Proceedings of the 27th Conference
on Decision and Control
Austin, Texas • December 1988

PERFORMANCE IN ADAPTIVE MANIPULATOR CONTROL

Günter Niemeyer and Jean-Jacques E. Slotine
Nonlinear Systems Laboratory
Massachusetts Institute of Technology
Cambridge, MA 02139, USA.

ABSTRACT

While many globally convergent adaptive control algorithms have been developed for linear time-invariant single-input single-output systems, extensions to nonlinear or multi-input systems have rarely been achieved. Yet, in the case of robot manipulators, which represent an important class of nonlinear, time-varying, multi-input multi-output dynamic systems, similar global convergence properties can indeed be obtained. [Slotine and Li, 1986] derived a simple, globally tracking-convergent, direct adaptive manipulator controller, which was demonstrated experimentally in subsequent work. The algorithm was further refined into a "composite" version, whose adaptation law is driven by both tracking error in joint motion and prediction error in joint torques, and therefore represents a combination of a direct and an indirect approach. In this paper, we further explore the performance issues linked to the effective implementation of adaptive manipulator controllers. Specifically, we detail issues of computational efficiency and recursive implementation, the treatment of closed chains, and minimal parametrizations. We also discuss extensions to interactions with mobile environments, whole-arm adaptive manipulation, adaptive impedance control, and adaptive control of spacecraft and space manipulators. The development is illustrated experimentally on a 4-degrees-of-freedom articulated robot arm, and suggests that the range of application of adaptive tracking controllers may extend well beyond adaptation to grasped loads.

1. INTRODUCTION

Adaptive control of linear time-invariant single-input single-output systems has been extensively studied, and a number of globally convergent controllers have been derived. Extensions of the results to nonlinear or multi-input systems have rarely been achieved. Yet, research in adaptive robot control has been very active in recent years (e.g., [Slotine and Li, 1986], [Craig, *et al*., 1986], [Middleton and Goodwin, 1986]; [Hsu, *et al*, 1987], [Sadegh and Horowitz, 1987], [Bayard and Wen, 1987], [Koditschek, 1987], [Slotine and Li, 1987a-e]; [Li and Slotine, 1988], [Walker, 1988]), and led to the remarkable result that global convergence properties similar to those of single-input linear systems can indeed be obtained for robot manipulators, which represent an important class of nonlinear, time-varying, multi-input multi-output dynamic systems. In [Slotine and Li, 1986], we derived a simple, globally tracking-convergent, direct adaptive manipulator controller, which was demonstrated experimentally in subsequent work [Slotine and Li, 1987b]. Based on the observation that parameter information is extracted from tracking error in direct adaptive control and from prediction error in indirect adaptive control, the algorithm was further refined into a "composite" version, whose adaptation law is driven by both tracking error in joint motion and prediction error in joint torques, and therefore represents a combination of a direct and an indirect approach [Slotine and Li, 1987e]. In this paper, we further explore the performance issues linked to the effective implementation of adaptive manipulator controllers, and illustrate the development experimentally on a 4-degrees-of-freedom articulated robot arm.

Following a brief review of the direct adaptive controller of [Slotine and Li, 1986], Section 2 details issues of computational efficiency and recursive implementation, the treatment of closed chains, minimal parametrizations, and the choice of design parameters. Section 3 presents experimental results using the 4-degree-of-freedom whole-arm manipulator developed at M.I.T. by [Salisbury, Townsend, et al., 1988; Townsend, 1988], and demonstrates applications to adaptive whole-arm manipulation. Section 4 discusses extensions to adaptive impedance control, and adaptive control of spacecraft and space manipulators. Section 5 offers brief concluding remarks.

2. RECURSIVE IMPLEMENTATION OF ADAPTIVE MANIPULATOR CONTROLLERS

In this section, we present the main results of this paper in terms of computational efficiency and effective implementation. Section 2.1 briefly reviews the direct adaptive controller of [Slotine and Li, 1986, 1987a]. The recursive implementation of the algorithm is detailed in section 2.2, while section 2.3 discusses minimal parametrizations. A recursive implementation of the composite adaptive controller of [Slotine and Li, 1987e] is developed in section 2.4. The development is then illustrated experimentally, in Section 3.

2.1 Direct Adaptive Control

In this section, we briefly summarize the direct adaptive manipulator control algorithm of [Slotine and Li, 1986, 1987a] (to which the reader is referred for details). In the absence of friction or other disturbances, the dynamics of a rigid manipulator (with the load considered as part of the last link) can be written as

$$\mathbf{H}(\mathbf{q})\,\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\,\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) = \boldsymbol{\tau}$$

where $\mathbf{q}$ is the $n\times1$ vector of joint displacements, $\boldsymbol{\tau}$ is the $n\times1$ vector of applied joint torques (or forces), $\mathbf{H}(\mathbf{q})$ is the $n\times n$ symmetric positive definite manipulator inertia matrix, $\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\,\dot{\mathbf{q}}$ is the $n\times1$ vector of centripetal and Coriolis torques, and $\mathbf{G}(\mathbf{q})$ is the $n\times1$ vector of gravitational torques. The adaptive controller design problem is as follows: given the desired trajectory $\mathbf{q}_d(t)$, and with some or all of the components of the (properly defined) $m\times1$ manipulator mass parameter vector $\mathbf{a}$ being unknown, and with the joint position and velocity measured, derive a control law for the actuator torques, and an adaptation (or estimation) law for the unknown parameters, such that the manipulator joint position $\mathbf{q}(t)$ closely track the desired trajectory $\mathbf{q}_d(t)$.

Define the tracking error measures

$$\tilde{\mathbf{q}} = \mathbf{q} - \mathbf{q}_d \qquad\qquad \mathbf{s} = \dot{\tilde{\mathbf{q}}} + \Lambda\,\tilde{\mathbf{q}} = \dot{\mathbf{q}} - \dot{\mathbf{q}}_r$$

where $\dot{\mathbf{q}}_r = \dot{\mathbf{q}}_d - \Lambda\tilde{\mathbf{q}}$ and Λ is a symmetric positive definite (*s.p.d.*) matrix (or, more generally, a matrix whose eigenvalues are strictly in the right-half complex plane). Also, define $\hat{\mathbf{a}}(t)$ as the current estimate of the constant parameter vector $\mathbf{a}$. Then, using the Lyapunov-like function

$$V(t) = \frac{1}{2}\left[\,\mathbf{s}^T\mathbf{H}\mathbf{s} + \tilde{\mathbf{a}}^T\mathbf{P}^{-1}\tilde{\mathbf{a}}\,\right]$$

and exploiting fundamental physical properties of the system such as conservation of energy and the positive-definiteness of the inertia matrix $\mathbf{H}$, it can be easily shown that the control and adaptation laws

$$\boldsymbol{\tau} = \mathbf{Y}\,\hat{\mathbf{a}} - \mathbf{K}_D\,\mathbf{s} \qquad\qquad \dot{\hat{\mathbf{a}}}(t) = -\,\mathbf{P}\,\mathbf{Y}^T\mathbf{s}$$

where $\mathbf{K}_D$ is a positive definite matrix and the matrix $\mathbf{Y}=\mathbf{Y}(\mathbf{q},\dot{\mathbf{q}},\dot{\mathbf{q}}_r,\ddot{\mathbf{q}}_r)$ is defined by

$$\tilde{\mathbf{H}} = \hat{\mathbf{H}} - \mathbf{H} \qquad\qquad \tilde{\mathbf{C}} = \hat{\mathbf{C}} - \mathbf{C} \qquad\qquad \tilde{\mathbf{G}} = \hat{\mathbf{G}} - \mathbf{G}$$

$$\tilde{\mathbf{H}}(\mathbf{q})\,\ddot{\mathbf{q}}_r + \tilde{\mathbf{C}}(\mathbf{q},\dot{\mathbf{q}})\,\dot{\mathbf{q}}_r + \tilde{\mathbf{G}}(\mathbf{q}) = \mathbf{Y}(\mathbf{q},\dot{\mathbf{q}},\dot{\mathbf{q}}_r,\ddot{\mathbf{q}}_r)\,\tilde{\mathbf{a}}$$

yield $\dot{V}(t) = -\,\mathbf{s}^T\mathbf{K}_D\mathbf{s} \le 0$. Using simple functional analysis arguments, this result can be shown to imply that $\mathbf{s}\rightarrow\mathbf{0}$ as $t\rightarrow\infty$, which in turn implies that $\tilde{\mathbf{q}}$ and $\dot{\tilde{\mathbf{q}}}$ both tend to $\mathbf{0}$, and therefore guarantees the global tracking convergence of the algorithm, independently of the initial parameter estimates. Note that the control law is composed of a "P.D." term $-\mathbf{K}_D\mathbf{s}$ and a particular type of "feedforward" $\mathbf{Y}\,\hat{\mathbf{a}}$.

