AN EXPERIMENTAL STUDY OF HIERARCHICAL CONTROL LAWS FOR GRASPING AND MANIPULATION USING A TWO-FINGERED PLANAR HAND

by

Karin Hollerbach, Richard M. Murray,
and S. Shankar Sastry

Memorandum No. UCB/ERL M91/90

4 October 1991

AN EXPERIMENTAL STUDY OF HIERARCHICAL CONTROL LAWS FOR GRASPING AND MANIPULATION USING A TWO-FINGERED PLANAR HAND

by

Karin Hollerbach, Richard M. Murray,
and S. Shankar Sastry

Memorandum No. UCB/ERL M91/90

4 October 1991

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

AN EXPERIMENTAL STUDY OF HIERARCHICAL CONTROL LAWS FOR GRASPING AND MANIPULATION USING A TWO-FINGERED PLANAR HAND

by

Karin Hollerbach, Richard M. Murray,
and S. Shankar Sastry

Memorandum No. UCB/ERL M91/90

4 October 1991

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720.

An Experimental Study of Hierarchical Control Laws for Grasping and Manipulation using a Two-Fingered Planar Hand

Karin Hollerbach
Department of Electrical Engineering and Computer Sciences
and Bioengineering Graduate Group
University of California Berkeley, CA 94720

Richard M. Murray
Department of Mechanical Engineering
California Institute of Technology Pasadena, CA 91125

S. Shankar Sastry
Department of Electrical Engineering and Computer Sciences
University of California Berkeley, CA 94720

1992 ICRA

Abstract

We compare the performance of hierarchical and single-level controllers in a grasping context, and we conclude that for rapid, planar grasping motions of heavy objects the performance of a hierarchical control structure is superior to that of the two single-level controllers tested. For slow movements of lighter objects the performance of the three controllers is similar; with an increase in movement speed and object mass, however, the hierarchical structure becomes increasingly important. Although the theory discussed here applies to grasping problems of arbitrary complexity, we focus on planar, two-fingered grasping for the sake of clarity and to simplify implementation and experimental testing of the proposed control algorithms.

1 Introduction

Interest in complex, multi-fingered robotic hands has seen an increase in the last few years, as advanced designs and a rigorous theory used to describe

them have been developed. Early research on multi-fingered hands tended to focus on the description of hand kinematics [13] and on the generation of stable grasps [20]. Later, researchers started to develop simple algorithms used in grasping control of single hands [4, 12, 15, 16] as well as control of associated groups of robots, cooperating to perform a single task [1, 2, 11, 19].

Advances in multi-fingered hand design are readily apparent when perusing the literature: the more well-known designs include the Utah-MIT hand [8], the Stanford/JPL hand [28], the NYU hand [5]. However, the problem of overcoming the computational burden associated with control of some of the more complicated hand designs has not been adequately addressed. In response to the computational difficulties surrounding control of complicated robotic hands we seek an approach to multi-fingered hand control that significantly reduces the computational burden placed on the controller while improving the grasping performance of the hand, and that is essentially design-independent. A beginning in the development of such an approach was made by Deno *et al.* in [7], and the experimental results presented here may be considered an implementation of the basic philosophy of hierarchical robot control as laid out there.

A motivating factor in this study of hierarchical grasping control is found in the highly effective, adaptable mammalian neuro-muscular control system and its hierarchy of spinal and cortical neural signals and control loops [14, 23]. Time delays inherent in biological motor systems indicate that control is likely to be hierarchical, occurring at many different levels of the central nervous system. For the same reasons, communication and computation delays make hierarchical controllers an attractive method for providing high system bandwidth while coordinating many degrees of freedom. These motivations are not limited to robotics: there is a large literature concerned with the use of the related, though not directly applicable, theory of decentralized control for general dynamic systems [21, 25].

Centralized control has been defined as a case in which every sensor's output influences every actuator [25]. The study of large scale systems led to a number of results concerning weakly coupled, systems, with decentralized control, and hierarchical systems, with controllers exhibiting a separation of time-scales. Graph decomposition techniques permitted the isolation of sets of states, inputs, and outputs that were weakly coupled. This decomposition simplified stability analyses and controller design. In multi-processor control systems, decentralized control is often mandated by restrictions on the communication rates between processors. Hierarchical controllers, as

exemplified by Clark's HIC [3], limit communication to adjacent levels of the hierarchy. HIC is an operating system intended to manage servo loops found in robot controllers in which Clark emphasizes distributed processing and interprocessor communication.

In this paper we compare the performance in a planar grasping task of a hierarchical control algorithm with that of single-level controllers containing no hierarchical structure. In doing so, we first selectively review the dynamics and control of robot systems and discuss the natural hierarchical structure that arises in biological grasping systems and may be created in robotic multi-fingered hands. We show that in the hierarchy used here, we have a control scheme that, for fast low-level controllers, induces the tracking error to converge to zero. We then proceed to describe the methods and the hardware used in the experimental comparison of the controllers' performance and subsequently present the experimental evidence demonstrating the superior performance in rapid, planar movements of heavy objects of the hierarchical controller. Finally, we discuss the merits of the hierarchical approach to the control of grasping and draw parallels between grasping in biological systems and in multi-fingered robotic hands.

2 Hierarchical Control of Grasp Dynamics

In this section we selectively review the dynamics and control of robot systems. Following the results of [17], we show that there is a natural hierarchical structure of the control system for multi-fingered hands that mirrors the physical structure of the system.

2.1 Grasp dynamics

The dynamics for a robot manipulator with joint angles $\theta \in \mathbb{R}^n$ and actuator torques $\tau \in \mathbb{R}^n$ can be derived using Lagrange's equations and written in the form

$$M(\theta)\ddot{\theta} + C(\theta, \dot{\theta})\dot{\theta} + N(\theta, \dot{\theta}) = \tau \tag{1}$$

where $M(\theta)$ is a positive definite inertia matrix and $C(\theta, \dot{\theta})\dot{\theta}$ is the Coriolis and centrifugal force vector. The vector $N(\theta, \dot{\theta}) \in \mathbb{R}^n$ contains all friction and gravity terms, and the vector $\tau \in \mathbb{R}^n$ represents generalized forces in the θ coordinate frame. For systems of this type, it can be shown that $\dot{M} - 2C$ is a skew symmetric matrix with proper choice of C (see [27]).

Robot systems with contact constraints, such as a multi-fingered hand grasping an object, can be represented by dynamic equations in the same form as equation (1). In the case of a multi-fingered hand grasping a box, we let θ be the vector of joint angles and x be the vector describing the position and orientation of the box. With these definitions, the grasping constraint may be written as

$$J(q)\dot{\theta} = G^T(q)\dot{x}, \tag{2}$$

where $q = (\theta, x) \in \mathbb{R}^m \times \mathbb{R}^n$, J is the Jacobian of the finger kinematic function and G is the "grasp map" for the system. We will assume that J is bijective in some neighborhood and that G is surjective. This form of constraint can also be used to describe a wide variety of other systems, including grasping with rolling contacts, surface following and coordinated lifting. For ease of exposition, we also assume that there exists a forward kinematic function between θ and x; that is, the constraint is holonomic. A more complete derivation of grasping kinematics can be found in [18].

