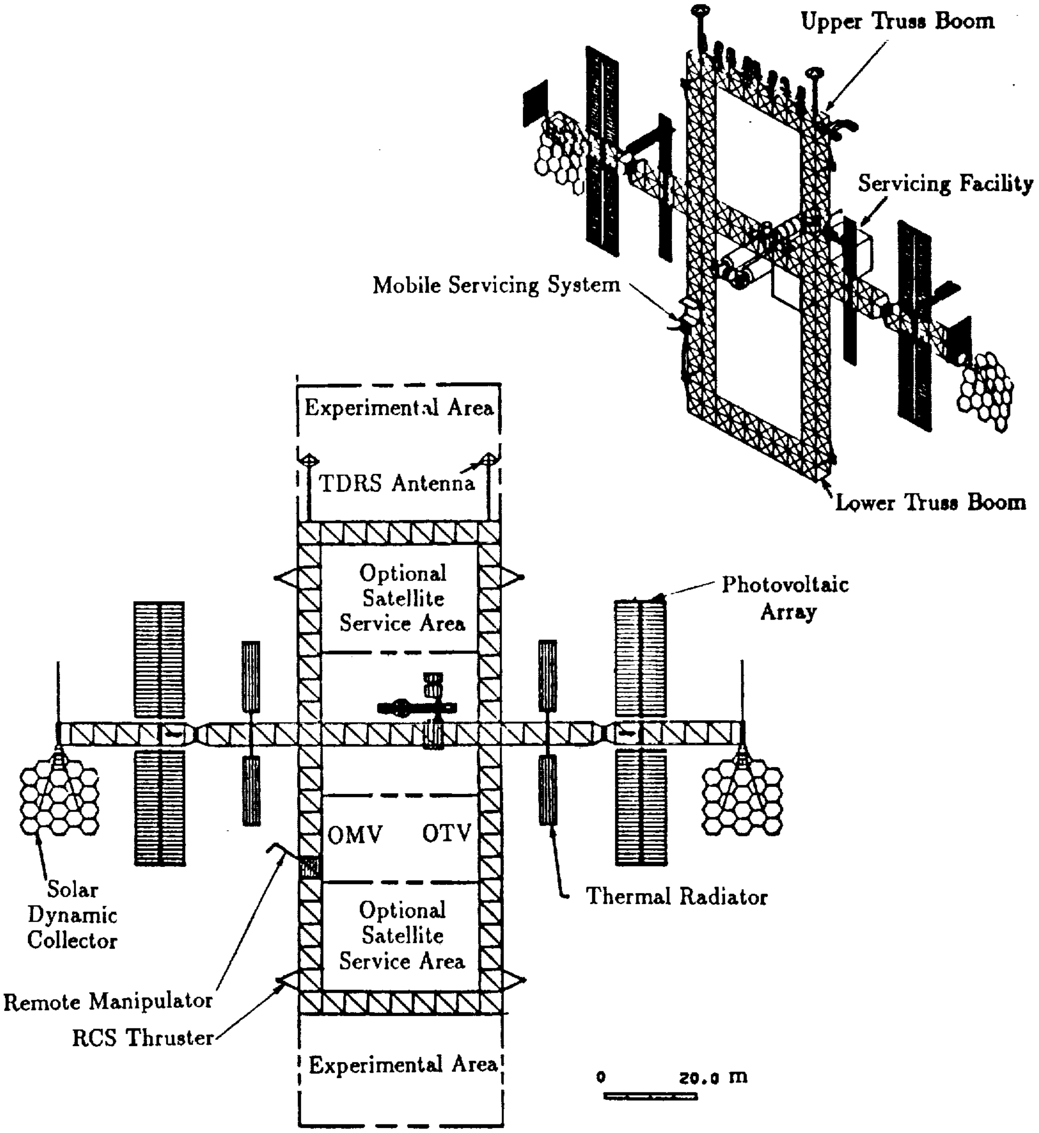


Fig.1 The U.S. Space Station.

2 DYNAMIC MODEL OF FLEXIBLE SPACE STATIONS

2.1 Introduction

We describe the dynamics of the space station by

Three Types of Motion:

- Rigid body translation perturbing the orbit.
- Rigid body rotation perturbing the orientation of the system.
- Vibration of the elastic members causing elastic deformation of the system.

To derive the dynamics we use the following coordinate systems:

2.2 Reference Coordinate Systems

- Body Coordinate System S_B: (i_b, j_b, k_b)
- Orbital Reference Coordinate System S_r: (i_r, j_r, k_r)
- Inertial Coordinate System S_I: (i_I, j_I, k_I)

Angular Velocities

$w_b = (w_x, w_y, w_z)'$: angular velocity of body frame w.r.t. the S_r.

$w_r = (w_X, w_Y, w_Z)'$:angular velocity of S_r w.r.t. S_I.

$w \equiv w_b + w_r = (w_1, w_2, w_3)'$:angular velocity of S_B w.r.t. S_I.

$$Q_i \equiv \begin{cases} \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, & \text{for i=1,2,3,4,} \\ \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}, & \text{for i=5,6.} \end{cases} \tag{1}$$

$$r_i = R + \bar{r}_i, \quad \text{with} \quad \bar{r}_i \equiv (\bar{x}_i, \bar{y}_i, \bar{z}_i)' = D_{i_0} + D_i, \tag{2}$$

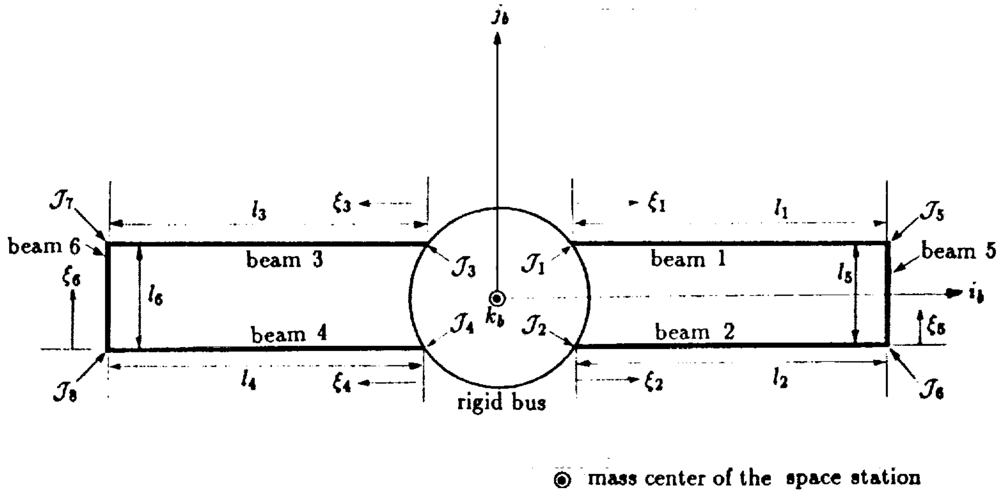
where $R \equiv (X, Y, Z)'$, $D_{i_0} \equiv (x_{i_0}, y_{i_0}, z_{i_0})'$, and $D_i \equiv Q_i\ (x_i, y_i, z_i)'$, i=1,2,...,6.

$\Omega_i \equiv (0, l_i)$, i=1,2,...,6 for l_i(the length of the beam i).

$$D_{1_o} = (s_{1_x} + \xi_1, s_{1_y}, 0)', \quad D_{2_o} = (s_{2_x} + \xi_2, s_{2_y}, 0)', \tag{3}$$

$$D_{3_o} = (-(s_{3_x} + \xi_3), s_{3_y}, 0)', \quad D_{4_o} = (-(s_{4_x} + \xi_4), s_{4_y}, 0)', \tag{4}$$

$$D_{5_o} = (s_{5_x}, -s_{5_y} + \xi_5, 0)', \quad D_{6_o} = (s_{6_x}, -s_{6_y} + \xi_6, 0)', \tag{5}$$



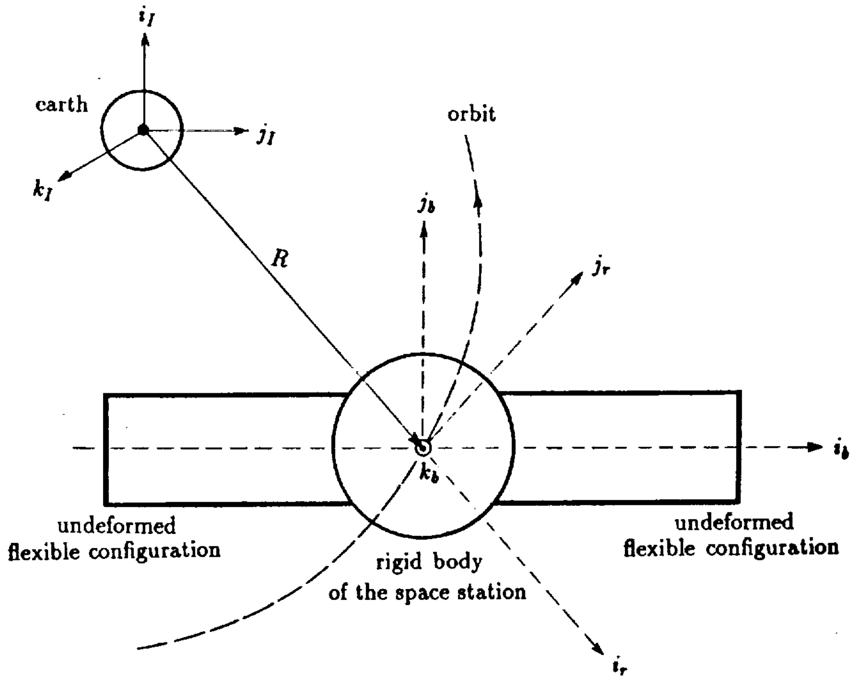


Fig.2 The space station and reference coordinate systems.

2.3 Inertia Tensor

The beam inertia tensor is given by

$$I_b \equiv \begin{pmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{xy} & I_{yy} & -I_{yz} \\ -I_{xz} & -I_{yz} & I_{zz} \end{pmatrix} \tag{6}$$

where

$$I_{xx} = \sum_{i=1}^{6} \int_{\Omega_i} \rho_i(\bar{y}_i^2 + \bar{z}_i^2) d\xi_i, \tag{7}$$

$$I_{yy} = \sum_{i=1}^{6} \int_{\Omega_i} \rho_i(\bar{x}_i^2 + \bar{z}_i^2) d\xi_i, \tag{8}$$

$$I_{zz} = \sum_{i=1}^{6} \int_{\Omega_i} \rho_i(\bar{x}_i^2 + \bar{y}_i^2) d\xi_i, \tag{9}$$

$$I_{xy} = \sum_{i=1}^{6} \int_{\Omega_i} \rho_i \bar{x}_i \bar{y}_i d\xi_i, \tag{10}$$

$$I_{xz} = \sum_{i=1}^{6} \int_{\Omega_i} \rho_i \bar{x}_i \bar{z}_i d\xi_i, \tag{11}$$

$$I_{yz} = \sum_{i=1}^{6} \int_{\Omega_i} \rho_i \bar{y}_i \bar{z}_i d\xi_i. \tag{12}$$

2.4 Dynamic Model of the Space Station

Let the Lagrangian of the whole system be denoted by L and external work by W. Then using the extended Hamilton's principle we obtain the relation[12,13]:

$$\begin{aligned} &\delta \int_{t_1}^{t_2} (L + W) dt \\ &\equiv - \int_{t_1}^{t_2} \left\{ (\delta R) \circ \mathcal{I}_1 - (\delta \theta) \circ \mathcal{I}_2 + \sum_{i=1}^{6} \int_{\Omega_i} (\delta D_i) \circ \mathcal{I}_3 d\xi_i + \mathcal{I}_4 \right\} dt \\ &\equiv 0, \end{aligned} \tag{13}$$

where

$$\mathcal{I}_1 = m_s \frac{d^2 R}{dt^2} + \frac{d}{dt} \sum_{i=1}^{6} \int_{\Omega_i} \left(\frac{dR}{dt} + \omega \times \bar{r}_i + \dot{D}_i \right) dm_i + \frac{G m_e m_\tau R}{|R|^3} - F_s, \tag{14}$$

$$\mathcal{I}_2 = T - \frac{d}{dt} \left\{ (I_s \circ \omega) + \sum_{i=1}^{6} \int_{\Omega_i} \bar{r}_i \times \frac{dr_i}{dt} dm_i \right\} + \sum_{i=1}^{6} \int_{\Omega_i} \left(\omega \times \bar{r}_i + \dot{D}_i \right) \times \left(\frac{dr_i}{dt} \right) dm_i, \tag{15}$$

$$\mathcal{I}_3 = \rho_i \frac{d^2 r_i}{dt^2} + \frac{\partial^2}{\partial {\xi_i}^2} \left(E I_i \frac{\partial^2 D_i}{\partial {\xi_i}^2} \right) - \bar{F}_{b_i}, \ i = 1, 2, ..., 6, \tag{16}$$

$$\mathcal{I}_4 = \sum_{i=1}^{6} \left\{ \left[E I_i \left(\frac{\partial^2 D_i}{\partial {\xi_i}^2} \right) \circ \frac{\partial}{\partial \xi_i} (\delta D_i) \right]_{\partial \Omega_i} - \left[\frac{\partial}{\partial \xi_i} \left(E I_i \frac{\partial^2 D_i}{\partial {\xi_i}^2} \right) \circ (\delta D_i) \right]_{\partial \Omega_i} \right\}, i = 1, 2, ..., 6. \tag{17}$$