2.2 Recursive Implementation

A well known problem of manipulator controllers including dynamic compensation is their computational complexity. The computationally-efficient, recursive Newton-Euler algorithm can be used to compute the inverse dynamics in "computed-torque" controllers. However, adaptive sliding controllers, as described above, modify the inverse dynamic expressions by introducing a reference velocity and a reference acceleration, so that standard Newton-Euler algorithms are no longer applicable directly. [Walker, 1988] suggests a solution to this problem by applying the ideas of [Slotine and Li, 1986] to the equations of motion expressed directly in a recursive Newton-Euler form. The resulting algorithm is recursive, hence efficient, and essentially has the same convergence properties as the original adaptive sliding controller. Yet certain differences exist. In particular, there is no joint-space representation of the calculated torques, and therefore no simple "closed-form" of the resulting multi-input controller.

In this section, we present a recursive algorithm whose overall control and adaptation laws are strictly identical to those of the original adaptive sliding controller of [Slotine and Li, 1986]. It thus represents a true recursive implementation of the original sliding controller, which can be easily analysed in closed-form. The algorithm turns out to be a generalization of the Newton-Euler algorithm, which simplifies to the standard version if the reference velocity is replaced by the actual velocity. As in Walker's formulation, a key simplifying element in the derivation is the use of the spatial vector notation of [Featherstone, 1987], which is briefly described in Appendix I.

While the algorithm can be extended easily to closed chains, as is actually done in the experiments, for clarity the derivation will focus on open chains. The links in an open chain are numbered from zero (base) to n (tip) with joint i connecting link i–1 to link i. The following cartesian variables are expressed in the spatial vector notation of [Featherstone, 1987]: $\mathbf{v}$ as velocity (both angular and linear), $\mathbf{w}$ as reference velocity, $\mathbf{e} = \mathbf{v} - \mathbf{w}$ as tracking error, $\mathbf{d}$ as the direction of a joint axis, $\mathbf{I}$ as inertia matrix (including 3x3 inertia , mass , position of center of mass), $\mathbf{F}$ as force (including torque), and $\mathbf{g}$ as gravity vector (linear acceleration pointing down). Subscripts on all variables identify the corresponding link or joint.

2.2.1 Control Law

We now develop the algorithm for calculating the control-torques, focusing on the terms needed for the dynamic compensation. The derivation is based on two properties. First, the relation between matrices $\mathbf{H}$ and $\mathbf{C}$ can be expressed as

$$c_{ki} = \frac{1}{2}\sum_{j=1}^{n}\left(\frac{\partial h_{ki}}{\partial q_j} + \frac{\partial h_{kj}}{\partial q_i} - \frac{\partial h_{ij}}{\partial q_k}\right)\dot{q}_j \tag{1}$$

as can be shown easily from the Lagrangian derivation of the equations of motion [Slotine and Li, 1987a]. Second, the kinetic energy can be written in both joint and cartesian variables:

$$E_{kin} = \frac{1}{2}\,\dot{\mathbf{q}}^T\,\mathbf{H}(\mathbf{q})\,\dot{\mathbf{q}} = \sum_{l=1}^{n}\frac{1}{2}\,\mathbf{v}_l^T\,\mathbf{I}_l\,\mathbf{v}_l \tag{2}$$

Using the spatial vector notation it is easy to give the relation between these two sets of variables:

$$\mathbf{v}_l = \sum_{k=1}^{l}\mathbf{d}_k\cdot\dot{q}_k \quad , \quad \mathbf{w}_l = \sum_{k=1}^{l}\mathbf{d}_k\cdot\dot{q}_{r_k} \tag{3}$$

Substituting this relation in equation (2) and changing the order of summation, we immediately get a relation between the $\mathbf{H}$ matrix and the individual inertia matrices $\mathbf{I}_l$

$$h_{ki} = \mathbf{d}_k^T \sum_{l=\max(k,i)}^{n} \mathbf{I}_l\,\mathbf{d}_i$$

Using equation (1) we can also obtain an expression for the $\mathbf{C}$ matrix

$$c_{ki} = \mathbf{d}_k^T \sum_{l=\max(k,i)}^{n}\left(\mathbf{I}_l\,\mathbf{v}_i\times\mathbf{d}_i + \frac{1}{2}\mathbf{v}_l\times\mathbf{I}_l\,\mathbf{d}_i + \frac{1}{2}\mathbf{d}_i\times\mathbf{I}_l\,\mathbf{v}_l + \frac{1}{2}\mathbf{I}_l\,\mathbf{d}_i\times\mathbf{v}_l\right)$$

The vector $\mathbf{G}(\mathbf{q})$ compensates for the gravitational acceleration and can thus be expressed as

$$G_k = -\mathbf{d}_k^T \sum_{l=k}^{n}\mathbf{I}_l\,\mathbf{g}$$

Combining the above expressions in the original matrix equation we get :

$$\tau_k = \left(\mathbf{H}(\mathbf{q})\,\ddot{\mathbf{q}}_r + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\,\dot{\mathbf{q}}_r + \mathbf{G}(\mathbf{q})\right)_k = \mathbf{d}_k^T\sum_{l=k}^{n}\mathbf{F}_l$$

$$\mathbf{F}_l = \frac{1}{2}\mathbf{v}_l\times\mathbf{I}_l\,\mathbf{w}_l + \frac{1}{2}\mathbf{w}_l\times\mathbf{I}_l\,\mathbf{v}_l + \frac{1}{2}\mathbf{I}_l\,\mathbf{w}_l\times\mathbf{v}_l + \mathbf{I}_l\,\dot{\mathbf{w}}_l - \mathbf{I}_l\,\mathbf{g} \tag{4}$$

$$\text{with} \qquad \dot{\mathbf{w}}_l = \sum_{k=1}^{l}\left(\mathbf{d}_k\,\ddot{q}_{r_k} + \mathbf{v}_k\times\mathbf{d}_k\,\dot{q}_{r_k}\right)$$

To implement this relation, the algorithm is divided into an upward and a downward part, a standard procedure in such recursive imlementations. First, starting from the base, the algorithm works its way up calculating the velocities of each link. Then, on the way down, it calculates and sums up the forces and computes the joint torques accordingly. One important aspect is the choice of coordinate frame. To minimize the numbers of transformations, velocities, inertias, and individual forces are expressed in the coordinate frames attached to their link. As we use the Denavit-Hartenberg convention, which places joint i in frame i–1, directions of joint-axes and summed forces are expreessed in the coordinate frames of the link below. Also it is of advantage to include $\mathbf{g}$ in $\dot{\mathbf{w}}$, as though the whole system were to accelerate upwards in a gravity-free environment. We can summarize the algorithm as follows with $\mathbf{X}_{l-1}|^l$ being the transformation matrix from l–1 to l.

initialize: $\dot{\mathbf{w}}_0 = -\mathbf{g}$

upward: $\mathbf{v}_l = \mathbf{X}_{l-1}|^l\left(\mathbf{v}_{l-1} + \mathbf{d}_l\,\dot{q}_l\right)$

$\mathbf{w}_l = \mathbf{X}_{l-1}|^l\left(\mathbf{w}_{l-1} + \mathbf{d}_l\,\dot{q}_{r_l}\right)$

$\dot{\mathbf{w}}_l = \mathbf{X}_{l-1}|^l\left(\dot{\mathbf{w}}_{l-1} + \mathbf{d}_l\,\ddot{q}_{r_l} + \mathbf{v}_{l-1}\times\mathbf{d}_l\,\dot{q}_{r_l}\right)$

downward: $\mathbf{F}_l = \left(\frac{1}{2}\mathbf{v}_l\times\mathbf{I}_l\,\mathbf{w}_l + \frac{1}{2}\mathbf{w}_l\times\mathbf{I}_l\,\mathbf{v}_l + \frac{1}{2}\mathbf{I}_l\,\mathbf{w}_l\times\mathbf{v}_l + \mathbf{I}_l\,\dot{\mathbf{w}}_l\right)$

$\mathbf{F}_{sum_l} = \mathbf{X}_l|^{l-1}\left(\mathbf{F}_{sum_{l+1}} + \mathbf{F}_l\right)$

$\tau_l = \mathbf{d}_l^T\cdot\mathbf{F}_{sum_l}$

Note that the above equations only account for the dynamic compensation part corresponding to $\boldsymbol{\tau} = \mathbf{H}(\mathbf{q})\ddot{\mathbf{q}}_r + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}}_r + \mathbf{G}(\mathbf{q})$. Friction terms and P.D. terms are calculated in joint-space and added to the above torques. Also, since the inertia matrix $\mathbf{I}$ and joint-axis vector $\mathbf{d}$ as well as some velocities have several zero elements, it is advisable for most terms not to use general cross-product rules, but to take advantage of their structure, i.e., "customize" the algorithm (similarly to e.g. [Koshla and Kanade, 1984]). This will become even more important when the adaptation is added to the algorithm. The major difference with the algorithm presented by [Walker, 1988] and with the standard Newton-Euler algorithm lies in the force calculated for each link; comparing with equation (4) the equivalent equations would be $\mathbf{F}_l = \left(\mathbf{w}_l\times\mathbf{I}_l\mathbf{w}_l + \mathbf{I}_l\dot{\mathbf{w}}_l - \mathbf{I}_l\mathbf{g}\right)$ and $\mathbf{F}_l = \left(\mathbf{v}_l\times\mathbf{I}_l\mathbf{v}_l + \mathbf{I}_l\dot{\mathbf{v}}_l - \mathbf{I}_l\mathbf{g}\right)$.

