545438
44P
N90-10096

EXPLICIT MODELING AND COMPUTATIONAL LOAD DISTRIBUTION FOR CONCURRENT PROCESSING SIMULATION OF THE SPACE STATION

By

R. Gluck
TRW Space and Technology Group
Redondo Beach, California

ABSTRACT

Analytical simulation of the dynamics/control interaction of large space structures is essential during the design process as full-scale tests of these vehicles in the laboratory are impractical. Furthermore, the operational manifests of large space systems on-orbit may call for significant changes in their mass and stiffness distributions as well as for substantial growth during the vehicles' lifetimes, and these can be studied only by analytical simulation.

Current methodologies for simulating large space structures involve implicit mathematical models and solutions on serial digital computers. These methodologies require unacceptably long computer processing time and exorbitant costs as the models become larger and more complex. Potential orders-of-magnitude reductions in simulation time and cost of multibody dynamic systems can be attained using: (1) enhanced analytical models for simulation, and (2) special-purpose, concurrent computational hardware and system software.

This paper deals with two important aspects of concurrent processing under development at TRW. These are: (1) the derivation of explicit mathematical models of multibody dynamic systems, and (2) a balanced computational load distribution (BCLD) among loosely coupled computational units (processors) of a concurrent processing system. The developed methodologies will be demonstrated in the paper by way of an application to the Phase 1 of the Space Station - a task being performed by TRW under NASA/JSC contract NAS9-17778.

The mathematical model of the Space Station consists of three interconnected flexible bodies capable of undergoing large, rigid-body motion with respect to each other. Body 1 is the main central body and contains the pressurized modules inboard of the two Alpha gimbals. Bodies 2 and 3 are the starboard and port bodies connected to Body 1 at the Alpha gimbals and include all components on the transverse booms outboard of the Alpha gimbals (including the solar arrays). The control systems in the model maintain Body 1 in a prescribed 3-axis attitude control mode, while producing large-angle rotations of the flexible solar arrays to position them normal to the sun-line.

PRECEDING PAGE BLANK NOT FILMED

PAGE 370 INTENTIONALLY BLANK

The BCLD methodology for concurrent processing developed by TRW enploys a static allocation strategy in which a separate software package is used off-line and at leisure prior to the execution of the simulation program. The load distribution, in this methodology, is carried out in a manner transparent to the user who, nevertheless, exercises control over the procedure with pre-selected constraint conditions.

The distributed model of the Space Station is now complete and ready to undergo benchmark tests on TRW's Custom Architectured Parallel System during the June-July 1988 timeframe.

Engineering & Test Division
TRW Space & Technology Group

NASA/OAST Workshop on Computational Aspects
in the Control of Flexible Systems

Explicit Modeling and Computational Load Distribution for Concurrent Processing Simulation of the Space Station

373

Dr. R. Gluck

July 12-14, 1988
Williamsburg, Virginia

Introduction

Engineering & Test Division
TRW Space & Technology Group

This paper presents the application of concurrent processing technology developed at TRW Space & Technology Group over the past several years to the simulation of the Space Station. The effort is funded by NASA Johnson Space Center under Contract NAS9-17778 and monitored by Mr. John W. Sunkel. The period of performance extends from April 1987 to November 1988

374

OBJECTIVE

This project was established to provide NASA with quantitative data to determine the cost effectiveness of utilizing a specialized processing system such as the Custom Architectured Parallel Processing System (CAPPS) for development and verification of the operational Space Station flight control system. The CAPPS is a concurrent processor consisting of loosely coupled, high speed array processors [computational units (CUs)] - each containing its own input/output capability and memory banks. The specially designed CUs are capable of concurrent computation and communication, thereby placing a very low overhead on the latter operation. Furthermore, the system's architecture provides for direct communication between each CU and any other CU, facilitating considerable flexibility in adapting the CAPPS architecture to a specific simulation problem.

Objective

Engineering & Test Division
TRW Space & Technology Group

The objective of this project is to develop, verify and demonstrate the simulation of an explicit model of the controls/structure interaction of the Space Station on CAPPS

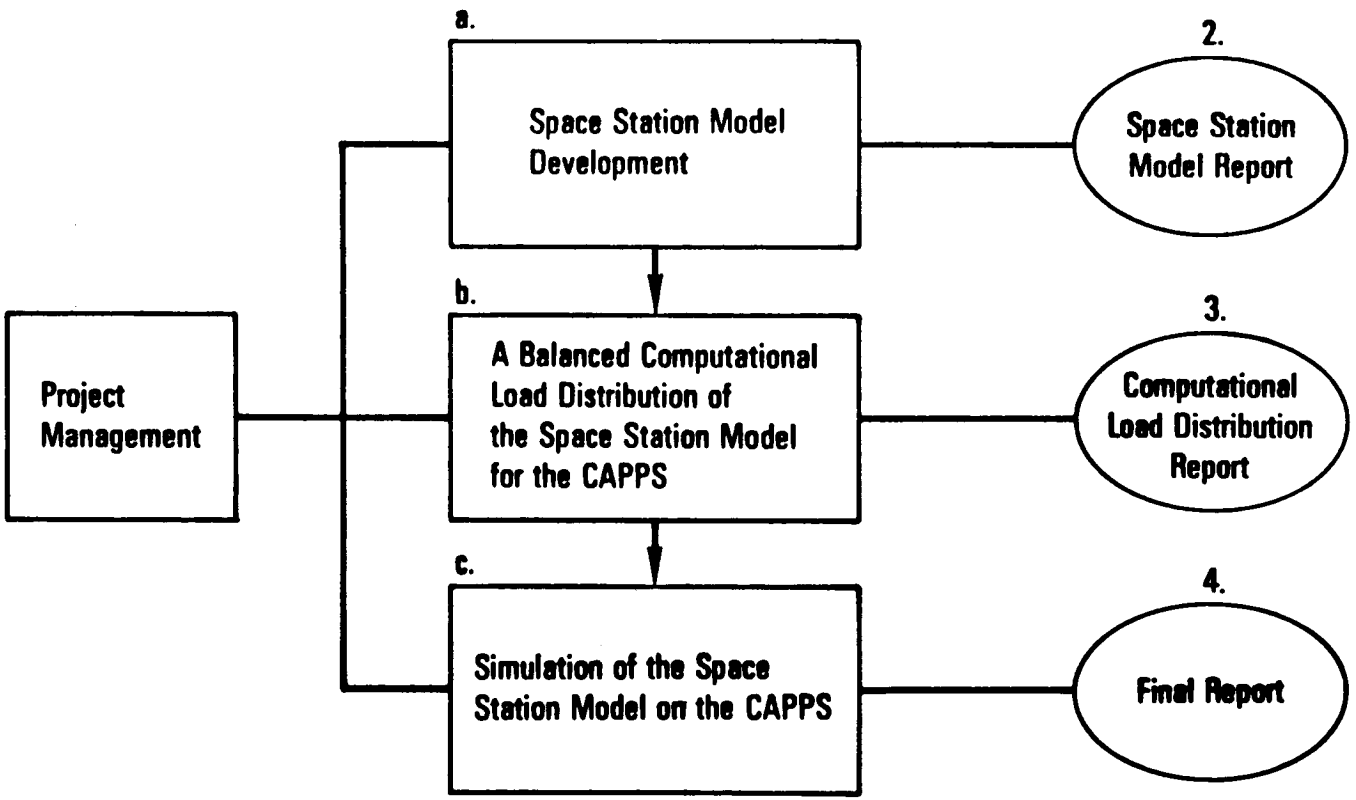
376

PROJECT WORK BREAKDOWN STRUCTURE

Applying previously developed application and system software at TRW to the concurrent processing of an explicit model of the control/structure interaction of the Space Station on CAPPS, the project described herein consists of three sequential tasks as stated in the figure. This paper focuses on the completed Tasks (a) and (b). Task (c), at the time of this writing, is in progress.

