Journal of Intelligent and Robotic Systems **5**: 5–24, 1992.
© 1992 *Kluwer Academic Publishers. Printed in the Netherlands.*

Experimental Evaluation of Cartesian Stiffness Control on a Seven Degree-of-Freedom Robot Arm

JOHN FIALA and ALBERT J. WAVERING
National Institute of Standards and Technology, Robot Systems Division, Gaithersburg, MD 20899, U.S.A.

(Received: 20 September 1990; revised: 12 February 1991)

Abstract. The programmability of Cartesian stiffness in Cartesian servo control algorithms that do not use explicit force feedback is examined. A number of Cartesian algorithms are implemented and evaluated on a commercial seven degree-of-freedom robot arm, using the NASREM robot control system testbed. It is found that Cartesian servo algorithms which use the transpose of the Jacobian and model-based gravity compensation, provide easy programmability and accurate reproduction of stiffnesses over a wide range. When dynamic behavior is a consideration, dynamic damping control, augmented to include a parameterization of the manipulator self-motion, provides superior performance and programmability.

Key words. Stiffness control, Cartesian control, robot control algorithms, control system architectures.

1. Introduction

A number of authors have described Cartesian-based servo algorithms for robot arm control in the recent literature. Cartesian-based servo algorithms compute tracking errors for the manipulator in a Cartesian coordinate system rather than in manipulator joint-space. Torques which provide the control input to the actuators, are obtained from the Cartesian errors by use of the manipulator Jacobian. There are two principal reasons for wanting to use a control algorithm of this type. First, the servo is in the coordinate frame in which the task geometry and dynamics are most easily expressed. Secondly, the fact that the servo is directly in the task coordinates eliminates the need to do explicit inverse kinematics, which can be expensive in the seven degree-of-freedom case.

By performing the servo in task coordinates, the control parameters can often be designed to provide the desired behavior of the manipulator end effector. For example, if the task dynamics require that the end effector be made to have a certain *impedance*, this can be achieved by a Cartesian servo algorithm having parameters sufficient to specify the mass-spring-damper behavior of the manipulator at the end effector [9]. Many manipulations can be performed by adjusting the compliance or *stiffness* of the manipulator. This paper investigates a class of Cartesian servo algorithms in which the gains of the Cartesian servo provide a convenient means for achieving end effector stiffness in task coordinates. This class of algorithms is termed *Cartesian stiffness control.*

Although the term *stiffness control* has been used frequently in the literature to refer to a variety of manipulator control algorithms, for the purposes of this discussion a stiffness control algorithm is one that contains modifiable parameters which directly encode the effective stiffness of the manipulator end effector in Cartesian coordinates without the use of force sensing at the end effector. Thus, stiffness control differs from *impedance control* in that impedance control requires that contact forces be sensed at the end effector and fed back as part of the control algorithm. In this paper, control algorithms which involve feedback of sensed end effector forces are not considered. Rather, contact forces are accommodated open-loop by the widely adjustable stiffness parameters in the control algorithm.

Programmable stiffness without force sensing requires a robot arm which is capable of producing accurate joint torques such that the appropriate force is realized at the tip. The control system testbed at the National Institute of Standards and Technology (NIST) uses the Robotics Research Corporation (RRC) K-1607* seven degree-of-freedom manipulator. This device has proven capable of providing programmable open-loop stiffness, as described in Section 5, with dynamic performance, as discussed in Section 6. The robot control systen testbed which operates the RRC manipulator is being developed at NIST in support of the National Aeronautics and Space Administration's (NASA's) Flight Telerobot Servicer project [16]. The testbed is based on the NASA/NIST Standard Reference Model Telerobot Control System Architecture (NASREM) [1]. The servo control level of the testbed allows for multiple servo algorithms and can be reconfigured in real-time. Specifics of the servo implementation are discussed in Section 4. First, Section 2 begins the discussion of Cartesian control by introducing notation and some computational aspects. Section 3 follows with a discussion of Cartesian stiffness control, relating a number of well-known algorithms.

2. Cartesian Control

Cartesian control algorithms generally require computing forward kinematics on the sensed joint positions,

$$x = \mathrm{kin}\,(\theta),$$

where θ is the joint position vector and x is the position of the manipulator tip with respect to a Cartesian coordinate system fixed at the base of the robot, as depicted in Figure 1. The six-dimensional quantity x includes orientation, which is represented by a quaternion as described in [4]. Such Cartesian quantities will be referred to simply as 'position' in this discussion. Lower-case symbols are used to represent vectors and upper-case symbols to represent matrices, unless otherwise stated.

* Certain commercial equipment is identified in this paper to describe the work adequately. Such identification does not imply recommendations or endorsement by the National Institute of Standards and Technology, nor does it imply that the equipment used is necessarily the best available for the purpose.

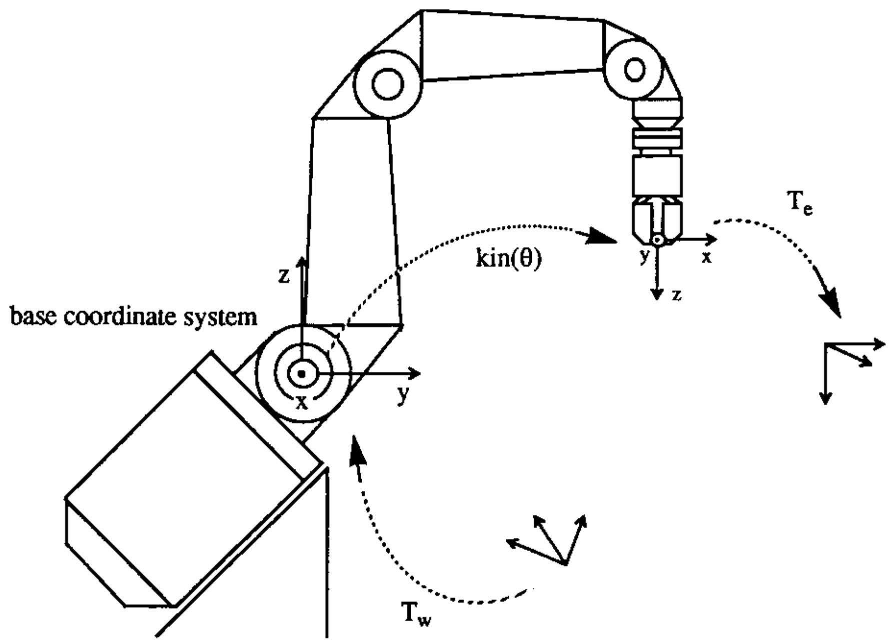

Fig. 1. Cartesian control coordinate systems.

The basic kinematics can be extended by the addition of two offset transformations [20]. Transformation T_w relates a reference coordinate system to the base coordinate system, as shown in the figure. Transformation T_e relates the offset of a tool from the end effector. The position of the tool with respect to the reference frame is

$$x = T_w \operatorname{kin}(\theta) T_e \tag{1}$$

Cartesian vectors such as this expressed with respect to the reference coordinate frame, are said to be in *world coordinates*. Vectors expressed with respect to the tool coordinate frame would be in *end effector coordinates*.