2.2.2 The Adaptation Law

We now turn our attention to the adaptation law, and again we focus on the dynamic-compensation-terms as above. The parameters involved are all included in the inertia matrices $\mathbf{I}$. An inertia matrix consists of ten parameters (six true inertias, three position components of the center of mass, and mass) placed at specific locations within the matrix. We can thus write

$$\mathbf{I}_l = \sum_{i=1}^{10}\mathbf{R}_i\;a_{l,i}$$

The 'placement' matrices $\mathbf{R}_i$ consist of ones and zeros and give the location of the appropriate parameter within the inertia matrix. Replacing this in the expression (4) we can write torques as a product of coefficients and parameters:

$$\tau_k = \mathbf{d}_k^T\sum_{l=k}^{n}\mathbf{F}_l = \mathbf{d}_k^T\sum_{l=k}^{n}\sum_{i=1}^{10}\mathbf{f}_{l,i}\;a_{l,i}$$

$$\mathbf{f}_{l,i} = \frac{1}{2}\mathbf{v}_l\times\mathbf{R}_i\,\mathbf{w}_l + \frac{1}{2}\mathbf{w}_l\times\mathbf{R}_i\,\mathbf{v}_l + \frac{1}{2}\mathbf{R}_i\,\mathbf{w}_l\times\mathbf{v}_l + \mathbf{R}_i\,\dot{\mathbf{w}}_l - \mathbf{R}_i\,\mathbf{g}$$

Therefore, we can write $\quad (\mathbf{Y}^T\mathbf{s})_{l,i} = \sum_{k=1}^{l} s_k\;\mathbf{d}_k^T\,\mathbf{f}_{l,i} = \mathbf{e}_l^T\;\mathbf{f}_{l,i}$

Assuming that $\mathbf{P}$ is diagonal the final step is to integrate the parameters. In summary the above algorithm has to be expanded into the following equations:

upward: $\mathbf{e}_l = \mathbf{v}_l - \mathbf{w}_l$

downward: $\mathbf{f}_{l,i} = \frac{1}{2}\mathbf{v}_l\times\mathbf{R}_i\,\mathbf{w}_l + \frac{1}{2}\mathbf{w}_l\times\mathbf{R}_i\,\mathbf{v}_l + \frac{1}{2}\mathbf{R}_i\,\mathbf{w}_l\times\mathbf{v}_l + \mathbf{R}_i\,\dot{\mathbf{w}}_l$

$\mathbf{F}_l = \sum_{i=1}^{10}\mathbf{f}_{l,i}\,a_{l,i} \qquad \dot{\hat{a}}_{l,i} = -P_{l,i}\,\mathbf{e}_l^T\,\mathbf{f}_{l,i}$

To compute $\mathbf{f}_{l,i}$ it is recommended to use the structure of the $\mathbf{R}_i$. Since these matrices are extremely sparse, the calculations then reduce to a few multiplications. Finally, note again that these are only the equations for the inertia parameters. Friction terms and P.D. tems are local to each joint, and thus their computational implementation is straightforward.

Note that closed chains can be treated by placing imaginary cuts at different joints, thus receiving an open but branched tree-type structure. The constraint equations are then used to calculate position, velocity and acceleration information for all joints. The algorithm then traverses the tree branch for branch and to sum the forces of sub-trees at a branch-point. Finally the constraint equations are again used to calculate the torques at the motorized joints. The approach of [Walker, 1988] for structuring closed chains and incorporating kinematic constraints can be used straightforwardly to extend the above development to closed chains.

2.3 Minimal Parameterization

As mentioned above, the mass properties of an arbitrary rigid body can be described by 10 parameters, all of which are included in the spatial inertia matrix. If, however, we connect two rigid bodies through a joint and thus restrict their motion with respect to each other, not all 20 parameters are needed to describe the mass properties of the connected system (e.g., [Khosla and Kanade, 1985; An et al., 1985]). [Mayeda, 1988] studies this redundancy analytically for robots with *rotational* joints whose axes are restricted to be either perpendicular or parallel. He concludes that the minimum number of parameters necessary is 7N-4B , where N is the number of links and B the number of parallel joints located at the base (which is at least one). If the first joint is vertical, then another two parameters can be removed.

Using the spatial notation it is possible to simply analyze parameter reductions involved with *both* rotational and translational joints, regardless of their intersection angle, as we now show. It will become clear what combinations of parameters affect the equations of motion and how to eliminate redundant sets of parameters in the implementation.

Let us investigate the combination of two links $i-1$ and i. Redundancy means we can change several parameters in a certain fashion without the system noticing. In the equations of motion, the individual forces are either summed or part of $\mathbf{F}_i$ is used to calculate τ_i. Therefore two conditions have to be satisfied while changing parameters:

$$\mathbf{F}_{i-1} + \mathbf{F}_i = constant \qquad \text{and} \qquad \mathbf{d}_i^T \mathbf{F}_i = constant$$

To study such changes, we subtract an extra inertia matrix $\delta\mathbf{I}$ from link $i-1$ and add it to link i. Doing this keeps the sum of both inertias constant, as required by the above conditions (as can be seen easily by considering the case when the connecting joint does not move). Using the relation between the velocities ($\mathbf{v}_i = \mathbf{v}_{i-1} + \mathbf{d}_i \dot{q}_i$ and equivalent equations) the above conditions can be rewritten in terms of the extra inertia $\delta\mathbf{I}$:

$$\mathbf{v} \times \delta\mathbf{I}\, \mathbf{d}_i + \mathbf{d}_i \times \delta\mathbf{I}\, \mathbf{v} + \delta\mathbf{I}\, \mathbf{v} \times \mathbf{d}_i = 0 \quad , \qquad \delta\mathbf{I}\, \mathbf{d}_i = 0$$
$$\mathbf{d}_i^T \; [\; \mathbf{v} \times \delta\mathbf{I}\, \mathbf{u} + \mathbf{u} \times \delta\mathbf{I}\, \mathbf{v} + \delta\mathbf{I}\, \mathbf{v} \times \mathbf{u} \;] = 0 \quad , \; \mathbf{d}_i^T \; [\; \delta\mathbf{I}\, \mathbf{v} \;] = 0$$

The vectors $\mathbf{u}$ and $\mathbf{u}$ are unknown velocities, which for now are assumed to be unrestricted, and $\mathbf{d}_i$ is the axis connecting both links. Using the standard Denavit-Hartenberg notation, $\mathbf{d}_i$ becomes the z-axis in frame $i-1$, in which we further study the above terms. It follows that $\delta\mathbf{I}$ may contain certain parameters without violating any conditions. Specifically:

For rotational joints	$\delta\mathbf{I} = \{\; J_{xx} = J_{yy} \neq 0 \;,\; p_z \neq 0 \;,\; m \neq 0 \;\}$
For translational joints	$\delta\mathbf{I} = \{\; J_{xx} \neq 0 \;,\; J_{yy} \neq 0 \;,\; J_{zz} \neq 0 \;,\; J_{xy} \neq 0 \;,\; J_{xz} \neq 0 \;,\; J_{yz} \neq 0 \;\}$

Using these results it is possible to cancel 3 parameters for each rotational joint, and 6 parameters for each translational joint. This is simply done by choosing the correct $\delta\mathbf{I}$, which sets several parameters to zero while adjusting others. In effect parameters are lumped into others through the joint which connects them. For rotational joints this procedure corresponds to having a thin rod on the joint-axis and being unable to distinguish to which link it belongs. We recommend lumping parameters upwards for the last joint at the tip of a robot. This guarantees that the parameters of the last link remain intact, so that adding a load to the system will only change a maximum of 10 parameters. However, this possibly leaves one degree of parametric redundancy in the system. For lower joints we recommend lumping downward, as restricted velocities at the base will eliminate further parameters. The exact transformation of $\delta\mathbf{I}$ from frame $i-1$ to frame i is given in appendix II. Note that for rotational joints one has the choice of eliminating J_{xx} or J_{yy}, since only their difference remains.

In the above development we assumed that the velocities components could appear along all axes. This occurs only when studying links connected to the base via two or more non-parallel rotational joints. At the base itself further parameters can be removed. For instance, for links connected to the base through parallel rotational joints all but J_{zz} , p_x , p_y (in the Denavit-Hartenberg notation) can be eliminated. And if the first joint rotates about the vertical axis, link 1 only needs J_{zz}. These reductions also become clear when customizing the algorithm.