To include velocity constraints in the dynamics formulation, we again appeal to Lagrange's equations. Following the approach in [18], the equations of motion for our constrained system can be written as

$$\tilde{M}(q)\ddot{x} + \tilde{C}(q,\dot{q})\dot{x} + \tilde{N}(q,\dot{q}) = F_e \tag{3}$$

where

$$\begin{aligned}
\tilde{M} &= M + GJ^{-T}M_\theta J^{-1}G^T \\
\tilde{C} &= C + GJ^{-T}\left(C_\theta J^{-1}G^T + M_\theta \frac{d}{dt}\left(J^{-1}G^T\right)\right) \\
\tilde{N} &= GJ^{-T}N \\
F_e &= GJ^{-T}\tau \\
M, M_\theta &= \text{inertia matrix for the box and fingers, respectively} \\
C, C_\theta &= \text{Coriolis and centrifugal terms}
\end{aligned}$$

Thus we have an equation with a form similar to that of our "simple" robot. In the box frame of reference, $\tilde{M}$ is the matrix of the effective mass of the box, and $\tilde{C}$ is the effective Coriolis and centrifugal matrix. These matrices include the dynamics of the fingers, which are being used to actually control the motion of the box. However the details of the finger kinematics and dynamics are effectively hidden in the definition of $\tilde{M}$ and $\tilde{C}$. The skew symmetry of $\dot{\tilde{M}} - 2\tilde{C}$ is preserved by this transformation.

Although the grasp map G was assumed to be surjective, it need not be square. From the equations of motion (3), we note that if the fingertip force $J^{-T}\tau$ is in the null space of G then the net force in the object's frame of reference is zero and causes no net motion of the object. These forces act against the constraint and are generally termed *internal* or *constraint* forces. We can use these internal forces to satisfy other conditions, such as keeping the contact forces inside the friction cone (to avoid slipping) or varying the load distribution of a set of manipulators rigidly grasping an object.

2.2 Control

To illustrate the control of robot systems, we look at two controllers which have appeared in the robotics literature. We start by considering systems of the form

$$M(q)\ddot{x} + C(q,\dot{q})\dot{x} + N(q,\dot{q}) = F \tag{4}$$

where $M(q)$ is a positive definite inertia matrix and $C(q,\dot{q})\dot{x}$ is the Coriolis and centrifugal force vector. The vector $N(q,\dot{q}) \in \mathbb{R}^n$ contains all friction and gravity terms and the vector $F \in \mathbb{R}^n$ represents generalized forces in the x coordinate frame.

Computed torque

The computed torque control law is a special case of the more general technique of feedback linearization. That is, through the use of nonlinear feedback, we wish to render the system dynamics linear in some appropriate set of coordinates. For a robot manipulator, given a desired trajectory x_d we use the control

$$F = M(q)\left(\ddot{x}_d + K_v\dot{e} + K_p e\right) + C(q,\dot{q})\dot{x} + N(q,\dot{q}) \tag{5}$$

where error $e = x_d - x$ and K_v and K_p are constant gain matrices. The resulting dynamics equations are linear, with exponential rate of convergence determined by K_v and K_p. Since the system is linear, we can use linear control theory to choose the gains (K_v and K_p) such that they satisfy some set of design criteria. In particular, if we choose K_v and K_p to be diagonal, we can analyze the system using the notions of system bandwidth and damping.

PD + feedforward control

PD controllers differ from computed torque controllers in that the desired stiffness (and potentially damping) of the end effector is specified, rather than its position tracking characteristics. Typically, control laws of this form rely on the skew-symmetric property of robot dynamics, that is to say $\alpha^T \left(\dot{M} - 2C\right) \alpha = 0$ for all $\alpha \in \mathbb{R}^n$. Consider the control law

$$F = M(q)\ddot{x}_d + C(q,\dot{q})\dot{x}_d + N(q,\dot{q}) + K_v\dot{e} + K_p e \tag{6}$$

where K_v and K_p are symmetric positive definite. Using a Liapunov stability argument, it can be shown that the actual trajectory of the robot converges to the desired trajectory asymptotically [9]. Extensions to the control law result in exponential rate of convergence [24, 26]. We note here that for a diagonal mass matrix, $M(q)$, the computed torque controller and the PD plus feedforward controller, when integrated into the upper level of the hierarchical structure, reduce to the same control algorithm. We make use of this fact later in determining the choice of gains by analyzing the system bandwidth and damping.

2.3 Primitives for robot control

A multi-fingered robot hand can be modeled as a set of robots that are connected to an object by a set of constraints. The analysis presented above allows us to model this interconnection and create a new dynamic system that encodes the constraints. In fact, this procedure is sufficiently straightforward that it may be automated: by specifying the contact constraint between the robots and the object, the new equations of motion for the composite robot can be derived using a symbolic manipulation program. In [6, 17], a system was proposed for building hierarchical control laws for complex interconnected robotic systems. We review that formulation here.

The fundamental objects in the robot specification environment are objects called robots. In a graph theoretic formalism these are nodes of a tree structure. At the lowest level of the tree are leaves which are instantiated by the `define` primitive. Robots are dynamical systems which are recursively defined in terms of the properties of their daughter robot nodes. Inputs to robots consist of desired positions and conjugate forces. The outputs of a robot consist of actual positions and forces. Robots also possess attributes such as inertial parameters and kinematics.

There are two other primitives that act on sets of robots to yield new robots, so that the set of robots is closed under these operations. These primitives (`attach` and `control`) may be considered links between nodes and result in composite robot objects. Nodes closer to the root may possess fewer degrees of freedom, indicating a compression of information upon ascending the tree.

The `attach` primitive reflects geometrical constraints among variables and, in the process of yielding another robot object, accomplishes coordinate transformations. `Attach` is also responsible for a bidirectional flow of information: expanding desired positions and forces to the robots below, and combining actual position and force information into an appropriate set for the higher level robot. In this sense the state of the root robot object is recursively defined in terms of the states of the daughter robots.

The `control` primitive seeks to direct a robot object to follow a specified "desired" position/force trajectory according to some control algorithm. The controller applies its control law (for example PD or computed torque) to the desired and actual states to compute expected states for the daughter robot to follow. In turn, the daughter robot passes its actual states through the controller to robot objects further up the tree.

