RECURSIVE ADAPTIVE CONTROL OF OPEN AND CLOSED KINEMATIC CHAINS

by

TRAVIS EUGENE BARNES

B.S., Kansas State University, 1985

A THESIS

submitted in partial fulfillment of the

requirements for the degree

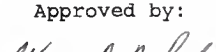
MASTER OF SCIENCE

MECHANICAL ENGINEERING

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1989

Approved by:



Major Professor

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

ACKNOWLEDGEMENTS

I would like to thank the people which have contributed to this thesis. Thanks go to my graduate committee for their time and support, Dr. J. Garth Thompson, Dr. Chi-Lung Huang, and Dr. Prakash Krishnaswami. Special thanks goes to my major advisor and committee head, Dr. Warren White Jr. for his support, direction and the confidence throughout the work. I would like to thank the Department of Mechanical Engineering, Kansas State University for the equipment and financial support. Thanks goes to Marie Kannowski for proof reading and to my parents for all their support and advice.

CHAPTER 1

INTRODUCTION

The Need for Adaptive Controls in Robotics

The technology of robot control has progressed significantly since the times of the initial Unimation arms of the early 1960's. Most commercial robots are controlled by a proportional (P) or a proportional-derivative (PD) controller. These controllers use the position and velocity errors to determine the control torques. The most common controllers are based upon the joint errors and are termed as "joint space" controllers. The position errors are the difference between the actual and desired joint positions and the velocity errors are the time derivative of the joint position errors. The vector of control torques, τ, for a PD controller can be described by the relation

$$\tau = K_p(q_d - q) + K_d(\dot{q}_d - \dot{q}), \qquad (1.1)$$

where K_p is the proportional control gain matrix and K_d is the derivative control gain matrix. The q_d and $\dot{q}_d$ terms are the desired joint position and velocity vectors, respectively and q and $\dot{q}$ terms are the actual joint position

and velocity vectors, respectively. Because the control is driven by error, the exact path of the robot's movement cannot be determined before motion begins. The method is both simple and easy to implement. Real time trajectory guidance can be achieved using either analog or digital control techniques.

There are several research robots that have been controlled using a computed torque method. A recursive computed torque controller developed by Luh, Walker, and Paul [1] uses an arm dynamic model of the form

$$\tau = H(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) \qquad (1.2)$$

to determine the torque vector, τ, needed to drive the robot. In equation 1.2 H(q) is the generalized mass matrix of the manipulator. The vector $C(q,\dot{q})\dot{q}$ contains the forces and torques due to centripetal and Coriolis accelerations. The vector G(q) consists of the forces and torques owing to gravity. The vectors q, $\dot{q}$, and $\ddot{q}$ contain the generalized joint position, velocity, and acceleration, respectively for the robot arm. If an accurate dynamic model of the robot exists and the mass, inertial values, and kinematic parameters for each link in the robot are known, then the motion of the robot can be accurately controlled and the path predicted with less error than that obtainable with PD control alone. Computed torque control, sometimes called feed forward compensation, is often implemented together

with PD control to increase the "robustness" of the controller. The computed torque method requires significant floating point computational resources to provide the control at a rate suitable for close path following. Until recently, the availability and cost of the control computers precluded this method of control as a viable alternative for industrial robots.

The dynamic model for serial link robots is well understood and efficient algorithms have been developed to obtain the driving torque. The required parameters for computing the torque include moments of inertia, location of mass center, mass, and viscous joint friction coefficient for each link. The parameter values are hard to obtain accurately and some parameters change by a large extent as the robot picks up or places an object and by a lesser extent as the drive mechanisms wear. The changing parameters make it difficult to maintain the desired control accuracy over the entire range of payloads and life of the robot.

To correct the problems associated with the changing or unknown parameters of a manipulator, an adaptive controller is used to produce the desired motion within a predictable tolerance. Adaptive controllers attempt to either modify parameters within the controller to compensate for the changing manipulator dynamics, or estimate the dynamic

manipulator parameters that the controller uses in the control law.