2.4 Recursive Implementation of Composite Adaptive Control

In the direct adaptive controller, parameter adaptation is driven only by the motion tracking error $\mathbf{s}$. Now a prediction error $\mathbf{e}$ on the parameter estimates could be obtained, since information on parametric uncertainty also appears linearly in the input (i.e. torque) error. While in practice it is undesirable or unfeasible to directly compare the actual torque with a predicted torque, since the joint accelerations would have to be known, the input error information can be obtained e.g. by using a filtered torque y instead. Indeed, this integrates the torque and thus remove the need for acceleration measurements while remaining linear in parameters, and yields an error relation of the form

$$\mathbf{e} = \hat{\mathbf{y}} - \mathbf{y} = \mathbf{W}(\mathbf{q},\dot{\mathbf{q}})\, \hat{\mathbf{a}} - \mathbf{y} = \mathbf{W}(\mathbf{q},\dot{\mathbf{q}})\, \tilde{\mathbf{a}}$$

where the $n \times m$ matrix $\mathbf{W}(\mathbf{q},\dot{\mathbf{q}})$ is known, as e.g. in the indirect adaptive controller of [Middleton and Goodwin, 1986]. Standard parameter estimation techniques may then be applied based on the above relation.

This duality motivates [Slotine and Li, 1987e] to derive a modified adaptation law, called *composite* adaptation law, which extracts information from *both* the motion tracking error and the torque error:

$$\dot{\hat{\mathbf{a}}}(t) = -\,\mathbf{P}(t)\, (\, \mathbf{Y}^T \mathbf{s} + \mathbf{W}^T \mathbf{R}(t)\, \mathbf{e} \,)$$

where $\mathbf{R}(t)$ is a $n \times n$ uniformly *s.p.d.* weighting matrix, and the adaptation gain $\mathbf{P}(t)$ is a (perhaps time-varying) *s.p.d.* gain matrix determined by the specific parameter estimation approach employed (gradient, least-squares, or e.g. the techniques described in [Li and Slotine, 1988]). As shown in [Slotine and Li, 1987e, 1988b], such a direct/indirect combination not only guarantees the global asymptotic tracking convergence of the system in general, but may yield global exponential convergence of the tracking and estimation errors if the desired joint trajectories are persistently exciting. For instance, defining the Lyapunov-like function V as before, we get with a constant gain matrix $\mathbf{P}$ (gradient estimation)

$$\dot{V} = -\, \mathbf{s}^T \mathbf{K}_D\, \mathbf{s} - \mathbf{e}^T \mathbf{R}\, \mathbf{e} \;\leq\; 0$$

which implies that both $\mathbf{s}$ and $\mathbf{e}$ tend to $\mathbf{0}$. With an exponentially-forgetting least-square gain update

$$\frac{d}{dt}\mathbf{P}^{-1} + \lambda_0\, \mathbf{P}^{-1} = \mathbf{W}^T \mathbf{R}\, \mathbf{W}$$

where $0 < \lambda_o \leq 2\,\lambda$, and using e.g. the " $\mathbf{K}_D = \lambda\, \hat{\mathbf{H}}$ " modification of [Slotine and Li, 1988a], which amounts to letting $\mathbf{K}_D = \mathbf{0}$ and formally replacing $\ddot{\mathbf{q}}_r$ with $\ddot{\mathbf{q}}_r$–As in the control and adaptation laws, we get the stronger result

$$\dot{V} + \lambda_0 V \;\leq\; 0$$

We now turn our attention to the calculation of $\mathbf{e}$ and $\mathbf{W}$. The torque can be predicted using the equations of motion, which are known to be linear in parameters $\mathbf{a}$. The friction torques are represented by $\mathbf{Fr}$.

$$\boldsymbol{\tau} = \mathbf{H}(\mathbf{q})\, \ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\, \dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) + \mathbf{Fr}(\dot{\mathbf{q}}) = \mathbf{Y}_1(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}})\, \mathbf{a}$$

To limit computations, one may want to use a *scalar* prediction error e rather than an $n \times 1$ vector $\mathbf{e}$ as above, at the price of some reduction in the convergence rate of the algorithm. This can be achieved easily, e.g. by replacing the prediction error vector $\mathbf{e}$ on the filtered joint torque vector $\boldsymbol{\tau}$ either by the scalar prediction error e on the filtered power input $\dot{\mathbf{q}}^T \boldsymbol{\tau}$, as suggested in [Slotine and Li, 1987e], or by that on the filtered input/output correlation $\mathbf{s}^T \boldsymbol{\tau}$. While the major theoretical properties of the algorithm are preserved by such choices, the computational advantage of using a scalar prediction error can be understood easily. Using a filtered torque error the adaptation calls for integrating $\mathbf{Y}_1$ by parts to remove the need for $\ddot{\mathbf{q}}$. As is clear from the discussion of the direct adaptive scheme, the $\mathbf{Y}$ matrix and equivalently the $\mathbf{Y}_1$ matrix very easily allow a recursive approach, since their elements are products of expressions which are constant in rows or columns. Integrating such a matrix, however, does not preserve this fortunate property. Thus, in the following, we describe the computational use of the filtered power input instead of the filtered torque. The use of the filtered input/output correlation is analogous.

Writing conservation of energy

$$\dot{\mathbf{q}}^T \tau = \frac{d}{dt}\left(\frac{1}{2}\dot{\mathbf{q}}^T \mathbf{H}(\mathbf{q})\,\dot{\mathbf{q}}\right) + \dot{\mathbf{q}}^T \mathbf{G}(\mathbf{q}) + \dot{\mathbf{q}}^T \mathbf{Fr}(\dot{\mathbf{q}})$$

filtering both sides, and substituting the spatial variables, we get:

$$\mathbf{y} = \int_0^t w(t-r)\,\dot{\mathbf{q}}^T \tau\, dr = \sum_{l=1}^{n} w(0)\,\mathbf{v}_l^T \mathbf{I}_l \mathbf{v}_l(t) - w(t)\,\mathbf{v}_l^T \mathbf{I}_l \mathbf{v}_l(0) +$$

$$\int_0^t \sum_{l=1}^{n} \left(w'(t-r)\,\mathbf{v}_l^T \mathbf{I}_l \mathbf{v}_l - w(t-r)\,\mathbf{v}_l^T \mathbf{I}_l \mathbf{g} \right) + \sum_{k=1}^{n} w(t-r)\,\dot{q}_k Fr_k(q_k)\ dr$$

and therefore we get for the inertia and friction parameters

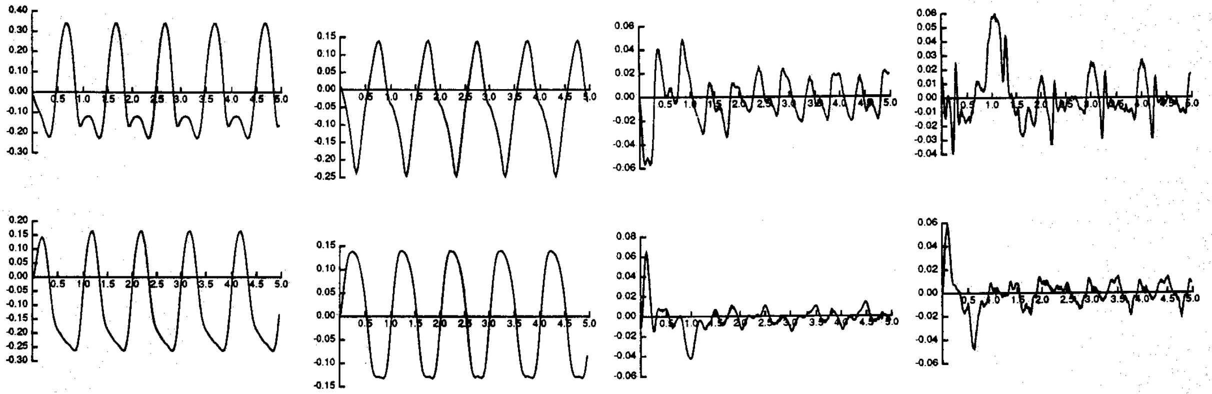
$$W_{l,i} = w(0)\,\mathbf{v}_l^T \mathbf{R}_i \mathbf{v}_l(t) - w(t)\,\mathbf{v}_l^T \mathbf{R}_i \mathbf{v}_l(0) + \int_0^t w'(t-r)\,\mathbf{v}_l^T \mathbf{R}_i \mathbf{v}_l - w(t-r)\,\mathbf{v}_l^T \mathbf{R}_i \mathbf{g}\ dr$$

$$W_k = \int_0^t w(t-r)\,\dot{q}_k fr_k(q_k)\ dr$$

where fr_k is the coefficient multiplying the friction parameter. The only additional information needed for these integrations is the gravitational vector, which now has to be explicitly converted to the different coordinate frames. Calculation of $\mathbf{e}$ and $\mathbf{W}^T\mathbf{e}$ as well as multiplication of ($\mathbf{Y}^T\mathbf{s} + \mathbf{W}^T\mathbf{e}$) by $\mathbf{P}$ are then done by standard matrix and vector schemes. This then results in an algorithm, which besides (optionally) updating $\mathbf{P}$, is still linear in the number of links (for open chains).