The block diagram portion of Figure 1 may be viewed as an example of a robot system comprised of these primitives. Starting from the bottom: a finger and thumb are `defined`; each digit is controlled by muscle tension and stiffness; the muscles and sensory organs of each form low-level, fast spinal reflex loops that go directly from the digit to the spinal cord and back to the digit. The two digits are `attached` to form a composite hand. Further up the hierarchy, the brainstem and cerebellum help `control` and coordinate motor commands and sensory information. Finally, at the highest level, the sensory motor cortex, where sensory information is perceived and where conscious motor commands originate, the fingers are thought of as a pincer which engages in high level tasks such as picking.

In designing controllers using the primitives described above, a key issue is how to properly model a robot which has a controller attached to it. In order to allow the recursive nature of the primitives to operate, we must describe the new robot as a dynamical system with equations of motion in the form given by equation (3). For controllers that are very fast relative to higher levels, it is often a good approximation to model the robot as an ideal force generator, with no mass. This approximation does not imply that the robot is no longer a dynamic object, but rather that controllers at higher levels can ignore the dynamic properties of the robot, since these properties

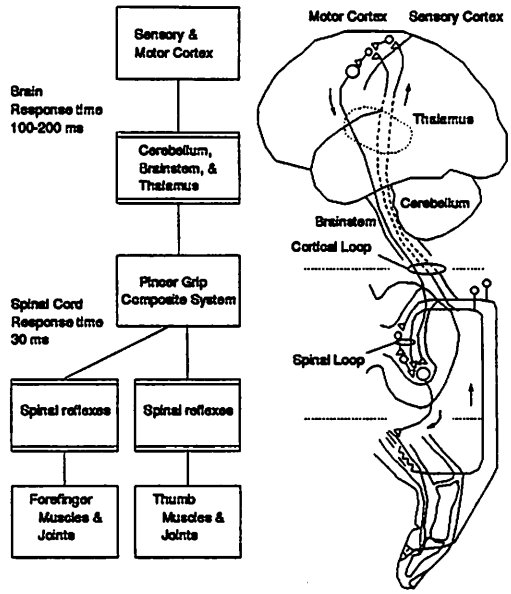

Figure 1: Hierarchical control scheme of a human finger and thumb [7]. (Figure courtesy of D. Curtis Deno)

are being compensated for at a lower level.

2.4 Stability of Hierarchical Control

In order to demonstrate stability of the hierarchical control scheme proposed here, we model the scheme as a two-step hierarchy: a low level PD at the motor level and a high level PD-type control for the "hand". If the low level scheme were infinitely fast, then the scheme would be a "conventional PD" type that is guaranteed to be exponentially convergent, i.e. the tracking error goes to zero exponentially. Standard singular perturbation arguments [10] may be used to show that the scheme is exponentially convergent, provided that the low level controller is fast enough, that is, provided the sample period is small enough.

3 Experimental Setup

What follows is a description of the hardware and software used in the control experiments presented here. First, we describe in some detail the

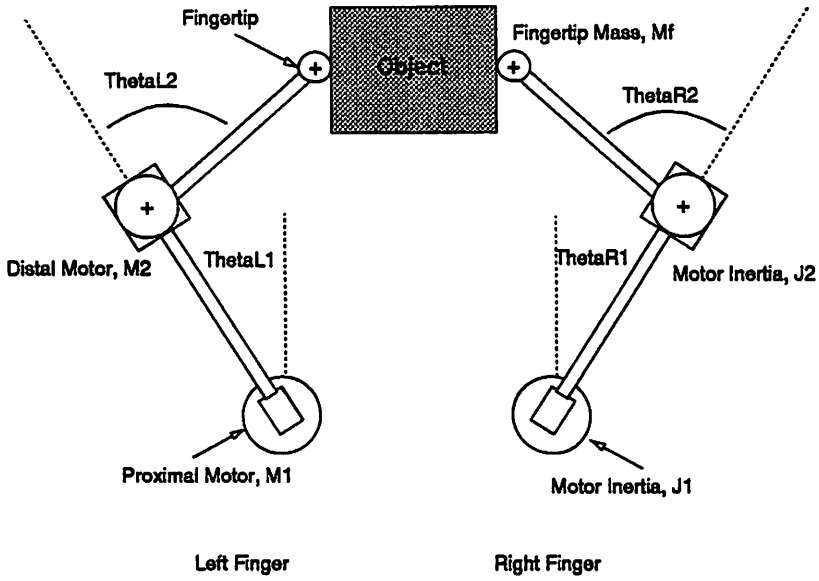

Figure 2: Top View of Styx

two-fingered hand as well as the objects that were manipulated by the hand. Next, we give an overview of the implementation of the hierarchical control structure and describe its software and hardware components.

3.1 Styx–Hardware

The control algorithms presented here have been implemented on a multi-fingered hand, known as Styx, that was designed and built to facilitate implementation and testing of control algorithms for multi-fingered hands [15]. Styx is a two-fingered, planar hand, with each finger consisting of two revolute joints and two links. The distal links are capped by small rubber cylinders that serve as fingertips and as contact "points" between the fingers and the object that is to be manipulated. A diagram of Styx is shown in Figure 2.

The motors used to drive Styx are direct-drive DC motors mounted at the base of each link and are driven with a pulse-width modulated 20 kHz square wave. Each motor contains a quadrature encoder used to sense joint position. The resolution for the proximal motors is 3600 counts per revolution and for the distal motors 2000 counts per revolution. Styx is connected to an IBM PC/AT running at 6 MHz with an 8087 floating point

Link Lengths	L_1, R_1	15.3	cm
	L_2	12.16	cm
	R_2	11.8	cm
Fingertip Radius	r_f	1.7	cm
Base Separation	B	20.0	cm
Link Mass	M_{L1}, M_{R1}	53	g
	M_{L2}	17	g
	M_{R2}	20	g
Distal Motor Mass	M_2	328	g
Fingertip Mass	M_f	3	g
Motor Inertia	J_1	18	g cm^2
	J_2	1.74	g cm^2

Table 1: Styx Parameters

coprocessor. The motors and encoders are interfaced to the AT using a set of four HP HCTL-1000 motion control chips interfaced to the AT bus. A view of the interconnection of the hardware supporting the Styx system is shown in Figure 3.

The parameters associated with Styx kinematics and dynamics are shown in Table 1.

Assumptions made to simplify implementation of the control algorithms presented here include:

1. Motor dynamics can be ignored—for small velocities, the torque generated by each motor is proportional to the input pulse width.

2. Fingertips can be modeled as fixed point contacts—in order to avoid the complexity associated with implementing a model of rolling contact dynamics, the fingertips were modeled as simple point contacts. As a result, the shifting of the contact points on the object was unmodeled. However, since the commanded trajectories all included zero orientation of the object, the effect of this shift was minimized.

