N90-29898

Computed Torque Control of a Free-Flying Cooperating-Arm Robot

Ross Koningstein Marc Ullman
Robert H. Cannon Jr.
Stanford University Aerospace Robotics Laboratory*

January 30, 1989

Abstract

This paper presents a unified approach to solving free-floating space robot manipulator endpoint control problems using a control formulation based on an extension of computed torque. Once the desired endpoint accelerations have been specified, the kinematic equations are used with momentum conservation equations to solve for the joint accelerations in any of the robot's possible configurations: fixed base or free-flying with open/closed chain grasp. The joint accelerations can then be used to calculate the arm control torques and internal forces using a recursive order n algorithm. Initial experimental verification of these techniques has been performed using our laboratory model of a two-armed space robot. This fully autonomous spacecraft system experiences the drag-free, zero-g characteristics of space in two dimensions through the use of an air cushion support system. Results of these initial experiments are included which validate the correctness of the proposed methodology. The final section addresses the further problem of control in the large where not only the manipulator tip positions but the entire system consisting of base and arms must be controlled. The availablity of a physical testbed has brought many benefits to this work—particularly a keener insight into the subtleties of the problem at hand.

1 Introduction

To achieve fast, precise control of a physical system, accurate dynamical modelling is required. Dynamical modelling quickly becomes complex and cumbersome for human derivation as controlled systems become more and more complex. This section will formalize the process of computed torque control specification for robotic manipulator dynamical systems, introducing terms easily generated by algorithmic means and suitable for computer implementation. The control technique will also present extensions and formalisms for dealing with free-flying and closed chain rigid body manipulator systems, all of which share the characteristic of being easily machine generated. The basic premise for this technique is derived from the computed torque control technique.[1]. This technique uses kinematics for determining joint acceleration inverse dynamics for obtaining the corresponding joint torques. Specification of desired controls in operational or cartesian space[2] requires that the inverse and derivative of the system's Jacobian J be used. The Jacobian is expressed by

$$\mathbf{v}^{\text{endpoint}} = \mathbf{J}\dot{q}$$

where $\mathbf{v}$ is a vector of the speeds of the manipulator endpoints, measured in some coordinate system and $\dot{q}$ are the derivatives of the joint angles. Research by Alexander[3] into the control of free-flying robots first showed that the Jacobian was non-square. Subsequently, Umetani and Yoshida[4] demonstrated that the system Jacobian could be augmented by momentum equations to enable solving for joint accelerations. Independent investigation has led to the formalization of the structure of the Jacobian Matrix, using Kane's [5] notational convention, and augmenting a system's Jacobian to include both momentum relations and kinematic constraints implied by closed chains. The procedure presented here for Jacobian generation makes it possible to solve for actuator joint torques without determining reduced order equations of motion. Instead, it is possible to solve for these torques directly with a simple recursive order n procedure.

*Work performed under NASA contract NCC-2-333

1.1 Concepts used in Analysis

This theory for serial chain manipulators is derived using Kane's dynamical analysis techniques. The analysis that follows assumes that the velocities $\mathbf{v}$ of points and angular velocities $\boldsymbol{\omega}$ of bodies in the system under consideration can be expressed in a Newtonian reference frame as follows:

$$\begin{aligned} \mathbf{v}^i &= \sum_{s=1}^{p} \mathbf{v}_s^i u_s \\ \boldsymbol{\omega}^i &= \sum_{s=1}^{p} \boldsymbol{\omega}_s^i u_s \end{aligned}$$

where the generalized speeds $u_{1..n}$ are linear combinations of the derivatives of the generalized coordinates $\dot{q}_{1..n}$. The partial angular velocities of bodies, and partial velocities of points, as defined by Kane[5], can be shown to be:

$$\begin{aligned} \mathbf{v}_r &= \frac{\partial}{\partial u_r}\mathbf{v} \\ \boldsymbol{\omega}_r &= \frac{\partial}{\partial u_r}\boldsymbol{\omega} \end{aligned}$$

2 Jacobian Structure

2.1 Desired Accelerations

First, a method will be demonstrated which formulates the system Jacobian using partial velocities. The desired endpoint accelerations will then be expressed using these partial velocities and their derivatives, which is the basis for the computed torque method. The Jacobian, expressed using generalized speeds[1], is used as follows:

$$\mathbf{v}^{\text{endpoint}} = \mathbf{J}u$$

The endpoint acceleration can then be expressed as:

$$\mathbf{a}^{\text{endpoint}} = \mathbf{J}\dot{u} + \dot{\mathbf{J}}u$$

and the joint accelerations can be solved for by rearranging these equations:

$$\dot{u} = \mathbf{J}^{-1}(\mathbf{a}^{\text{endpoint}} - \dot{\mathbf{J}}\ u)$$

[1]If one chooses $u \triangleq \dot{q}$ then this is the standard Jacobian. If not, it becomes a more generalized Jacobian. The theory is valid for either case.