Let

$$EI_i \equiv \begin{pmatrix} EI_{ix} & 0 & 0 \\ 0 & EI_{iy} & 0 \\ 0 & 0 & EI_{iz} \end{pmatrix}, \quad i = 1,2,...,6. \tag{18}$$

$$\bar{F}_{b_i} \equiv \begin{cases} (0, F_{i_y}, F_{i_z})', & \text{for } i\text{=1,2,3,4,} \\ (F_{i_x}, 0, F_{i_z})', & \text{for } i\text{=5,6.} \end{cases} \tag{19}$$

For boundary conditions we introduce the following assumptions:

Assumptions for the Joints and Beams:

(i) The structures are in xy plane in undisturbed state,

(ii) The beams 1,2,3,4 are clamped at the joints $\mathcal{J}_1, \mathcal{J}_2, \mathcal{J}_3, \mathcal{J}_4$,

(iii) Displacements of the beams 1,2,...,6 are small,

(iv) The beams 5,6 are rigidly jointed with the other four beams as shown in Fig.2.

For convenience of presentation we use the following notations:

$$\alpha_i \equiv \dot{\omega} \times \bar{r}_i + 2\omega \times \dot{D}_i + \omega \times (\omega \times \bar{r}_i), \; i = 1,2,...,6.$$

$$F_{b_i} \equiv \begin{cases} (F_{i_y}, F_{i_z})', & \text{for } i\text{=1,2,3,4,} \\ (F_{i_x}, F_{i_z})', & \text{for } i\text{=5,6.} \end{cases} \tag{20}$$

$$\tilde{Q}_i \equiv \begin{cases} \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, & \text{for i=1,2,3,4,} \\ \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}, & \text{for i=5,6,} \end{cases} \tag{21}$$

$$\text{mass of beams} : m_b = \sum_{i=1}^{6} \int_{\Omega_i} dm_i, \qquad \text{mass of the body} = m_s \tag{22}$$

$$\text{mass density of the beam} : \rho_i \equiv \rho_i(\xi_i), \qquad m_T \equiv m_s + m_b \tag{23}$$

$$I_b \circ \omega = \sum_{i=1}^{6} \int \bar{r}_i \times (\omega \times \bar{r}_i) dm_i \tag{24}$$

$$I_T \equiv I_s + I_b, \qquad I_s : \text{inertia of the body} \quad I_b : \text{inertia of the beams.} \tag{25}$$

Since the variations δR, $\delta\theta$ and δD_i are arbitrary in (13), $\mathcal{I}_i, i = 1,2,3$ are all zero provided $\mathcal{I}_4$=0 from the boundary conditions. Hence we obtain from this fact the following equations of motion[12,13]:

$\mathcal{I}_1$=0, that is,

$$m_T \frac{d^2R}{dt^2} + \sum_{i=1}^{6} \int_{\Omega_i} \rho_i \Big\{ (\dot{\omega} \times \bar{r}_i) + \omega \times (\omega \times \bar{r}_i) + \ddot{D}_i + 2\omega \times \dot{D}_i \Big\} d\xi_i + \frac{G m_e m_T R}{|R|^3} = F_s, \tag{26}$$

$\mathcal{I}_2$=0, that is,

$$I_T \circ \dot{\omega} + \omega \times (I_T \circ \omega) + \dot{I}_b \circ \omega + \sum_{i=1}^{6} \int_{\Omega_i} \rho_i \left(\bar{r}_i \times \frac{d^2R}{dt^2} \right) d\xi_i$$

$$+ \sum_{i=1}^{6} \int_{\Omega_i} \rho_i (\bar{r}_i \times \ddot{D}_i) d\xi_i + \sum_{i=1}^{6} \int_{\Omega_i} \rho_i \Big\{ \omega \times (\bar{r}_i \times \dot{D}_i) \Big\} d\xi_i = T, \tag{27}$$

and for $\xi_i \in \Omega_i$, $\xi_j \in \Omega_j$; i=1,2,3,4, j=5,6, $\mathcal{I}_3$=0, that is,

$$\rho_i \frac{\partial^2}{\partial t^2}\begin{pmatrix} y_i \\ z_i \end{pmatrix} + \rho_i \tilde{Q}_i\left(\alpha_i + \frac{d^2 R}{dt^2}\right) + \frac{\partial^2}{\partial {\xi_i}^2}\left(\begin{pmatrix} EI_{iy} & 0 \\ 0 & EI_{iz} \end{pmatrix} \frac{\partial^2}{\partial {\xi_i}^2}\begin{pmatrix} y_i \\ z_i \end{pmatrix}\right) = F_{b_i}, \tag{28}$$

$$\rho_j \frac{\partial^2}{\partial t^2}\begin{pmatrix} x_j \\ z_j \end{pmatrix} + \rho_j \tilde{Q}_j\left(\alpha_j + \frac{d^2 R}{dt^2}\right) + \frac{\partial^2}{\partial \xi_j^2}\left(\begin{pmatrix} EI_{jx} & 0 \\ 0 & EI_{jz} \end{pmatrix} \frac{\partial^2}{\partial \xi_j^2}\begin{pmatrix} x_j \\ z_j \end{pmatrix}\right) = F_{b_j}. \tag{29}$$

Let C_B^I denote the coordinate transformation matrix from body to inertial frame and $(a_i, a_j, a_k)' \equiv (C_B^I)'$ $\frac{d^2}{dt^2}R$. Then the equations of motion given in (26-29) can be written in explicit form for computer simulation as follows.

BODY DYNAMICS(TRANSLATION)

$$m_\tau \frac{d^2}{dt^2}\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} + C_B^I \begin{pmatrix} f_1(t) \\ f_2(t) \\ f_3(t) \end{pmatrix} + \begin{pmatrix} h_4(t) \\ h_5(t) \\ h_6(t) \end{pmatrix} = \begin{pmatrix} F_1 \\ F_2 \\ F_3 \end{pmatrix}, \tag{30}$$

where

$$\begin{aligned} f_1(t) = \sum_{i=1}^{6} \int_{\Omega_i} \rho_i \Big\{ &(\bar{z}_i \dot{\omega}_2 - \bar{y}_i \dot{\omega}_3) + 2(\omega_2 \frac{\partial z_i}{\partial t} - \omega_3 \frac{\partial y_i}{\partial t}) \\ &+ \omega_1 \omega_2 \bar{y}_i + w_1 w_3 \bar{z}_i - (w_2^2 + w_3^2)\bar{x}_i + \frac{\partial^2 x_i}{\partial t^2} \Big\} d\xi_i \end{aligned} \tag{31}$$

$$\begin{aligned} f_2(t) = \sum_{i=1}^{6} \int_{\Omega_i} \rho_i \Big\{ &\bar{x}_i \dot{w}_3 - \bar{z}_i \dot{w}_1 + 2(w_3 \frac{\partial x_i}{\partial t} - w_1 \frac{\partial z_i}{\partial t}) \\ &+ w_2 w_3 \bar{z}_i + w_2 w_1 \bar{x}_i - (w_3^2 + w_1^2)\bar{y}_i + \frac{\partial^2 y_i}{\partial t^2} \Big\} d\xi_i, \end{aligned} \tag{32}$$

$$\begin{aligned} f_3(t) = \sum_{i=1}^{6} \int_{\Omega_i} \rho_i \Big\{ &\bar{y}_i \dot{w}_1 - \bar{x}_i \dot{w}_2 + 2(w_1 \frac{\partial y_i}{\partial t} - w_2 \frac{\partial x_i}{\partial t}) \\ &+ w_3 w_1 \bar{x}_i + w_3 w_2 \bar{y}_i - (w_1^2 + w_2^2)\bar{z}_i + \frac{\partial^2 z_i}{\partial t^2} \Big\} d\xi_i, \end{aligned} \tag{33}$$

$$h_4(t) = \frac{G m_e m_\tau}{|R|^3} X, \tag{34}$$

$$h_5(t) = \frac{G m_e m_\tau}{|R|^3} Y, \tag{35}$$

$$h_6(t) = \frac{G m_e m_\tau}{|R|^3} Z, \tag{36}$$

BODY DYNAMICS(ROTATION)

$$I_\tau \begin{pmatrix} \dot{\omega}_1 \\ \dot{\omega}_2 \\ \dot{\omega}_3 \end{pmatrix} + \begin{pmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{pmatrix} I_\tau \begin{pmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{pmatrix} + \begin{pmatrix} f_4 + f_7 \\ f_5 + f_8 \\ f_6 + f_9 \end{pmatrix} = \begin{pmatrix} T_1 \\ T_2 \\ T_3 \end{pmatrix}, \tag{37}$$

where

$$f_4(t) = \sum_{i=1}^{6} \int_{\Omega_i} \rho_i \Big\{ \bar{y}_i \frac{\partial^2 z_i}{\partial t^2} - \bar{z}_i \frac{\partial^2 y_i}{\partial t^2} + 2w_1 \bar{y}_i \frac{\partial y_i}{\partial t} + 2w_1 \bar{z}_i \frac{\partial z_i}{\partial t} - 2w_2 \bar{y}_i \frac{\partial x_i}{\partial t} - 2w_3 \bar{z}_i \frac{\partial x_i}{\partial t} \Big\} d\xi_i, \tag{38}$$

$$f_5(t) = \sum_{i=1}^{6} \int_{\Omega_i} \rho_i \left\{ \bar{z}_i \frac{\partial^2 x_i}{\partial t^2} - \bar{x}_i \frac{\partial^2 z_i}{\partial t^2} + 2\omega_2 \bar{z}_i \frac{\partial z_i}{\partial t} + 2w_2 \bar{x}_i \frac{\partial x_i}{\partial t} - 2w_3 \bar{z}_i \frac{\partial y_i}{\partial t} - 2w_1 \bar{x}_i \frac{\partial y_i}{\partial t} \right\} d\xi_i, \tag{39}$$

$$f_6(t) = \sum_{i=1}^{6} \int_{\Omega_i} \rho_i \left\{ \bar{x}_i \frac{\partial^2 y_i}{\partial t^2} - \bar{y}_i \frac{\partial^2 x_i}{\partial t^2} + 2w_3 \bar{x}_i \frac{\partial x_i}{\partial t} + 2w_3 \bar{y}_i \frac{\partial y_i}{\partial t} - 2w_1 \bar{x}_i \frac{\partial z_i}{\partial t} - 2w_2 \bar{y}_i \frac{\partial z_i}{\partial t} \right\} d\xi_i, \tag{40}$$

$$f_7(t) = \sum_{i=1}^{6} \int_{\Omega_i} \rho_i \left\{ \bar{y}_i a_k - \bar{z}_i a_j \right\} d\xi_i, \tag{41}$$

$$f_8(t) = \sum_{i=1}^{6} \int_{\Omega_i} \rho_i \left\{ \bar{z}_i a_i - \bar{x}_i a_k \right\} d\xi_i, \tag{42}$$

$$f_9(t) = \sum_{i=1}^{6} \int_{\Omega_i} \rho_i \left\{ \bar{x}_i a_j - \bar{y}_i a_i \right\} d\xi_i, \tag{43}$$