Project Work Breakdown Structure

Engineering & Test Division
TRW Space & Technology Group



378

The Space Station Model Development

Engineering & Test Division
TRW Space & Technology Group

379

- Methodology
- Model's Contents

ADVANTAGES OF EXPLICIT EQUATIONS OF MOTION

The derivation of the equations of motion by symbol manipulation has several important attributes compared to the conventional (implicit) methodology. Symbol manipulation, i.e., the nonnumerical computation with a digital computer, yields scalar equations of motion specifically tailored to complex dynamical systems, where the analyst has the freedom and insight to incorporate any required fidelity in the model. Furthermore, the output of symbol manipulation is a completely portable FORTRAN code in the format of $A(x,t)\dot{x} = b(x,t)$, which can be delivered via file to either serial or concurrent processors without requiring any programming. This reduces development cost by at least one order-of-magnitude or more compared to that of a special-purpose implicit code. Finally, symbolically derived scalar equations of motion require a substantially reduced simulation time compared to those of conventional codes. Benchmark tests conducted at TRW yielded improvements in run times by factors of approximately 4 and 3 for rigid-body and flexible-body systems, respectively.

Advantages of Explicit Equations of Motion

Engineering & Test Division
TRW Space & Technology Group

- Useful engineering insights into the dynamic characteristics of the system
- No major programming effort required to perform simulation
- Large reduction of time required for simulation as compared to that required for implicit formulation
  - — Implicit formulation requires the derivation of the equations of motion to be performed numerically at each integration step
  - — Explicit formulation requires the derivation to be performed only once

381

ADVANTAGES OF USING PROGRAM SMP TO DERIVE EXPLICIT EQUATIONS OF MOTION

Program SMP was selected for the TRW symbol manipulation methodology following a thorough analysis which proved it superior in both versatility and speed to other available symbol manipulation codes such as MACSYMA, Reduce and FORMAC. The SMP program is implemented in the C language and is available on a variety of mainframes and work-stations. Its capability of handling very large amounts of data is ideally suited for the derivation of explicit equations of motion of multibody spacecraft. The program's other attributes are listed in the accompanying figure.

Advantages of Using SMP to Derive Explicit Equations of Motion

Engineering & Test Division
TRW Space & Technology Group

- Relieves the drudgery and distasteful tasks of manual algebraic manipulation
- Reduces cost and time by orders-of-magnitude as compared to manual derivation
- Allows the analysts to fully participate in the derivation process to achieve the most efficient mathematical model
- Leads to equations with no wasteful operations, such as additions of zeros, multiplications by unity, and dot product of orthogonal vectors

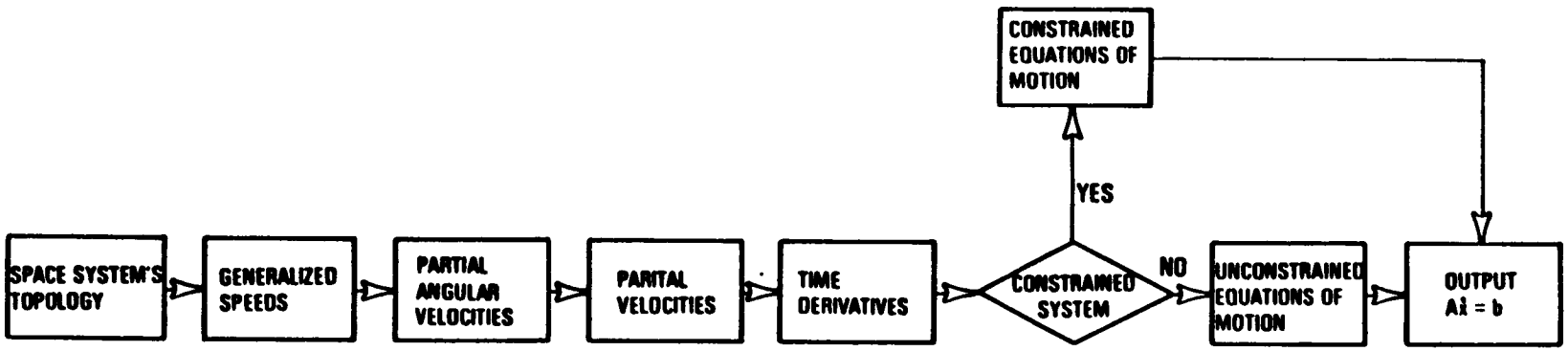
PROGRAM SYMBOD

Program SYMBOD for the derivation of the explicit equations of motion of multibody flexible dynamical systems was developed within the general framework of SMP. A menu is provided to the analyst for introducing the system's topology and appropriate logic is available for the definition/computation of the essential elements of Kane's Dynamical Equations as shown in the figure; however, the analyst can, at his discretion, override the logic imbedded in each of the program's modules. The procedure is considered to be optimal in the sense that it leaves with the analyst the tasks he is best trained to perform, while transferring to the computer the manually prohibitive algebraic manipulation and long derivation operations.

Two methodologies were incorporated in Program SYMBOD for dealing with the presence of m geometrical and velocity constraints in a multibody dynamic system. In the first methodology (by Wampler et al, Reference 1), the dynamic equations governing a constrained system are generated symbolically directly from expressions comprising the equations governing the system without constraints. This methodology for constraint elimination (which requires a symbolic inversion of a matrix of order m) is applied when the number of constraint equations in the system is small ($m<6$) and no working constraints are involved; otherwise, the Lagrange multiplier methodology is used, where the stabilized penalty procedure of Reference 2 offers an attractive way for stabilizing the constraint equations now retained in the mathematical model.

Top Level Flow Diagram for the Explicit Formulation of Kane's Dynamical Equations with the SMP Program (Program SYMBOD)

Engineering & Test Division
TRW Space & Technology Group



ATTRIBUTES OF PROGRAM SYMBOD

Program SYMBOD contains several innovations which combine to produce an efficient mathematical model. These are listed in the figure and explained briefly below.

The generation of equations of motion by symbol manipulation requires a systematic method of reducing the number of algebraic operations in the formation of Kane's equations. Frequently, the intermediate computations of expressions, such as the velocity terms, produce a multitude of expressions so large that their storage requirements exceed the computer's capacity. A procedure for systematically introducing new intermediate symbols to replace recurring combinations of algebraic subexpressions was developed. This procedure eliminates repetitious calculations and results in efficient computational algorithms requiring fewer arithmetic operations.

The formulation of Kane's dynamical equations associated with the flexible-body degrees-of-freedom (dof) of a body are iterative in the number of assumed admissible functions required to represent the flexibility. The totality of the flexible-body dof for each body was, therefore, represented in Program SYMBOD by a single dof of that body. This allows postponement of the final selection of the required number of assumed admissible functions until after the development of the explicit mathematical model (including the control system) is completed, i.e., the assumed admissible functions in this formulation need not be selected prematurely.

Program SYMBOD provides for direct elimination (by command) of superfluous higher order terms in the explicit equations of motion when these terms are inconsistent with basic assumptions made in the formulation or with the variance of input parameters.