The Jacobian matrix's components are dependent upon the partial velocities and partial angular velocities of the endpoint of the manipulator(s) in the system. An endpoint velocity can be expressed in terms of its partials as:

$$\mathbf{v}^{\text{endpoint}} = \sum_{r=1}^{n} \mathbf{v}_r^{\text{endpoint}} u_r$$

and therefore 3D endpoint velocity can be expressed in terms of speeds along some established inertial x,y and z directions, for example, along unit vectors which we define as $\mathbf{x}, \mathbf{y}$ and $\mathbf{z}$:

$$\begin{aligned} \mathbf{v}^{\text{endpoint}} \cdot \hat{\mathbf{x}} &= \sum_{r=1}^{n} \mathbf{v}_r^{\text{endpoint}} \cdot \hat{\mathbf{x}}\ u_r \\ \mathbf{v}^{\text{endpoint}} \cdot \hat{\mathbf{y}} &= \sum_{r=1}^{n} \mathbf{v}_r^{\text{endpoint}} \cdot \hat{\mathbf{y}}\ u_r \\ \mathbf{v}^{\text{endpoint}} \cdot \hat{\mathbf{z}} &= \sum_{r=1}^{n} \mathbf{v}_r^{\text{endpoint}} \cdot \hat{\mathbf{z}}\ u_r \end{aligned}$$

the elements of the Jacobian due to an endpoint's velocity is therefore:

$$\begin{aligned} J_{1r} &= \mathbf{v}_r^{\text{endpoint}} \cdot \hat{\mathbf{x}} \\ J_{2r} &= \mathbf{v}_r^{\text{endpoint}} \cdot \hat{\mathbf{y}} \\ J_{3r} &= \mathbf{v}_r^{\text{endpoint}} \cdot \hat{\mathbf{z}} \\ &\vdots \end{aligned}$$

As shown above, desired endpoint accelerations can be expressed in terms of the Jacobian, its derivative, and the generalized speeds and their derivatives. The derivatives of the elements of the Jacobian can also be determined from the partial velocities:

$$\begin{aligned} \dot{J}_{1r} &= \dot{\mathbf{v}}_r^{\text{endpoint}} \cdot \hat{\mathbf{x}} \\ \dot{J}_{2r} &= \dot{\mathbf{v}}_r^{\text{endpoint}} \cdot \hat{\mathbf{y}} \\ \dot{J}_{3r} &= \dot{\mathbf{v}}_r^{\text{endpoint}} \cdot \hat{\mathbf{z}} \\ &\vdots \end{aligned}$$

where the derivatives, taken in a Newtonian reference frame, of the partial velocities are

$$\dot{\mathbf{v}}_r^{\text{endpoint}} \triangleq \frac{{}^{\mathrm{N}}\mathrm{d}}{\mathrm{d}t}\mathbf{v}_r^{\text{endpoint}}$$

which can be calculated very easily given the angular velocity of the body that the partial velocity vectors are based in. This completes the formal description of the Jacobian elements for desired accelerations. Note that desired angular accelerations are treated in an identical manner, allowing body angular acceleration specification.

2.2 Momentum Conservation

In any free-flying system of bodies, the linear and angular momenta vary according to the external forces on the system. On a free-flying robot, these are the system thrusters. If assume that these thruster settings are known a priori, we are able to predict the rate of change of the system momenta. The Jacobian can be augmented with linear and angular momenta equations to include these system states in the calculation of the desired generalized accelerations. Inclusion of these relations can make a Jacobian full rank, and suitable for application of the computed torque method.