3. The Coriolis and frictional forces are ignored—for trajectories resembling the slower movements tested here, these forces have been shown to be negligible [15]. More extensive testing regarding the relative significance of the Coriolis and frictional forces in the fast trajectories is

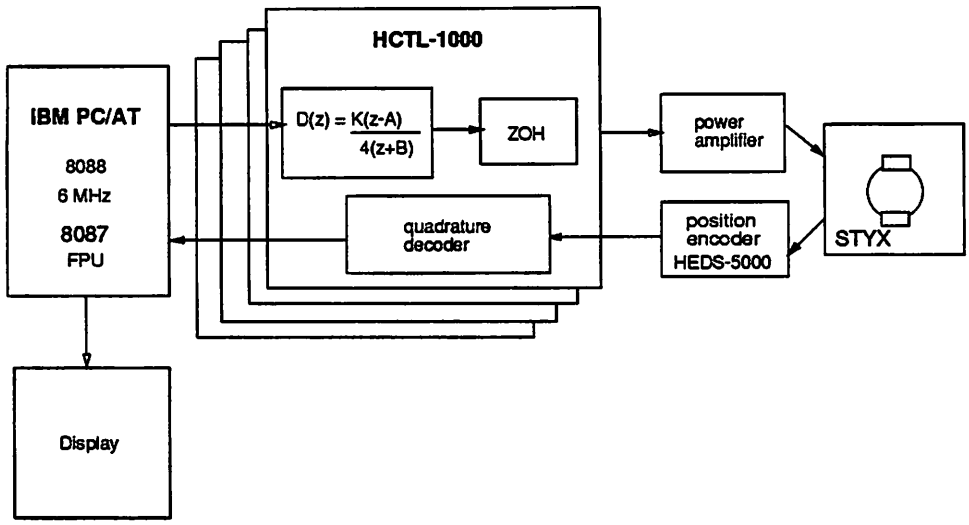

Figure 3: Hardware Supporting Styx

required.

3.2 Control Hierarchy

In the discussion of the tracking performance of the hierarchical structure we assumed the existence of two continuous time controllers, a high level PD plus feedforward controller in object coordinates as well as a low level PD controller in joint angle coordinates. Here we describe in more detail the implementation details of the hierarchical structure, including the points of departure of the implementation from the theory.

In order to implement the two-level hierarchical structure, we actually used the three-level structure shown in Figure 4. The upper two levels, consisting of a primary and a secondary control loop, are written in the C programming language, using the Microsoft 5.1 Optimizing C compiler. An assembly language scheduler controls the sample rates of the control loops. In order to more closely relate to each other the hierarchical control structures shown in Figures 4 and 1 we chose the sampling periods of the control loops to be roughly equivalent to the time delays present in the low-level, spinal reflex loops and the high-level, cortical feedback loops.

At the top of the figure we see the highest control level, the secondary control loop running at 10 Hz. At this level we calculate the inverse kine-

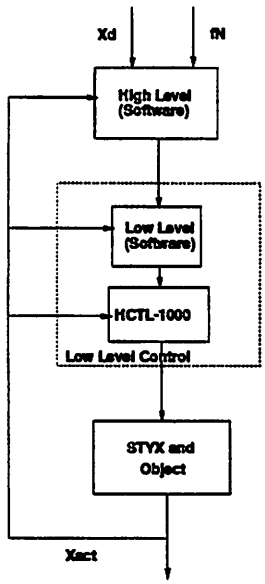

Figure 4: Control hierarchy for Styx, showing control of the desired and actual object trajectories, X_d and X_{act}, respectively, and the specified internal force, f_N.

matics of the object's desired trajectory (X_d in Figure 4) and perform the high level control functions when we put Styx into the hierarchical control mode. The high level software can be made to implement any desired control algorithm that is to be superimposed upon the lower levels. Here we present results of the high level PD +feedforward controller in object coordinates only.

The lower level of the control hierarchy consists of the low level software block shown in Figure 4 in addition to the hardware block below it. The lower software level is implemented by the primary control loop. The purpose of the primary control loop is to write at a frequency of 100 Hz directly to the motion control hardware the current commanded joint angles received from the secondary loop. In addition, the low level controller is capable of joint angle interpolation, calculating a new, updated commanded joint position by adding incrementally the joint velocity multiplied by a time step based on the ratio of the primary and secondary control frequencies. This interpolation allows the low-level controller to gradually command the joint position to move from the position commanded by the high-level controller

at one time step to the position commanded at the following time step.

At the lowest level of the control hierarchy exists the HP HCTL-1000, a digitally sampled, general purpose motor controller. Although we assumed in the stability analysis that the lower control level operated in continuous time, we used the HCTL-1000 in the implementation, because it had already been built into the existing hardware. In particular, the HCTL-1000 was used in its position control capacity only, because the fast response relative to the software control loops allowed us to make comparisons with the human control system and its multi-level structure. In the analysis, our assumption was that the digital sampling of the controller was fast enough so that we could approximate it by a continuous time PD controller.

Programmable variables in the HCTL's position control include the sampling time, T, as well as the parameters associated with the digital filter used to compensate for closed loop system stability:

$$D(z) = \frac{K(z - A/256)}{4(z + B/256)} \tag{7}$$

The position control mode performs point to point position moves. A position command is specified, which the controller compares with the actual position, calculating the position error. The full digital compensation is applied to the position error, and the calculated motor command is output until the position error changes or a new position command is given.

Since the HCTL-1000 is used in its position control mode, we do not directly send desired torque commands to the HCTL controller. Although the chip's position control is entirely adequate for specifying a desired trajectory with no high level control, a problem arises when we wish to apply a certain internal force, as, by definition, the internal force evokes no change in the system's position variables. We cannot induce the motors to apply the torques required for the specified internal force by specifying the torques directly. Instead, we calculate a virtual position error which, when multiplied by the DC gain of the position controller, yields the desired joint torques. The same difficulty arises when we wish to specify additional torques based on correctional terms calculated by the high level controller. Instead of adding in the torques directly, we calculate a "correctional" position error, by assuming that the trajectory is slow enough relative to the time constants associated with the HCTL controller that we can use the DC approximation in the torque to position conversion with some reasonable degree of accuracy.

The control system was tested with each of three different progressively more complicated control schemes, each building on the one(s) before: set-

	Beam	Box	
Mass	245	33	g
Moment of Inertia	1.8×10^3	1.3×10^3	g cm^2
Length	12	17	cm

Table 2: Object Parameters

Gain, K_d	64	
Zero, A	229	
Pole, B	64	
Sample Freq, 1/T	1.9	KHz

Table 3: HCTL-1000 Parameters

point control without joint interpolation, setpoint control with joint interpolation, and hierarchical control (including setpoint control with joint interpolation at the lower level).