Attributes of Program SYMBOD

Engineering & Test Division
TRW Space & Technology Group

- Systematic introduction of intermediate variables for algebraic subexpressions
  - — Eliminates repetitious calculations
  - — Reduces computer memory requirements
  - — Produces efficient computational equations involving fewer arithmetic operations
- Totality of flexible-body degrees-of-freedom (dof) of each flexible body is represented by a single dof of that body. Selection of desired flexible functions in simulation is performed **after** control system is specified
- Elimination of superfluous higher-order terms in equations
- Error-free translation of explicit equations of motion into FORTRAN
  - — Eliminates manual mistyping of equations
  - — Eliminates debugging of code

Engineering & Test Division
TRW Space & Technology Group

The Space Station Model Contents

(A joint effort of NASA & TRW)

388

Space Station Model Simulation Objective

Engineering & Test Division
TRW Space & Technology Group

Simulate a transient maneuver involving a large-angle rigid-body motion of the flexible solar arrays connected to their respective transverse booms, while the central body is maintained in a three-axis attitude control mode

386

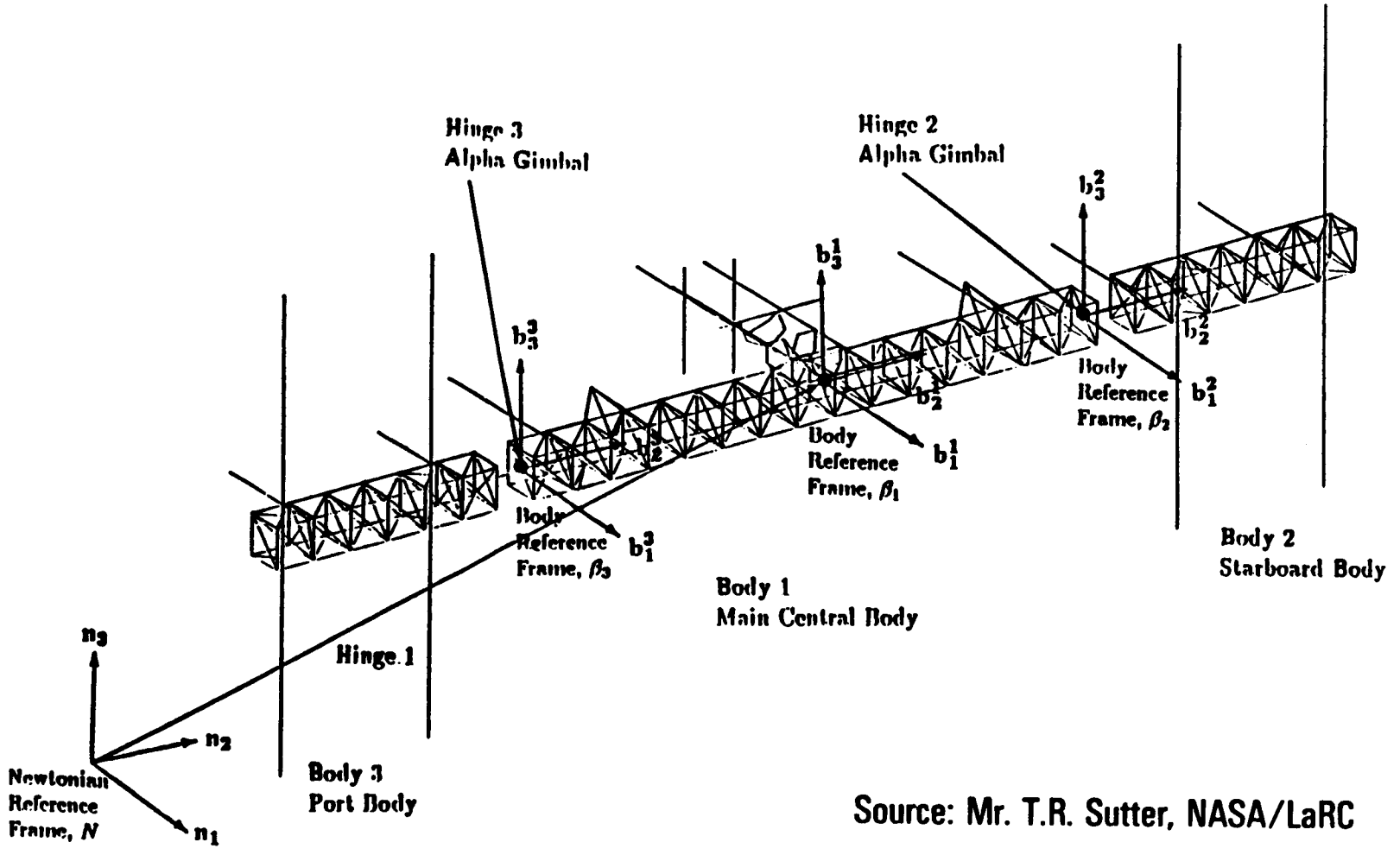
SPACE STATION MODEL

The mathematical model of the Space Station consists of three interconnected flexible bodies capable of undergoing large, rigid-body motion with respect to each other. Body 1 is the main central body and contains the pressurized modules inboard of the two Alpha gimbals. Bodies 2 and 3 are the starboard and port bodies connected to Body 1 at the Alpha gimbals, and include all components on the transverse booms outboard of the Alpha gimbals (including the solar arrays).

The three-body Space Station model contains eight (8) large-motion, rigid-body degrees of freedom, 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 degrees of freedom are facilitated in the model. The flexibility of Body 1 is described by 1 to 45 "free-free" natural modes, excluding the six rigid-body modes, used here as assumed admissible functions. The flexibilities of Bodies 2 and 3 are each described by 1 to 45 "fixed-free" natural modes serving also as assumed admissible functions. The assumed spatial admissible functions are general 3-dimensional functions which satisfy the boundary conditions of the body in question but do not provide solutions to its differential equations of motion.

Space Station Model

Engineering & Test Division
TRW Space & Technology Group



Source: Mr. T.R. Sutter, NASA/LaRC

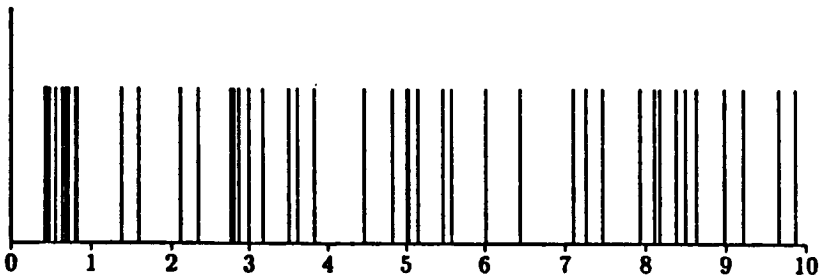
391

FREQUENCY SPECTRA OF THE SPACE STATION MODEL

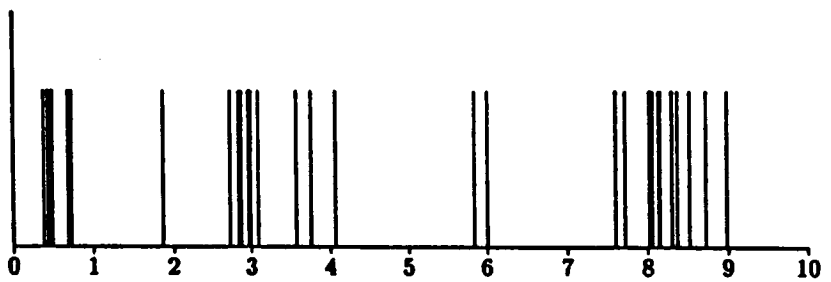
The assumed admissible functions in the Space Station model were obtained from finite element models developed for each of the bodies. These included an unconstrained (free-free) model for the central body and two constrained (fixed-free) models for the starboard and port bodies, cantilevered at the Alpha gimbals. A MSC/NASTRAN code was used to obtain the natural modes of vibration within a 10 Hz frequency bandwidth. The spectrum of natural frequencies for each of the three finite element models is shown in the figure. Note that these are characterized by a number of low frequency modes (below 1 Hz) spaced closely together.