First, the linear momentum, then the angular momentum of the system will be examined. The linear momentum $\mathbf{L}^i$ of a body i in the system is

$$\begin{aligned} \mathbf{L}^i &= m^i \mathbf{v}^{i*} \\ &= m^i \sum_{s=1}^{n} \mathbf{v}_s^{i*} u_s \\ &= \sum_{s=1}^{n} \mathbf{L}_s^i u_s \end{aligned}$$

where the *partial linear momentum* of body i is defined by

$$\mathbf{L}_s^i \triangleq m^i \mathbf{v}_s^{i*}$$

The linear momentum $\mathbf{L}$ of a system of ν bodies is the sum of the linear momenta of each body i in the system:

$$\begin{aligned} \mathbf{L} &= \sum_{i=1}^{\nu} \mathbf{L}^i \\ &= \sum_{i=1}^{\nu} m^i \mathbf{v}^{i*} \\ &= \sum_{i=1}^{\nu} m^i \sum_{s=1}^{n} \mathbf{v}_s^{i*} u_s \\ &= \sum_{i=1}^{\nu} \sum_{s=1}^{n} m^i \mathbf{v}_s^{i*} u_s \\ &= \sum_{i=1}^{\nu} \sum_{s=1}^{n} \mathbf{L}_s^i u_s \\ &= \sum_{s=1}^{n} \mathbf{L}_s u_s \end{aligned}$$

where the *partial linear momentum* of the system of ν bodies is defined by

$$\mathbf{L}_s \triangleq \sum_{i=1}^{\nu} m^i \mathbf{v}_s^{i*}$$

The *partial linear momenta* of the system can be formulated using the mass and center of mass partial velocity of each body in the system. The process of building an augmented Jacobian using these vector quantities is similar to the process used for the partial velocities discussed in the previous section, and will be discussed after the angular momentum terms are examined.

The angular momentum $\mathbf{H}^i$ of each body i, about its center of mass is:

$$\begin{aligned} \mathbf{H}^i &= \mathbf{I}^{i/i*} \omega^i \\ &= \mathbf{I}^{i/i*} \sum_{s=1}^{n} \omega_s^i u_s \\ &= \sum_{s=1}^{n} \mathbf{I}^{i/i*} \omega_s^i u_s \\ &= \sum_{s=1}^{n} \mathbf{H}_s^i u_s \end{aligned}$$

where the *partial angular momentum* $\mathbf{H}_s^i$ of each body is defined as

$$\mathbf{H}_s^i \triangleq \mathbf{I}^{i/i*} \omega_s^i$$

The central angular momentum $\mathbf{H}$ of the system of ν bodies about the system's center of mass point, is:

$$\begin{aligned} \mathbf{H} &= \sum_{i=1}^{\nu} \mathbf{H}^i + \sum_{i=1}^{\nu} (\mathbf{r}^{i*} - \mathbf{r}^{cm}) \times m^i \mathbf{v}^{i*} \\ &= \sum_{i=1}^{\nu} (\mathbf{I}^{i/i*} \omega^i + (\mathbf{r}^{i*} - \mathbf{r}^{cm}) \times m^i \mathbf{v}^{i*}) \\ &= \sum_{i=1}^{\nu} (\sum_{s=1}^{n} \mathbf{I}^{i/i*} \omega_s^i u_s + \sum_{s=1}^{n} (\mathbf{r}^{i*} - \mathbf{r}^{cm}) \times m^i \mathbf{v}_s^{i*} u_s) \\ &= \sum_{i=1}^{\nu} \sum_{s=1}^{n} (\mathbf{H}_s^i u_s + (\mathbf{r}^{i*} - \mathbf{r}^{cm}) \times \mathbf{L}_s^i u_s) \\ &= \sum_{s=1}^{n} \mathbf{H}_s u_s \end{aligned}$$

where the *partial angular momentum* $\mathbf{H}_s$ of the system is defined as

$$\mathbf{H}_s \triangleq \sum_{i=1}^{\nu}(\mathbf{H}_s^i + (\mathbf{r}^{i*} - \mathbf{r}^{cm}) \times \mathbf{L}_s^i)$$

A set of Jacobian augmentation equations can be set up which describe the relation between the momenta and the generalized speeds.

$$\begin{aligned} \mathbf{L} &= \mathbf{J}^L u \\ \mathbf{H} &= \mathbf{J}^H u \end{aligned}$$

The elements of the Jacobian due to the linear and angular momenta are therefore:

$$\begin{aligned} J_{1r}^L &= \mathbf{L}_r \cdot \hat{\mathbf{x}} \\ &\vdots \end{aligned}$$

and

$$\begin{aligned} J_{1r}^H &= \mathbf{H}_r \cdot \hat{\mathbf{x}} \\ &\vdots \end{aligned}$$

The *partial momenta* can be formulated automatically using the partial velocities in the system.