BEAM DYNAMICS(**BEAM VIBRATION**)

$$\begin{aligned} &\rho_i \frac{\partial^2}{\partial t^2} \begin{pmatrix} y_i \\ z_i \end{pmatrix} + \rho_i \begin{pmatrix} 0 & -2\omega_1 \\ 2\omega_1 & 0 \end{pmatrix} \frac{\partial}{\partial t} \begin{pmatrix} y_i \\ z_i \end{pmatrix} + \frac{\partial^2}{\partial {\xi_i}^2} \left(\begin{pmatrix} EI_{iy} & 0 \\ 0 & EI_{iz} \end{pmatrix} \frac{\partial^2}{\partial {\xi_i}^2} \begin{pmatrix} y_i \\ z_i \end{pmatrix} \right) \\ &\quad + \rho_i \begin{pmatrix} -\omega_1^2 - \omega_3^2 & -\dot{\omega_1} + \omega_2\omega_3 \\ \dot{\omega_1} + \omega_2\omega_3 & -\omega_1^2 - \omega_2^2 \end{pmatrix} \begin{pmatrix} \bar{y_i} \\ \bar{z_i} \end{pmatrix} + \rho_i \bar{x}_i \begin{pmatrix} \dot{\omega_3} + \omega_1\omega_2 \\ -\dot{\omega_2} + \omega_1\omega_3 \end{pmatrix} \\ &\quad + \rho_i \tilde{Q}_i C_I^B \frac{d^2 R}{dt^2} = \begin{pmatrix} F_{i_y} \\ F_{i_z} \end{pmatrix}, \qquad \text{for } \xi_i \in \Omega_i,\ i = 1,2,3,4, \end{aligned} \tag{44}$$

$$\begin{aligned} &\rho_j \frac{\partial^2}{\partial t^2} \begin{pmatrix} x_j \\ z_j \end{pmatrix} + \rho_j \begin{pmatrix} 0 & 2\omega_2 \\ -2\omega_2 & 0 \end{pmatrix} \frac{\partial}{\partial t} \begin{pmatrix} x_j \\ z_j \end{pmatrix} + \frac{\partial^2}{\partial \xi_j^2} \left(\begin{pmatrix} EI_{jx} & 0 \\ 0 & EI_{jz} \end{pmatrix} \frac{\partial^2}{\partial \xi_j^2} \begin{pmatrix} x_j \\ z_j \end{pmatrix} \right) \\ &\quad + \rho_j \begin{pmatrix} -w_2^2 - w_3^2 & \dot{w}_2 + w_3 w_1 \\ -\dot{w}_2 + w_3 w_1 & -w_1^2 - w_2^2 \end{pmatrix} \begin{pmatrix} \bar{x_j} \\ \bar{z_j} \end{pmatrix} + \rho_j \bar{y}_j \begin{pmatrix} -\dot{w}_3 + w_2 w_1 \\ \dot{w}_1 + w_2 w_3 \end{pmatrix} \\ &\quad + \rho_j \tilde{Q}_j C_I^B \frac{d^2 R}{dt^2} = \begin{pmatrix} F_{j_x} \\ F_{j_z} \end{pmatrix}, \qquad \text{for } \xi_j \in \Omega_j,\ j = 5,6. \end{aligned} \tag{45}$$

BOUNDARY CONDITIONS FOR BEAM DYNAMICS

Since the beams(1-4) are clamped to the central body and the end beams(5,6) are rigidly jointed to the first four beams as shown in Fig.2, the following boundary conditions must hold.

Clamped at $\xi_i = 0$, i=1,2,3,4 for the joints $\mathcal{J}_1, \mathcal{J}_2, \mathcal{J}_3, \mathcal{J}_4$:

$$y_i(0,t) = 0, \qquad z_i(0,t) = 0, \tag{46}$$

$$\frac{\partial y_i}{\partial \xi_i}(0,t) = 0, \qquad \frac{\partial z_i}{\partial \xi_i}(0,t) = 0, \tag{47}$$

Continuity of displacements at the joints $\mathcal{J}_5, \mathcal{J}_6, \mathcal{J}_7, \mathcal{J}_8$:

$$x_5(l_5,t) = 0, \ \ z_1(l_1,t) = z_5(l_5,t), \ \ at\ \mathcal{J}_5, \tag{48}$$

$$x_5(0,t) = 0, \ \ z_2(l_2,t) = z_5(0,t), \ \ at\ \mathcal{J}_6, \tag{49}$$

$$x_6(l_6,t) = 0, \ \ z_3(l_3,t) = z_6(l_6,t), \ \ at\ \mathcal{J}_7, \tag{50}$$

$$x_6(0,t) = 0, \ \ z_4(l_4,t) = z_6(0,t), \ \ at\ \mathcal{J}_8, \tag{51}$$

Continuity of slopes at the joints $\mathcal{J}_5, \mathcal{J}_6, \mathcal{J}_7, \mathcal{J}_8$:

$$\frac{\partial y_1}{\partial \xi_1}(l_1,t) = -\frac{\partial x_5}{\partial \xi_5}(l_5,t), \ at\ \mathcal{J}_5, \tag{52}$$

$$\frac{\partial y_2}{\partial \xi_2}(l_2,t) = -\frac{\partial x_5}{\partial \xi_5}(0,t), \ at\ \mathcal{J}_6, \tag{53}$$

$$\frac{\partial y_3}{\partial \xi_3}(l_3,t) = \frac{\partial x_6}{\partial \xi_6}(l_6,t), \ at\ \mathcal{J}_7, \tag{54}$$

$$\frac{\partial y_4}{\partial \xi_4}(l_4,t) = \frac{\partial x_6}{\partial \xi_6}(0,t), \ at\ \mathcal{J}_8, \tag{55}$$

Equilibrium equations:

$$at\ \mathcal{J}_5, \quad EI_{5z}\frac{\partial^2 z_5}{\partial \xi_5^2}(l_5,t) = 0, \quad EI_{1z}\frac{\partial^2 z_1}{\partial \xi_1^2}(l_1,t) = 0, \tag{56}$$

$$at\ \mathcal{J}_6, \quad EI_{5z}\frac{\partial^2 z_5}{\partial \xi_5^2}(0,t) = 0, \quad EI_{2z}\frac{\partial^2 z_2}{\partial \xi_2^2}(l_2,t) = 0, \tag{57}$$

$$at\ \mathcal{J}_7, \quad EI_{6z}\frac{\partial^2 z_6}{\partial \xi_6^2}(l_6,t) = 0, \quad EI_{3z}\frac{\partial^2 z_3}{\partial \xi_3^2}(l_3,t) = 0, \tag{58}$$

$$at\ \mathcal{J}_8, \quad EI_{6z}\frac{\partial^2 z_6}{\partial \xi_6^2}(0,t) = 0, \quad EI_{4z}\frac{\partial^2 z_4}{\partial \xi_4^2}(l_4,t) = 0, \tag{59}$$

$$EI_{1y}\frac{\partial^3 y_1}{\partial \xi_1^3}(l_1,t) = 0, \qquad EI_{3y}\frac{\partial^3 y_3}{\partial \xi_3^3}(l_3,t) = 0, \tag{60}$$

$$EI_{2y}\frac{\partial^3 y_2}{\partial \xi_2^3}(l_2,t) = 0, \qquad EI_{4y}\frac{\partial^3 y_4}{\partial \xi_4^3}(l_4,t) = 0, \tag{61}$$

$$EI_{1z}\frac{\partial^3 z_1}{\partial \xi_1^3}(l_1,t) + EI_{5z}\frac{\partial^3 z_5}{\partial \xi_5^3}(l_5,t) = 0, \tag{62}$$

$$EI_{3z}\frac{\partial^3 z_3}{\partial \xi_3^3}(l_3,t) + EI_{6z}\frac{\partial^3 z_6}{\partial \xi_6^3}(l_6,t) = 0, \tag{63}$$

$$EI_{2z}\frac{\partial^3 z_2}{\partial \xi_2^3}(l_2,t) - EI_{5z}\frac{\partial^3 z_5}{\partial \xi_5^3}(0,t) = 0, \tag{64}$$

$$EI_{4z}\frac{\partial^3 z_4}{\partial \xi_4^3}(l_4,t) - EI_{6z}\frac{\partial^3 z_6}{\partial \xi_6^3}(0,t) = 0, \tag{65}$$

$$EI_{1y}\frac{\partial^2 y_1}{\partial \xi_1^2}(l_1,t) - EI_{5x}\frac{\partial^2 x_5}{\partial \xi_5^2}(l_5,t) = 0, \tag{66}$$

$$EI_{3y}\frac{\partial^2 y_3}{\partial \xi_3^2}(l_3,t) + EI_{6x}\frac{\partial^2 x_6}{\partial \xi_6^2}(l_6,t) = 0, \tag{67}$$

$$EI_{2y}\frac{\partial^2 y_2}{\partial \xi_2^2}(l_2,t) + EI_{5x}\frac{\partial^2 x_5}{\partial \xi_5^2}(0,t) = 0, \tag{68}$$

$$EI_{4y}\frac{\partial^2 y_4}{\partial \xi_4^2}(l_4,t) - EI_{6x}\frac{\partial^2 x_6}{\partial \xi_6^2}(0,t) = 0. \tag{69}$$

3. STABILIZATION OF SPACE STATIONS

3.1 The Reference Trajectory

Let R_0 denote the reference trajectory(nominal trajectory) governed by the equation

$$\frac{d^2 R_0}{dt^2} + \frac{G m_e R_0}{|R_0|^3} = 0 \,, \tag{70}$$

with the initial conditions:

$$\frac{dR_0}{dt} \times \left(R_0 \times \frac{dR_0}{dt}\right) = \frac{G m_e R_0}{|R_0|}, \qquad \text{at } \ t = 0. \tag{71}$$

Denoting the excursion of the radial vector R around the nominal trajectory by $\tilde{R} \equiv R - R_0 = (\tilde{X}, \tilde{Y}, \tilde{Z})'$ we obtain from eqs.(26) and (70) the perturbed radial dynamics:

$$m_\tau \frac{d^2 \tilde{R}}{dt^2} + \sum_{i=1}^{6} \frac{d}{dt} \int_{\Omega_i} \rho_i \Big(\omega \times \bar{r}_i + \dot{D}_i\Big) d\xi_i + \frac{G m_e m_\tau \tilde{R}}{|R_0|^3} = F_s. \tag{72}$$

3.2 The Rest State

For stability of the system subject to external disturbances we consider the rest state:

$$w_1 = 0, \;\; w_2 = 0, \;\; w_3 = 0, \tag{73}$$

$$\tilde{X}(t) = \tilde{Y}(t) = \tilde{Z}(t) = 0, \quad \tilde{v}_1(t) = \tilde{v}_2(t) = \tilde{v}_3(t) = 0, \tag{74}$$

and for $k = 5, 6$; $j = 1, 2, 3, 4$; $i = 1, 2, ..., 6$,

$$x_k(\xi_k, t) = y_j(\xi_j, t) = z_i(\xi_i, t) = 0, \;\; \frac{\partial x_k}{\partial t}(\xi_k, t) = \frac{\partial y_j}{\partial t}(\xi_j, t) = \frac{\partial z_i}{\partial t}(\xi_i, t) = 0, \tag{75}$$

where $\tilde{v} \equiv \frac{d\tilde{R}}{dt} \equiv (\tilde{v}_1, \tilde{v}_2, \tilde{v}_3)'$.

Theorem 1 **(Distributed Control)**

Consider the perturbed system described by the equations (30-69) around the reference trajectory. Suppose that the controls applied to the system are given by the feedback law:

$$T = (T_x - k_1\omega_1, \; T_y - k_2\omega_2, \; T_z - k_3\omega_3)', \quad k_1, k_2, k_3 > 0, \tag{76}$$

$$F_{b_i} = Q_i\Big(\tilde{A}_{i_x} - d_{i_1}\frac{\partial x_i}{\partial t}, \tilde{A}_{i_y} - d_{i_2}\frac{\partial y_i}{\partial t}, \;\; \tilde{A}_{i_z} - d_{i_3}\frac{\partial z_i}{\partial t}\Big)', \quad d_{i_1}, d_{i_2}, d_{i_3} > 0, \; i = 1, 2, ..., 6, \tag{77}$$

$$F_s = (-e_1\tilde{v}_1, \; -e_2\tilde{v}_2, \; -e_3\tilde{v}_3)', \qquad e_1, e_2, e_3 > 0, \tag{78}$$

where $(\tilde{A}_{i_x}, \tilde{A}_{i_y}, \tilde{A}_{i_z})' \equiv -\rho_i \frac{G m_e}{|R_0|^3} R_0$ and $(T_x, T_y, T_z)' \equiv -\frac{G m_e}{|R_0|^3}\big(\sum_{i=1}^{6} \int_{\Omega_i} \rho_i \bar{r}_i d\xi_i\big) \times R_0$. Then the system is asymptotically stable (in the sense of Lyapunov) with respect to the rest state (73-75).
Proof see the refs.[12,13].