Frequency Spectra of the Space Station Model

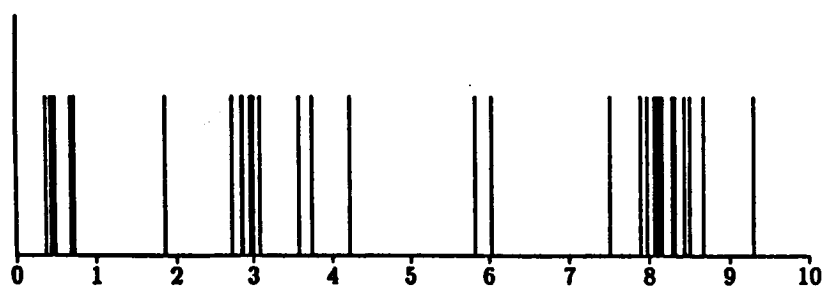
Engineering & Test Division
TRW Space & Technology Group



Central Body Frequencies, Hertz



Starboard Body Frequencies, Hertz



Port Body Frequencies, Hertz

393

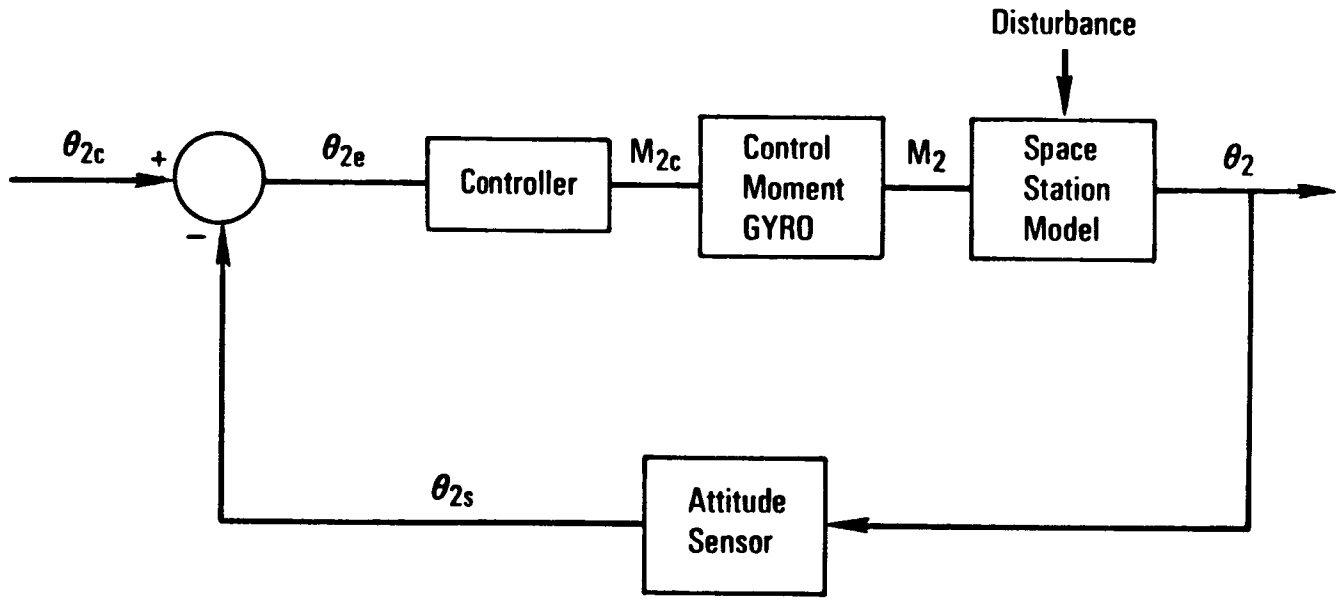
ATTITUDE CONTROL SYSTEM FOR SPACE STATION MODEL

The attitude control system of the Space Station was designed to regulate its orientation and keep its longitudinal axis aligned with the local vertical vector while maintaining its plane perpendicular to the velocity vector. The control system consists of attitude sensing instrumentation, control moment gyros, and electronics to cause corrective control moments to be applied to the Space Station's central body whenever it moves away from the commanded attitude. The attitude and rate sensors and the control moment gyros (CMG's) are co-located at the origin of the coordinate system of the central body placed at its undeformed center of mass.

A block diagram of the control law for the X_2 axis is shown in the figure. Similar control laws were designed for the remaining two axes (the three control laws are uncoupled from each other). Attitude sensing instrumentation provides electronic representations of the attitude angle θ_2 and its time rate of change. The sensed attitude angle is subtracted from the commanded attitude angle (θ_{2c}) to form the attitude error signal (θ_{2e}). The electronic controller mechanizes a control law, specified in the form of a transfer function, to produce a commanded control moment (M_{2c}) based on the error signal. The CMG generates control moments (M_2) according to the commanded moments to drive the attitude error towards zero. External disturbances are not considered in this simulation and the commanded attitude is set nominally to zero.

Attitude Control System for Space Station Model - X_2 Axis

Engineering & Test Division
TRW Space & Technology Group



395

θ_2 — Central-Body Attitude

θ_{2s} — Senced Attitude

θ_{2c} — Commanded Attitude

θ_{2e} — Attitude Error

M_2 — Control Moment

M_{2c} — Commanded Contrtol Moment

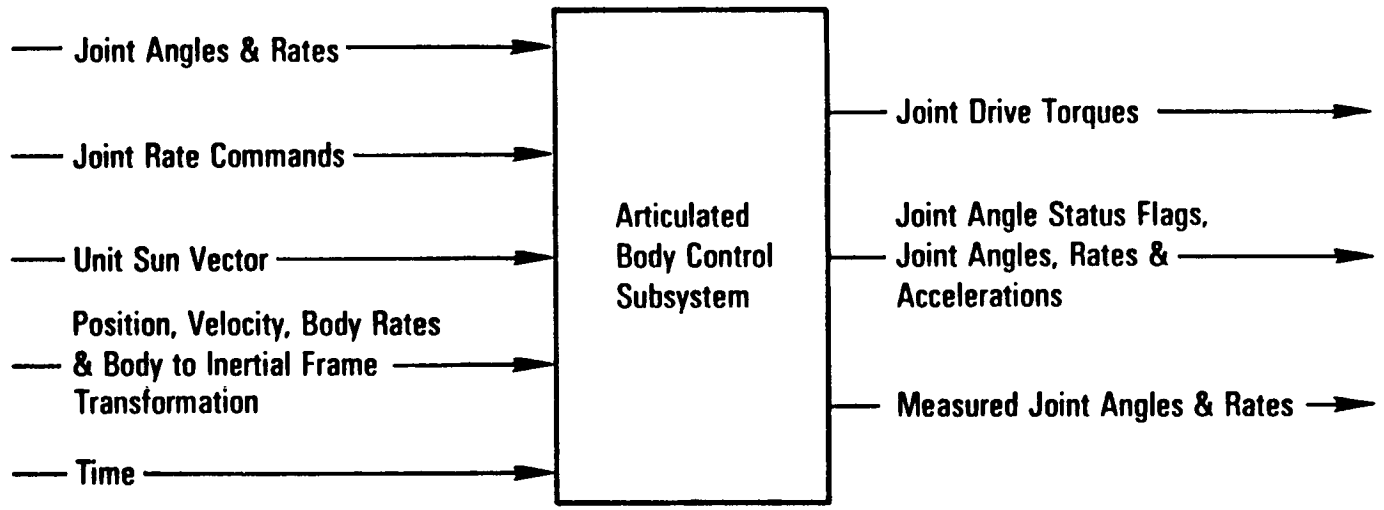
Source: Mr. J.W. Young, NASA/LaRC

ARTICULATED-BODY CONTROL FOR SPACE STATION MODEL

In addition to the attitude control system, the Space Station model includes a second control system to maintain the solar arrays pointing in a direction perpendicular to the sun line. The control law is based on an angular position and rate feedback scheme with options provided to rewind the solar arrays during eclipse. The commanded angular position of the Alpha gimbal is utilized in a second order transfer function to calculate the controller's motor torque. Input and output parameters for the articulated body control system are shown in the figure.