Expected momentum rates (due to external forces and torques) can be expressed in terms of these Jacobian augmentation equations and their derivatives along with the generalized speeds and their derivatives.

$$\begin{aligned} \dot{\mathbf{L}} &= \mathbf{J}^L \dot{u} + \dot{\mathbf{J}}^L u \\ \dot{\mathbf{H}} &= \mathbf{J}^H \dot{u} + \dot{\mathbf{J}}^H u \end{aligned}$$

The derivatives of the elements of the augmented Jacobian can be determined from the partial momenta:

$$\begin{aligned} \dot{J}_{1r}^L &= \dot{\mathbf{L}}_r \cdot \hat{\mathbf{x}} \\ &\vdots \end{aligned}$$

and

$$\begin{aligned} \dot{J}_{1r}^H &= \dot{\mathbf{H}}_r \cdot \hat{\mathbf{x}} \\ &\vdots \end{aligned}$$

where the derivatives, taken in a Newtonian reference frame, of the partial momenta are:

$$\begin{aligned} \dot{\mathbf{L}}_r &\triangleq \frac{{}^{\mathrm{N}}\mathrm{d}}{\mathrm{dt}}\mathbf{L}_r \\ \dot{\mathbf{H}}_r &\triangleq \frac{{}^{\mathrm{N}}\mathrm{d}}{\mathrm{dt}}\mathbf{H}_r \end{aligned}$$

and the rate of change of the momenta are given by:

$$\begin{aligned} \dot{\mathbf{L}} &= \sum \mathbf{F}^{\mathrm{ext}} \\ \dot{\mathbf{H}} &= \sum \mathbf{T}^{\mathrm{ext}} + \sum (\mathbf{r}^{ext} - \mathbf{r}^*) \times \mathbf{F}^{\mathrm{ext}} \end{aligned}$$

This completes the formal description of the Jacobian elements for momentum conservation.

2.3 Closed Chains

In a dynamical system with nonholonomic constraints, the generalized speeds $u_{1..n}$ are not independent, rather, one (or more) are dependent on the rest. In the system considered, a manipulator system, this condition can arise when two ends of a chain touch and are held together, either by a pin joint, or rigidly. The case of a velocity constraint on the a manipulator, a nonholonomic constraint situation, will be analyzed, and the constraint equations will be expressed in terms of quantities used in the kinematics derivations.

The constraint of endpoint closure is described by:

$$\mathbf{v}^{\mathrm{endpoint}_1} = \mathbf{v}^{\mathrm{endpoint}_2}$$

expanding this into partial velocities,

$$\sum_{r=1}^{n} \mathbf{v}_r^{\mathrm{endpoint}_1} u_r = \sum_{r=1}^{n} \mathbf{v}_r^{\mathrm{endpoint}_2} u_r$$

defining a constraint velocity[2]:

$$\begin{aligned} \mathbf{C} &\triangleq \mathbf{v}^{\mathrm{endpoint}_1} - \mathbf{v}^{\mathrm{endpoint}_2} \\ &= 0 \end{aligned}$$

and the *constraint partial velocities*[3] evaluate to:

$$\mathbf{C}_r = \mathbf{v}_r^{\mathrm{endpoint}_1} - \mathbf{v}_r^{\mathrm{endpoint}_2}$$

[2]The concept of a constraint velocity is not dependent upon having a free-floating base and hence works for all instances of closed kinematic chains

[3]Although the constraint velocity is zero, the individual *constraint partial velocities* are non-zero.

It is evident that by dot multiplication with inertial basis vectors, as was done with endpoint velocity, this vector equation can be reduced to scalar equations for incorporation into the system Jacobian.

$$0 = \mathbf{J}^C u$$

where the elements of these Jacobian augmentation equations are:

$$\begin{aligned} J^C_{1r} &= \mathbf{C}_r \cdot \hat{\mathbf{x}} \\ &\vdots \end{aligned}$$

These constraint partial velocities can be formulated automatically using the partial velocities of the endpoints of the manipulator which are touching.