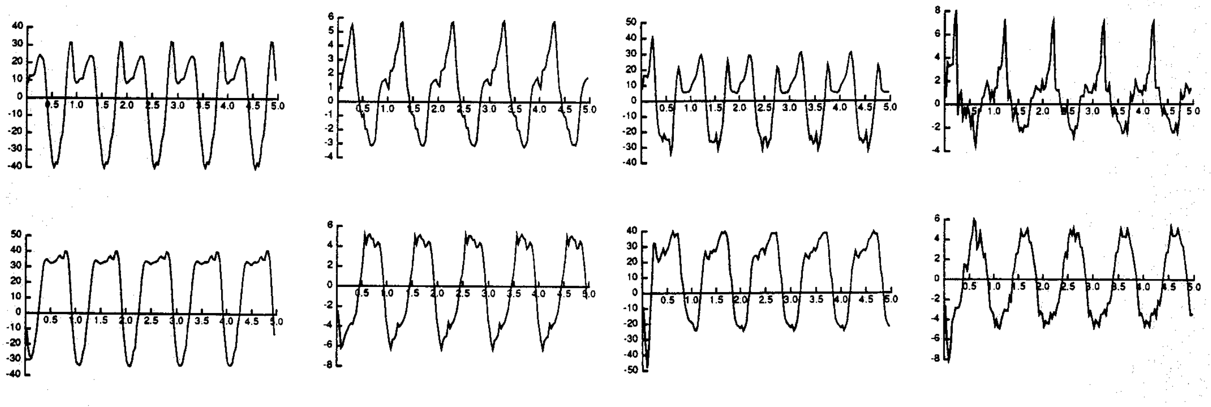
3. EXPERIMENTAL RESULTS

The above development is tested on a 4 d.o.f. cable-driven "whole-arm" manipulator designed at the M.I.T. Artificial Intelligence Laboratory [Townsend, 1988; Salisbury, Townsend, et al., 1988]. The manipulator configuration is similar to that of a human arm, with an extended length of 1 meter. The torques are



Tracking errors $\tilde{q}$, P.D. **Tracking errors $\tilde{q}$, Direct Adaptive Controller**

Tracking errors during the first five seconds of the trajectory, in radians (clockwise: joints 1,3,4,2). The adaptive controller achieves a factor 10 to 20 improvement over the (well tuned) P.D. after a transient of about one second. Furthermore, the maximum tracking error of the adaptive controller during the transient is 2 to 10 times smaller than that of the P.D.



Joint torques τ , P.D. **Joint torques τ , Direct Adaptive Controller**

The control activity and authority of both contollers are comparable, i.e., the P.D. and adaptive controllers generate control torques of similar smoothness and amplitudes. Plots are in N.m.

Figure 1: Experimental results, $q_{di} = (\pi/4)\,(1 - \cos 2\pi t)$

generated by four pulse-width modulated motors, capable of delivering a maximum of 1.5 Nm each. The motors are located close to the base and connected to the lightweight links via cables, introducing a transmission-ratio varying between 1:20 and 1:30. Reduction is done at the joints. Position measurements are made at the motor shafts using resolvers. In order to exploit the wide dynamic range that the manipulator can potentially achieve, the arm is connected to a VME-Bus based multiprocessor system [Narasimhan, *et al.*, 1988], consisting of up to eight 68020 based processor boards, a D/A and A/D board, a parallel interface, as well as other boards. This system is interfaced with the network via a Sun-3 Workstation of Sun Microsystems, Inc. The different processors are used to implement high-speed input/output routines, the basic controller algorithm, the adaptation algorithm, the P-update, and other higher level tasks. One i/o processor performs both the input acquisition and filtering at 4 KHz and the state estimation at 1 KHz, while the controller and adaptation algorithms are executed on separate processors at 100 Hz, with the P.D. component of the control input executed at 200 Hz on the controller processor. The velocity signal on the i/o processor is created by filtered differentiation of the 16 bit position signal (12 bits at the motors). The integration in the adaptation algorithm is performed using a 2nd order Adams-Bashforth scheme. All programs are written in C.

The friction model used for the manipulator consists of viscous and Coulomb friction, where Coulomb friction is allowed to be direction-dependent. We therefore deal with 12 friction parameters in addition to the 24 inertial parameters, and thus we are adapting to 36 unknown values. While dealing with unknown loads only requires in principle to adapt to 10 unknown parameters, the capability of effectively adapting to all 36 parameters allows the range of application of the algorithm to be considerably extended, as discussed later. Also, closed chains are used in the kinematics, as the cable-transmission involves a differential introducing two extra drums.

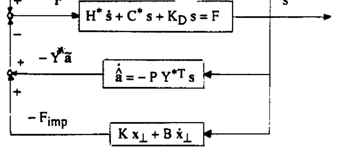
The trajectory performed in the experiment of Figure 1 consists of sinusoids of period 1 second and amplitude equal to $\pm 45^o$ per joint, thus allowing the tip to travel approximately 5 meters per period, with maximum tip velocities and accelerations of 7m/s and 5g. This was done in (i) with a simple well-tuned P.D. (whose performance is, of course, considerably helped by the presence of the transmission ratios), and in (ii) with the adaptive scheme starting as a P.D., i.e. with initially $\hat{\mathbf{a}}(0) = \mathbf{0}$. The plots clearly demonstrate a factor 10 improvement in tracking error after about 1 second when adapting to all 36 parameters. Adaptation dead-zones of 0.15 rad/s in the components of s (or about 1°, with $\lambda = 10$) are used in order to account for residual parametric uncertainty (e.g., inaccurate friction models, torque ripple) and enhance robustness to noise and unmodelled dynamics.

In other experiments, the adaptation was kept on while using the last link to push a large box (of weight about ten times that of the last link, and of volume about 0.2 m^3) located on the floor, at speeds of about .75 m/s. Despite the fact that accurate models of the relative motion of the box with respect to the arm or of the friction between the box and the floor were not developed, this strategy allowed accurate "whole-arm" manipulation of the box (converging again to precisions of about 1° per joint angle after 0.5 second), the presence of the box being in this case largely interpreted by the algorithm as a change in the inertial and friction coefficients of the base link. We believe that this robustness is quite remarkable for a reasonably complex high-performance algorithm, and that it should extend the range of applications of adaptive tracking manipulator controllers well beyond adaptation to grasped loads.

4. EXTENSIONS

4.1 Adaptive Impedance Control

Since many important robotics applications involve *constrained* interaction with some environment (as opposed to the the case of accurately pushing an object on an open surface, as discussed above), it is important to be able to secure contact to an


F_{env}
Environment
+
F
$H^* \dot{s} + C^* s + K_D s = F$
s
−
+
$-Y^*\tilde{a}$
$\dot{\hat{a}} = -P Y^{*T} s$
+
$-F_{imp}$
$K x_\perp + B \dot{x}_\perp$

open surface, as discussed above), it is important to be able to secure contact to an arbitrary surface in a stable and controlled fashion. Conceptually, it is then often easier to rewrite the adaptive algorithm directly in cartesian space, as in [Slotine and Li, 1987c]. However, such an approach introduces extra computations, in particular computations of Jacobian transformation matrices. Now using the spatial notation, the Jacobian matrix consists of the different joint-axis direction vectors $\mathbf{d}_i$. Since these vectors and their derivatives already appear in the joint-space algorithm, the implementation of cartesian-space versions seems computationally promising.

Furthermore, adaptive sliding control has the interesting property of introducing a passive mapping between external forces and the motion tracking error measure **s**. Passivity approaches are useful for contact problems, since the environment is largely unknown yet often passive in its force-velocity relationship. However, in order to have a sliding controller properly interact with an arbitrary environment, it is necessary to redefine the vector s. To do so we use a standard compliance frame, with travels with the end-effector along the surface and whose axes are either parallel or perpendicular to the surface. Removing the proportional terms from tracking errors perpendicular to the surface, i.e. defining the components of s as $s_i = \dot{x}_i - \lambda_i x_i$ parallel to the surface and $s_j = \dot{x}_j$ perpendicularly to the surface, results in a sliding controller that can be coupled to an arbitrary surface in a passive and hence stable fashion. To reinstall proportional feedback perpendicular to the surface and to secure contact we further combine the system with an impedance controller limited to the perpendicular direction. An impedance controller (e.g., [Hogan, 1985]) is automatically passive, since it imitates a spring-damper system. We thus obtain a controller which in principle combines adaptive properties, precise motion, and stable contact.