Articulated-Body Control for Space Station Model

Engineering & Test Division
TRW Space & Technology Group



Source: Mr. J. Mapor, LEMSCO; Houston, Texas

397

Engineering & Test Division
TRW Space & Technology Group

A Balanced Computational Load Distribution Methodology

3988

MATHEMATICAL CHARACTERISTICS OF THE SPACE STATION MODEL

The balanced computational load distribution methodology described herein is aimed at a broad class of multibody dynamic systems, which includes every variety of spacecraft, robot, rotary aircraft and mechanism. This class is characterized by a set of first-order, ordinary differential equations, known as Kane's Dynamical Equations, as depicted in the figure.

The methodology for computational load allocation adopted here takes advantage of the fact that the mathematical model involved, although generally very complex, remains essentially unchanged for many hundreds (if not thousands) of simulation runs made in the course of the development and verification of the dynamic system in question. For these simulation runs, which feature different combinations of initial conditions, input functions and parameter values, it is possible to distribute the computational load statically, off-line, and thereby gain a significant advantage in execution speed during simulation compared to that achievable with a dynamic load allocation methodology.

MATHEMATICAL CHARACTERISTICS OF THE SPACE STATION MODEL

Engineering & Test Division
TRW Space & Technology Group

- KANE'S DYNAMICAL EQUATIONS CONSTITUTE A SET OF FIRST-ORDER ORDINARY DIFFERENTIAL EQUATIONS IN THE MATRIX FORM

$$\mathbf{A}(\mathbf{q},t)\,\dot{\mathbf{u}} = \mathbf{b}(\mathbf{q},\mathbf{u},t)$$
$$\dot{\mathbf{q}} = \mathbf{f}(\mathbf{q},\mathbf{u},t)$$

WHERE **u** IS THE VECTOR OF GENERALIZED SPEEDS, **q** IS THE VECTOR OF GENERALIZED COORDINATES, t IS THE TIME, AND A DOT INDICATES TIME DIFFERENTIATION

- THE FORM OF THE EQUATIONS IS QUITE GENERAL, AND IS THE FORM ADDRESSED BY THE LOAD BALANCING METHODOLOGY

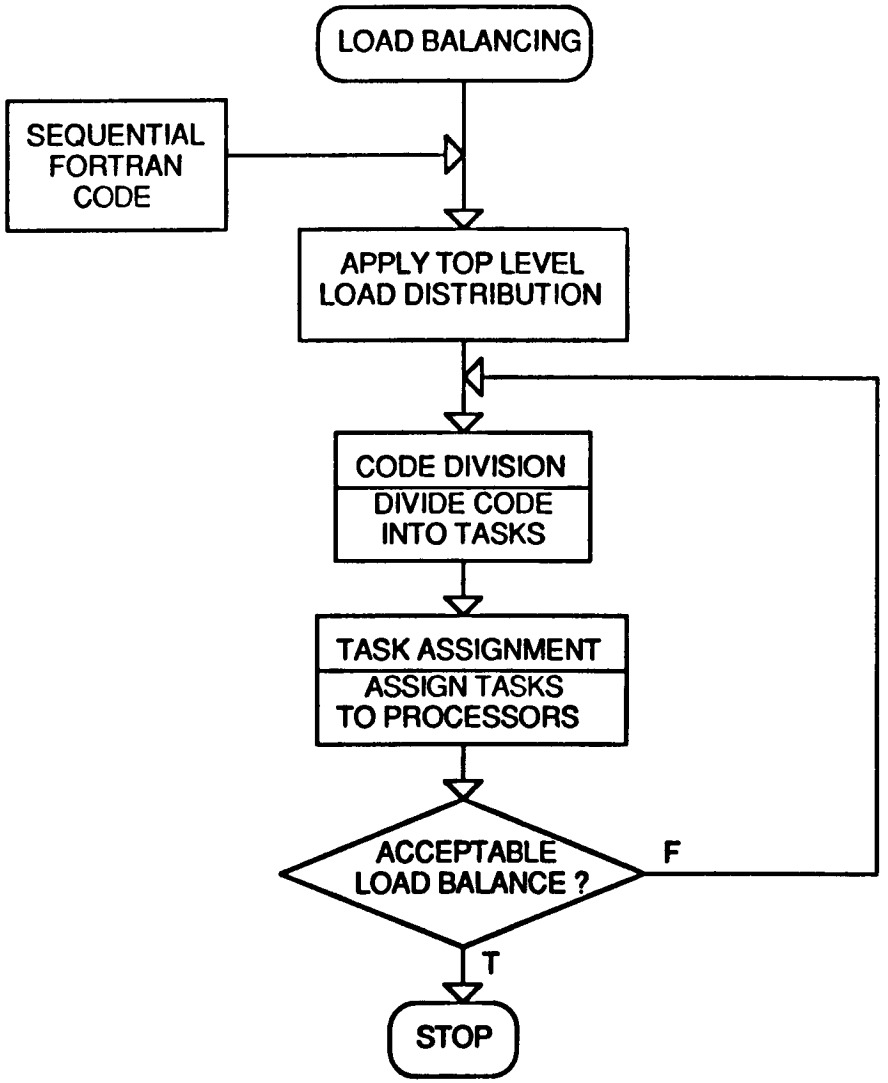
400

LOAD BALANCING DIAGRAMS

The basic input to the load balancing software is the sequential FORTRAN code developed using Kane's Dynamical Equations and the symbol manipulation program SYMBOD/SMP.

The code is inspected by the user to determine large scale operations that may be done in parallel. This provides the software with a top level load distribution that it may then refine and balance. Mathematical models of the sort considered here willl have certain computational features that are ideally suited to parallel execution, and these may be used to provide a preliminary code division into tasks for each processor.