Differentiating the constraint augmentation equations automatically expresses the acceleration constraints:

$$0 = \mathbf{J}^C \dot{u} + \dot{\mathbf{J}}^C u$$

The derivatives of the constraint augmentation equations can also be determined from the partial velocity derivatives:

$$\begin{aligned} \dot{J}1r &= \dot{\mathbf{C}}_r \cdot \mathbf{x} \\ &\vdots \end{aligned}$$

where the derivatives, taken in a Newtonian reference frame, of the constraint partial velocities are

$$\begin{aligned} \dot{\mathbf{C}}_r &\triangleq \frac{\mathrm{d}}{\mathrm{dt}} \mathbf{C}_r \\ &= \dot{\mathbf{v}}_r^{\mathrm{endpoint}_1} - \dot{\mathbf{v}}_r^{\mathrm{endpoint}_2} \end{aligned}$$

This completes the formal description of the Jacobian elements for closed chain constraints. Note that angular velocity constraints can be treated in an identical manner.

3 Joint Acceleration Solution

The full system Jacobian $\mathbf{J}^S$ can now be constructed using the following components: A regular Jacobian which relates the linear and angular velocities of the manipulator endpoint(s) to the the system's generalized speeds. Next augmentation equations describing the rates of change of system momenta are added. Finally, augmentation equations which ensure that the chain closure constraint is met are added. This process results in a full rank Jacobian that looks like:

$$\mathbf{J}^S = \begin{bmatrix} \mathbf{J} \\ \mathbf{J}^L \\ \mathbf{J}^H \\ \mathbf{J}^C \end{bmatrix}$$

A corresponding set of control objectives can be formulated:

$$\mathbf{a}^S = \begin{bmatrix} \mathbf{a}^{\mathrm{endpoint}} \\ \sum \mathbf{F}^{\mathrm{ext}} \\ \sum \mathbf{T}^{\mathrm{ext}} + \sum (\mathbf{r}^{ext} - \mathbf{r}^*) \times \mathbf{F}^{\mathrm{ext}} \\ 0 \end{bmatrix}$$

Relating these two quantities is the equation:

$$\mathbf{a}^S = \mathbf{J}^S u$$

from which we can solve for the derivatives of the generalized speeds corresponding to this set of control objectives:

$$\dot{u} = \mathbf{J}^{S^{-1}} (-\dot{\mathbf{J}}^S u + \mathbf{a}^S)$$

The resulting derivatives of the generalized speeds can then be used in an inverse dynamics routine to obtain corresponding joint control torques.

4 Order n Inverse Dynamics

In this section a simple and straightforward algorithm to solve the inverse dynamics equation for the control torques along a serial chain with revolute joints will be presented. Traditional computed torque control schemes have used the following equation to compute the joint torques:

$$\mathbf{M}\ddot{q} = \mathbf{V}(q, \dot{q}) + \mathbf{T}$$

This method requires $\mathcal{O}(n^2)$ computations, and requires that the mass matrix and non-linear terms of the system $\mathcal{S}$ be computed, then desired joint accelerations and known joint rates be used to generate a vector from which the control torques are easily derived. We will present an alternate method of computing these joint torques in $\mathcal{O}(n)$ computations. This method is based on the Newton-Euler method of formulating robot equations of motion, but instead of generating equations symbolically, we will generate numerical values for accelerations, joint forces and torques, and actuator torques as we traverse the robot's chain manipulator.

As we recurse down the rigid body chain, endpoint accelerations are used to determine the accelerations of all the joints and each of the center of mass points of the ν bodies in the system. We can use the link recursion relation that the acceleration at the start of a link is related to the acceleration at the end of a link as follows:

$$\begin{aligned}\mathbf{a}^{\text{start}} &= \mathbf{a}^{\text{end}} \\ &\quad -\boldsymbol{\alpha}^{\text{link}} \times \mathbf{r}^{\text{start to end}} \\ &\quad -\boldsymbol{\omega}^{\text{link}} \times \boldsymbol{\omega}^{\text{link}} \times \mathbf{r}^{\text{start to end}}\end{aligned}$$

where the following components are derived as follows:

$$\boldsymbol{\alpha}^{\text{link i}} = \boldsymbol{\alpha}^{\text{link i}-1} + \ddot{q}_i \cdot \lambda^{\mathbf{i}}$$

The axis direction $\lambda^{\mathbf{i}}$ is a positive rotation, in a right handed sense, along q_i. The forces and moments are propagated back from the end of each chain. We assume the force and moment at the end of the chain is a known value, typically zero. If the chain is closed, then a desired 'squeeze' force can be assumed as a starting internal force at the link end, and conceptually cutting the closed chain, into two.