4.2 Adaptive Control of Spacecraft and Space Manipulators

Approaches similar to those of the manipulator case can be exploited to control other important classes of nonlinear systems [Slotine, 1988; Slotine and Ydtsie, 1988]. Consider for instance, following [Slotine and Di Benedetto, 1988] (to which the reader is referred for details), the attitude control of a rigid spacecraft driven by reaction wheels. The spacecraft is treated as a rigid body whose attitude can be described by two sets of equations, namely, *kinematic* equations, which relate the time derivatives of the angular position coordinates to the angular velocity vector, and *dynamic* equations, which describe the evolution of the angular velocity vector. The development can be directly applied to the case of a spacecraft having rigidly secured a possibly large load of unknown mass properties. The results are also directly applicable to a spacecraft having itself inadequately known mass properties, due to e.g., reconfiguration, fuel variations in the gas-jet systems, thermal deformation, and so on.

Dynamic Equations: Let us first define the (classical) reference frames in which our attitude control problem shall be described. We assume that the control torques are applied through a set of three reaction wheels along orthogonal axes. Based on these axes, we define an arbitrary orthonormal reference frame linked to the spacecraft, which we shall refer to as the *spacecraft frame*. The origin of this frame is not necessarily the center of mass of the system, nor are the axes necessarily the principal axes of the spacecraft. We also assume that an arbitrary *inertial frame* has been defined, with respect to either fixed stars or to a reference that can be considered inertial for the duration of the attitude maneuver (e.g., a space station). Let $\boldsymbol{\omega}$ denote the angular velocity vector of the spacecraft, expressed in the spacecraft frame. The equations describing the evolution of $\boldsymbol{\omega}$ in time may be written as (e.g., [Crouch, 1984])

$$\mathbf{H}\,\dot{\boldsymbol{\omega}} = \mathbf{p} \times \boldsymbol{\omega} + \boldsymbol{\tau}$$

The "inertia" matrix **H** is symmetric positive definite, and can be written $\mathbf{H} = \mathbf{H}^0 - \mathbf{H}^A$, where $\mathbf{H}^0$ is the total (spacecraft with reaction wheels) central inertia matrix, $\mathbf{H}^A$ is the (diagonal) matrix of axial wheels' inertias, and **p** is the total spacecraft angular momentum, all expressed in spacecraft coordinates. Note that since $\boldsymbol{\tau}$ is the torque vector applied to the spacecraft by the reaction wheels, $-\boldsymbol{\tau}$ is the vector of control torques actually applied by the reaction wheels motors.

Kinematic Equations: The angular position of the body may be described in various ways. For example, one can consider the so-called Gibbs vector $\mathbf{q} = \tan(\rho/2)\,\mathbf{e}$ which derives from a transformation of the quaternion parametrization (e.g., [Dwyer and Batten, 1985]). The vector **q** represents the result of a virtual rotation of ρ radians about a virtual unit axis **e**, with reference to the inertial reference frame. In that case, one can write

$$\dot{\mathbf{q}} = \mathbf{J}(\mathbf{q})\,\boldsymbol{\omega} \qquad \text{where} \quad \mathbf{J}(\mathbf{q}) = \frac{1}{2}[\,\mathbf{I} + \mathbf{q}\,\mathbf{q}^T + \mathbf{q}\times\,]$$

and $\mathbf{I}$ is the 3×3 identity matrix. This description is valid for $-\pi < \rho < \pi$. Using this representation, the momentum $\mathbf{p}$ can be expressed as a function of $\mathbf{q}$ by noting that $\mathbf{p} = \mathbf{R}(\mathbf{q})\,\mathbf{p}^I$, where $\mathbf{p}^I$ is the (constant) inertial angular momentum, and the matrix $\mathbf{R}(\mathbf{q})$ represents the coordinate transformation from the inertial frame to the spacecraft frame: $\mathbf{R}(\mathbf{q}) = 2\,(1 + \mathbf{q}^T\mathbf{q})^{-1}\,[\mathbf{I} + \mathbf{q}\mathbf{q}^T - \mathbf{q}\times] - \mathbf{I}$.

Adaptive Attitude Tracking Control: Let us choose state-space coordinates as the components of the vectors $\mathbf{x}$ and $\dot{\mathbf{x}}$ defined by

$$\mathbf{x} = \mathbf{q} \qquad \dot{\mathbf{x}} = \mathbf{J}(\mathbf{q})\,\boldsymbol{\omega}$$

This set of coordinates is well-defined, since the matrix $\mathbf{J}$ remains invertible in the domain of validity of the kinematic representation. Writing the control torque in the form $\boldsymbol{\tau} = \mathbf{J}^T\mathbf{F}$, and differentiating the expression of $\dot{\mathbf{x}}$, the equations of motion can be written, in the new coordinates, as

$$\mathbf{H}^*(\mathbf{x})\,\ddot{\mathbf{x}} + \mathbf{C}^*(\mathbf{x},\dot{\mathbf{x}})\,\dot{\mathbf{x}} = \mathbf{F}$$

with $\mathbf{H}^*(\mathbf{x}) = \mathbf{J}^{-T}\mathbf{H}(\mathbf{x})\,\mathbf{J}^{-1}$ and $\mathbf{C}^*(\mathbf{x},\dot{\mathbf{x}}) = -\,\mathbf{J}^{-T}\mathbf{H}\,\mathbf{J}^{-1}\dot{\mathbf{J}}\mathbf{J}^{-1} - \mathbf{J}^{-T}[\mathbf{p}\times]\mathbf{J}^{-1}$. As in the robotic case, two properties of the above dynamics are exploited in the adaptive controller design. First, the matrix $(\dot{\mathbf{H}}^* - 2\mathbf{C}^*)$ is again *skew-symmetric*, as can be verified easily. Second, the dynamics is *linear* in terms of a properly defined constant parameter vector $\mathbf{a}$. While many such parametrizations are possible, we choose $\mathbf{a}$ to consist of the 6 independent components of the symmetric central inertia matrix $\mathbf{H}$, and of the 3 components of the constant inertial angular momentum $\mathbf{p}^I$. Given the above expressions of $\mathbf{H}^*$ and $\mathbf{C}^*$ and the relation $\mathbf{p} = \mathbf{R}(\mathbf{q})\,\mathbf{p}^I$, the matrices $\mathbf{H}^*$and $\mathbf{C}^*$ are indeed linear in the constant parameter vector $\mathbf{a}$.

It is now straightforward to derive an adaptive attitude tracking controller for the spacecraft, using the same proof as in the robot manipulator case. We assume that the system's state vector, namely $\mathbf{x}$ and $\dot{\mathbf{x}}$, is available (or computable) from measurements, and that the desired $\mathbf{x}_d$, $\dot{\mathbf{x}}_d$, and $\ddot{\mathbf{x}}_d$ are all bounded. Then, using the Lyapunov-like function

$$V(t) = \frac{1}{2}[\,\mathbf{s}^T\mathbf{H}^*\mathbf{s} + \tilde{\mathbf{a}}^T\mathbf{P}^{-1}\tilde{\mathbf{a}}\,]$$

with the vector $\mathbf{s}$ defined as $\mathbf{s} = \dot{\tilde{\mathbf{x}}} + \Lambda\tilde{\mathbf{x}} = \dot{\mathbf{x}} - \dot{\mathbf{x}}_r$, the same proof as in the manipulator case shows that the control law (recall that $\boldsymbol{\tau} = \mathbf{J}^T\mathbf{F}$) and adaptation law

$$\mathbf{F} = \hat{\mathbf{H}}^*(\mathbf{x})\,\ddot{\mathbf{x}}_r + \hat{\mathbf{C}}^*(\mathbf{x},\dot{\mathbf{x}})\,\dot{\mathbf{x}}_r - \mathbf{K}_D\,\mathbf{s} \qquad \dot{\hat{\mathbf{a}}} = -\,\mathbf{P}\,(\,\mathbf{Y}^*\,)^T\mathbf{s}$$

yield $\dot{V}(t) = -\,\mathbf{s}^T\mathbf{K}_D\mathbf{s} \le 0$ and therefore guarantee tracking convergence.

Note that, as in [Slotine in Li, 1987a], the controller can be modified easily to be robust to bounded time-varying disturbances, such as those that may be created by vibrations of flexible appendages (e.g., solar panels). Also, since the state $(\mathbf{x}, \dot{\mathbf{x}})$ of the system is bounded (given the above Lyapunov proof), the singularity of the quaternion-based representation is not reached as long as the rotation corresponding to the desired maneuver does not exceed 180° (if this is not the case, one can simply decompose the control problem in subtasks, and change reference frames between tasks [Dwyer and Batten, 1985]). Also, note that while we are estimating the *central* inertia matrix $\mathbf{H}$ as part of the adaptation process, the actual position of the center of mass of the system is not assumed to be known. In addition, note that we only assumed that $\mathbf{x}$ and $\dot{\mathbf{x}}$ (i.e., the state vector) are available from measurements. If the initial angular velocity $\boldsymbol{\Omega}(0)$ of the reaction wheels about their axes can also be measured, then, using the fact that

$$\mathbf{P}^I = \mathbf{R}^T(\mathbf{x}(0))\,\mathbf{H}^A\,[\,\boldsymbol{\omega}(0) + \boldsymbol{\Omega}(0)\,] + \mathbf{R}^T(\mathbf{x}(0))\,\mathbf{H}\,\boldsymbol{\omega}(0)$$

the above adaptive controller can be expressed (under the mild assumption that the matrix $\mathbf{H}^A$ of axial wheels inertias is known) in terms of only the 6 independent components of the inertia matrix $\mathbf{H}$. Finally, note that, as in the robotic case (see e.g., [Takegaki and Arimoto, 1981]), a simple P.D. controller in $\mathbf{F}$, namely

$$\boldsymbol{\tau} = \mathbf{J}^T\mathbf{F} = -\,\mathbf{J}^T[\,\mathbf{K}_P\tilde{\mathbf{x}} + \mathbf{K}_D\dot{\mathbf{x}}\,]$$

would guarantee stable *position* control (i.e., that $\tilde{\mathbf{x}}$ converges to $\mathbf{0}$ if $\mathbf{x}_d$ is *constant*), as can be shown easily using the Lyapunov function

$$V(t) = \frac{1}{2}[\,\dot{\mathbf{x}}^T\mathbf{H}^*\dot{\mathbf{x}} + \tilde{\mathbf{x}}^T\mathbf{K}_P\tilde{\mathbf{x}}\,] \quad , \quad \dot{V}(t) = -\,\dot{\mathbf{x}}^T\mathbf{K}_D\dot{\mathbf{x}} \le 0$$

and the invariant set theorem.