LOAD BALANCING DIAGRAMS



Engineering & Test Division
TRW Space & Technology Group

402

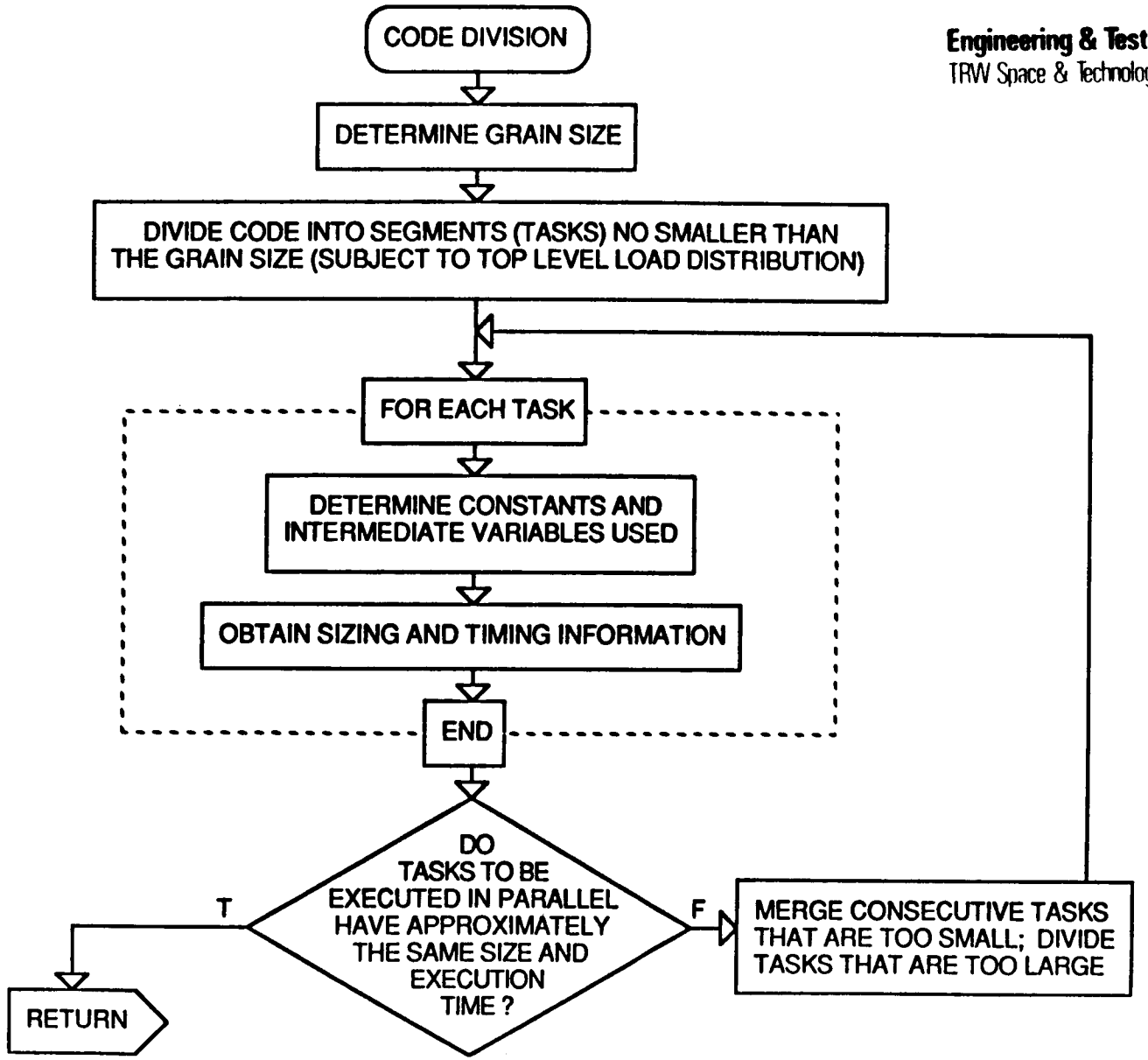
LOAD BALANCING DIAGRAMS (CONT'D)

The code is then divided into smaller tasks using the program TASK. This partitioning incorporates the divisions already given in the top level distribution, and results in tasks which are no smaller in size than some predetermined "grain" size. The user chooses whether or not the code division is to be fine-grained or coarse-grained. Different choices will be a result of, e.g., a different number of available processors, the need to examine values for intermediate variables in a convenient way, etc.

Once tasks are obtained, TASK checks to see that the size and execution time for tasks that are to be executed in parallel are approximately the same (according to criteria determined, in part, by the user), in order to have a balanced computational load. Those tasks found to be too large are divided while those that are too small are merged with tasks to be executed before or afterwards. This division and merging continues until the criteria mentioned above are satisfied.

LOAD BALANCING DIAGRAMS (CONTINUED)

TRW
Engineering & Test Division
TRW Space & Technology Group



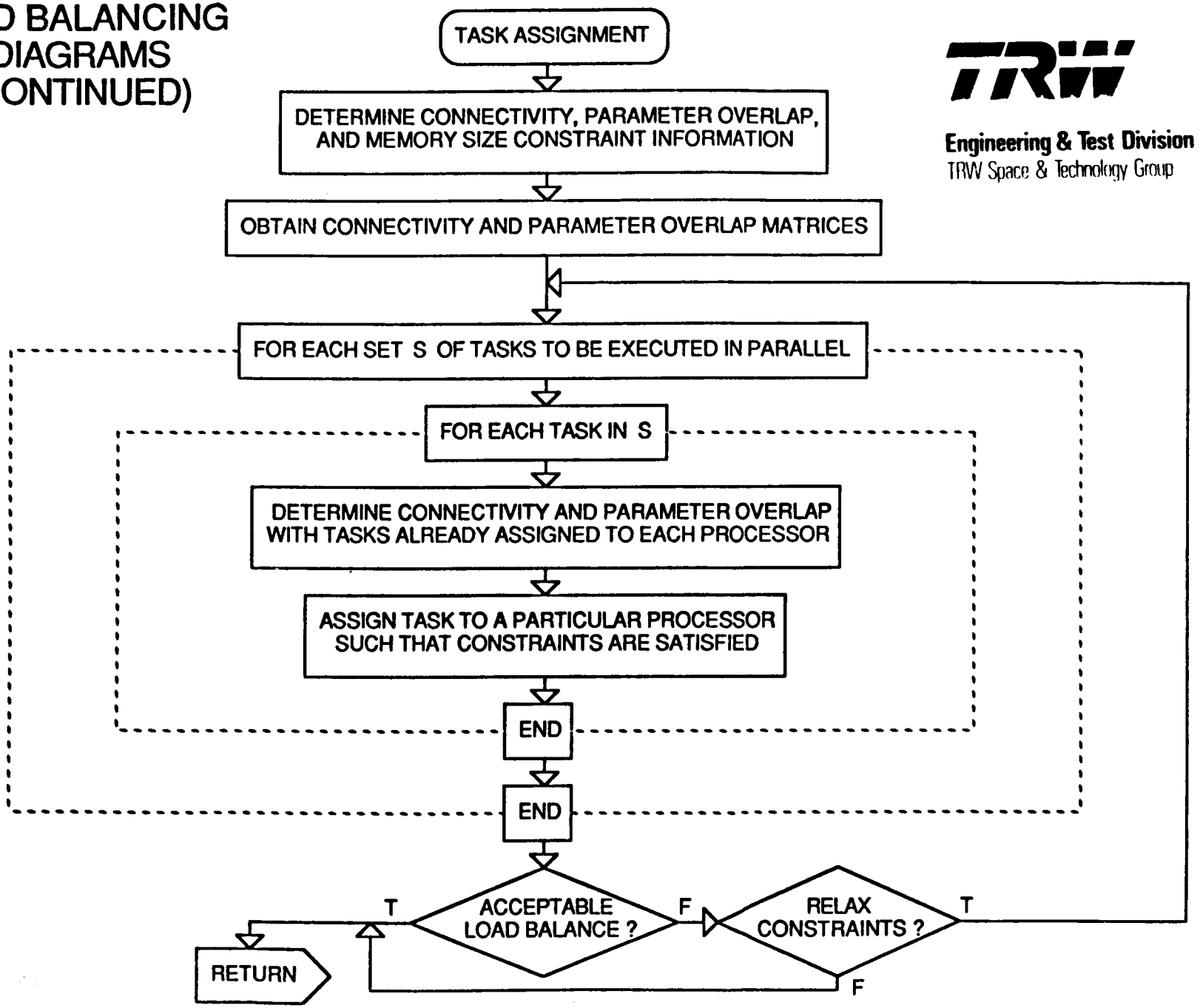
LOAD BALANCING DIAGRAMS (CONT'D)

At the conclusion of the code division, the tasks are evaluated in terms of how much communication they require with other tasks, and how many variables and parameters they share with other tasks. The program ASSIGN takes the results of the first of these evaluations and constructs what is called the connectivity matrix, with each entry indicating how much communication from task S, say, to task T is required, where S and T range over all tasks. ASSIGN uses the second evaluation to produce the parameter overlap matrix, where each entry indicates the number of parameters shared by the two tasks.