We take moments about the joint at the start of the link, and consider only the components along the joint's axis $\lambda^{\mathbf{i}}$. The moments due to the center of mass acceleration and the link's angular acceleration are easily evaluated given its mass properties. The joint motor torque will be the only unknown in the equation

$$\begin{aligned}\mathbf{T}^i \cdot \lambda^{\mathbf{i}} &= -(\mathbf{T}^{\text{link end}} \\ &\quad +\mathbf{r}^{\text{start to end}} \times \mathbf{F}^{\text{end}} \\ &\quad -\mathbf{r}^{\text{start to }*} \times m^i \mathbf{a}^{i*}) \cdot \lambda^{\mathbf{i}}\end{aligned}$$

Now take moments about the link start point, which are the moments applied to the end of the next link in. Likewise, the sum of the forces will yield the forces applied by this link to the end of the next link in. The focus of reference can now be shifted to the next link in, where this process can be repeated until all of the control torques have been determined.

The process of solving for the joint control torques or forces is fairly straightforward, and if the robot has two or more arms, the solution for the control values for the various arms can be done in parallel.

5 Implementation

The Jacobian formulation method introduced here has been used to generate the joint acceleration specification matrix equation necessary in order to solve the computed torque control problem for the general 3D case of a free-flying robot with kinematic chain manipulators. The $\mathcal{O}(n)$ inverse dynamics solution has also been derived for this general 3D case. A specialized and partially optimized derivation for 2D has been done to allow testing on our experimental free-flying robot model.

The dynamical system under study, a dual arm satellite manipulator model, is essentially a serial chain of rigid bodies, and undergoes only minor changes (in terms of structure) when it grasps an object: it either becomes a longer chain, or it becomes a closed chain. If the equations of motion of a chain system have a certain form, then the addition of extra links to the system should result in a small change in the computation of the equations of motion - and not necessitate the rederivation of the system's equations of motion from scratch. The possibility of generating equations of motion and control algorithmically is desirable, since this task is then no longer a manual procedure. For our 2D robot testbed, the algorithms, given the joint accelerations, to compute the control torques are $\mathcal{O}(n)$.

Continuing work in the analysis of robot dynamics by Rosenthal[6], Rodriguez[7] and others have shown that robot dynamics for simulation can be solved in $\mathcal{O}(n)$ computations. In the spirit of this process, we have presented an algorithm for control which is also $\mathcal{O}(n)$.

6 Experimental Hardware

We have built a laboratory model of a dual-armed space robot which experiences in two-dimensions the drag-free, zero-g characteristics of space. These characteristics are achieved through the use of air cushion technology which allows our vehicle to float on a $9'x12'$ granite surface plate with a drag-to-weight ratio of about 10^{-4} and gravity induced accelerations below $10^{-5}g$—a very good approximation to the actual conditions of space. The robot is a fully self contained spacecraft containing

- an onboard gas subsystem (used both for flotation and for propulsion via thrusters)
- a complete electrical power system with plug-in rechargeable batteries packs[4] and power condi-

[4]The battery packs can also be recharged while on board the vehicle through the use of an umbilical power cord

tioning, distribution, and monitoring circuitry

- a full complement of sensors and signal conditioning electronics
- a high speed microprocessor based computer system with a floating point coprocessor
- a complete set of digital and analog data acquisition and control interfaces
- a fiber optic based data/communications link to a network of off-board computers

The robot measures $50cm$ in diameter and stands $65cm$ high with a total mass of just under $50kg$. In order to simplify maintenance operations as well as to facilitate future design modifications the robot was designed as a series of independent modules. These modules take the form of layers (see figure 1) which each perform a distinct task. The layers can be easily separated[5] when necessary for servicing or repair.