Although Cartesian position must always be given in world coordinates it is possible to express Cartesian velocity in either world or end effector coordinates. Forward kinematics for velocity is given by

$$\dot{x} = J(\theta)\dot{\theta}$$

where the Cartesian six-vector, $\dot{x}$, gives the rate of change of the Cartesian directions x, y, z, roll, pitch, yaw, in that order [20], and $J(\theta)$ is the appropriate Jacobian matrix for world or end effector coordinates. Note that $J(\theta)$ includes a differential transform corresponding to T_e in the case of end effector coordinate velocity representation, and a differential transform corresponding to T_w for world representation [7]. In the following, explicit representation of position-dependency will not be made (i.e., $J(\theta)$ will be written as J, for simplicity).

Cartesian control also involves computing Cartesian errors, which are represented by $(x_d - x)$, the desired position minus the sensed position. The objective of stiffness control is to make end effector forces proportional to position errors for each Cartesian degree of freedom. Ideally, the stiffness of each translational and rotational degree of freedom would be independently specifiable and completely decoupled from the others. For translational deflections of any size and for small rotations this ideal may be approached. The translational components of $(x_d - x)$ are determined using vector subtraction, and the rotational components are computed in two steps. First, the quarternion which represents the error between the desired and actual orientations is computed

$$^{w}r_e = {}^{w}r_d \cdot \mathrm{inv}\,(^{w}r)$$

where $^{w}r_e$ is the error quaternion with respect to world coordinates, and $^{w}r_d$ and ^{w}r are the desired and actual orientations with respect to world coordinates. (The definition of inverse and multiplication operations for quarternions may be found in [4].) Next, the error quaternion is converted to a three-dimensional differential rotation vector by multiplying the components of the axis of the quaternion by the magnitude of the angle of the quaternion. The equivalence of these two representations for small rotations is demonstrated in [20].

This technique may be applied for large rotational displacements as well. However, large rotational displacements do not behave as vectors, and cannot be resolved into independent components about each axis. Independent (different) stiffnesses may not be truly realized for arbitrarily large displacements along rotational degrees of freedom. Yet, if uniform rotational stiffnesses are used with the above computation of rotational error, the result is that the desired rotational stiffness will be applied about the axis of the error rotation, which is as desired.

Cartesian control errors can also be computed with respect to end effector coordinates instead of world coordinates. In this scheme, x_d is still specified in world coordinates, but the operation $(x_d - x)$ is modified to give the error relative to the end effector frame.

$$^{\mathrm{end\,eff}}r_e = \mathrm{inv}\,(^{w}r) \cdot {}^{w}r_d$$
$$^{\mathrm{end\,eff}}p_e = \mathrm{inv}\,(^{w}r) \cdot (^{w}p_d - {}^{w}p)$$

Here, the symbols involving r represent the rotational part of the Cartesian vectors and the symbols involving p represent the positional part of the Cartesian vectors.

Thus, by substituting the appropriate $(x_d - x)$ and Jacobian computations, Cartesian control algorithms can be made to allow the realization of stiffness frames which are aligned with either world or end effector coordinates [8]. In addition, the use of offsets T_w and T_e allow the control coordinates to be placed arbitrarily anywhere in the robot workspace [7].

3. Cartesian Stiffness Control

The term *stiffness control* originates from Salisbury's 'active stiffness control' algorithm. Salisbury's original algorithm is quite complicated and involves extensive use of force feedback [21]. However, after simplification to open-loop form, the algorithm can be written

$$\tau_{\text{act}} = J^t K_p J(\theta_d - \theta) + K_v M(\dot{\theta}_d - \dot{\theta}) + \tau_{\text{gravity}} + \tau_{\text{friction}} \tag{2}$$

where τ_{act} is the control torque to the joint actuators, K_p is the Cartesian stiffness specification, K_v is the damping gain, J^t is the transpose of the Jacobian, and M is the model inertia matrix. The vectors τ_{gravity} and τ_{friction} provide model-based compensation for the effects of gravity and friction.

Salisbury's algorithm has a number of interesting features. First, there is feed-forward compensation for the effect of gravity, which is essential for the success of all stiffness control algorithms. Without this compensation, the position gain K_p must always be large enough to hold the manipulator at the goal position against the pull of gravity and, thus, the stiffness is not independently programmable. Another feature of Salisbury's algorithm is that the control damping is specified with respect to joint coordinates and includes compensation for the variable inertia of the mechanism. The amount of joint damping is adjusted according to the effective inertia at each joint, but the damping ratio of the dynamic response at the end effector still varies with arm configuration.

Algorithm (2) is a *joint stiffness control* algorithm, since the servo error is computed in joint space. The Cartesian stiffness is specified by K_p which is transformed to a joint-space stiffness by

$$K_\theta = J^t K_p J \tag{3}$$

This produces a constant Cartesian stiffness K_p about the end effector for any manipulator position. However, the algorithm is based on the assumption that the joint-space displacement $(\theta_d - \theta)$ is small so that as the distance between θ_d and θ increases, the effective Cartesian spring origin is shifted away from that corresponding to θ_d. This means that the specified stiffness will not be achieved for large displacements.

To compare with Salisbury's approach, one can examine the form of other well-known manipulator control algorithms with force feedback removed. For example, removing force feedback from Whitney's versions [24] of stiffness control and hybrid position/force control results in

$$\tau_{\text{act}} = K_p J^{-1}(x_d - x) - K_v \dot{\theta}. \tag{4}$$

Here, K_p specifies a joint-space quantity since it acts on the joint vector $J^{-1}(x_d - x)$. In addition, there is no explicit compensation for gravity. Thus, (4) cannot be considered to be a stiffness control algorithm by the current definition.

Two modifications of (4) seem likely to produce a more reasonable algorithm. The simplest modification is to interchange J^{-1} and K_p, which results in an algorithm similar to resolved acceleration control [15].

$$\tau_{\text{act}} = J^{-1} K_p(x_d - x) - K_v \dot{\theta}. \tag{5}$$

Now, K_p clearly acts in Cartesian coordinates, but J^{-1} does not correctly relate the desired Cartesian force $K_p(x_d - x)$ to joint torque. Again, Cartesian stiffness control is not realized.

Since $J^{-1}(x_d - x)$ in (4) is a joint-space quantity, perhaps an equivalent joint-space stiffness, K_θ of (3), can be used for K_p in (4). The algorithm resulting from this substitution and the addition of friction and gravity compensation is called the *Jacobian-transpose* algorithm

$$\tau_{\text{act}} = J^t K_p(x_d - x) - K_v\dot{\theta} + \tau_{\text{gravity}} + \tau_{\text{friction}}. \tag{6}$$

This algorithm is a *Cartesian stiffness control*, since the servo error is in Cartesian coordinates and Cartesian stiffness K_p is obtained over the entire workspace for displacements of any magnitude.