Remark

Defining the state variable x appropriately and incorporating the boundary conditions in the differential operator one can rewrite the system equation as an ordinary differential equation in a Banach space (probably Orlicz space) as follows

$$\dot{x} = Ax + F(x, \dot{x}), \tag{79}$$

where A can be proved to be the infinitesimal generator of a C_0- semigroup in the Banach space and F is a strongly nonlinear operator having polinomial growth. This is a descriptor system in an infinite dimensional space and very little is known about these systems concerning the existence and uniqueness of nonlinear semigroups.

4 NUMERICAL RESULTS

For numerical simulation we assume that (i) the bus inertia tensor is diagonal. (ii) the flexible members consist of three beams (i.e., beams 1,2 and 5 in Fig.4) and each beam is uniform.

The following data and parameters were used.

System Parameters:

I_s=diag(I_1, I_2, I_3) = diag(31750, 5000, 33450) $slug\ ft^2$

the length of the beams: l_1 =l_2 =187.7 ft, l_5 =66 ft,

Flexural rigidity: EI_{5x} =EI_{5z} =$10^5 lb\ ft^2$, EI_{iy} =EI_{iz} =3565.5 $lb\ ft^2$, for i=1,2,5,

Mass density: ρ_i =8.25 × 10^{-2} $slug/ft$,

D_{1_0} =(3+ ξ_1, 33, 0)′, D_{2_0} =(3+ ξ_2, -33,0)′, D_{5_0} =(190.7, -33+ ξ_5, 0)′ft, m_s =300 $slug$.

Initial Conditions:

ω_1 (0)=0.03, $\omega_2(0)$ =0.02, $\omega_3(0)$ = 0.01 rad/sec and for ξ_i ∈[0,l_i] and i=1,2, 5, $\frac{\partial x_i}{\partial t}(\xi_i,0)$ =$\frac{\partial y_i}{\partial t}(\xi_i,0)$ =$\frac{\partial z_i}{\partial t}(\xi_i,0)$ =0, $x_i(\xi_i,0)$= $y_i(\xi_i,0)$ =$z_i(\xi_i,0)$ =ϕ_o (ξ_i), where ϕ_o satisfies the boundary conditions (equations 3.97-3.120).

Control Gains:

k_1 =30000, k_2 =10000, k_3 =100 for t ∈[0,2.5] and k_3 =50000 for t ∈[2.5,10]; d_{i_1} =d_{i_2} =d_{i_3} =0.05 for i= 1,2,5; e_1=9000, e_2=3000, e_3=6000. $F_1(t)$ =$F_2(t)$ =$F_3(t)$ =0.

Slew maneuver using Bang -Pause -Bang control with the parameters:

T_{s_1}=10035, t_{s_1}=0.5sec, t_{s_2}=2.0sec, t_{s_3}=2.5sec.

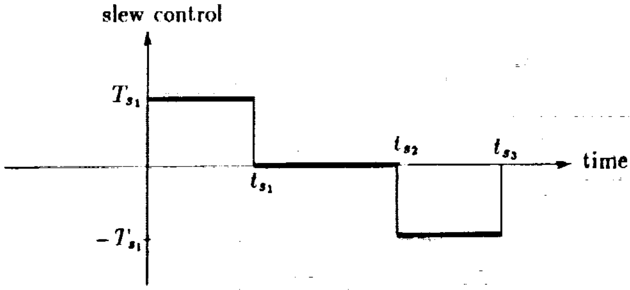


Fig.3 Bang-Pause-Bang Control.

Detailed numerical results showing stabilization of the various state trajectories were obtained from the simulation and presented in Figs. 4-23. Figs.4 and 5 represent the slew angle and the angular rate of the spacecraft corresponding to the slew control (Fig.3). We note from the Fig.4 that slew maneuver was successfully achieved by the slew control commands. As one can see from Eqs.(30-45), the body translation, body rotation and vibration of beams are strongly coupled. This implies that any perturbation in one part of the system will induce disturbances in the other members of the system. Hence the slew maneuver induced significant perturbations leading to beam vibrations and body rotation. This is clearly observed from the responses without controls. From Figs.4-23 it is clear that without stabilizing controls beam vibrations, body oscillations and radial excursions persist or grow. However, with application of the proposed feedback controls, oscillations induced by the slew maneuver are effectively eliminated throughout the entire system.

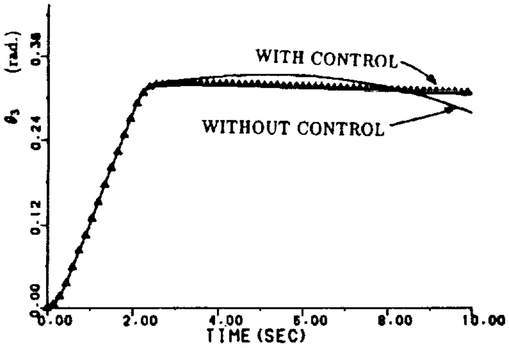


Fig. 4 Slew angle θ_3 in k_b direction.

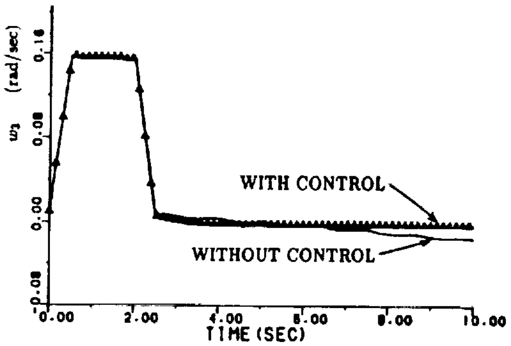


Fig. 5 Bus angular velocity w_3.

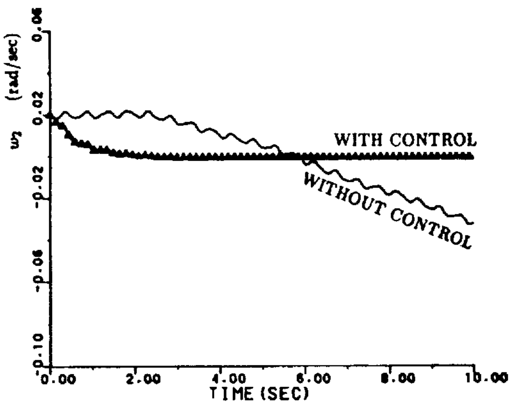


Fig. 6 Bus angular velocity w_2.

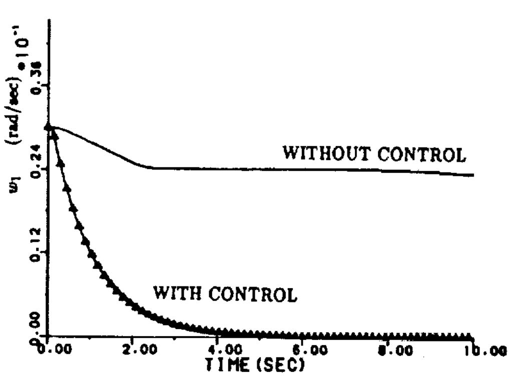


Fig. 7 Bus angular velocity w_1.

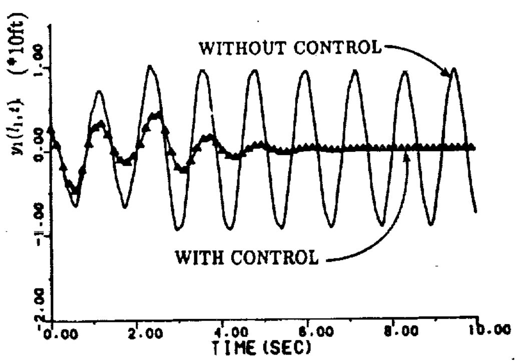


Fig. 8 Beam displacement $y_1(l_1,t)$.

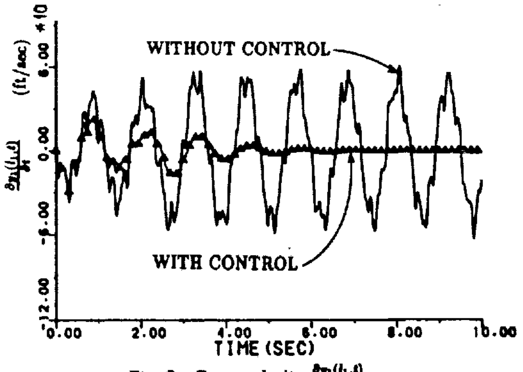


Fig. 9 Beam velocity $\frac{\partial y_1(l_1,t)}{\partial t}$.

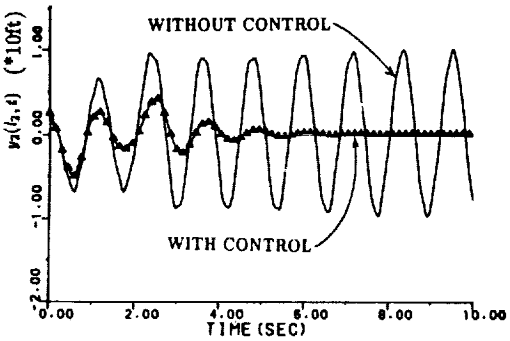


Fig. 10 Beam displacement $y_2(l_2,t)$.

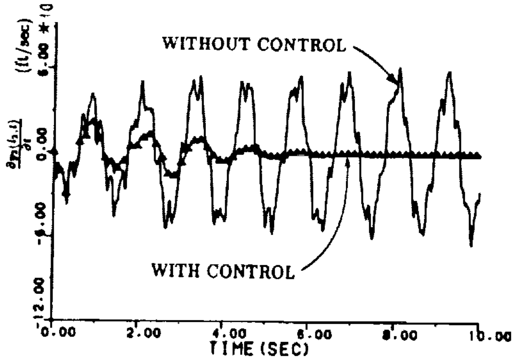


Fig 11 Beam velocity $\frac{\partial y_2(l_2,t)}{\partial t}$.

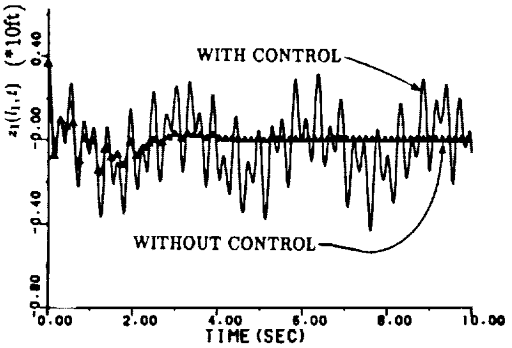


Fig.12 Beam displacement $z_1(l_1,t)$.

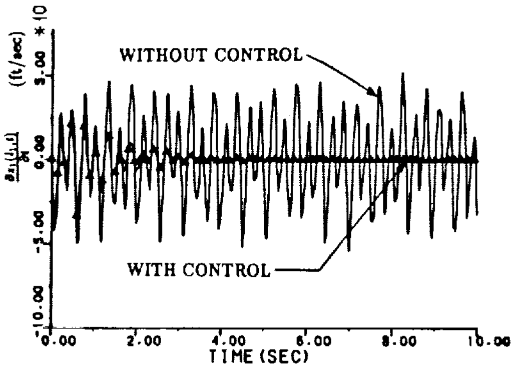


Fig.13 Beam velocity $\frac{\partial z_1(l_1,t)}{\partial t}$.

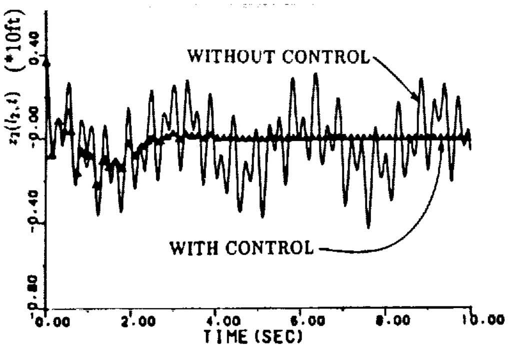


Fig.14 Beam displacement $z_2(l_2,t)$.

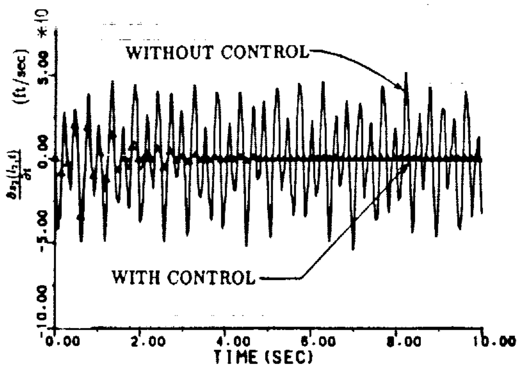


Fig 15 Beam velocity $\frac{\partial z_2(l_2,t)}{\partial t}$.