4 Experimental Results

In the experimental results presented here, we use two different objects, each suited to meet the needs of the particular trajectory being tested. In the first set of trajectories, we used a metal bar that was loosely attached to the fingertips via pin joints. Mounted on top of the bar was a heavy mass, which the fingers then were required to move. The "object" thus consisted of both the beam and the mass attached to it; the parameters associated with this object are shown in the column labeled "beam" in Table 2. The second object used in the experiments was a cardboard box, the dimensions of which are also shown in Table 2, in the column labeled "box".

The HCTL-1000 parameters used in the generation of the figures shown were the chips' default parameters, listed in Table 3.

The commanded trajectories for the objects' centers of mass were circular trajectories of radius 2.5 cm, centered at $x = 1.3$ cm, $y = 21.2$ cm relative to the midpoint between the two proximal motors, as shown in Figure 2, with frequencies of 1.0 Hz for the "beam" and 0.25 Hz for the "box". The

orientation of the objects was to remain at zero throughout each movement. The tracking performance along these trajectories was tested for consistency between trials. We determined that collecting data for a period of 20 seconds was entirely sufficient for our purposes, resulting in series of 5 trials each for the slow trajectories and 20 trials each for the fast motions. Additional trials merely served to duplicate the results.

Although we are interested primarily in grasping control rather than coordinated robot control, we used the beam in the rapid movements because of hardware limitations: In the slow movements we applied an internal force of 3×10^4 dyne. The maximum motor torque, however, was limited to the extent that we were not able to exert the large internal forces required to grasp and move a heavy object during rapid movement. Thus, we used an object that we could loosely attach to the fingertips, enabling us to use smaller internal forces (3×10^3 dyne) without losing contact with the object.

Figures 5-7 depict the performance of the three different controllers, the setpoint controller (non-hierarchical), the setpoint controller with low-level joint velocity interpolation, and the hierarchical controller. The setpoint controller consisted of a high level piece that existed solely to calculate the inverse kinematics, directing the low-level controller to a new setpoint at a frequency of 10 Hz. Position control was carried out only at the fast, low level by the HCTL-1000 and was done strictly in joint angular coordinates. The rather poor performance of this controller can be judged quickly by examining Figure 5(a), which shows the actual trajectory of the object's center of mass. A large overshoot, primarily in the horizontal, x, direction is evident.

The second controller that was used, the joint interpolation controller, was non-hierarchical as well, as the high level software, again, existed merely to solve the inverse kinematics, with the additional control piece, the joint velocity interpolation, added only at the low level, as described above. Although the trajectory shown in Figure 5(b), was found to be smoothed out more, we found the same overshooting of the goal trajectory that occurred in Figure 5(a). Further efforts aimed at improving overall performance by introducing more and more complicated versions of a single-level, fast controller seemed unwarranted. The problem that remained to be addressed was the object's mass, a parameter that was not taken into account at the lower level.

A significant improvement in the trajectory tracking performance was found when a high-level controller that corrected for the object's tracking errors was superimposed upon the existing low-level control structure. By

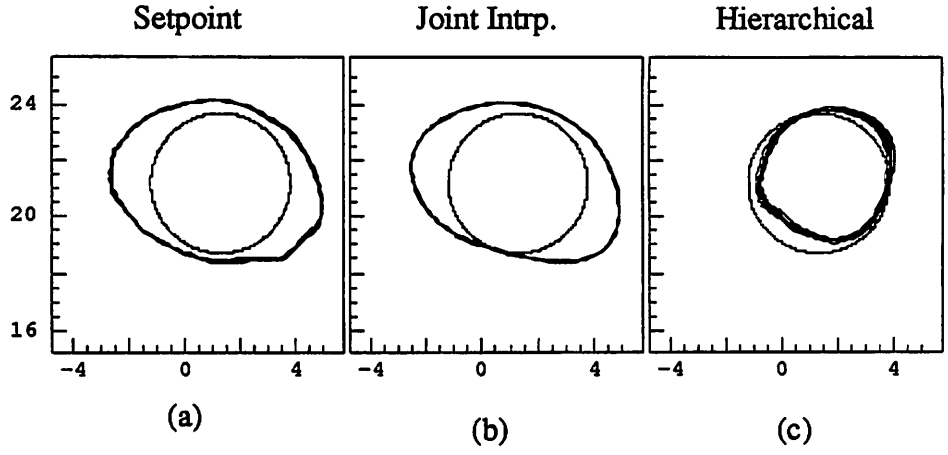

Figure 5: Actual and commanded object trajectories with setpoint controller (a), joint interpolation controller (b), and hierarchical controller (c) in Cartesian space with weighted beam; commanded circular trajectories are shown within each graph (fast trajectory).

calculating the torques required to bring the object back to the desired trajectory, the high-level controller was able to compensate for and minimize the trajectory errors in object's (Cartesian) coordinates, as shown in Figure 5(c). The overshoot that so grossly disfigured the trajectories of the two non-hierarchical controllers disappeared completely, resulting in a much better overall tracking performance.

In Figure 6 we show the calculated box position error, for the setpoint controller (a), the setpoint controller with interpolation (b), and the hierarchical controller (c) for the same trajectories that were depicted in Figure 5. For each controller, the error is given by $\sqrt{(X_{act} - X_{des})^2 + (Y_{act} - Y_{des})^2}$. Again, the insignificant improvement afforded by the addition of the joint interpolation to the setpoint controller, and the marked improvement of the performance of the hierarchical controller over the performance of both single-level controllers is evident.

In the figures showing data in "box-plot" form (Figures 6-8), the top and bottom of the boxes correspond to the twenty-fifth and the seventy-fifth percentiles of the given variables, while the horizontal lines through the boxes correspond to the median values of the variables. The vertical lines have ends that extend beyond the quartiles by a distance equal to one and

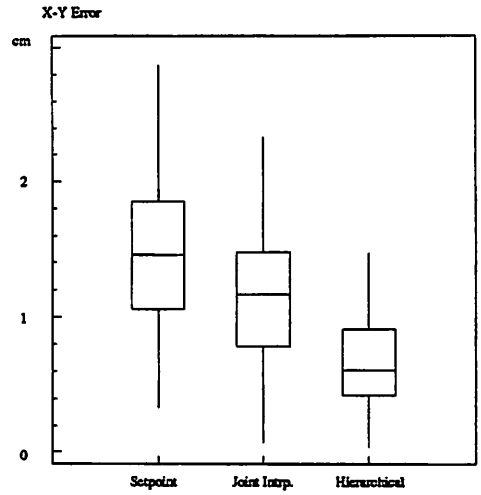

Figure 6: Deviation (in cm) from commanded trajectories of actual object trajectories with setpoint controller, joint-interpolation controller, and hierarchical controller when using weighted beam (fast trajectory).

one-half times the inter-quartile range. Approximately ninety-nine percent of normally distributed data are within the range covered by the vertical lines; outliers are identified by an asterisk, '*'.