The Jacobian-transpose control scheme given by (6) has a number of nice properties. It has been proven [22] that, in noncontact situations, the algorithm produces asymptotically stable behavior for positive-definite symmetric matrices K_p and K_v. The algorithm is also probably passive [2] such that the manipulator will be stable in contact with all passive environments. Another benefit of (6) is that the control algorithm, since it uses only the transpose of the Jacobian, cannot 'blow-up' and produce infinite joint rates as in many inverse control schemes. Other authors have even found the algorithm to be robust with respect to errors in the Jacobian [17].

When the manipulator being controlled by (6) has seven degrees of freedom, it is redundant with respect to the Cartesian position goal. This means that the manipulator can execute *self-motions* while still satisfying the Cartesian stiffness task. Since (6) provides no control for this self-motion, except gravity compensation and damping, the self-motion may assume an arbitrary position. It may be desirable to control the position (and stiffness) of the self-motion. This can easily be done by adding a self-motion parameter to the Cartesian goal and feedback vectors and augmenting the Jacobian to relate changes in this parameter to joint space [8, 14, 19]. The *augmented Jacobian-transpose* algorithm is

$$\tau_{\text{act}} = J_a^t K_p(x_d - x) - K_v\dot{\theta} + \tau_{\text{gravity}} + \tau_{\text{friction}}, \tag{7}$$

where J_a^t is the transpose of the 7×7 augmented Jacobian matrix, and the x vectors are now seven-dimensional quantities which include the position of the manipulator tip *and* the self-motion parameter. One possibility for the self-motion parameter is to use the position of a joint such as the first joint. This will work fine except for those configurations in which the first joint no longer lies in the nullspace of the Jacobian. Kreutz *et al.* [14] have devised a kinematic expression for the K-1607 *elbow angle* as a self-motion parameter. This provides a good parameterization of the nullspace when the manipulator is far from kinematic singularities, although it does introduce some additional mathematical singularities. Most algorithms discussed in this section can be augmented as just described.

One drawback of the Jacobian-transpose algorithm is that the Cartesian dynamic response is not uniform throughout the workspace. Cartesian motions will be more heavily damped for some arm configurations than others. To get a uniform dynamic response, sophisticated algorithms such as the operational space approach have been proposed [5, 11].

$$\tau_{\text{act}} = J^t M_x[K_p(x_d - x) - K_v\dot{x}] + \tau_{\text{cent}} + \tau_{\text{gravity}} + \tau_{\text{friction}}. \tag{8}$$

Here, τ_{cent} is the model torque which compensates Coriolis, centrifugal, and other velocity effects. M_x is the model manipulator inertia in task coordinates. The model-based nonlinear compensation effectively linearizes and decouples the system when the manipulator is not in contact with the environment. Constant K_p and K_v in this case provide constant bandwidth over the workspace. Unfortunately, the algorithm does not provide uniform stiffness. The effective Cartesian stiffness is $M_x K_p$, which varies with manipulator configuration [2]. Thus, (8) cannot function as a Cartesian stiffness control algorithm.

By proper inclusion of force feedback, algorithms like (8) can provide *impedance control* in which a uniform mass-spring-damper behavior can be specified for the end effector [9, 18, 19]. This technique can have some disadvantages, however. One problem is that it may be difficult to maintain the passivity of the control with the complete model-based compensation and for arbitrary mass specification [2, 19]. This means that the control may be unstable when the manipulator is in contact with certain environments. Also, without accurate sensing of the external forces acting on the manipulator, a Cartesian stiffness cannot be programmed in this approach. Obtaining accurate feedback of all interaction forces may be difficult, as in the case of external forces acting on the self-motion of a redundant manipulator.

When accurate models of the manipulator dynamics are not available, such that $M_x = I$, and $\tau_{\text{cent}} = 0$, algorithm (8) simplifies to

$$\tau_{\text{act}} = J^t[K_p(x_d - x) - K_v\dot{x}] + \tau_{\text{gravity}} + \tau_{\text{friction}}. \tag{9}$$

Note the similarity of this algorithm to that of (6). The only difference is that the damping term is in Cartesian space instead of joint space. Although it may sometimes be useful to specify Cartesian damping, this approach does have a drawback. Assume that K_v is diagonal and uniform, such that $K_v = k_v I$. Then (9) can be rewritten

$$\tau_{\text{act}} = J^t K_p(x_d - x) - k_v J^t J\dot{\theta} + \tau_{\text{gravity}} + \tau_{\text{friction}}. \tag{10}$$

Clearly, the joint velocities get mapped through the Jacobian so that any velocities in the nullspace of the Jacobian will not be damped by the control algorithm. The redundant self-motion due to using only the tip Jacobian or encountering a singular configuration is unstable for this algorithm [11, 13]. Experiments at NIST have shown that this control law is not very robust near singular regions. Limit cycle behavior is generated for a large region near the wrist singularity of the K-1607 manipulator at lower sampling rates and higher K_v.

Even though K_v specifies a Cartesian damping in (9), a constant damping ratio is not achieved for the Cartesian dynamic response because the variable inertia of the mechanism is uncompensated. It is possible to achieve a constant damping ratio with

Cartesian stiffness control by use of the *dynamic damping* approach developed by Anderson [2]. The form of this algorithm

$$\tau_{\text{act}} = J^t K_p(x_d - x) - B\dot{\theta} + \tau_{\text{gravity}} + \tau_{\text{friction}} \tag{11}$$

is similar to (6), but here B is computed based on the specified K_p and the manipulator inertia to provide a constant damping ratio over the workspace. This algorithm has the advantage of only requiring specification of K_p and having the appropriate B determined automatically. In the NIST implementation, the Jacobian is augmented using the elbow angle parameterization as described for (7) to provide a dynamic damping algorithm which allows control of the manipulator self-motion. The damping term is computed as in [2],

$$B = 2\zeta(K_p^{1/2}J_a)^t V\Sigma V^t(K_p^{1/2}J_a) \tag{12}$$

where V and Σ are obtained from the Singular Value Decomposition (SVD)

$$U\Sigma V^t = M^{1/2}(K_p^{1/2}J_a)^{-1}$$

and ζ is the desired damping ratio. This computation is expensive, but by using the technique of saving SVD rotations from the previous cycle as described in [25], the computation of B (after M and J have already been obtained), is performed in 50–60 ms in the NIST testbed control system.

To conclude this section, Cartesian stiffness control without the use of force feedback requires an algorithm which incorporates, at a minimum, gravity compensation and the Jacobian transpose, such as in algorithms, (6), (9), and (11). While (6) offers a good combination of computational efficiency and robustness, determination of appropriate damping gains can be very difficult. One one hand, it is not difficult to select gains which result in a system that is stable both in free space and in contact. However, the dynamic performance is highly configuration-dependent and nonuniform. This may not be a severe problem in applications where the algorithm is used for quasistatic motions, such as part-mating during assembly.

The procedure used at NIST to select damping gains for (6) is to determine the maximum velocity gain which may be used for each joint by locating the high-gain instability value, as in [12]. This instability, which is primarily affected by the sample rate and joint inertia, is performed with the minimum-inertia configuration for the joint. The velocity gain which results is then decreased by about 20% to give the maximum allowable value. The maximum position gains are then limited by the damping available when the maximum velocity gains are used. The manipulator response is usually over-damped with this approach. This problem of joint velocity gain selection may be avoided altogether with the use of an algorithm such as dynamic damping control (11). For this reason, algorithm (11), augmented as in (7), is considered to be a more useful Cartesian stiffness control when dynamic performance is a consideration.