Tasks to be executed in parallel are now assigned to processors by ASSIGN. The connectivity and parameter overlap of each task with tasks already assigned to the processors is examined and a task is assigned to a particular processor according to a set of previously specified constraints.

At the end of the assignments, ASSIGN checks the computational load for balance among the processors. If the result is acceptable, then the software is finished. If not, then the constraints may be relaxed (in a way predefined by the user) and task assignment may be attempted again. If the constraints have been relaxed as far as possible, and the load is still not acceptably balanced, then TASK will attempt a different code division (and subsequent task assignment using ASSIGN) where the grain size may be different than before.

LOAD BALANCING DIAGRAMS (CONTINUED)



DATA FLOW GRAPH

The data dependencies in the Space Station model are shown in the figure. The graph depicts the functional form of the model's equations.

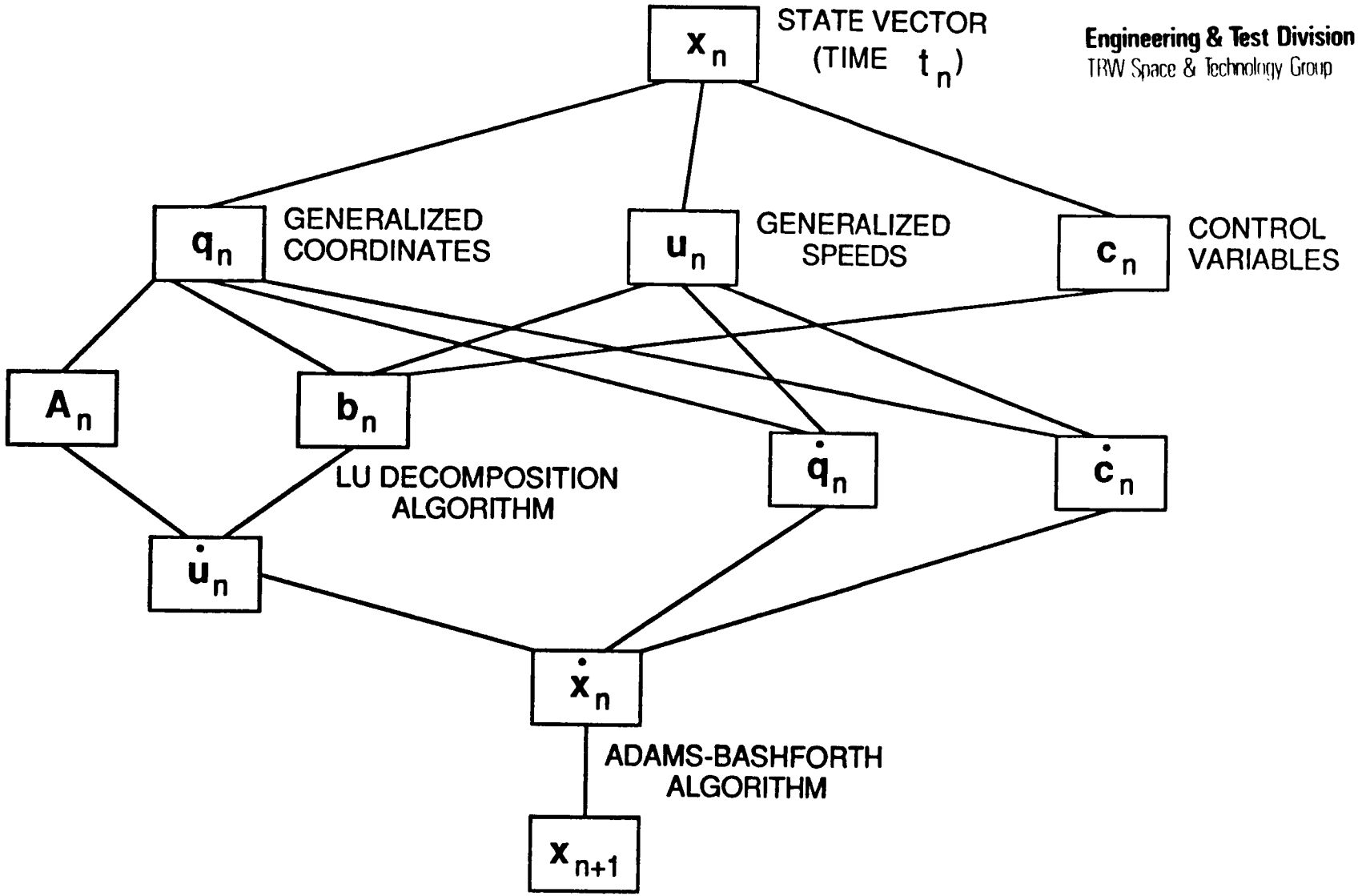
The state vector, $\underline{x}_n$, at a given time t_n is composed of the generalized speeds ($\underline{u}_n$), the generalized coordinates ($\underline{q}_n$) and the control variables ($\underline{c}_n$).

The generalized coordinates are used in calculating time derivatives for all the state variables, as are the generalized speeds (though these are not used in the computation of the matrix $\underline{A}_n$). The control variables affect only the control torques and thus influence only the vector $\underline{b}_n$ and the derivative of $\underline{u}_n$.

Time derivatives of $\underline{q}_n$ and $\underline{c}_n$ are found directly from the generalized coordinates and speeds. Gaussian elimination is used to solve the matrix equation $\underline{A}_n \dot{\underline{u}}_n = \underline{b}_n$ giving the derivative of $\underline{u}_n$.

The time derivatives of each of the components of the state vector are collected to give the derivative of $\underline{x}_n$. This is then integrated using the Adams-Bashforth algorithm to give the state vector at time t_{n+1}:$\underline{x}_{n+1}$.

DATA FLOW GRAPH



TOP LEVEL LOAD BALANCING (SPACE STATION MODEL)

Some opportunities for parallel execution of the Space Station model code are immediately apparent from even a casual inspection of the model, as shown in the figure.

The coordinate transformation matrices between frames in the three bodies, and between the body frames and an inertial frame are used frequently and must be calculated first. Each matrix, however, is calculated by a processor only if that processor will subsequently use it.