The improvement in the object's position tracking that we found when using the hierarchical controller was, unfortunately, not mirrored in the object's orientation, as can be seen in Figure 7. One possible cause is a poor approximation of the moment of inertia of the object. Due to the configuration of the system, a very slight shift in the fingertip position can cause a relatively large orientation error in the object. Thus, any error in the calculation of the object's moment of inertia can cause an error in the orientation portion of the feedback control, which, in turn, can cause a significant orientation error in the object. We expect that the same controller, when furnished with a more accurate measure of the object's moment of inertia, will show an improvement in the orientation error similar to that found in the position error. Further tests with more sophisticated measurements of the object's moment of inertia are expected to confirm this prediction.

As discussed above, the reasons for using the beam in the controller comparisons were to enable us to track high-speed trajectories with heavy objects without saturating the motor output. However, we wish to point out that the system is, in fact, capable of standard "grasping" manipulation. To

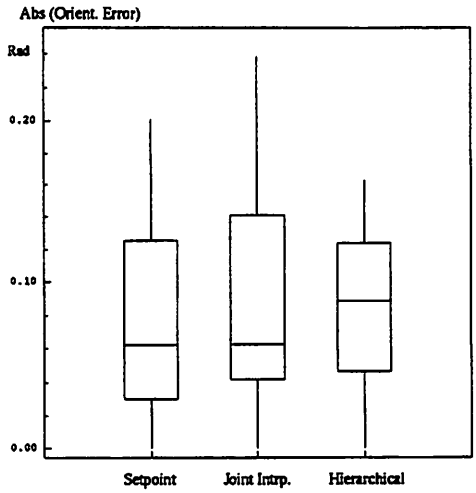

Figure 7: Deviation (in radians) from commanded object orientation of actual object orientation with setpoint controller, joint interpolation controller, and hierarchical controller from commanded trajectories when using weighted beam (fast trajectory).

this end, we show in Figure 8 the object's position, the orientation error as a function of time, and the position error when the three controllers were used to manipulate the lighter object ("box") in tracking the slow (0.25 Hz) trajectory. As expected, at slow speeds and when using small object masses, the differences between the single-level controllers, which did not take into account the object's parameters, and the hierarchical controller became insignificant. The position control used in the setpoint controller appeared to be entirely adequate when the object was commanded to move slowly enough for the controller to reach each specified position before receiving the next position command.

The commanded internal force exerted on the object by the hand was kept at a constant value, 3×10^4 dyne, in all of the slow trajectory experiments shown here. We currently have not measured the actual internal force on the object; hence, this quantity is not a controlled variable at this time. The value of the constant internal force was chosen, because it seemed an adequate compromise between having enough internal force to hold the object throughout the movement with a minimum of slipping and not having so much internal force that the grasp became unstable. This potential instability due to a large "internal force" was a real problem that needed to be

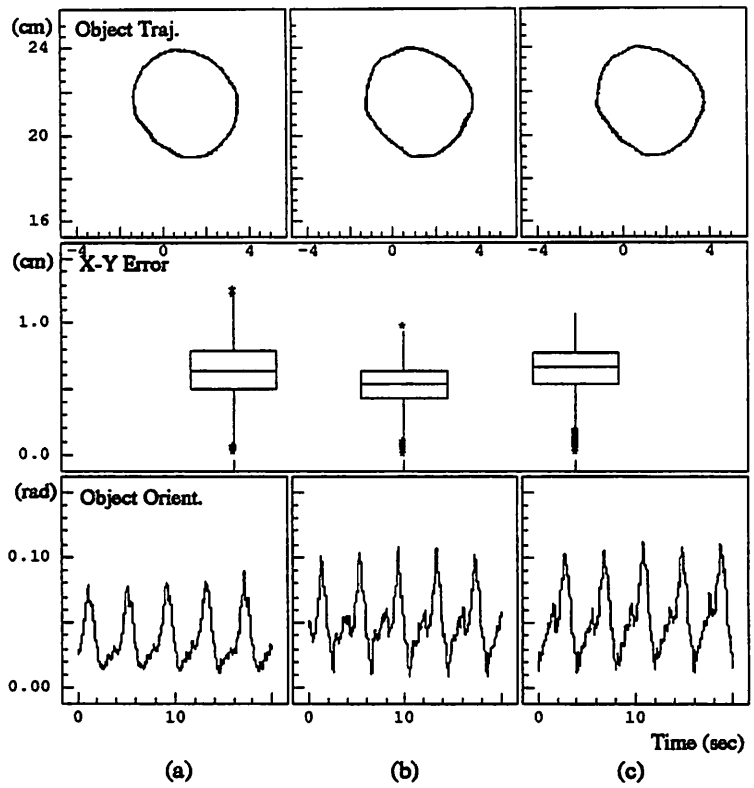

Figure 8: Object trajectory, box plot of object position error, and object orientation error as a function of time for setpoint controller (a), setpoint controller with joint interpolation (b), and hierarchical controller (c) (slow trajectory).

addressed: the grasp and Jacobian matrices, used in the internal force calculation, were updated at the relatively slow speed of the high-level control loop. The calculated "internal" force, therefore, contained a small, at times significant portion that lay outside of the grasp matrix's null space. Similarly, the torque calculated to produce the internal force differed slightly, at times significantly, from the "true" value of the torque that would have been required to produce an internal force. A possible cause of this difference may be traced to differences between the Jacobian matrix that was calculated in the slow control loop and the "true" current value of the Jacobian. An even more significant contributing factor, perhaps, was the amplifier gain "drift" over time, which, when combined with the slight kinematic differ-

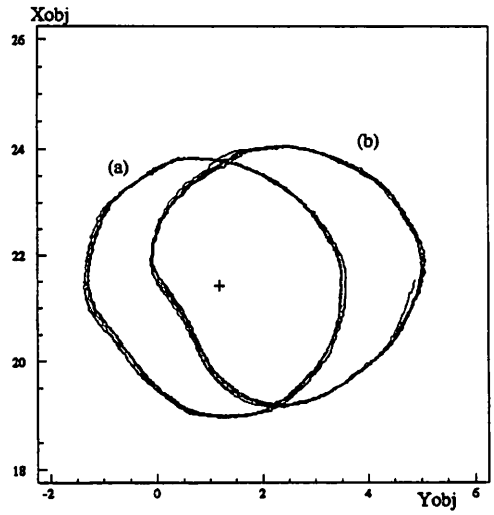

Figure 9: Effect of increased specified internal force on trajectory errors using the setpoint controller; trajectories (a) and (b) were followed with a specified internal force of 3×10^4 and 5×10^4 dyne, respectively; the plus sign ("+") marks the center of the commanded trajectory.

ences between the two fingers, made precise calibration of the system an extremely difficult task, and one that would need to be repeated throughout the life of the robot components. We stress this difficulty in calibration, because it goes beyond the need to precisely measure the kinematic parameters associated with the robotic hand and the object. In general, the higher the desired internal force, the higher the errors associated with applying the proper torques. We found that relatively high commanded internal forces interfered significantly with proper tracking behavior, while even higher forces caused the grasp to become unstable.