4. Implementation

The Intelligent Controls Group (ICG) within the Robot Systems Division at NIST is developing a robot control testbed based on the NASREM Architecture [1, 16]. The purpose of this facility is to test the usefulness of NASREM and various robot control approaches for NASA's Flight Telerobotic Servicer project. The robot control system testbed consists of two major hardware components, the robot arm and the multiprocessor control system.

The robot arm, an RRC K-1607 dextrous manipulator, is a seven degree-of-freedom manipulator which incorporates joint torque loops to minimize the effects of drive train nonlinearities (especially motor friction and harmonic drive compliance), and to reduce the apparent motor inertia seen at the actuator output. The manipulator is supported by its commercial controller which provides an interface which allows an external computer system to issue joint torque commands to the manipulator every 2.5 ms [6].

The testbed multiprocessor control system consists of seven Motorola 680x0 processor boards in a VME backplane. This hardware executes the Primitive and Servo Level code and interfaces to the RRC controller using an IEEE-488 parallel link. Code development is in Ada on a Sun-3 host, which cross-compiles the code and downloads it, along with a run-time kernel, to the target system.

Although there are many software components of the testbed, this discussion will focus primarily on the Servo Level of NASREM as described in [7]. The Servo Level computes the servo control loops that provide the static and dynamic behavior in the small. In autonomous mode, the Servo Level receives commands from the Primitive Level [23]. These commands include set point trajectories which determine the large dynamic motions of the manipulator. In addition, Primitive selects the coordinate system in which the errors will be computed and the Servo algorithm which will be used. The algorithm selection is made from a large number of resident algorithms. Thus, the Servo algorithm can be changed in real-time to meet the needs of the current task. The ability to support multiple algorithms greatly simplified the experimental evaluation of the Cartesian control algorithms described in Section 3. All of these algorithms are implemented in the current version of the Servo Level.

The implementation of the Servo Level as related to Cartesian stiffness control is achieved by the software processes shown in Figure 2. The boxes in the figure represent the software processes; the ovals represent the data which interfaces processes. The processes are labeled according to their functional role in the NASREM Architecture, Sensory Processing (SP), World Modeling (WM), or Task Decomposition (TD) [1, 7, 10]. These processes can be thought of as virtually concurrent processes which are distributed to four of the processors in the multiprocessor system. The processes implement the descriptions given in [7, 10] for the interfaces and functions of the Servo Level. For the most part, the symbols used in the figure correspond to those of the previous section, however, some new symbols will be defined below.

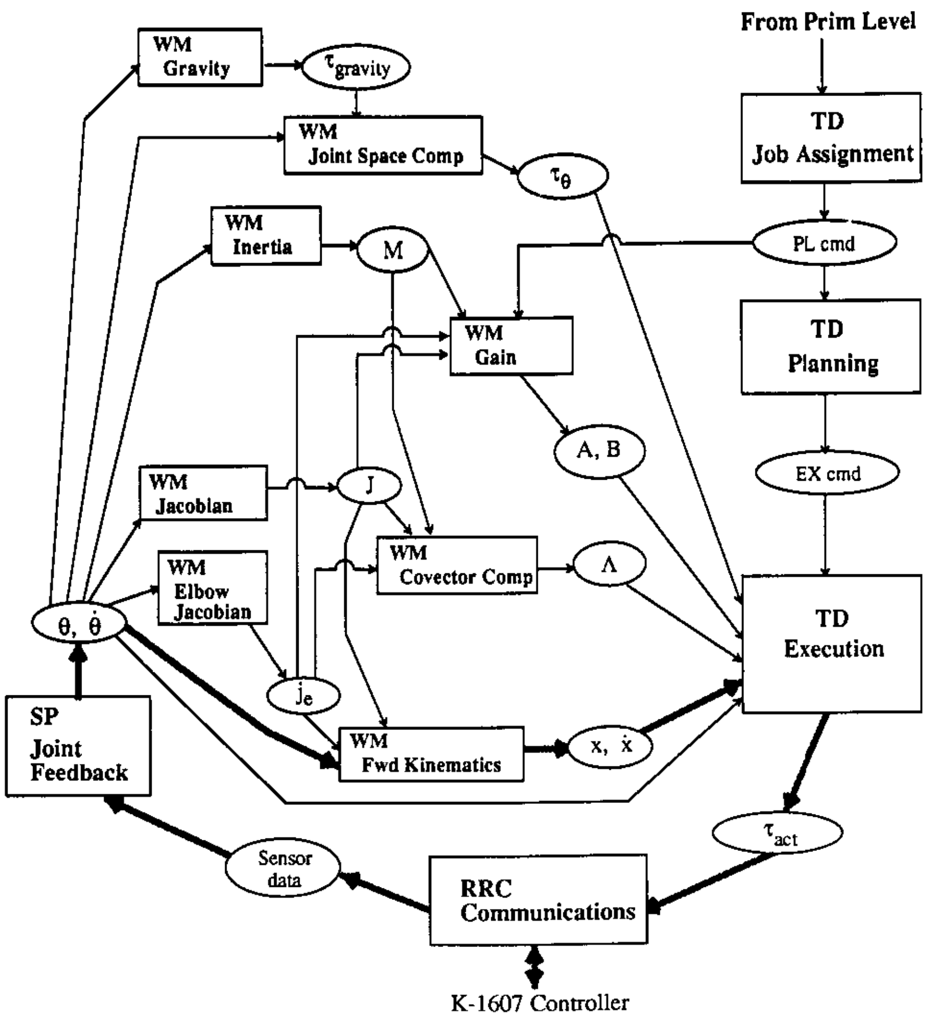

Fig. 2. Servo Level processes and interfaces.

The RRC Communications process communicates with the RRC controller using the IEEE-488 link. The joint position feedback is obtained from the RRC controller every 2.5 ms. Velocity feedback is derived from the position feedback by the Joint Feedback process. The RRC Communications process also reads the command torque from the Execution process interface every cycle and transmits the values to the RRC controller. The RRC controller, running in torque mode, sends the torque commands to the torque control loops active at the joints.

Joint feedback is used by the WM processes to compute elements of the control algorithm. The Gravity process computes the model gravity compensation torque, $\tau_{gravity}$. The Inertia process computes the model inertia matrix, M. The manipulator mass and inertia models were derived from data provided by the manufacturer. The Jacobian process computes the Jacobian J which relates changes in world coordinates or end effector coordinates to joint space. The selection of which Jacobian is to be computed is made based on the Primitive command [7], however this interface is not shown for simplicity. The Elbow Jacobian process computes the Jacobian vector j_e which relates changes in elbow-angle self-motion parameter to joint space. These Jacobians are used by the Forward Kinematics process in converting the joint feedback into Cartesian feedback.