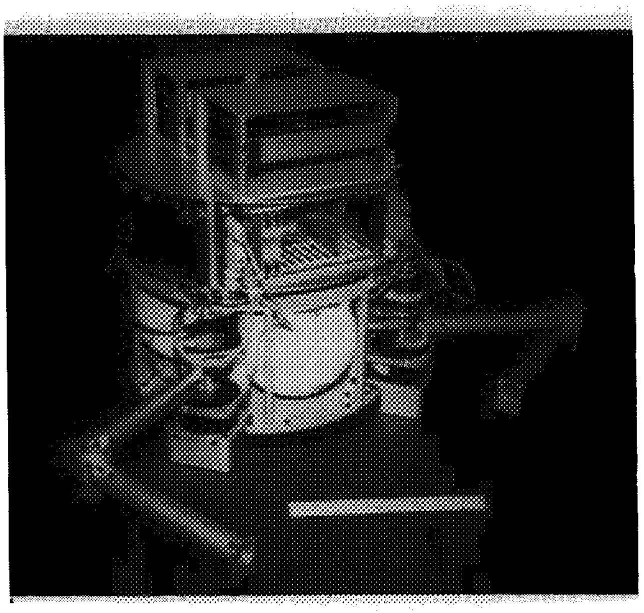

Figure 1: Stanford University Aerospace Robotics Laboratory Dual-Arm Space Robot

Figure 2 shows the nomenclature used for modeling the dynamics and characterizing the mass properties of the robot. The base body has three degrees of freedom (x, y, θ) and sports eight gas jet thrusters mounted as four ninety-degree pairs sitting at the corners of a square inscribing its outer circumference.

A pair of two-link planer arms aligned with a set of ninety-degree separated rays are attached to the base.

[5]The main layers can be separated without requiring the use of any tools.

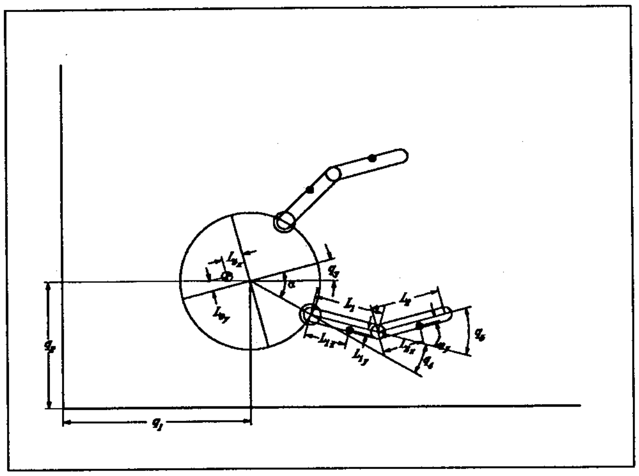

Figure 2: Free body diagram of space robot indicating nomenclature used for dynamic modelling.

These manipulator arms are driven by a coaxial set of limited angle DC torque motors—the shoulder joint being driven directly while the elbow joint is driven though a cable from the elbow motor which rides on the shoulder link. Both joints are instrumented with RVDTs for sensing joint angles. Analog differentiators provide corresponding rate signals in hardware. The manipulators are equipped with pneumatically actuated grippers which possess a single degree of freedom along the z-axis. Objects can be grasped by lowering the gripper plungers into cup-like grasp points mounted on the objects.

The onboard computer system runs the VxWorks[6] real time operating system. This operating system allows us to develop code on our Sun Workstations which can then be downloaded to the target processor via a fiber optic Ethernet link. Since the real time OS contains a complete implementation of TCP/IP and NFS our target processor can access files and data on our host server. We have configured our system with a set of subnets so that we can communicate between on and off board processors without incurring delays due to traffic on our workstation LAN.

We will ultimately be adding an on-board vision system in order to allow us to perform endpoint control. This addition will enabling us to capture and manipulate free-floating targets.

7 Experimental Results

We have implemented the control methodology described above on our space robot system and it works!

[6]*VxWorks*^{TM} is a product of Wind River Systems, Emeryville, CA

ORIGINAL PAGE
BLACK AND WHITE PHOTOGRAPH

8 Large Motion Control

In order for free flying space robots to be effective instruments in the space automation arsenal they must be capable of autonomously executing large motions which involve the coordinated motion of their base body and their manipulators. It is for this reason that the task of gross motion control of a space robot poses a set of interesting and unique challenges which differ from the typical satellite positioning/attitude control problem or the uncontrolled free-floating base situation presented above. In most satellite control problems we are interested in achieving two principal goals: 1) keeping the satellite as a whole on its proper orbit path, and 2) keeping various sensors and/or communications devices pointed in desired directions. These objectives amount to requirements for holding the center of mass of the satellite on track while servoing the attitude of the main body so as to keep the pointing actuators within their allowable ranges of motion. As noted above, the linear and angular momentum principles tell us that the total linear and total angular momenta are unaffected by the internal actuators so this problem divides nicely into three distinct parts: 1) controlling the position of the system center of mass, 2) controlling the attitude of the main body, and 3) controlling the orientations of the respective sensors. Clearly part 1 is independent of parts 2 and 3; however part 3 acts as a disturbance to part 2 and visa versa.