Methods of Adaptive Control

There are two major types of adaptive control methods. The first method is known as model reference adaptive control (MRAC). This method uses a controller with some simple mathematical model to which the response of the system is compared. A second order system is a common choice for the reference model. The resulting tracking error is used to update the constants within the simple model in an attempt to drive the difference between the two responses to zero. Many variations of the MRAC controller have been tried. Craig [2] cites some of these many controller variations using the model reference adaptive control technique. Craig [2] goes on to state that robust stable control can not be proven for a controller using the "traditional" MRAC approach.

The second major method uses a detailed mathematical model of the system and estimates the model parameters. The tracking error is used to update the estimate of the model parameters. In addition to the parameter update scheme the controller consists of a computed torque calculation together with a PD compensator.

Robotic Adaptive Control

In 1986 Craig [2] developed a method using the computed torque calculations with a model reference adaptive controller to determine the torques necessary for a robot to closely follow a given trajectory. The controller adapted all of the system dynamic parameters. The adaptive algorithm was developed from the tracking error so that it was Lyapunov stable. This method required the knowledge of the actual joint accelerations together with the actual joint positions and velocities. The method also required the inverse of the generalized mass matrix, which presents considerable computational overhead for real time applications. Craig justified this computationally intensive approach with the reasoning that soon we will have processors capable of the computing speeds necessary to produce the control calculations in real time.

In 1987 Slotine and Li [3,4,5] developed a method that did not require the knowledge of the joint acceleration or the inverse of the mass matrix. A Lyapunov argument was used to prove stability by showing that the tracking error converged to a sliding surface. The method achieved a very robust adaptive control; however, the method does require the knowledge of the complete dynamic model of the arm as did Craig's approach. The complete dynamic model of a multi-axis manipulator is difficult to find owing to the algebraic burden of the task as demonstrated by Snyder [6].

In 1988 Walker [7] presented a variation of Slotine's work that was a recursive procedure for both open and closed kinematic chains. Walker developed a special data structure to handle the closed loop chains, and used spatial notation to describe the kinematics and dynamics of the manipulator. The spatial notation that Featherstone [8] originally presented provides both a unique method of representing the manipulator and compact algebraic expressions of the recursive arm dynamics. The kinematic constraint equations for the closed loops were embedded into the manipulator's computed torque control equations. The dynamic parameters used in the torque control equations were updated using intermediate results of the torque calculations. In 1988 Slotine and Niemeyer [9] also developed a recursive procedure using spatial notation for serial link arms which included parameter estimation techniques. Slotine and Niemeyer only indicated that their technique could be extended to include the closed kinematic chain geometry.

Project Overview

This thesis is an investigation into developing a recursive adaptive control algorithm. The algorithm uses a computed torque method for control and performs adaptation on the estimates of the dynamic parameters of the manipulator. The algorithm will handle both serial link manipulators and manipulators with closed kinematic chains.

The final form of the control law reduces to

$$\tau = Y \hat{a} - K_d s \tag{1.3}$$

and the update of the estimated parameters is achieved by

$$\dot{\hat{a}} = -\Gamma Y^T s. \tag{1.4}$$

The $\hat{a}$ term is a vector that contains the estimates of the np dynamic parameters that are being adapted. There are a minimum of 10 parameters per link to be adapted. The Y matrix is an nd by np matrix where nd is the number of driven joints. The K_d term is a positive definite matrix containing control constants of a derivative controller. The Γ term is a positive definite matrix and s is a vector of tracking errors derived from the errors in joint velocity and position. The main thrust of this research is to develop a recursive method to find the Y matrix of the control law modeled after the well known recursive dynamic procedure developed by Luh, Walker, and Paul [1].

Chapter Two develops the recursive adaptive control theory for a serial link manipulator. The development modifies the Newton-Euler recursive control technique to generate the Y matrix. The modifications involve factoring the equations in terms of the dynamic parameters and developing procedures to overcome problems in the recursive procedure associated with Coriolis and centripetal accelerations.