The Joint Space Compensation process generates the total feedforward compensation torque τ_θ. This includes gravity and friction compensation. A simple friction model is used to compensate for viscous and dynamic Coulombic friction. The parameters for this model were determined by experimentation similar to that described in [3]. Significant static Coulombic frictional (stictional) disturbances remain however. Breakaway values of stiction range from about 2 Nm for joint seven to 22 Nm for joint one.

The Covector Compensation process computes the total covector field compensation matrix Λ. In the case of algorithm (6), Λ would just be J^t. For (8), Λ would be MJ^{-1}, an equivalent form of $J^t M_x$. For some algorithms, such as (2) or (11), two model-based values are needed instead of a single covector operator. One value is needed to multiply position error and a second, different value is needed for multiplying the velocity feedback. These two values are essentially generalized position and velocity gains A and B, and are obtained from the diagonal commanded gains by the Gain process. For example, for algorithm (11), A is $J^t K_p$, while B is computed using (12).

The Execution process computes the servo algorithm, e.g. (6) or (11). The process reads the elements of the equation from the interfaces written by the World Modeling processes and the Planning process. The Planning process does any interpolation of Primitive set point commands that may be required. The Job Assignment process receives the Servo commands from the Primitive Level and is responsible for configuring the Servo Level to run the algorithm commanded by Primitive. The interfaces which provide reconfiguration are not shown to simplify the diagram.

The Servo Level processes are distributed to processors as follows. The RRC Communication process runs on one 68030 processor. The Execution, Planning, and Joint Feedback processes run on another 68030 processor. These processes repeat their execution cycle every 2.5 ms. The Job Assignment and Forward Kinematics processes are grouped on another 68030 processor and, again, run on a 2.5 ms cycle. The remaining WM processes are not time-critical and are lumped together on a 68020 processor. Depending on how many of these processes are active for a particular algorithm, the WM processes cycle every 5 to 100 ms. The critical path for Cartesian control is depicted in Figure 2 by the thick arrows. Based on the distribution of processes and their resulting execution times, the loop rate achieved for Cartesian control is about 133 Hz.

5. Static Performance Tests

In order to evaluate the effectiveness of the Cartesian stiffness algorithms which have been implemented, a number of experiments have been performed to determine the range of stiffnesses which may be achieved and the accuracy with which the desired

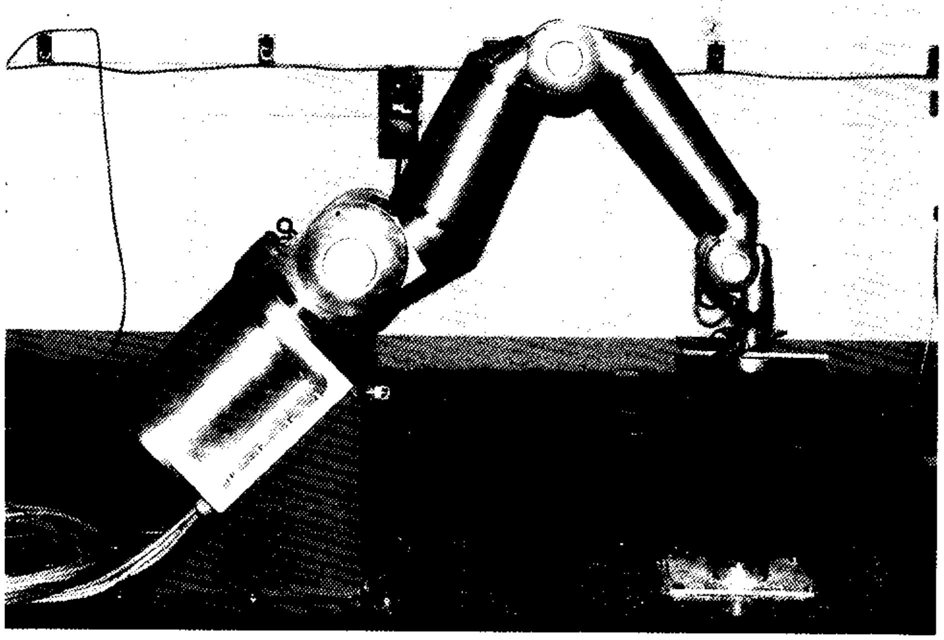

Fig. 3. Experimental setup for stiffness measurement.

stiffness matrix is realized by the manipulator. This section describes these static performance experiments.

5.1. SET-UP AND PROCEDURE

The experimental setup for measuring Cartesian stiffnesses is shown in Figure 3. A rigid pedestal, which consists of aluminum plates welded to the top and bottom of a hollow aluminum cylinder, was fixed to the floor about 1 m in front of the robot pedestal. A JR3 6-axis force/moment sensor is attached to the robot tool plate, and an aluminium bar is bolted to the opposite end of the force sensor.

The test procedure is as follows. First, the robot is commanded very low position and velocity gains to allow the robot to be manually positioned with only gravity compensation torques being applied. The robot is positioned as desired and the bar is then bolted to the pedestal. After bolting, the position and damping gains are set to obtain the desired Cartesian stiffness. Next, the robot is commanded to move to step offsets from the clamped position in each of the six Cartesian directions of the world coordinate system. A positive and a negative offset is used for each direction, resulting in 12 data points for each set of data. The magnitude of the offset is chosen to achieve a given force or moment, so that larger offsets will be used for low gains, and smaller offsets for large gains. The resulting forces are recorded for each position offset. Nominal forces and moments are read before each offset command and subtracted to obtain the net effect of the offset. The forces and moments are rotated to be aligned with world coordinates using

$$f_w = J^t_{T7r} f_s$$

where J^t_{T7r} is the transpose of the differential transform [20] corresponding to the orientation of the world coordinate system with respect to the sensor coordinate system, and f_s is the six-dimensional vector of forces and moments in sensor coordinates. The stiffness matrix is determined row-by-row by performing a multiple linear regression fit to each column of force data using the entire matrix of commanded position offsets.

5.2. RESULTS

Two sets of static test results will be presented. In the first set, the desired stiffness matrix is varied while other conditions are held constant. In the second set, the effect of variations in arm configuration, joint velocity gain selection, and the addition of elbow angle position control are examined for a particular desired stiffness matrix. For all of the static performance tests, the position offsets were chosen to result in forces of 75 N and moments of 15 Nm/rad.

For the first set of tests, the Jacobian-transpose algorithm (6) is used with translational stiffnesses (k_{pt}) of 500, 5000, and 15000 N/m and rotational stiffnesses (k_{pr}) of 50, 500, and 1500 Nm/rad. Joint velocity gains of (600, 600, 400, 250, 70, 70, 50) Nm/rad/s were used. The manipulator configuration for these tests was as shown in Figure 3, with the elbow in a vertical position. The actual stiffness matrices which result from the fit to experimental data for the different desired stiffnesses are given in Table I. Ideally, stiffness matrices would be diagonal with the upper left three elements equal to k_{pt} and the lower right three elements equal to k_{pr}. It is seen that the translational stiffness was obtained to within 10% in most cases, and frequently to within 5%. The rotational stiffness achieved is typically 60–80% of the desired value. For this configuration, desired stiffnesses are realized quite well for a wide range of stiffnesses. The lowest stiffness value, 500 N/m, is comparable to that of a soft sponge, while 15000 N/m is about as stiff as the manipulator is when operated with its commercial controller running individual joint servo loops.