By way of contrast, in the space robot gross motion control problem we are interested in controlling the base body position and orientation so as to place or maintain the manipulator arm tip position(s) in a desired workspace. Since we are interested in the actual positions of the manipulators (as opposed to the orientation of sensors in the case of a satellite) we must control both the base position and orientation rather than just the system center of mass position. In particular, if we are operating in a densely populated workplace (e.g. in the middle of space station construction) we must know the exact extents of our base body and all of its appendages. There are, of course, certain circumstances were we might be executing a large motion slew (one which requires base motion in order to complete) away from any potential obstacles. In this case we may not be concerned with the manipulator tip positions or the precise base position and thus can control the position of the system center of mass. Therefore, a number of different control situation may arise and enumerated below:

- The robot is in position to perform some task; however, since their is no way for it to anchor itself to the environment it is working in[7] we must perform station keeping of the base position and orientation to keep the manipulators within the required workspace.
- The robot is executing a large motion slew along some prescribed trajectory with a large corridor of unobstructed space surrounding it. In this case, we can control position of the system center of mass without concerning ourselves with the actual location or orientation of the base and the manipulators.
- The robot is attempting to intercept a free floating object such as a satellite and must execute a trajectory which will rendezvous with the target in such a way as to match both its position and velocity at the intercept point. In planning and performing such a trajectory we must assure that the base position and orientation allow the manipulators sufficient freedom of reach so that they can successfully grapple the target without running into the limits of their workspace.

Clearly it is this last case which is the most challenging and therefore the most interesting. In order to successfully capture a free floating target we must simultaneously control our manipulator tip positions as well as the robot base position and orientation. Since the corresponding states are coupled with each other it becomes necessary to view the system as whole rather than as decoupled parts. Simply generating an intercept trajectory which is realizable given the limited actuator authority available, the ever present dynamic constraints imposed by a free floating robot[8], and any temporal constraints which might exist(e.g. the object might float out of reach if we do not get to it soon enough) presents a formidable problem. Various trajectory generation, validation, and control algorithms which address these issues will be the focus of a future paper.

References

[1] John J. Craig. *Introduction to Robotics Mechanics and Control.* Addison-Wesley, Reading, MA, 1986.

[2] Oussama Khatib. Dynamic control of manipulators in operational space. In *Proceedings of the*

[7] If there was some way for the robot to anchor itself to its environment, we would not have the "free-flying" robot control problem at all.

[8] Imagine driving a car where the only way to slow down was to put it in reverse.

Sixth CISM-IFToMM Congress on Theory of Machines and Mechanisms, pages 1128–1131, New Delhi, India, December 1983.

[3] Harold L. Alexander. *Experiments in Control of Satellite Manipulators.* PhD thesis, Stanford University, Department of Electrical Engineering, Stanford, CA 94305, December 1987.

[4] Y. Umetani and K. Yoshida. Continuous path control of space manipulators mounted on omv. *ACTA Astronautica*, 15(12):981–986, 87.

[5] Thomas R. Kane and David A. Levinson. *Dynamics: Theory and Application.* McGraw-Hill Series in Mechanical Engineering. McGraw-Hill Book Company, New York, NY, 1985.

[6] Dan E. Rosenthal. Order N Formulation for Equations of Motion of Multibody Systems. In G. Man and R. Laskin, editors, *Proceedings of the Workshop on Multibody Simulation*, pages 1122–1150, Pasadena, CA, April 1988. NASA Jet Propulsion Laboratory. JPL D-5190, Volume III.

[7] G. Rodriguez and K. Kreutz. Recursive mass matrix factorization and inversion. Technical report, NASA Jet Propulsion Laboratory, Pasadena, CA, March 1988. JPL Publication 88–11.