A brief exploration of the effect of different levels of internal force is shown to furnish the reader with some understanding of the magnitude of this effect. The commanded trajectory was the slow circle described above. Figure 9 shows the increase in the tracking error accompanying an increase in the specified internal force from 3×10^4 dyne to 5×10^4 dyne. The primary cause of the shift in the trajectory appears to have been the calibration difficulty discussed above. While the figure shows the data for the non-hierarchical setpoint controller only, the results were very similar when the other controllers were tested.

4.1 Discussion of Results

The data presented here show a noticeable improvement in the performance of the hierarchical controller as compared with that of the single-level controllers when the controllers were used in fast movements of a heavy object. The most basic controller tested, the setpoint controller, was a simple, fast, single-level controller operating only on the joint position variables. The performance of this controller, even when operating at high frequencies relative to the trajectory frequencies, was clearly inadequate, demonstrating a large overshoot in the positioning of the object throughout much of the trajectory. In an attempt to improve the performance of the single-level control scheme, we introduced an additional level of complexity at the joint level by calculating the joint angular velocity and implementing an interpolation scheme that was designed to smooth the commanded trajectory between adjacent points as specified by the high level routines calculating the inverse kinematics. Although some smoothing was observed in the object's trajectory when the joint interpolation scheme was used, there was no significant change in the tracking performance of the system. Evidently, adding some degree of complexity to the existing single-level controller was not sufficient to overcome the tracking errors of the setpoint controller. We surmise that the poor tracking performance was a result of neglecting the object's dynamics in the overall control scheme. The speed at which the single-level controllers were operating, however, placed strict limits on the amount of computation that could be completed within the control loop, thereby effectively excluding the possibility of moving the computation of the dynamics of the entire system, including both the robot hand and the object, into the lower level. We add here that, in addition to testing the fast, simple single-level controller, we implemented a more complicated single-level controller that incorporated both the object and the hand dynamics and that was run at the slow speed of the high level in the hierarchical scheme. The results, when the controller was tested in the fast movement experiments, were instability to the point where collecting data became entirely unnecessary. Clearly, a scheme making use of the best of both controller complexity and controller speed was required.

A solution to the computational problem associated with manipulating a heavy object at high speeds was the hierarchical control scheme presented here. The hierarchical scheme allowed us to incorporate in a single control scheme the advantages of fast control loops as well as the complexity required to adequately model the system dynamics. As expected, incorporating the

object's dynamics in the control scheme became more important as both the object's mass and its acceleration along the trajectory increased. Our conclusion regarding the need for hierarchical control in certain movements was supported by the relative similarity in the performance of the three controllers when operating on light moving at slow speeds. With an increase in mass and movement speed, differences between the controllers became more apparent.

We exploited the system's similarity when using the computed torque and the PD plus feedforward controllers and based the particular choice of the high-level controller parameters on the second-order linear error dynamics resulting from applying a computed torque controller. This analysis enabled us to select the cutoff frequency and the damping factor for the system. In order to obtain a rolloff above the trajectory frequencies and a damping ratio of 0.5, we chose the high-level control gains to be $K_v = 2$ and $K_p = 4$, with acceptable performance.

A subject for further inquiry is the choice of the control parameters at the low level of the hierarchy. The difficulties encountered in this choice were the limitations of the existing hardware and a lack of understanding of how and to what extent, the choices made at one level of the hierarchy were affected by the choices made at higher or lower levels. At this time, our choice of the HCTL-1000 parameters was based upon the assumption of complete separability of the two control levels, which allowed us to analyze the closed loop performance of only the low level system. The parameter values resulting from this analysis were then subjected to experimental testing to determine the validity of the assumptions, and a final choice was then made, based upon the performance of different combinations of parameters. However, we stress that no extensive testing was performed to determine the *best* choice. What is required is a more rigorous approach, including an analysis of the extent to which the adjacent levels of control interact, leading to an analytical solution and an experimental confirmation of the results.

The results discussed here represent an experimental confirmation of the predicted stability of the hierarchical control structure under the conditions given. In particular, the assumptions regarding the separability of the choice of control parameters as well as the time scales of adjacent control structures have been validated experimentally for the planar, two-fingered system used here. Furthermore, the validity of the assumption that the frequencies inherent in the trajectories were low enough to merit the steady-state approximation of the HCTL-1000 gain was confirmed. What remains to be gained is a better understanding of what constitutes adequate limits on the

ratios of adjacent time scales and how these limits relate to the trajectories of interest.

5 Conclusion

Based upon the experiments performed on Styx thus far, the most effective control scheme in fast movement of heavy objects was the hierarchical control scheme. Hierarchical control schemes have the advantage of being able to run simple, lower levels at high speeds, thus rapidly correcting for tracking errors in fast movements, while running at lower speeds more complicated higher levels that improve the overall performance by incorporating system dynamics far removed from the low level actuators. Although the experimental results presented here are based solely upon work done with a simple, planar system, the advantages of using a hierarchical control scheme can easily be applied to more complicated system, as we, in the implementation of the control structure, in no way made use of simplifying assumptions based upon the simplicity of the system. In fact, the differences between the performance of the hierarchical and single-level controllers ought to become greater as the system complexity is increased and the computational complexity of the higher level dynamics becomes greater.

An example of an extraordinarily complicated grasping control system is the human motor control system and its feedback pathways from the skin surrounding and the muscles controlling the fingers to the spinal cord and to various parts of the brain. In a study of two-handed grasping control in humans Reinkensmeyer [22] suggests the use of a simple control structure that takes advantage of the spring-like properties of muscle and of the similarity between the dynamics of the single robot hand (in our case "finger") and the robot hand-object system. In the work presented here, there are no such simplifying properties of the actuators. However, by configuring the system in such a way as to resemble the relative feedback delays of the human motor control system ($20 - 30$ ms for the single-reflex spinal feedback loop and up to 200 ms and more for the highest-level, voluntary control feedback loops [14, 23]), we were able to examine some connections between the two systems that merit further exploration. In both systems the structure of the lower levels enables the control algorithms at the higher levels to make simplifying assumptions about the systems that lie beneath them, thereby allowing the higher levels to control on a more abstract level the entire movement, leaving the lower levels free to implement rapidly the

details of locally interfacing directly with the actuators. We hope to further understanding of possible and plausible control structures in both systems by exploring the parallels as well as the differences between them.