Although many of the off-diagonal terms in the measured stiffness matrices are reasonably small (compared with the diagonal terms), there are several which indicate significant coupling between certain degrees of freedom. Several factors contribute to inaccuracies in the diagonal terms, as well as the presence of off-diagonal terms (which should ideally all be 0). Sources of error fall into two categories, torque generation errors (which are responsible for the difference between the desired and the actual endpoint forces) and force measurement errors (which are responsible for the difference between the actual and the measured endpoint forces).

Ideally, the joint torques corresponding to the Cartesian deflection and the gravity forces on the links would be produced exactly, resulting in the desired endpoint forces.

Table I. Experimentally-determined stiffness matrices for Jacobian-transpose algorithm

k_{pt} (N/m)	k_{pr} (Nm/rad)	$K_{p\,\mathrm{fit}}$
500	50	$\begin{bmatrix} 524 & -15 & -11 & -9.2 & -24 & -1.5 \\ 0.11 & 554 & 50.5 & -4.5 & 7.0 & -0.09 \\ -8.0 & 13 & 483 & 24 & -13 & -26 \\ 3.7 & -17 & -14 & 41 & 1.7 & -1.1 \\ 9.6 & 2.2 & 1.4 & 2.2 & 32 & -2.8 \\ -1.0 & -0.05 & -0.99 & -0.53 & -1.5 & 40 \end{bmatrix}$
5000	500	$\begin{bmatrix} 4950 & -144 & -111 & -105 & -224 & 13 \\ -83 & 5390 & 508 & -19 & 70 & 9.6 \\ -47 & 159 & 4770 & 196 & -117 & -284 \\ 43 & -157 & -147 & 399 & 17 & -12 \\ 73 & 28 & 16 & 19 & 307 & -30 \\ 11 & -0.79 & -1.2 & -7.3 & -22 & 365 \end{bmatrix}$
15000	1500	$\begin{bmatrix} 13900 & -469 & -356 & -258 & -589 & 48 \\ -98 & 15700 & 230 & -135 & 196 & 9.2 \\ -120 & 161 & 14200 & 610 & 353 & -775 \\ 112 & -460 & 405 & 1180 & 45 & 36 \\ 278 & 62 & 43 & 63 & 902 & -111 \\ 78 & -2.7 & -1.2 & -15 & -56 & 1070 \end{bmatrix}$

Constant conditions:
Jacobian-transpose algorithm of Equation (6),
arm configuration as shown in Figure 3,
joint velocity gains = (600, 600, 400, 250, 70, 70, 50) Nm/rad/s.

However, errors in joint torque loop offsets, joint torque conversion constants, and gravity model errors contribute to deviations from the desired torques. The joint torques which are produced interact in a complex, statically-indeterminate manner to determine the static friction torques and the structural deformation of the manipulator and the force sensor. For the most part, errors between expected and measured forces mapped to joint space are within the range of static friction. Contributions to force measurement errors include force sensor noise and coupling. Although a decoupling calibration matrix is used to process the force sensor output, any discrepancy between the calibration matrix and actual sensor coupling will show up directly as off-diagonal terms in the stiffness matrix.

Now that the nominal stiffness characteristics have been established, we can look at the effect of varying some of the conditions held constant in the previous set of tests. First, it is important to know how much the stiffness matrix can change for a different

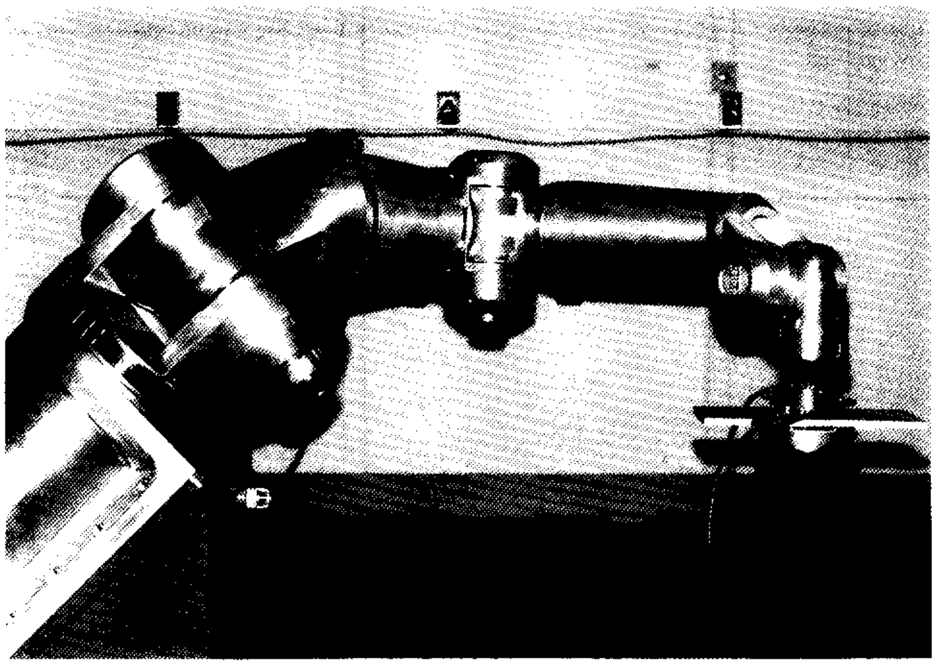

Fig. 4. Alternative manipulator configuration.

elbow configuration. Since the K-1607 is a kinematically redundant manipulator, it is possible for the arm to assume different configurations while maintaining a constant end effector pose. Indeed, with the basic Jacobian-transpose algorithm of (6), there is no control exercised over the nullspace of the manipulator, and the elbow may be pushed around with the end effector bolted to the pedestal. To examine the effect of a different self-motion configuration, the 5000 N/m stiffness test was repeated with the elbow rotated down to a horizontal position as shown in Figure 4. The resulting stiffness matrix fit is given by the first entry in Table II.

It is seen that some coupling terms have become more prominent in this configuration. There is an especially strong coupling between deflections in the x-direction and forces in the y-direction as evidenced by element $K_{p\,\mathrm{fit}}(2, 1)$. The diagonal stiffness terms are similar, for the most part, to those given previously, although the stiffness in the x-direction suffers some additional error in this configuration. In this configuration, the coupling may be more pronounced due to larger gravity model errors (especially for joints 1, 2 and 3) and because of the lack of alignment between joint axes and world coordinate directions.