Robotic spacecraft potentially represent a safe and economical alternative or complement to man in the construction, maintenance, and operation of the space structures to be deployed in the next decade. Furthermore, while such systems can potentially be expected to easily handle objects of masses and sizes comparable to or larger than their own (as, e.g., in releasing a payload from a shuttle orbiter, retrieving a sattelite, or performing docking or construction operations), thanks to weightlessness, such tasks involve by nature large dynamic uncertainties. The above development can in principle be extended easily to include translational control of the spacecraft using gas-jets. Of course, the necessity of averaging on-off gas-jet action presents well-known problems of its own, as does the presence of low-frequency structural modes that often results from the lightness requirements in space components. In particular, while the previous discussion can be extended easily to control the rigid dynamics of manipulators mounted on the spacecraft, e.g. applying the "virtual manipulator" formalism of [Vafa and Dubowsky, 1987], practical implementation will require flexibility issues to be explicitly addressed.

5. CONCLUDING REMARKS

We believe that advanced control concepts have exceptional potential in the development of robust, reliable, high-performance robotic systems. We hope studies such as the one presented here will accelerate such implementations in common industrial and scientific practice, and thus allow the full potential of ever increasing computational capabilities to be exploited effectively.

Acknowledgements: We are grateful to Ken Salisbury and Bill Townsend for allowing us to test the above results on the fine manipulator they developed, and for stimulating discussions. We would also like to thank Brian Eberman for his help in the implementation. This work was supported in part by a generous student Grant from the Perry Foundation, and in part by Grant 8803767-MSM from the National Science Foundation. Development of the arm and control hardware was supported by Grants N00014-86-K-0685 and N00015-85-K-0214 from the Office of Naval Research.

REFERENCES

An, C.H., Atkeson, C.G. and Hollerbach, J.M., 1985. Estimation of inertial parameters of rigid body links of manipulators,*I.E.E.E. Conf. Decision and Control*, Fort Lauderdale.

Asada, H., and Slotine, J.J.E., 1986. Robot Analysis and Control, *Wiley*.

Bayard, D.S., and Wen, J.T., 1987. Simple Adaptive Control Laws for Robotic Manipulators, *Proceedings of the Fifth Yale Workshop on the Applications of Adaptive Systems Theory*.

Craig, J.J., Hsu, P. and Sastry, S., 1986. Adaptive Control of Mechanical Manipulators, *I.E.E.E. Int. Conf. Robotics and Automation*, San Francisco.

Crouch, P., 1984. Spacecraft Attitude Control and Stabilization: Application of Geometric Control Theory to Rigid Body Models, *I.E.E.E. Trans. Autom. Control*, AC-29 (4).

Dwyer, T.W.A., and Batten, A.L., 1985. Exact Spacecraft Detumbling and Reorientation Maneuvers with Gimbaled Thrusters and Reaction Wheels, *A.A.S. J. of the Astronautical Sciences*, vol.3.

Featherstone, R., 1987. Robot Dynamics Algorithms, *Kluwer Academic Publishers*.

Hsu, P., Sastry, S. , Bodson, M. and Paden, B. 1987. Adaptive Identification and Control of Manipulators Without Joint Acceleration Measurements, *I.E.E.E. Int. Conf. Robotics and Automation*, Raleigh, NC.

Khosla, P.,and Kanade, T., 1985. Parameter Identification of Robot Dynamics, *I.E.E.E. Conf. Decision and Control*, Fort Lauderdale.

Koditschek, D.E., 1987. Adaptive Techniques for Mechanical Systems, *Proceedings of the Fifth Yale Workshop on the Applications of Adaptive Systems Theory*.

Li, W. and Slotine, J.J.E. 1987. Parameter Estimation Strategies for Robotic Aplications. *A.S.M.E. Winter Annual Meeting*, Boston, MA.

Li, W. and Slotine, J.J.E. 1988a. Indirect Adaptive Robot Control, *5th I.E.E.E. Int. Conf. Robotics and Automation*, Philadelphia, PA. Revised version with complete convexity proof now in preprint.

Mayeda, H., Yoshida, K., and Osuka, K., 1988. Base Parameters of Manipulator Dynamic Models, *5th I.E.E.E. Int. Conf. Robotics and Automation*, Philadelphia, PA.

Middleton, R.H. and Goodwin, G.C. , 1986. Adaptive Computed Torque Control for Rigid Link Manipulators, 25th *I.E.E.E. Conf. on Dec. and Contr.*, Athens, Greece.

Sadegh, N., and Horowitz, R., 1987. Stability Analysis of an Adaptive Controller for Robotic Manipulators. *I.E.E.E. Int. Conf. Robotics and Automation*, Raleigh, NC.

Salisbury, J.K., et al., 1988. Preliminary Design of a Whole-Arm Manipulator System, *I.E.E.E. Int. Conf. Robotics and Automation*, Philadelphia, PA.

Slotine, J.J.E., 1988. Putting Physics in Control, I.E.E.E. Control Systems Magazine, 8-5.

Slotine, J.J.E., and Di Benedetto, M.D., 1988. Hamiltonian Adaptive Control of Spacecraft, MIT-NSL-880603 Report, May 1988, submitted to I.E.E.E. Trans. Autom. Control.

Slotine, J.J.E., and Li, W., 1986. On The Adaptive Control of Robot Manipulators, *A.S.M.E. Winter Annual Meeting*, Anaheim, CA.

Slotine, J.J.E., and Li, W., 1987a. On the Adaptive Control of Robot Manipulators, *Int. J. Robotics Res.*, vol. No.3

Slotine, J.J.E., and Li, W., 1987b. Adaptive Robot Control, A Case Study, *I.E.E.E. Int. Conf. Robotics and Automation,* Raleigh, NC.

Slotine, J.J.E., and Li, W., 1987c. Adaptive Strategies in Constrained Manipulation, *I.E.E.E. Int. Conf. Robotics and Automation*, Raleigh, NC.

Slotine, J.J.E. and Li, W., 1987d. Theoretical Issues In Adaptive Manipulator Control. In *the Proceedings of the Fifth Yale Workshop on Applications of Adaptive Systems Theory.*

Slotine, J.J.E. and Li, W., 1987e. Adaptive Robot Control - A New Perspective, *I.E.E.E. Conf. Decision and Control* L.A., CA.

Slotine, J.J.E., and Li, W., 1988a. Adaptive Manipulator Control: A Case Study, *I.E.E.E. Trans. Autom. Control* 33, 11.

Slotine, J.J.E., and Li, W., 1988b. Composite Adaptive Robot Control, MIT-NSL-880501, submitted to Automatica.

Slotine, J.J.E., and Ydtsie, B.E., 1988. Control of Nonlinear Chemical Processes: A Physically-Motivated Approach, submitted to *Automatica.*

Takegaki, M., and Arimoto, S., 1981. A New Feedback Method for Dynamic Control of Manipulators, *A.S.M.E. J. Dynamic Systems, Measurement, and Control,* 102.

Townsend, W., 1988. "The Effect of Transmission Design on Force-Controlled Manipulator Performance", Ph.D. Thesis, M.I.T., Department of Mechanical Engineering.

Vafa, Z., and Dubowsky, S., 1987. On the Dynamics of the Manipulators in Space Using the Virtual Manipulator Approach, *Proc. of the 1987 I.E.E.E. Int. Conf. on Robotics and Automation*, Raleigh, NC.

Walker, M.W., 1988. An Efficient Algorithm for the Adaptive Control of a Manipulator, *5th I.E.E.E. Int. Conf. Robotics and Automation*, Philadelphia, PA.