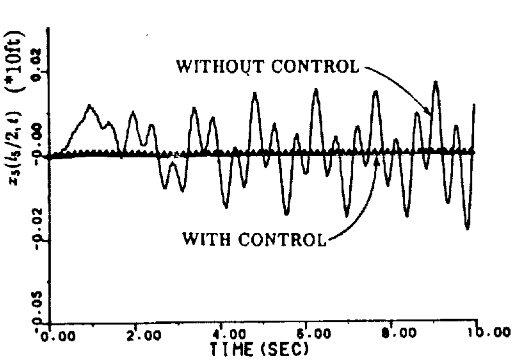


Fig. 16 Beam displacement $z_5(l_5/2,t)$.

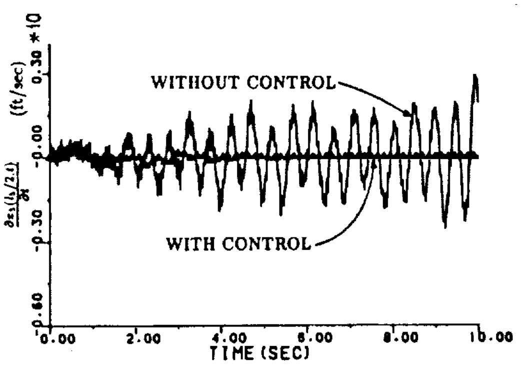


Fig. 17 Beam velocity $\frac{\partial z_5(l_5/2,t)}{\partial t}$.

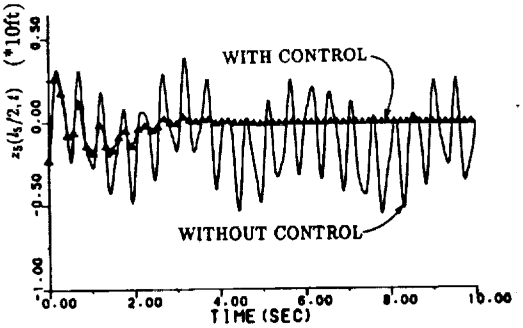


Fig. 18 Beam displacement $z_5(l_5/2,t)$.

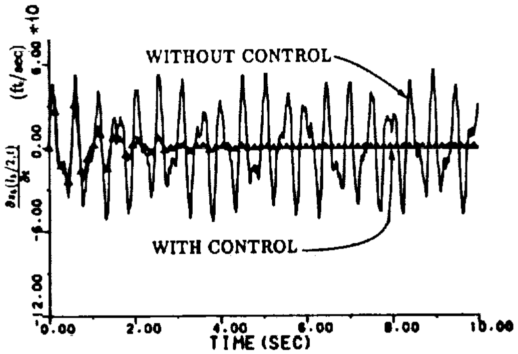


Fig. 19 Beam velocity $\frac{\partial z_5(l_5/2,t)}{\partial t}$.

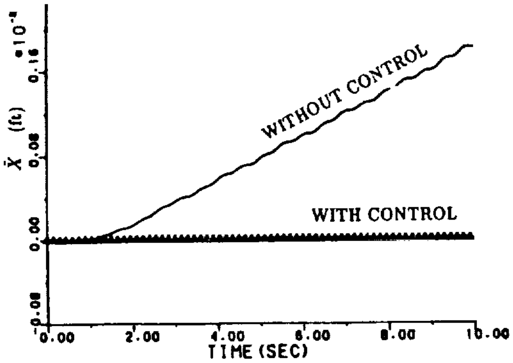


Fig. 20 Radial perturbation $\bar{X}$.

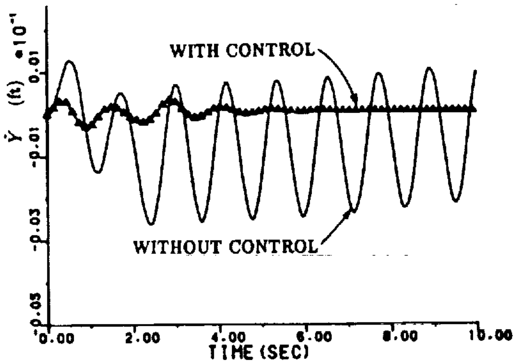


Fig. 21 Radial perturbation $\bar{Y}$.

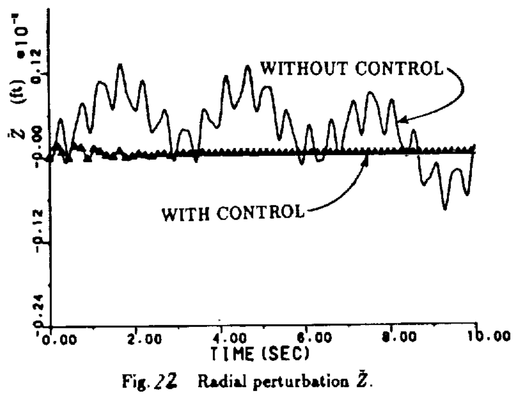


Fig. 22 Radial perturbation $\bar{Z}$.

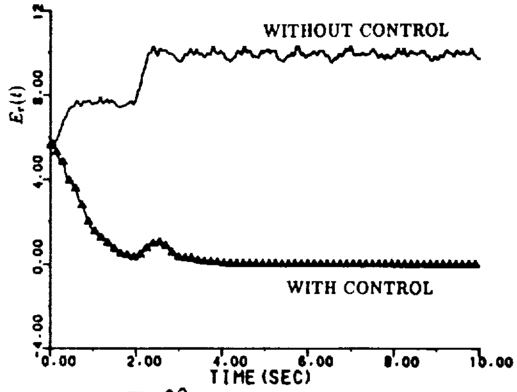


Fig. 23 Relative beam energy.

5. CONCLUSIONS

In this paper, suppression of vibration induced by slew maneuver in flexible space stations has been considered. Based on the dynamic model developed by the authors[13], asymptotic stability of the system subject to perturbation is investigated. The rotational perturbing forces were applied to the system and their corresponding stabilizations have been demonstrated. From the numerical results it is clearly observed that (i)if the disturbances following external perturbing forces persist, then in the absence of proper controls, these small motions may build up leading to instability of the entire system, (ii)during the slew maneuver, vibration in the beams and oscillations in the angular velocities of the body are induced and it has been shown that the stabilizing control can effectively eliminate the oscillations and stabilize the entire system.

REFERENCES

[1] L. Taylor Jr, "4th Annual NASA SCOLE Workshop", NASA TM 101503, 1987.

[2] J.G. Lin, "Control Design Challenges of Large Space Systems ans SCOLE", NASA CR-178392, Oct. 1987.

[3] A.V. Balakrishnan, "A Mathematical Formulation of the SCOLE Control problem, Part I", NASA CR 172581, 1985.

[4] M.J. Balas,"Trends in Large Space Structure Control Theory-Fondest Hopes, Wildest Dreams", IEEE Trans. on Auto. Control, AC-27, pp.522-535, 1982.

[5] L. Meirovitch, "Stability of a Spinning Body Containing Elastic Parts via Lyapunov Direct Method", AIAA J., Vol 8, pp.1193-1200, 1970.

[6] N.U. Ahmed and S.K. Biswas, "Mathematical Modeling and Control of Large Space Structures with Multiple Appendages", Mathematical and Computer Modelling, Vol.10, pp891-900, 1988.

[7] N.U. Ahmed and S.S. Lim,"Modelling and Stabilization of Flexible Spacecraft Under the Influence of Orbital Perturbation", Proc. of the 26th Conf. on Decision and Control, Los Angeles, CA, pp.2331-2336, Dec. 1987.

[8] S.K. Biswas and N.U. Ahmed,"Modelling of Flexible Spacecraft and Their Stabilization", Int. J. of Systems Science, Vol.16, pp. 535-551, 1985.

[9] S.K. Biswas and N.U. Ahmed, "Stabilization of A Class of Hybrid System Arising in Flexible Spacecraft", J. of Optimization Theory and Applications, Vol. 50, No. 15, pp.83-103, 1986.

[10] S.S. Lim and N.U. Ahmed, "Stabilizing Controls for Flexible Spacecraft under the Influence of Orbital Perturbation", Int. J. of Science and Technology, Vol.1, No.2, pp5-20, 1988.

[11] S.S. Lim and N.U. Ahmed,"Mathematical Modeling and Stabilization of Large Flexible Spacecraft Subject to Orbital Perturbation", Mathematical and Computer Modelling. (to appear)

[12] S.S. Lim, "Modeling and Control of Large Space Structures", Ph.D. thesis, University of Ottawa, Ottawa, Ontario, July 1989.

[13] N.U. Ahmed and S.S. Lim, "Towards Modeling and Control of Large Space Stations", Proc. of the 28th Conf. on Decision and Control, Tampa, Florida, Dec.13-15, 1989.(to appear)

Mini-Mast Dynamic Analysis Using the Truss-Beam Model

Elias G. Abu-Saba
William M. McGinley
Raymond C. Montgomery

Abstract

The Mini-Mast is a generic space truss designed by Astro Aerospace Corporation of California. A truss 210-meter-long located in the Structural Dynamics research Laboratory of NASA Langley is used in comprehensive active-vibration-control experiments on a realistic large space structure. Some predictions of the natural frequencies and mode shapes have been made based on the Finite Element model.

The purpose of this paper is to use the Truss-Beam model developed by Elias G. Abu-Saba to predict the natural frequencies of the Mini-Mast and then compare the results with those obtained from the FE model. Imperfections of the joints will be modeled, and joint contribution to the flexibility matrix of the structure will be noted. The natural frequencies will be obtained. An iterative procedure will be used to enhance the Truss-Beam model by comparing these results with experimental ones.

7
N90-23088

Neural Networks in Support of Manned Space

Dr. Paul J. Werbos
Administrator, Neuroengineering
National Science Foundation

Abstract

Many lobbyists in Washington have argued that artificial intelligence (AI) is an alternative to manned space activity. In actuality, this is the opposite of the truth, especially as regards artificial neural networks (ANNs), that form of AI which has the greatest hope of mimicking human abilities in learning, ability to interface with sensors and actuators, flexibility and balanced judgement.

This talk will begin by briefly reviewing ANNs and their relation to expert systems (the more traditional form of AI), and the limitations of both technologies. It will give a few highlights of recent work on ANNs, including an NSF-sponsored workshop on ANNs for control applications. It will then discuss current thinking on ANNs for use in certain key areas — the National Aerospace Plane, teleoperation, the control of large structures, fault diagnostics, and docking — which may be crucial to the long-term future of man in space.

N90-23089

A VERIFICATION LIBRARY FOR MULTIBODY SIMULATION SOFTWARE

Sung-Soo Kim and Edward J. Haug
The Center for Simulation and Design Optimization of Mechanical Systems
The University of Iowa
Iowa City, Iowa 52242

Harold P. Frisch
NASA Goddard Space Flight Center
Greenbelt, Maryland 20771

Abstract

A multibody dynamics verification library, that maintains and manages test and validation data is proposed, based on RRC Robot arm and CASE backhoe validation and a comparative study of DADS, DISCOS, and CONTOPS that are existing public domain and commercial multibody dynamic simulation programs. Using simple representative problems, simulation results from each program are cross checked, and the validation results are presented. Functionalities of the verification library are defined, in order to automate validation procedure.

1. Introduction

Multibody simulation software programs are currently used for an extremely broad range of applications; e.g., robotics, space structures, automotive vehicles, farm machinery, spacecraft, etc. Most multibody programs in active use have passed an exhaustive series of theoretical tests. However, none have been subject to the rigors of an extensive laboratory test and validation program. A project supported by NASA has been established to validate and evaluate multibody simulation programs through experimental testing and theoretical cross checking, so that engineers can utilize simulation software with confidence. Moreover, through this validation and evaluation procedure, modeling and analysis capabilities that must be developed can be identified for future code enhancements.

To carry out validation for current and future flexible multibody simulation programs, there is a need to define and to perform a series of laboratory tests that can be used as references. There is also a need to set up a library of test and validation data that are maintained in a format that is compatible with input and output data requirements of commercially available and public domain multibody simulation programs.

The verification procedure envisioned involves (1) defining actual mechanical systems and tests, (2) performing the series of tests, (3) modeling the mechanical systems, (4) simulation and test data processing, and (5) comparison between simulation data from different software and experimental data for validation. To alleviate the engineer's burden in modeling, simulation, and data post-analysis, a systematic tool; i.e., a verification library system, is being developed to automate the verification procedure by integrating software modules to store models, launch simulation software, and manage data.

To develop this verification library system requires (1) a survey of multibody simulation software to investigate modeling and analysis capability and at the same time to identify a standard input and output data format for the verification library, (2) theoretical cross verification among simulation software and validation of multibody programs with generic multibody problems through experimental tests, and (3) definition of engineering capabilities for the verification library system, based on experience obtained from tasks (1), and (2).