There are many areas in the design, analysis, and testing of hierarchical control algorithms pertaining to grasping that need to be investigated more fully. Lacking at this time is a rigorous theory of the interconnection of the various levels of grasping control as well as a theory of the conditions under which we can expect stability of the overall grasping control scheme. The need for more sophistication in the development of approaches to the choice of system parameters is clear. With increased sophistication, the potential for improved performance at a lower computational cost than is associated with single-level control is, in our opinion, undeniable. On the experimental side, we would like to see an implementation of other controllers, such as a computed torque controller, in the higher level of the hierarchical structure. Furthermore, extensive testing of rapid movements with high, possibly time-varying, internal forces would afford greater understanding of grasping control and might enable us to draw more conclusions about the biological as well as the roboticist's solution to the problem of controlling rapid grasping movements. In turn, the introduction of varying internal forces, especially in rapid movements, may act to compound the problem of the object slipping within the fingers' grasp and may, therefore, precipitate the implementation of models that include the dynamics of rolling contacts.

Our comparison of hierarchical and single-level control schemes has at once provided an indication of the advantages of using hierarchical control algorithms in grasping control and introduced a variety of open research questions relating to the theory and the application of hierarchical control in multi-fingered grasping situations.

Acknowledgements

This research was supported in part by NSF under grant DMC-84-51129 and under grant ECS-87-19298 and by the U.S. Department of Health and Human Services under grant PHS GM07379-14. In addition, the authors would like to thank Dr. S. Lehman, of the Bioengineering Graduate Group at the University of California at Berkeley, for many helpful discussions and comments relating to the work presented here.

References

[1] C. O. Alford and S. M Belyen. Coordinated control of two robot arms. In *International Conference on Robotics*, pages 468–473, 1984.

[2] S. Arimoto, F. Miyazaki, and S. Kawamura. Cooperative motion control of multiple robot arms or fingers. In *IEEE International Conference on Robotics and Automation*, pages 1407–1412, 1987.

[3] D. Clark. Hic: An operating system for hierarchies of servo loops. In *IEEE International Conference on Robotics and Automation*, pages 1004–1008, 1989.

[4] A. B. A. Cole, J. E. Hauser, and S. S. Sastry. Kinematics and control of multifingered hands with rolling contact. *IEEE Transactions on Circuits and Systems*, 34(4):398–404, 1989.

[5] J. Demmel, G. Lafferrier, J Schwartz, and M. Sharir. Theoretical and experimental studies using a multifinger planar manipulator. *IEEE International Conference on Robotics and Automation*, pages 390–395, 1988.

[6] D. C. Deno, R. M. Murray, K. S. J. Pister, and S. S. Sastry. Control primitives for robot systems. In *IEEE International Conference on Robotics and Automation*, pages 1866–1871, 1990.

[7] D. C. Deno, R. M. Murray, K. S. J. Pister, and S. S. Sastry. Primitives for robot control. In M. A. Kaashoek, J. H. van Schuppen, and A. C. N. Ran, editors, *Proceedings of the 1989 International Symposium on the Mathematical Theory of Networks and systems (MTNS-89)*, pages 13–32. Birkhäuser, 1990.

[8] S. Jacobsen, J. Wood, K. Bigger, and E. Iverson. The Utah/MIT hand: Work in progress. *International Journal of Robotics and Control*, 4(3):221–250, 1986.

[9] D. Koditschek. Natural motion for robot arms. In *IEEE Control and Decision Conference*, pages 733–735, 1984.

[10] P.V. Kokotovic, H. K. Khalil, and J. O'Reilly. *Singular Perturbation Methods in Control: Analysis and Design.* Academic Press, New York, 1986.

[11] Kader Laroussi, Hooshang Hemami, and Ralph E. Goddard. Coordination of two planar robots in lifting. *IEEE Journal on Robotics and Automation*, 4(1):77–85, 1988.

[12] Z. Li, P. Hsu, and S. Sastry. On kinematics and control of multifingered hands. In *IEEE International Conference on Robotics and Automation*, pages 384–389, 1988.

[13] M. T. Mason and Jr. J. K. Salisbury. *Robot Hands and the Mechanics of Manipulation.* The MIT Press, Cambridge, Massachusetts, 1985.

[14] T. A. McMahon. *Muscles, Reflexes, and Locomotion.* Princeton University Press, Princeton, New Jersey, 1984.

[15] R. Murray and S. S. Sastry. Control experiments in planar manipulation and grasping. *International Conference on Robotics and Automation*, 1989.

[16] R. M. Murray. Experimental results in planar grasping. Master's thesis, University of California at Berkeley, 1988.

[17] R. M. Murray, D. C. Deno, K. S. J. Pister, and S. S. Sastry. Control primitives for robot systems. *IEEE Transactions on Systems, Man and Cybernetics*, 1992.

[18] R. M. Murray and S. S. Sastry. Grasping and manipulation using multifingered robot hands. In R. W. Brockett, editor, *Robotics: Proceedings of Symposia in Applied Mathematics, Volume 41*, pages 91–128. American Mathematical Society, 1990.

[19] Y. Nakamura, K. Nagai, and T. Yoshikawa. Mechanics of coordinative manipulation of multiple robot mechanisms. In *IEEE International Conference on Robotics and Automation*, pages 991–998, 1987.

[20] Van-Duc Nguyen. Constructing force-closure grasps. *International Journal of Robotics and Control*, 7(3):3–16, 1988.

[21] Jr. N.R. Sandell, P. Varaiya, M. Athans, and M.G. Safonov. Survey of decentralized control methods for large scale systems. *IEEE Transactions on Automatic Control*, AC-23:108–128, 1978.

[22] D. J. Reinkensmeyer. Human control of a simple two-hand grasp. Master's thesis, University of California at Berkeley, May 1991.

[23] J. C. Rothwell. *Control of Human Voluntary Movement.* Aspen Publishers, Inc., 1987.

[24] N. Sadegh. *Adaptive Control of Mechanical Manipulators: Stability and Robustness Analysis.* PhD thesis, Department of Mechanical Engineering, University of California, Berkeley, California, 1987.

[25] N. R. Sandell, Jr., P. Varaiya, M. Athans, and M. G. Safonov. Survey of decentralized control methods for large scale systems. *IEEE Transactions on Automatic Control*, AC-23:108–128, 1978.

[26] J. E. Slotine and W. Li. On the adaptive control of robot manipulators. *International Journal of Robotics and Control*, 6:49–59, 1987.

[27] M. W. Spong and M. Vidyasagar. *Robot Dynamics and Control.* John Wiley and Sons, New York, 1989.

[28] S. T. Venkataraman and T. E. Djaferis. Multivariable feedback control of the JPL/Stanford hand. In *IEEE International Conference on Robotics and Automation*, pages 77–82, 1987.