Chapter Three describes the extension of the recursive adaptive control method to manipulators with closed loop kinematics. The dynamics are developed along the scheme that Luh and Zheng [10] developed for closed kinematic chains. The closed chains are cut at certain joints and the remaining links and joints are treated as serial link manipulators. Then Lagrangian multipliers are used to obtain the closed loop dynamics from the results of the open loop dynamic calculations.

The method developed in this work is very flexible and does not require any explicit evaluations of the dynamic equations to obtain the closed loop results. The closed kinematic loops are cut open and treated as serial links to calculate the open loop dynamics. Then the closed loop dynamics are found by determining the force at the cut location so that the driving torques of the free joints vanish. This flexibility is made possible by the development of a data structure to define closed kinematic chains. The manipulator description used the conventional Denavit and Hartenberg [11] parameters and did not use the spatial notation of Walker [7]. The only part of the routine that must be customized by the operator is the equations to obtain the kinematic position, velocity and acceleration of the dependent joints from the independent joints.

The adaptive control algorithm initially opens the kinematic chains to form serial links and calculates the Y matrix using the serial link algorithm. The kinematic chains are then closed and the Y matrix corrected.

Chapter Four describes both the data structures necessary to define a manipulator with closed kinematic chains and the method used to model the manipulator for the purpose of performing simulations. The manipulator model is a generalized model that will mechanisms with both open and closed kinematic chains. The simulations generate the generalized mass matrix and force vectors by using the recursive computed torque algorithm with selected inputs.

Chapter Five presents the results of the simulations. The simulations for serial link manipulators included a R-theta manipulator, a double pendulum, and a three joint manipulator. Simulations for manipulators with closed kinematic chains included a slider crank mechanism, an offset slider crank mechanism and a Cincinnati Milacron T^3 776 industrial robot.

Chapter Six presents the conclusions of the investigation and gives recommendations for further work.

CHAPTER 2

DEVELOPMENT OF A RECURSIVE ALGORITHM FOR SERIAL LINKS

Non-Recursive Serial Link Adaptation

Slotine and Li [3,4,5] used a Lyapunov stability argument along with a sliding mode formulation to develop the manipulator control law. The control law for an n degree of freedom manipulator is based upon the dynamic model

$$\tau = H(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) \qquad (2.1)$$

where $H(q)$ is the n x n symmetric, positive definite, generalized mass matrix for an n link manipulator. The matrix $C(q,\dot{q})$ is the n x n matrix containing terms stemming from the forces and torques created by the centripetal and Coriolis accelerations. The vector $G(q)$ is of length n and contains the forces and torques due to gravity. The quantity τ is the n x 1 vector of applied forces and torques. The vector q is the n x 1 vector of joint positions while $\dot{q}$ and $\ddot{q}$ are the n x 1 vectors of joint velocity and acceleration, respectively.

The time varying estimate of the dynamic parameters $\hat{a}$ is used to form $\hat{H}$, $\hat{C}$, and $\hat{G}$ which are approximations to the

corresponding quantities of equation (2.1). A formal structure for the vector $\hat{a}$ will be presented in a later section. Craig [2] points out that H, C, and G are linear with respect to the link dynamic parameters which consist of the center of mass with respect to the link frame, the link mass, and the six unique elements of the link inertia tensor referenced to the link frame. A similar fact is true for the matrices $\hat{H}$, $\hat{C}$, and $\hat{G}$ in terms of the approximate model parameters. Exploiting this fact, part of a controller presented by Slotine and Li [3,4,5] consisted of

$$\hat{H}(q)\ddot{q}_r + \hat{C}(q,\dot{q})\dot{q}_r + \hat{G}(q) = Y(q,\dot{q},\dot{q}_r,\ddot{q}_r)\hat{a} \qquad (2.2)$$