The purposes of this paper are the to (1) present current verification activities, based on a comparative study of the flexible multibody simulation programs DADS, DISCOS, and CONTOPS and validation of those programs through theoretical cross checking and experimental testing, and (2) to define a verification library system; i.e., its functionality and software architecture. Note that DADS, DISCOS and CONTOPS are multibody simulation software that treat flexible body components and have integrated capabilities for simulation of the mechanical subsystems and the control subsystems. In Section 2, current validation activity is presented. Section 3 presents a summary of current multibody simulation capabilities. Finally, the concept of the verification library system is defined in Section 4.

2. Current Validation Activities and Status

2.1 Validation of Manipulator System

Manipulator arms have been chosen as generic multibody problems, since they are actively controlled variable kinematic topology systems, when the end effector contacts ground, and their joints have nonlinear effects such as friction and flexibility. Two manipulator systems have been simulated and tested for verification. One is the RRC (Robot Research Corporation) robot arm and the other is a CASE construction backhoe.

The RRC arm shown in Fig. 1 has 7 revolute joints, each with a harmonic drive gear transmission. This arm is actively driven by DC servo-motors with position, velocity, and torque feedback controllers. Due to the high gear ratio of the harmonic drive, effective rotor inertia effects and gyroscopic forces are significant.

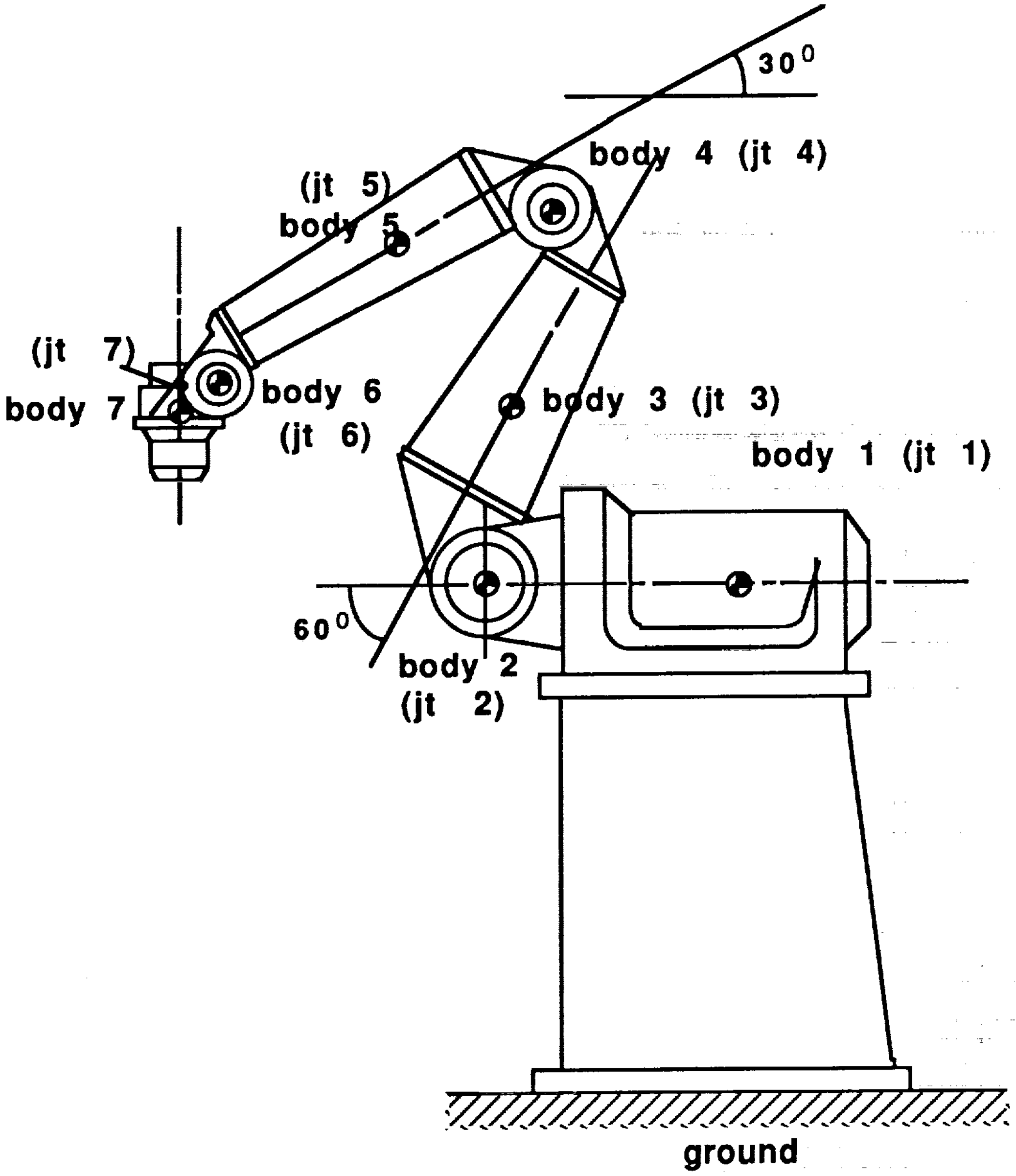


Figure 1. RRC Robot Arm

Several simple and moderately complicated experimental tests have been performed at NASA Goddard robotics laboratory. Test data have been collected and processed, to be compared with simulation results. RRC arm simulation models have been created with different degrees of fidelity, according to inclusion of the gear reducer, friction, and joint controllers, using DADS and the Order N Iowa program [1,2]. Validation of the dynamic and dynamic/control simulation is under way. For dynamics validation, experimentally obtained joint control torques have been imposed in the simulation model. Joint displacements and velocities from experiments and simulations can thus be compared. In this way, dynamic simulation can be isolated from dynamic/control simulation. For dynamic/control validation, the same controller reference input is imposed in the simulation model to obtain displacement, velocity, and control torque of each joint, to be compared with experimental data. Details will be presented in Ref. 3.

The CASE backhoe system is manipulated by hydraulic actuators and consists of topological closed loops. Joint frictions are important dynamic effects. A simulation model has been created using the DADS program. Piston displacements and forces in hydraulic actuator have been validated through experimental tests. Static strains of several interest points in the boom has been also validated. Detailed validation results are presented in Ref. 4.

2.2 Theoretical Cross Verification with DADS, DISCOS, and CONTOPS

To validate multibody simulation codes such as DADS, DISCOS, and CONTOPS by cross checking simulation results, four representative simple multibody problems were selected; i.e., rigid body open and closed systems and flexible body open and closed loop systems. Since details are presented in Refs. 5-9, only validation results are summarized in this paper.

As a simple rigid body open loop multibody problem, the double pendulum system shown in Fig. 2 was simulated with all three programs. Springs and dampers are attached to joints 1 and 2. The simulation was carried out under the influence of gravitational force in negative y direction. All of three programs generated essentially the same solution.

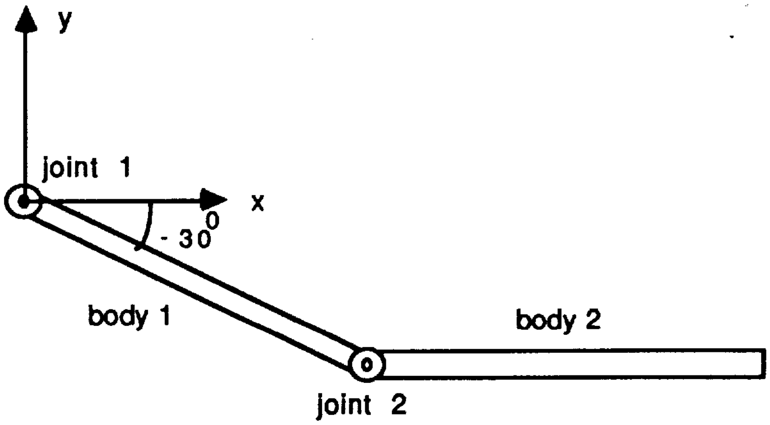


Figure 2. Double Pendulum

As a rigid body closed loop multibody problem, the four bar linkage mechanism shown in Fig. 3 was cross validated. Springs and dampers are mounted at joints 1 and 4. Under the influence of gravitational force in the negative y direction, simulations were carried out. Since the DISCOS program cannot handle rigid body closed loop systems, cross verification was done only between DADS and CONTOPS, which yielded the same simulation results.

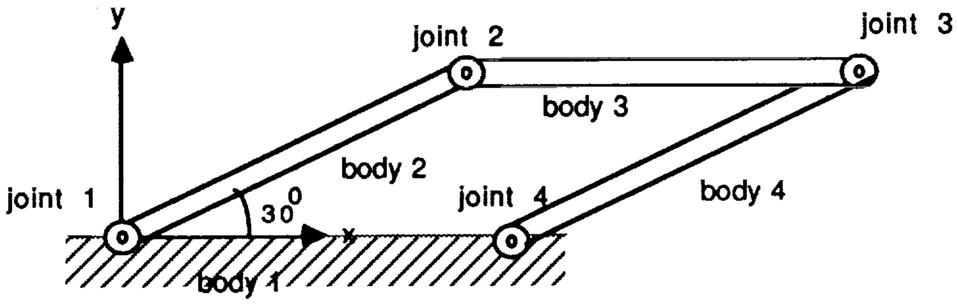


Figure 3. Rigid Four Bar Linkage

As a flexible open loop multibody problem, a flexible beam that are attached to a moving body was tested. A schematic diagram of the flexible beam is presented in Fig. 4. Body 2 rotates about the z axis with constant angular velocity. The flexible beam is initially deformed. In order to represent flexibility of the beam, the first two vibrational normal modes with clamped boundary conditions were employed. Simulations were carried out without gravitational force. Since CONTOPS has no provision for imposing a pre-strained initial configuration, only DADS and DISCOS were cross validated. Essentially the same results were obtained with both codes.

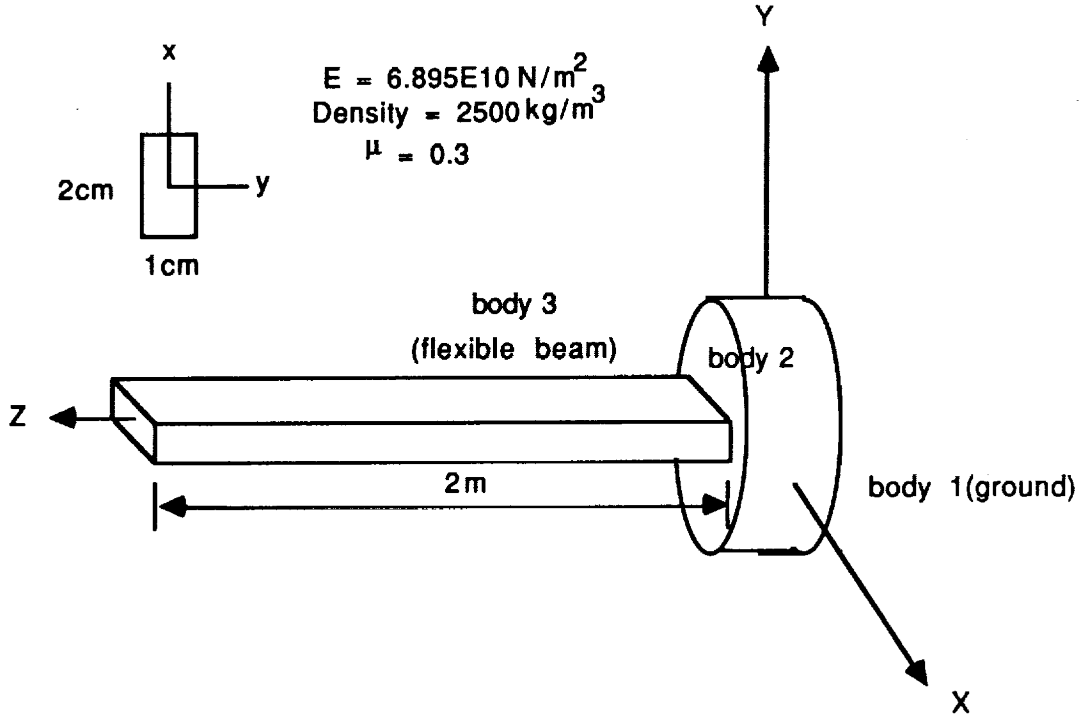


Figure 4. Rotating Flexible Beam

A four bar linkage with a flexible coupler was tested, as shown in Fig. 5. It is difficult to use CONTOPS program for this application, since the user must provide time independent coefficient terms related to the flexible body [9], and no provision is made for imposing initial modal coordinates and rates. Thus, only DADS and DISCOS simulations were carried out, without gravity force. Since the DISCOS program requires at least 6 vibrational modes for any closed loop system, six vibrational modes including an axial direction mode were used to represent flexibility of the coupler. Gross motion and dominant deformation motion (lateral bending) were the same for both DADS and DISCOS solutions. However, slightly different results were obtained in axial motion of the coupler. With a moderate integration step size, DISCOS generated axial motion values that were approximately the average of the oscillatory motion values obtained by DADS. With a smaller integration step size, the axial motion values of DISCOS tended to converge with those of DADS. Thus, DISCOS required a small integration step size, in order to produce the same results as DADS.