Appendix I: A Brief Introduction To Spatial Vector Notation

Individual points in space have three degrees of freedom and are well described by three-dimensional vectors, which we shall denote in this appendix by arrows ($\vec{v}$). Rigid bodies, however, have six degrees of freedom, and are normally described by sets of two three-dimensional vectors corresponding to linear and angular quantities. The spatial notation of [Featherstone, 1987] combines these two vectors into a single six-dimensional vector. While this obviously reduces the number of equations, it also reduces their complexity, since the coupling between linear and angular quantities can be taken care of automatically. Spatial vectors will be denoted as bold characters. For instance

$$\mathbf{v}|^1 = \begin{bmatrix} \vec{\omega}|^1 \\ \vec{v}|^1 \end{bmatrix} \qquad \text{or} \qquad \mathbf{F}|^1 = \begin{bmatrix} \vec{F}|^1 \\ \vec{\tau}|^1 \end{bmatrix}$$

The spatial velocity $\mathbf{v}|^1$ combines the angular velocity and linear velocity measured at the origin of and expressed in the coordinate frame 1. Changing coordinate frame from 1 to 2 involves shifting the origin from $\vec{r}_1$ to $\vec{r}_2$ and rotating according to the 3x3 rotation matrix $\mathbf{R}_1|^2$. As in standard vector notations this matrix consists of frame 1 base vectors expressed in frame 2. The resulting transformation matrix $\mathbf{X}_1|^2$ in spatial notation is

$$\mathbf{v}|^2 = \begin{bmatrix} \vec{\omega}|^2 \\ \vec{v}|^2 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_1|^2 & 0 \\ \vec{r}_1^{\,2}|^2 \times \mathbf{R}_1|^2 & \mathbf{R}_1|^2 \end{bmatrix} \cdot \begin{bmatrix} \vec{\omega}|^1 \\ \vec{v}|^1 \end{bmatrix} = \mathbf{X}_1|^2\, \mathbf{v}|^1$$

where $\qquad \vec{r}_1^{\,2} = \vec{r}_1 - \vec{r}_2$

The notation $\vec{r}\times$ denotes a matrix which multiplied by a vector $\vec{s}$ is equivalent to the cross-product $\vec{r} \times \vec{s}$. This is also true for the spatial cross-product, which is defined as follows.

$$\mathbf{a} \times \mathbf{b} = \begin{bmatrix} \vec{m}\times & 0 \\ \vec{n}\times & \vec{m}\times \end{bmatrix} \cdot \mathbf{b} \qquad \text{with} \quad \mathbf{a} = \begin{bmatrix} \vec{m} \\ \vec{n} \end{bmatrix}$$

The transpose operator is defined differently from what one might expect, so that the product $\mathbf{F}^T\!\cdot\mathbf{v}$ be equal to power. Nevertheless both operators have the same properties as their counterparts in standard vector notation.

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}^T = \begin{bmatrix} D^T & B^T \\ C^T & A^T \end{bmatrix} \quad , \quad \begin{bmatrix} \vec{m} \\ \vec{n} \end{bmatrix}^T = [\ \vec{n}^T\ \vec{m}^T\ ]$$

The transformation matrices **X** have the following properties:

$$\mathbf{X}_c|^a = \mathbf{X}_b|^a \cdot \mathbf{X}_c|^b \quad , \quad (\mathbf{X}_c|^a)^T = \mathbf{X}_a|^c \quad , \quad \frac{d}{dt}\mathbf{X}_c|^a = \mathbf{v}_c^a|^a \times \mathbf{X}_c|^a$$

To express rigid body dynamics we also need to define an inertia matrix **I**, which is a 6x6 matrix mapping spatial velocity to spatial momentum. It consists of the standard 3x3 inertia matrix **J**, the mass m, and the product of mass and position of the center of mass $\vec{p}$.

$$\mathbf{I}_c|^c = \begin{bmatrix} -\vec{p} & m\cdot\mathbf{1} \\ \mathbf{J} & \vec{p} \end{bmatrix} \qquad \text{with } \mathbf{1} \text{ being the 3x3 unity matrix}$$

$$\mathbf{I}_c|^a = \mathbf{X}_c|^a \cdot \mathbf{I}_c|^c \cdot \mathbf{X}_a|^c \quad , \quad \frac{d}{dt}\mathbf{I}_c|^a = \mathbf{v}_c^a|^a \times \mathbf{I}_c|^a - \mathbf{I}_c|^a\, \mathbf{v}_c^a|^a \times$$

With these tools it is possible to write the equation of motion for an arbitrary rigid body. These include both Newton's and Euler's equation and account for any coupling automatically. This also avoids problems such as appropriate choices of the coordinate frame origin. We have

$$\mathbf{F} = \frac{d}{dt}(\mathbf{I}\,\mathbf{v}) = \mathbf{I}\,\mathbf{a} + \mathbf{v} \times \mathbf{I}\,\mathbf{v} \quad \text{where } \mathbf{a} = \frac{d}{dt}\mathbf{v} \text{ is the acceleration.}$$

Appendix II: Transformations for Lumping Parameters

The following equations determine how parameters may be moved from link i-1 to link i. The Denavit-Hartenberg parameters (a , α , d , θ) involved correspond to joint i, which lies in frame i-1 between link i-1 and link i.

Rotational Joints:

$\boldsymbol{\delta}\mathbf{I}|^{i-1}$ $\quad \delta J_{xx} = \delta J_{yy} = J \;\; \delta p_z = p_z \;\; \delta m = m$

$\boldsymbol{\delta}\mathbf{I}|^{i}$

$$\delta J_{xx} = J + m\,d^2 - 2\,d\,p_z$$
$$\delta J_{yy} = (\,J + m\,d^2 - 2\,d\,p_z\,)\cos^2(\alpha) + m\,a^2$$
$$\delta J_{zz} = (\,J + m\,d^2 - 2\,d\,p_z\,)\sin^2(\alpha) + m\,a^2$$
$$\delta J_{xy} = (\,a\,p_z - m\,a\,d\,)\sin(\alpha) \;\; \delta J_{xz} = (\,a\,p_z - m\,a\,d\,)\cos(\alpha)$$
$$\delta J_{yz} = -(\,J + m\,d^2 - 2\,d\,p_z\,)\sin(\alpha)\cos(\alpha)$$
$$\delta p_x = -m\,a \;\; \delta p_y = (\,p_z - m\,d\,)\sin(\alpha) \;\; \delta p_z = (\,p_z - m\,d\,)\cos(\alpha)$$

Translational Joints:

$\boldsymbol{\delta}\mathbf{I}|^{i-1}$

$$\delta J_{xx} = J_{xx} \;\; \delta J_{yy} = J_{yy} \;\; \delta J_{zz} = J_{zz}$$
$$\delta J_{xy} = J_{xy} \;\; \delta J_{xz} = J_{xz} \;\; \delta J_{yz} = J_{yz}$$

$\boldsymbol{\delta}\mathbf{I}|^{i}$

$$\delta J_{xx} = J_{xx}\cos^2(\theta) + 2\,J_{xy}\sin(\theta)\cos(\theta) + J_{yy}\sin^2(\theta)$$
$$\delta J_{yy} = (\,J_{xx}\sin^2(\theta) - 2\,J_{xy}\sin(\theta)\cos(\theta) + J_{yy}\cos^2(\theta)\,)\cos^2(\alpha) - 2\,(\,J_{xz}\sin(\theta) - J_{yz}\cos(\theta)\,)\sin(\alpha)\cos(\alpha) + J_{zz}\sin^2(\alpha)$$
$$\delta J_{zz} = (\,J_{xx}\sin^2(\theta) - 2\,J_{xy}\sin(\theta)\cos(\theta) + J_{yy}\cos^2(\theta)\,)\sin^2(\alpha) + 2\,(\,J_{xz}\sin(\theta) - J_{yz}\cos(\theta)\,)\sin(\alpha)\cos(\alpha) + J_{zz}\cos^2(\alpha)$$
$$\delta J_{xy} = (\,-(\,J_{xx} - J_{yy}\,)\sin(\theta)\cos(\theta) + J_{xy}(\,\cos^2(\theta) - \sin^2(\theta)\,)\,)\cos(\alpha) + (\,J_{xz}\cos(\theta) + J_{yz}\sin(\theta)\,)\sin(\alpha)$$
$$\delta J_{xz} = (\,(\,J_{xx} - J_{yy}\,)\sin(\theta) - J_{xy}(\,\cos^2(\theta) - \sin^2(\theta)\,)\,)\sin(\alpha) + (\,J_{xz}\cos(\theta) + J_{yz}\sin(\theta)\,)\cos(\alpha)$$
$$\delta J_{yz} = -(\,J_{xx}\sin^2(\theta) - 2\,J_{xy}\sin(\theta)\cos(\theta) + J_{yy}\cos^2(\theta)\,)\sin(\alpha)\cos(\alpha) - (\,J_{xz}\sin(\theta) + J_{yz}\cos(\theta)\,)(\,\cos^2(\alpha) - \sin^2(\alpha)\,) + J_{zz}\sin(\alpha)\cos(\alpha)$$