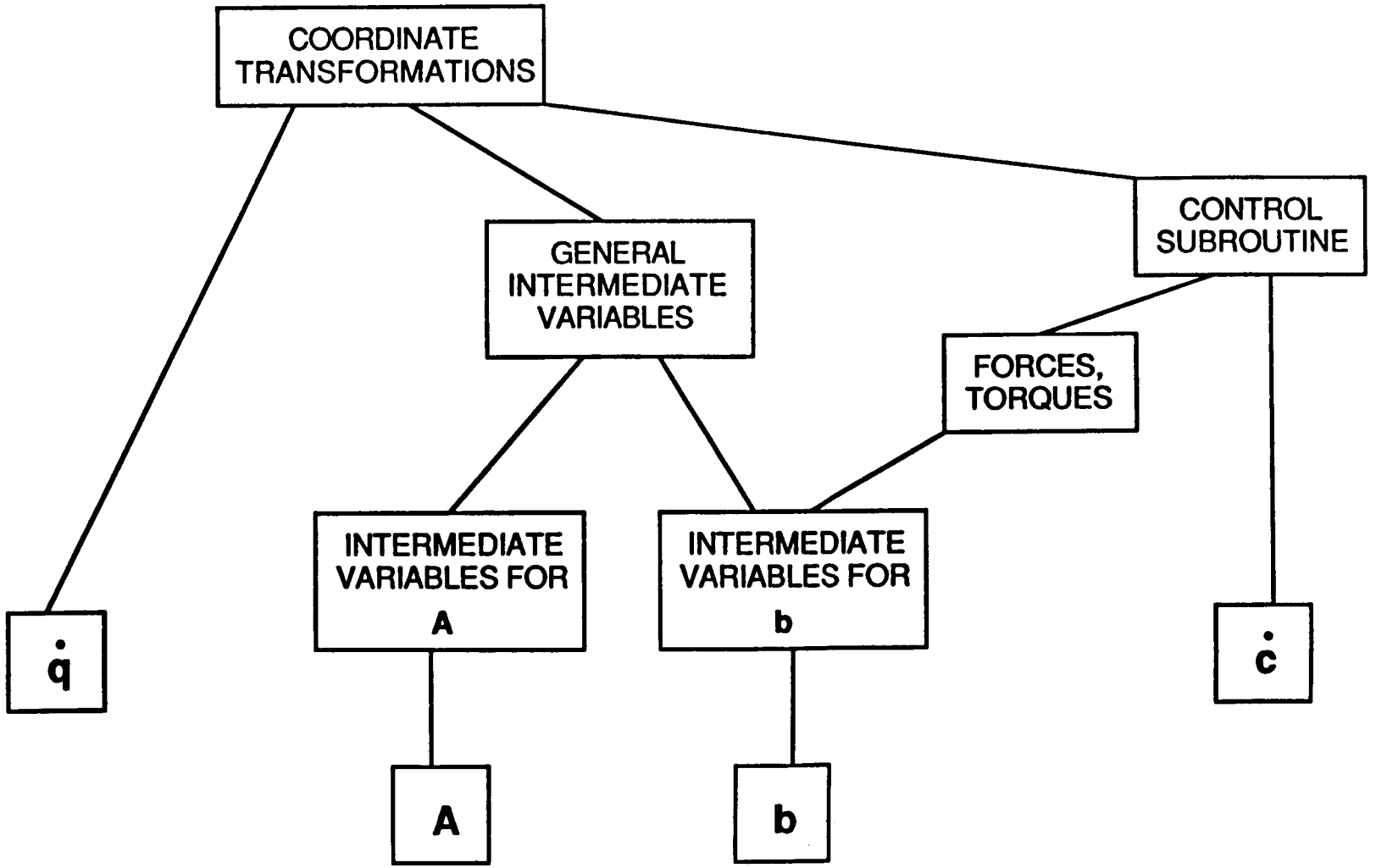
The outputs of the control subroutine are used only in the computation of the vector $\underline{b}$. Thus, this subroutine may be executed in parallel with sections of code computing general intermediate variables used by both $\underline{A}$ and $\underline{b}$ (such as partial angular velocities, partial velocities, etc.).

The calculation of elements of $\underline{A}$ and $\underline{b}$ may also be done in parallel, as may be the computation of the time derivative of $\underline{q}$.

A general division of the code may also be made according to whether computations involving Body 2 or Body 3 are needed. When possible, therefore, a given processor will compute quantities related only to Body 2 or only to Body 3, thus reducing interprocessor communication.

TOP LEVEL LOAD BALANCING (SPACE STATION MODEL)

Engineering & Test Division
TRW Space & Technology Group



410

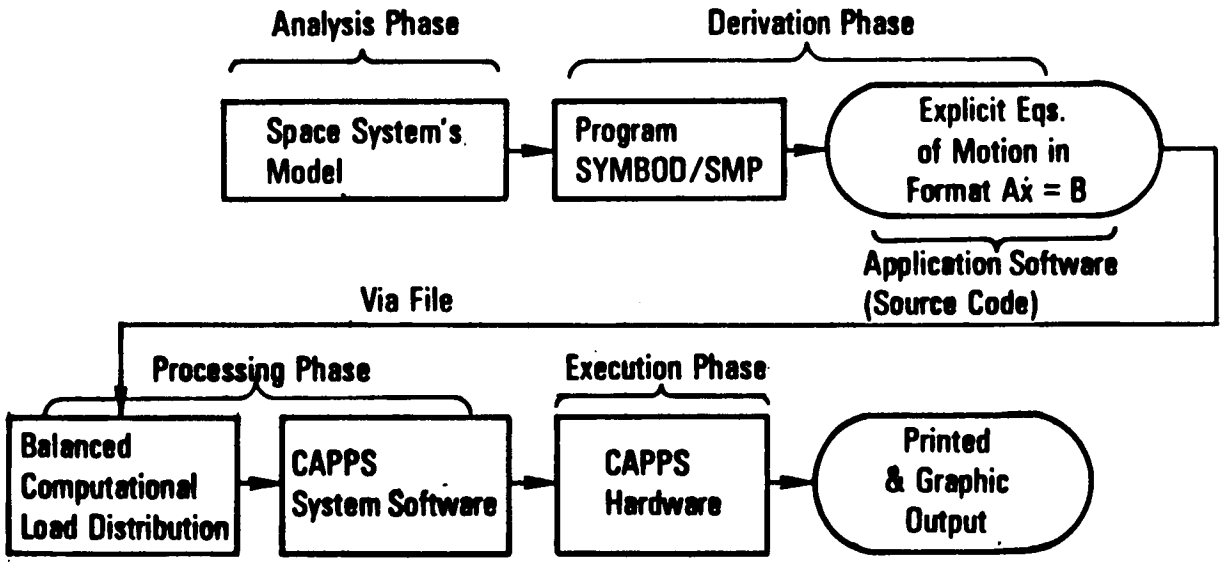
AN OVERVIEW OF THE CAPPS SIMULATION METHODOLOGY

An overview of the CAPPS simulation methodology is shown in the figure. The methodology is divided into four phases. The analysis phase constitutes the development of the mathematical model and requires an intensive interaction between the analyst and the previously described Program SYMBOD/SMP. The derivation phase follows, in which Program SMP carries out the instructions imbedded in SYMBOD to yield a matrix of ordinary differential equations in FORTRAN format. In addition to providing an accurate reflection of the analyst's intentions in the derivation of the equations of motion, this procedure also leads to equations which are virtually free of wasteful operations (such as additions of zeros, multiplications by unity and taking dot products of orthogonal vectors), as well as superfluous high order terms. The FORTRAN equations are delivered via file to the CAPPS computational load distribution software to begin the processing phase which is described in more detail below. It should be noted that the procedure completely eliminates the costly and time consuming programming effort which is normally required at this stage. The CAPPS system software transforms the derived equations from their original FORTRAN format to a binary format executable in concurrent operations by the CAPPS's CUs.

An Overview of the CAPPS Simulation Methodology

Engineering & Test Division
TRW Space & Technology Group

An Efficient Mathematical Model is Fed into a very Fast Computing System



412

REFERENCES

(1) C. Wampler, K. Buffinton, and J. Shu-hui, "Formulation of Equations of Motion for Systems Subject to Constraints"; ASME Journal of Applied Mechanics, Vol. 52, June 1985.

(2) K. C. Park and J. C. Chiou, "Stabilization of Computational Procedures for Constrained Dynamical Systems"; University of Colorado, College of Engineering, Report No. Cu-CSSC-87-05, June 1987.