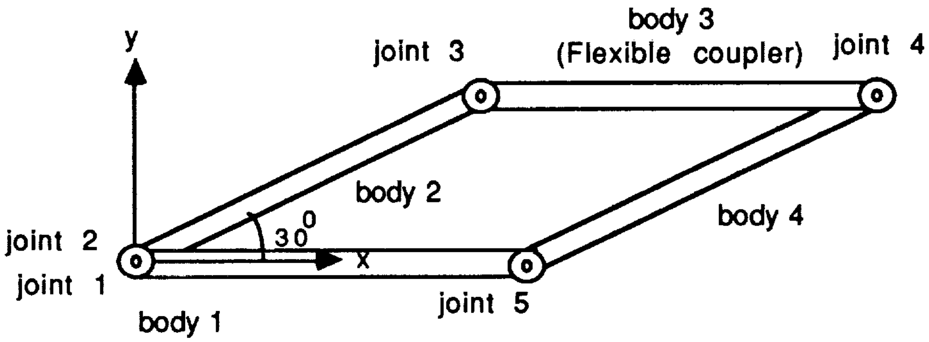


Figure 5. Flexible Four Bar Linkage

3. Current Status of Multibody Simulation Software

A comparative study [10] has been made among the simulation codes such as DADS, DISCOS, and CONTOPS, based on the experience of theoretical cross verification. These three programs are the first candidate simulation software to be validated for verification library. Difficulties in defining standard data for verification library, such as simulation and test data and their format were identified through this comparative study. This comparative study illustrates modeling and analysis capabilities of each simulation code.

3.1 Generality

The current multibody simulation programs are biased to generic problem classes. Thus, modeling and analysis capabilities of each code are different. DADS has been developed for dynamics of mechanisms and ground vehicles, whereas DISCOS and CONTOPS are spacecraft oriented. Thus, there are several differences in modeling and analysis approaches. DADS handles closed loop mechanical systems such as ground vehicle suspensions and mechanisms as easily as open loop systems. In DISCOS, to solve a closed loop system, at least one flexible body with at least six vibrational normal modes must be employed to satisfy loop closure constraint equations. Since CONTOPS uses relative coordinates, the user must specify cut joints to generate a spanning tree system. However, only spherical joints can be cut. Thus, DISCOS and CONTOPS are somewhat limited in treating closed loop mechanical systems.

DADS provides six standard joints (bracket, revolute, universal, spherical, cylindrical, and translational joints) and several non-standard joints (revolute-revolute, revolute-translational joints etc). Most of joints represent physical objects encountered in mechanisms and machines. They are treated as passive joints that transmit motion and force from one body to the adjacent body, thus active controllers are attached only to revolute joints. In DISCOS and CONTOPS, general joints that can have from 0 to 6 relative degrees of freedom are used and any generalized coordinate associated these joints can be actively controlled.

DADS provides a library of force elements such as springs, dampers, actuators, and a user defined force element. In addition to these basic force elements, there are vehicle oriented force elements such as tire force, leaf spring, and bushing elements. A rotational spring, damper, and actuator element is applicable in any revolute or cylindrical joint. A translational spring, damper, and actuator can be defined between pairs of bodies. DISCOS offers several different kinds of user subroutines to compute force in a system. Springs and dampers can be attached in general joints along any joint coordinate. The translational spring and damper can only be attached to a translational joint. CONTOPS also provides spring and damper elements for joints. Translational springs and dampers are also available between pair of bodies.

Gyrostats (momentum wheels) are often used for attitude control of satellites. Thus, DISCOS provides gyrostats that can be attached to any body without introducing extra-bodies. However, in DADS and CONTOPS, extra-bodies must be introduced with a revolute joint and a driver to make an equivalent model.

CONTOPS provides a library of sensor elements that are related to spacecraft dynamics, such as sun and star sensors. Modeling capabilities are summarized in Table 1.

Table 1. Modeling Capabilities of DADS, DISCOS, and CONTOPS for Dynamic Simulation

	DADS	DISCOS	CONTOPS
Type of Motion	Spatial/Planar	Spatial	Spatial
Body Type	Rigid/Flexible	Rigid/Flexible	Rigid/Flexible
Gyrostat	No	Yes	No
Joint Type	Library	General	General
Topology	Open/Closed	Open/Closed(flex)	Open/Closed(limited)
Initial Assembly	Yes	No	No
Force Element	Library	RSD1, TSD2(Joint), user	RSD1, TSD2(s)
Gravity	Yes	Yes (user)	Yes
Driver	Library	Joint	Library
Sensor	Point of interest	Sensor	Library
Curve Element	Yes	No (user)	Yes
Reaction Force	Yes	Yes	Yes

1. Rotational-spring-damper. 2. Translational-spring-damper.

The DADS program can perform kinematic, dynamic, and inverse dynamic analyses, whereas DISCOS and CONTOPS can only treat dynamic analysis. Since DADS and DISCOS use Cartesian coordinates in forming the equations of motion, sparse matrix solvers for linear equations are used to obtain accelerations. In contrast, CONTOPS uses relative joint or abstract coordinates to form state space equations of motion (the number of equations of motion is the same as the number of degrees of freedom), and a full matrix solver is used for solving linear equations.

Analysis capabilities and formulation methods for these three dynamic simulation codes are summarized in Table 2.

Table 2. Analysis Capabilities of DADS, DISCOS, and CONTOPS

	DADS	DISCOS	CONTOPS
Analysis Mode	K[1]/D[2]/ID[3]	D[2]	D[2]
Linearization	Yes	Yes	Yes
Formulation	Virtual Work	Lagrange Eq.	Kane's Eq.
Coordinate Systems	Cartesian	Cartesian	abstract/joint
Identification Igc[4], dgc[5]	Yes	No	No
Linear Equation Solver	Sparse Matrix	Sparse Matrix	Full Matrix
Constraint Force	Yes	Yes	Yes
Speed-Up Options	No	No	Yes
Integrator	A(v)[6]	RK(f)[7]	RK(f)[7],A(v)[6],U[8]
CPU Time Report	Yes(binary output)	Yes	No
Restarting Options	No	No	Yes

1. Kinematic analysis
2. Dynamic analysis
3. Inverse dynamic analysis
4. Independent generalized coordinates
5. Dependent generalized coordinates
6. Adams Bashforth and Adams Moulton variable order and variable step method
7. Runge Kutta fourth order constant step method
8. User provide integration method

3.2 Ease of Use and Code Automation

DADS, DISCOS, and CONTOPS have alpha-numeric interactive pre- and post-processing capabilities. The post-processor provides basically x-y plots. An interactive pre-processor helps the user to define necessary input data. However, it is difficult for the user to find mistakes in input data for complicated spatial mechanical systems. Some graphics oriented user interfaces are provided by DADS for post-analysis and animation.

Flexible body dynamic analysis can be carried out with each of these three programs. Data associated with flexible body components can be obtained from finite element analysis. DADS provides interfaces with NASTRAN and ANSYS. DISCOS and CONTOPS can be integrated with NASTRAN.

DADS uses a variable order-variable step integration method. Step size is automatically selected by the program, according to the system characteristic of the equations of motion. Thus, the user does not need to choose step size. DISCOS uses a constant step Runge Kutta fourth order method. Thus, the user must have an idea of how small a step size is required for a certain mechanical system simulation. CONTOPS can have three different integration methods, such as a constant step Runge Kutta fourth order method, an Adams family variable step method, and a user defined integration method.

Imposing initial conditions on a closed loop system is challenging, since generalized coordinates and velocities in closed loop systems are not independent. A kinematically admissible initial state of the mechanical system must be imposed. DADS provides initial assembly and initial velocity computation routines, so that from user's initial estimate of the configuration and definition of initial conditions, a mechanical system is assembled to satisfy all kinematic relations. However, DISCOS and CONTOPS require the user to provide kinematically consistent initial conditions, which can be difficult for complicated closed loop systems.

In order to use current multibody simulation codes, a dynamics work station [11] is being developed to automate dynamic simulation modeling and post-processing by integrating a graphics oriented modeler, an initial assembly program, finite element codes, a graphics oriented post-processor, and an animator.

3.3 Input and Output

In order to systematically compare the simulation data from different simulation software with experimental data from a validation library, it is important to study input and output data definition for each program, to identify standard data for the verification library. Input data for each code are dictated by the formulation used. Since Cartesian coordinates are used in DADS and DISCOS, and interrelationships between pair of bodies due to joints are treated as constraints, there is no concept of inboard, outboard, base bodies, and cut joints. However, for CONTOPS, these are necessary data for its relative joint coordinate approach.

The flexible body formulation in each code studied is based on lumped mass and modal coordinate approaches. Thus, most data required to define flexible bodies are the same. However, CONTOPS does not need a lumped mass matrix. Instead, it requires the user to provide a so called h-parameter array [9], which is function of nodal masses and mode shapes. Such parameters are internally computed with given nodal masses and mode shapes in DADS and DISCOS.

The items required to describe a body and a joint are essentially the same for each code. However, the way a conceptual item is defined is quite different. For example, in order to define a joint triad, DADS and CONTOPS require the user to specify two unit vectors of the joint triad with respect to the body reference frame, whereas DISCOS requires Euler angles for the triad.

An input data comparison is summarized in Table 3.

Table 3. Input Data Comparison among DADS, DISCOS, and CONTOPS.

Body data

	DADS	DISCOS	CONTOPS
Initial Position & Orientation	Yes	No	No
Inertia properties	Centroidal/Body frame	Body frame	arbitrary
Nodal Coord.	Yes	Yes	Yes
Nodal Mass	Yes	Yes	No
Nodal Inertia	No	Yes	No
Modal Stiffness	Yes	Yes	Yes
Modal Damping	No	Yes	Yes
Modal Mass	Yes	No	Yes
Mode Shapes	Eigen/Static Vector	Shape function	Shape function

Topology data

	DADS	DISCOS	CONTOPS
Base Body	No	Yes	Yes
Inboard/Outboard Body	No	No	Yes
Cut Joint	No	No	Yes

Joint data

	DADS	DISCOS	CONTOPS
Joint Type	Library	General	General
Joint position/Velocity	No	Yes	Yes
Joint Reference Frame	P, Q, R	X, Y, Z, Euler angles	Direction cosine vector

Initial condition data

DADS	Initial independent coordinates and velocities
DISCOS	Initial relative coordinates and velocities
CONTOPS	Initial relative coordinates and velocities

Output data that represent physical quantities are different for each code. DADS provides the position and orientation of the body with respect to an inertial reference frame. However, DISCOS reports body center of mass position with respect to the first body reference frame (a kind of reference body for the mechanical system). Translational velocity of a body is reported in an inertial reference frame in DADS, whereas it is reported in body reference frames in DISCOS. DISCOS also reports total linear and angular momentum of the system and each body's contribution to total kinetic and potential energy.

An output data comparison is presented in Table 4.

Table 4. Comparison of output data among DADS, DISCOS, and CONTOPS

	DADS	DISCOS	CONTOPS
Body Positions	Yes	Yes	No
Body Orientations	Yes	Yes	No
Body Velocities	Yes	Yes	No
Body Accelerations	Yes	Yes	No
Modal Coordinates	Yes	Yes	Yes
Modal Velocities	Yes	Yes	Yes
Modal Accelerations	Yes	Yes	No
Relative Displacements	No	Yes	Yes
Relative Velocities	No	Yes	Yes
Relative Accelerations	No	No	No
Constraint Forces	Yes	Yes	Yes
Sensor Frame Positions	Yes	Yes	Yes
Sensor Frame Velocities	Yes	Yes	Yes
Sensor Frame Accelerations	Yes	Yes	Yes
Position of System C.M.	No	Yes	No
Total Momentum	No	Yes	No
Total Energy	Yes	Yes	No

4. Verification Library System

4.1 Verification Procedure

The conceptual verification procedure is presented in Fig. 6. Through parameter estimation for the mechanical system, system parameters can be defined for the model. At the same time, experimental tests can be defined, identifying what kind of physical quantities can be measured through experimental test, according to the availability of measuring devices. From this test plan, simulation model initial conditions and simulation scenarios can be defined. Simulation input data for a particular simulation code can then be set up, with initial condition, simulation scenario, and mechanical system parameter such as geometric dimensions, and inertia properties.