The second entry in Table II gives the stiffness matrix obtained when augmented Jacobian-transpose control is used with self-motion position and velocity gains of 2000 Nm/rad and 50 Nm/rad/s, again for the robot configuration of Figure 3. Comparing with the second entry of Table I, it is clear that the addition of the nullspace control term did not significantly affect the achieved Cartesian end point stiffness. For

Table II. Experimental stiffness matrices for Cartesian stiffness variations

Variation	$K_{p\,\mathrm{fit}}$
all conditions same as in Table I except arm configuration as shown in Figure 4	$\begin{bmatrix} 3690 & -427 & -43 & 207 & -252 & 8.8 \\ -1580 & 4840 & 153 & 298 & -203 & 121 \\ 819 & 156 & 5060 & 171 & -0.17 & -551 \\ 308 & -69 & -39 & 334 & 33 & -21 \\ -137 & -128 & 52 & 6.4 & 424 & 9.4 \\ 5.4 & -9.4 & -7.0 & -19 & -6.1 & 374 \end{bmatrix}$
augmented Jacobian-transpose algorithm equation (7) self motion position gain = 2000 Nm/rad arm configuration as shown in Figure 3	$\begin{bmatrix} 4770 & -151 & -126 & -88 & -208 & 33 \\ -38 & 5420 & 514 & -0.46 & 71 & 1.6 \\ -53 & 129 & 4770 & 216 & -127 & -274 \\ 35 & -152 & -145 & 398 & 18 & -12 \\ 33 & 23 & 14 & 22 & 320 & -38 \\ 5.2 & -0.51 & -1.1 & -5.6 & -18 & 366 \end{bmatrix}$
dynamic damping algorithm equation (11) arm configuration as shown in Figure 3	$\begin{bmatrix} 5010 & -149 & -123 & -97 & -193 & 25 \\ -20 & 5450 & 475 & -15 & 77 & -3.2 \\ -67 & 115 & 4780 & 233 & -138 & -310 \\ 40 & -167 & -137 & 402 & 16 & -11 \\ 133 & 22 & 16 & 21 & 347 & -23 \\ 4.5 & 0.69 & 0.23 & -5.8 & -6.8 & 366 \end{bmatrix}$

Constant conditions:
$k_{pt} = 5000$ N/m, $k_{pr} = 500$ Nm/rad,
nominal forces = 75 N translation, 15 Nm rotation

this test, the only point of contact was the end effector. If forces were exerted on the elbow, Cartesian stiffness results would be affected.

Experimental stiffness results have also been obtained for other variations of the basic Jacobian-transpose algorithm. The third entry in Table II gives the result when dynamic damping (11) is used with the manipulator in the configuration of Figure 3. Again comparing with the second entry in Table I, it is evident that the method of providing system damping has little effect on the static performance, as expected. Dynamic damping (11) and inertia-scaled damping as in (2), therefore give similar stiffness results to the Jacobian-transpose algorithm with constant joint-space damping (6).

6. Dynamic Performance Tests

In addition to the static performance characteristics discussed above, it is also important to be able to assess the dynamic capabilities of a manipulator under

Cartesian stiffness control. Such measurements as frequency and step response provide insight into the tracking capabilities and suitability of the algorithms for different tasks. In this section, the results of preliminary bandwidth and step response tests for Cartesian stiffness algorithms are presented.

6.1. SET-UP AND PROCEDURE

For the dynamic tests, the pedestal is removed from the robot work volume and the force/moment sensor is removed from the tool plate. The robot is moved to the configuration shown in Figure 3. Frequency response is observed by commanding a sinusoidal motion with a frequency that increases linearly with time. Magnitude and phase response are determined by comparing corresponding peaks in the command and feedback position data for the frequency sweep.

6.2. RESULTS

As an example of the dynamic performance which can be achieved by the system under Cartesian stiffness control, the magnitude and phase response of the manipulator in the world z-direction under dynamic damping control is shown in Figure 5. The amplitude of the commanded sine wave was ± 0.04 m. The plots show curves for two different sets of position gains. The translational stiffness in the z-direction is 3000 N/m for one, and 12 000 N/m for the other. For both sets of plots, the translational stiffness in the x- and y-directions was 12 000 N/m, the rotational stiffnesses were all 1200 Nm/rad and the self motion stiffness was 2000 Nm/rad. A damping ratio of $\zeta = 0.7071$ is used to compute the dynamic damping matrix of (12).

For $k_{pz} = 3000$ N/m, the magnitude-limited bandwidth is just over 1 Hz, and for $k_{pz} = 12\,000$ N/m it is about 2 Hz. The magnitude plot thus indicates that when the stiffness is increased by a factor of 4, the bandwidth is approximately doubled. For this configuration the manipulator response in the z-direction approximates that of a second-order system with a damping ratio of 0.7071, in which bandwidth is proportional to $\sqrt{k_{pz}}$. Note also that increasing the stiffness does not result in a response

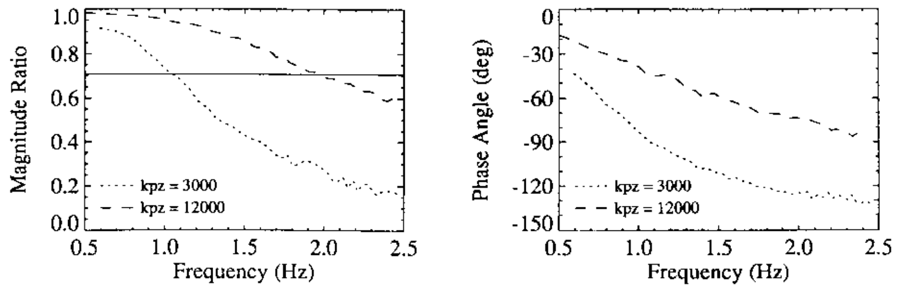

Fig. 5. Magnitude and phase plots for dynamic damping algorithm (world z-direction).

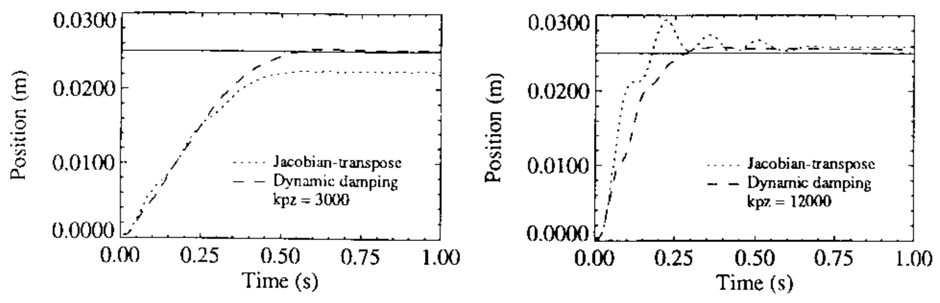

Fig. 6. Response to 0.025 m step (world z-direction).

peak, since the dynamic damping algorithm automatically adjusts the velocity gains in response to the increased position gain.

To look more closely at the effectiveness of dynamic damping, the step response for this algorithm may be compared with that obtained for Jacobian-transpose control with constant joint-space damping. Figure 6 gives the response for both algorithms for a step of 0.025 m in the world z-direction. The joint velocity gains for Jacobian-transpose control are again (600, 600, 400, 250, 70, 70, 50) Nm/rad/s. In the first graph $k_{pz} = 3000$ N/m, and it is seen that the response is more damped with Jacobian-transpose than with dynamic damping. In fact, one or more of the joint velocity gains used with Jacobian-transpose is large enough to cause a significant steady-state error. The second graph in Figure 6 shows what happens when k_{pz} is increased to 12 000 N/m. Now, the Cartesian response under Jacobian-transpose control is underdamped, resulting in am 18% overshoot and oscillation. The time response under dynamic damping control, however, remains as desired.