Experimental tests can be performed and data for measurable physical quantities can be acquired, according to the test plan and availability of measuring devices. Experimental test data are then processed and investigated to determine whether they are meaningful.

Simulations can be carried out according to simulation scenarios defined. The simulation data associated with observable physical quantities are extracted from the simulation output data. Through x-y plots, simulation and experimental data can be compared. Engineers can then evaluate simulation results. If simulation results are quite different from experimental results, the engineer can refine the simulation model. With the refined model, the mechanical system can be re-analyzed. An evaluation report for a validated multibody simulation can then be provided.

The test plan, observable physical quantities, and processed test data, simulation input and output data, and the evaluation report are then stored for reference.

4.2 Verification Library System Functionality

In Subsection 4.1, a conceptual verification procedure is introduced. However, this verification procedure may involve tedious data preparation and manipulation effort. For example, If an engineer wants to validate his simulation, he can set up the simulation model by retrieving the test plan and simulation input data for previously validated simulation software. He must understand previous simulation input data, which may not be easy. After carrying out a simulation, results can be compared by retrieving test data and evaluation reports from the verification library. The engineer should provide simulation data that have compatible format with existing experimental data.

In order to alleviate these burdens, a verification library system is desired, which can automate following procedures; modeling, carrying out simulations, and storing and retrieving data for verification of the multibody simulation software. For systematic verification, several functionalities are being considered for the verification library. The first functionality of the verification library system is to store and retrieve the following data; test plan, observable physical quantities, processed test data, simulation input and output data, and the associated evaluation report. The second functionality is to model a mechanical system for different simulation programs. Using a graphics oriented mechanical system modeler, a neutral input that contains generic mechanical system data can be created and modified. Neutral input data can be translated into input data for DADS, DISCOS, CONTOPS, and other multibody simulation programs. The third functionality is to launch simulation software to obtain simulation results. An interface program is required to integrate the verification library system and simulation software. The fourth functionality is to display simulation and experimental results together, using computer graphics, to help in the evaluation procedure. The final functionality is to create and edit evaluation reports.

To achieve these functionalities (engineering capabilities), software integration [12] is required. The verification library system being designed will integrate a dynamics workstation, x-y plots, visualization software [13], and the simulation codes DADS, DISCOS, and CONTOPS with a database management system. A schematic of the verification library system is presented in Fig. 7.

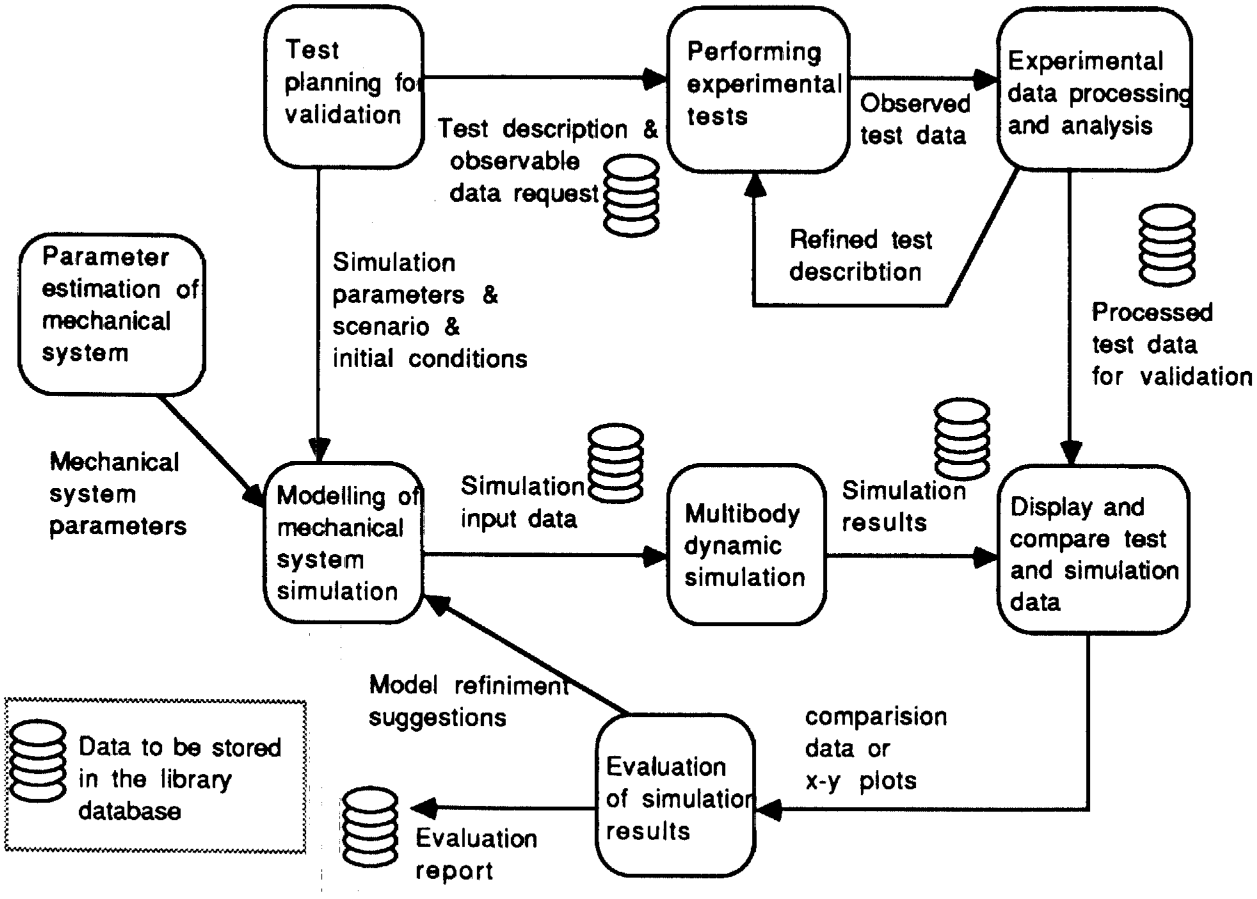

Test planning for validation
Performing experimental tests
Experimental data processing and analysis
Test description & observable data request
Observed test data
Refined test describtion
Parameter estimation of mechanical system
Simulation parameters & scenario & initial conditions
Processed test data for validation
Mechanical system parameters
Modelling of mechanical system simulation
Simulation input data
Multibody dynamic simulation
Simulation results
Display and compare test and simulation data
Model refiniment suggestions
comparision data or x-y plots
Evaluation of simulation results
Evaluation report
Data to be stored in the library database

Figure 6. Validation Procedure Data Flow

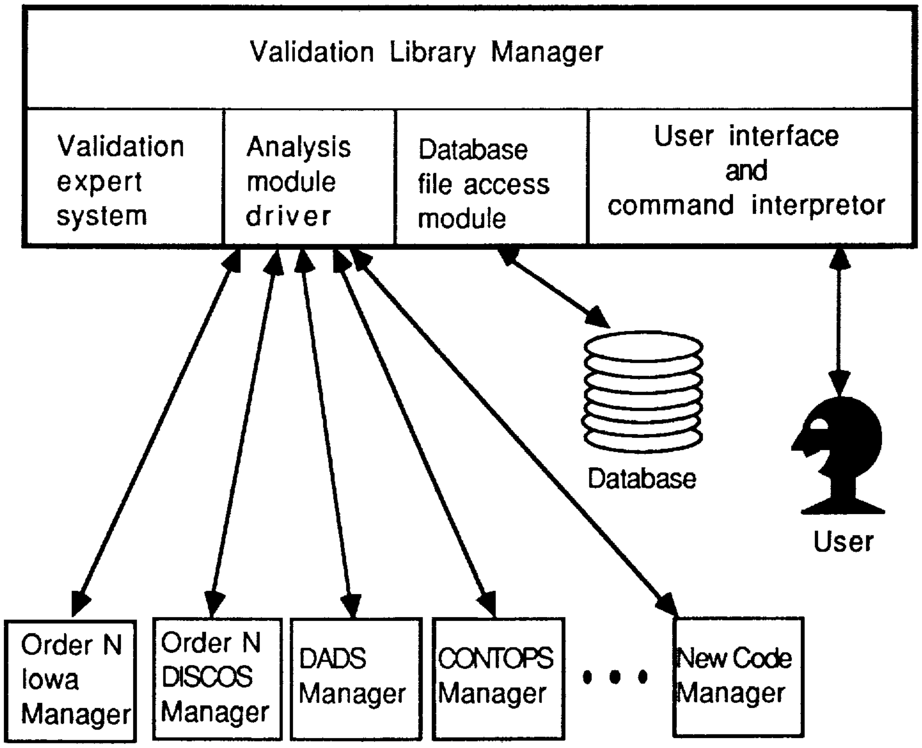


Figure 7. Verification Library Software Architecture

5. Conclusions

The verification library system concept and engineering requirements have been introduced based on experience gained in RRC arm and CASE backhoe validations and theoretical cross verifications of DADS, DISCOS, and CONTOPS. A systematic software integration technique will be utilized to achieve an integrated capability to help potential users to validate their multibody simulation software.

References

1. Yang, F.C., Ryu, J., Jeong, K., and Kim, S.S., "System Performance Specification of the New General Purpose Dynamic Simulation Code," in preparation, The Center for Simulation and Design Optimization, The University of Iowa, 1989.

2. Yang, F.C., Ryu, J., Jeong, K., Choong, F.N., and Kim, S.S., "Software Requirement Specification of the New General Purpose Dynamic Simulation Code," in preparation, The Center for Simulation and Design Optimization, The University of Iowa, 1989.

3. Chuang L.P. and Kim, S.S., "Experimental Verification of Telerobot Arm Simulation," in preparation, The Center for Simulation and Design Optimization, The University of Iowa, 1989.

4. Yae, K.H., Hwang, H., and Chern, S., "Experimental Verification of Dynamics Simulation," to appear Proceedings of 3rd Annual Conference on Aerospace Computational Control, August 28-30, 1989.

5. Lai, H.J., Kim, S.S., and Haug, E. J., "Double Pendulum Test Example," Multibody Simulation Verification Library Working Paper 2, The Center for Simulation and Design Optimization, The University of Iowa, December, 1988.

6. Lai, H.J., Kim, S.S., and Haug, E. J., "Four Bar Linkage Test Example," Multibody Simulation Verification Library Working Paper 4, The Center for Simulation and Design Optimization, The University of Iowa, January, 1989.

7. Lai, H.J., Kim, S.S., and Haug, E. J., "Flexible Rotating Beam Test Example," Multibody Simulation Verification Library Working Paper 5, The Center for Simulation and Design Optimization, The University of Iowa, February, 1989.

8. Hwang, H., Kim, S.S., and Haug, E. J., "Flexible Four Bar Linkage Test Example," Multibody Simulation Verification Library Working Paper 6, The Center for Simulation and Design Optimization, The University of Iowa, July, 1989.

9. User's Manual for CONTOPS, Honeywell Space & Strategic Avionics Division, Clearwater, Florida.

10. Kim, S.S., Hwang, H., and Haug, E. J., "A Comparative Analysis of DADS, DISCOS, and CONTOPS, for Flexible Multibody Dynamics," Multibody Simulation Verification Library Working Paper 7, The Center for Simulation and Design Optimization, The University of Iowa, August, 1989.

11. Wu, J.K., Fogle, M.A., Wang, J.Y., and Lu, J.K., "A Dynamic Work Station," presented in The 1989 ASME Design Technical Conferences-15th Design Automation Conference, Montreal, Quebec Canada, September 17-21, 1989.

12. Dopker, B., Murray, P., and Choong, F.N., "An Object Oriented Data Base and Application Management System for Integrated, Interdisciplinary Mechanical System Simulation," presented in The 1989 ASME Design Technical Conferences-15th Design Automation Conference, Montreal, Quebec Canada, September 17-21, 1989.

13. Visualization of Dynamic System User Documentation, Version 1.0, The Center for Simulation and Design Optimization, October, 1988.

14. DADS User's Manual, Rev. 5.0, Computer Aided Design Software, Inc. Oakdale, Iowa, 1988.

15. Bodley, C., Devers, A., Park, A., and Frisch, H., "A Digital Computer Program for Dynamic Interaction Simulation of Controls and Structures (DISCOS), NASA Technical Paper 1219, 1978.

INDEX

INDEX