Of course, since the manipulator Cartesian inertia is not uniform, the dynamic behavior in other directions may be different, even if the same stiffness is used. For example, the Cartesian inertia in the x-direction is quite large for the configuration of Figure 3, which results in a bandwidth of only about 0.3 Hz for a position gain of 12 000 N/m. The dynamic performance will also vary for different end effector positions and for different self-motion configurations at the same end effector position. Changes in payload inertia will also affect the dynamic response.

7. Conclusions

Several algorithms for Cartesian stiffness control have been implemented in a laboratory manipulator control system and experimentally evaluated. The Jacobian-transpose algorithms do a good job of achieving desired translational stiffnesses. Rotational stiffness accuracy, while not quite as good, is probably sufficient for many applications. The algorithms have been found to be stable for a wide range of gain combinations both in free space and in contact with a wide range of environmental stiffneses. End effector stiffnesses ranging from 500 to 14 000 N/m for translation and 50 to 1500 Nm/rad have been achieved. Significant off-diagonal coupling terms were apparent in the measured stiffness matrices, particularly when joint axes were not aligned with world coordinate directions. This is not surprising, considering that we are trying to obtain Cartesian stiffnesses from a 7 degree-of-freedom revolute-joint arm with substantial uncompensated joint friction. It should also be pointed out that although very low stiffnesses may be achieved, as stiffness is decreased frictional disturbances become quite significant. The residual friction not compensated by the friction model is a primary contributor to stiffness and trajectory following errors.

The combination of Cartesian stiffness control with elbow angle specification of self-motion and dynamic damping velocity gain determination represents and excellent package for manipulator control in contact situations. Although substantial computational resources are required to implement all of these features, the result is the ability to specify Cartesian stiffnesses with uniform time response throughout the work volume and to make effective use of the kinematic redundancy of the manipulator. On the other hand, the computational simplicity of Jacobian-transpose algorithms which use constant joint velocity gains makes them a straightforward way to achieve desired Cartesian stiffness characteristics with limited resources. However, selecting appropriate constant velocity gains is not trivial. One approach to solving this problem would be to use the dynamic damping computational off-line to determine constant velocity gains that would give acceptable response for a limited range of stiffnesses in a particular region of the workspace.

Acknowledgements

This work was performed for NASA Goddard Space Flight Center in support of the Flight Telerobotic Servicer Project under Contract No. S-28187-D.

References

1. Albus, J.S., McCain, H.G., and Lumia, R., 1987, NASA/NBS Standard Reference Model Telerobot Control System Architecture (NASREM), NIST Technical Note 1235, NIST, Gaithersburg, MD.
2. Anderson, R.J., 1990, Dynamic damping control: Implementation issues and simulation results, *IEEE Intl. Conf. on Robotics & Automation, Cincinnati*, pp. 68–77.
3. Armstrong, B., 1988, Friction: Experimental determination, modeling, and compensation, *IEEE Intl. Conf. on Robotics & Automation, Philadelphia.*
4. Brady, M., 1982, Trajectory planning, in *Robot Motion: Planning and Control* (eds M. Brady, *et al.*). MIT Press, Cambridge MA.
5. Craig, J.J., 1986, *Introduction to Robotics: Mechanics and Control*, Addision-Wesley, Reading, Mass.
6. Eissmann, P., 1989, *Servo Level Interface Operators Manual*, Robotics Research Corp., Milford, Ohio.
7. Fiala, J., 1988, Manipulator servo level task decomposition, NIST Technical Note 1255, NIST, Gaithersburg, MD.
8. Fiala, J. and Wavering, A., 1990, Implementation of a Jacobian-transpose algorithm, NIST Internal Report, NISTIR 90-4286, National Institute of Standards and Technology, Gaithersburg, MD.
9. Hogan, N., 1985, Impedance control: An approach to manipulation, *J. Dynamic, Systems, Measurement and Control* **107**, March, pp. 1–24.
10. Kelmar, L., 1989, Manipulator servo level world modeling, NIST Technical Note 1258, NIST, Gaithersburg, MD.

11. Khatib, O., 1987, A unified approach for motion and force control of robot manipulators: The operational space formulation, *IEEE J. Robotics Automat.* **RA-3**, No. 1, pp. 43–53.
12. Khosla, P.K., 1987, Choosing sampling rates for robot control, Technical Report, CMU-RI-TR-87-5, Dept. of Electrical and Computer Engineering, CMU.
13. Kim, J.-O., Chung, W.K., and Khosla, P.K., 1989, On using a redundant m..nipulator in force control, CMU Technical Report.
14. Kreutz, K., Long, M., and Seraji, H., 1989, Kinematic functions for the 7 DOF robotics research arm, NASA Conf. on Space Telerobotics, Pasadena, CA.
15. Luh, J.Y.S., Walker, M.W., and Paul, R.P., 1980, Resolved-acceleration control of mechanical manipulators, *IEEE Trans. Automat. Control* **AC-25**, No. 3, pp. 468–474.
16. Lumia, R., Fiala, J., and Wavering, A., 1988, The NASREM robot control system and testbed, *2nd Intl. Symp. Robotics & Automated Manufacturing, Albuquerque, NM.*
17. Miyazaki, F. and Masutani, Y., 1989, Robustness of sensory feedback control based on imperfect Jacobian, *5th Intl. Symp. on Robotics Research, Tokyo, Japan.*
18. Newman, W.S., Dohring, M.E., Farrell, J.D., Eismann, P.H., and Vold, H.I., 1989, Preliminary work in impedance control on a kinematically redundant manipulator, *Proc. ASME Winter Annual Meeting*, ASME, San Francisco.
19. Newman, W.S. and Dohring, M.E., 1990, Augmented impedance control: An approach to compliant control of kinematically redundant manipulators, Case Western Reserve Univ., Center for Automation and Intelligent Systems, Technical Report TR 90-152.
20. Paul, R.P., 1981, *Robot Manipulators: Mathematics, Programming, and Control*, MIT Press, Cambridge, Mass.
21. Salisbury, J.K., 1980, Active stiffness control of a manipulator in Cartesian coordinates, *19th IEEE Conf. on Decision & Control, Albuquerque, NM.*
22. Takegaki, M. and Arimoto, S., 1981, A new feedback method for dynamic control of manipulators, *J. Dynamic Systems, Measurement, and Control* **102**, June, pp. 119–125.
23. Wavering, A., 1988, Manipulator primitive level task decomposition, NIST Technical Note 1256, NIST, Gaithersburg, MD.
24. Whitney, D.E., 1985, Historical perspective and state of the art in robot force control, *IEEE Intl. Conf. on Robotics & Automation, St. Louis.*
25. Maciejewski, A.A. and Klein, C.A., 1989, The singular value decomposition: computational and applications to robots, *Intl. J. Robotics Res.* **8**, No. 6, pp. 63–79.