where

$$\dot{q}_r = \dot{q}_d - \Lambda\tilde{q} \ , \qquad (2.3)$$

$$\ddot{q}_r = \ddot{q}_d - \Lambda\dot{\tilde{q}} \ , \qquad (2.4)$$

$$\tilde{q} = q_d - q \ , \qquad (2.5)$$

and

$$\dot{\tilde{q}} = \dot{q}_d - \dot{q} \ . \qquad (2.6)$$

The partial control law of equation 2.2 consist of the approximate dynamic equations and an additional term that resembles a PD controller with variable coefficients. The vectors q_d, $\dot{q}_d$, and $\ddot{q}_d$ are n x 1 vectors containing the desired joint positions, velocities, and accelerations, respectively. The Λ matrix is a positive definite matrix that is used to develop the sliding mode control.

The complete control law of Slotine and Li [3,4,5] for a manipulator is given by

$$\tau = \hat{H}(q)\ddot{q}_r + \hat{C}(q,\dot{q})\dot{q}_r + \hat{G}(q) - K_d s \qquad (2.7)$$

where $$s = \dot{q} - \dot{q}_r = \dot{\tilde{q}} + \Lambda\tilde{q}. \qquad (2.8)$$

The quantity s will be referred to in later sections as the tracking error. Substituting equation 2.2 into equation 2.7 yields

$$\tau = Y(q,\dot{q},\dot{q}_r,\ddot{q}_r)\ \hat{a} - K_d s \qquad (2.9)$$

where K_d is a positive definite matrix.

The update for the parameter estimation is provided by

$$\dot{\hat{a}} = -\Gamma Y^T s \qquad (2.10)$$

where Γ is a constant positive definite matrix. Both Slotine and Li [3,4,5] and Craig [2] have produced equation 2.10 from a Lyapunov stability analysis. The estimated dynamic parameters $\hat{a}$ can be updated by using some appropriate integration to provide a recursive evaluation scheme for $\hat{a}$.

The work of Slotine and Li [3,4,5] indicated that the equations of motion for the manipulators were evaluated explicitly. These two investigators stated that the presence of both $\dot{q}_r$ and $\dot{q}$ in the system model precluded the direct implementation of a Newton-Euler scheme to evaluate the arm control law.

Newton-Euler Recursive Torque Calculations

Luh, Walker, and Paul [1] developed a fast recursive algorithm for calculating the torque required to control a serial link manipulator. The method provides torques

equivalent to that found with equation 2.1. The method is broken into a forward pass and a backward pass in which Newton's and Euler's equations of motion are applied.

The forward pass starts at the base, link 0, and works out to the end of the manipulator. The vectors for angular velocity ω_i, angular acceleration $\dot{\omega}_i$, frame acceleration $\dot{v}_i$, mass center acceleration $\dot{\hat{v}}_i$, mass center inertial forces F_i, and the time rate of change of the angular momentum about the mass center, N_i, are found in the forward pass. These vectors are calculated for link i in the frame attached to link i. These values are calculated from the joint position vector q, joint velocity vector $\dot{q}$ and joint acceleration vector $\ddot{q}$. The vector q consists of the angular position of revolute joints, and the linear position of prismatic joints. The forward pass equations are presented in Table 2.1.

The rotational coordinate transformation A_i^j is found from the kinematic variables and from the position variable q as described by Paul, Shimano, and Mayer [12]. The coordinate transformation is from the coordinate system for link j to the coordinate system for link i when the coordinate systems for the manipulator follow the Denavit-Hartenberg [11] convention. See Appendix 1 for a summary of the coordinate transformation. The position of the current

TABLE 2.1

RECURSIVE NEWTON-EULER FORWARD PASS EQUATIONS

Angular Velocity

$$\omega_i = \begin{cases} A_i^{i-1}\,(\omega_{i-1} + z_0\dot{q}_i) & (R) \\ A_i^{i-1}\,(\omega_{i-1}) & (P) \end{cases}$$

Angular Acceleration

$$\dot{\omega}_i = \begin{cases} A_i^{i-1}[\dot{\omega}_{i-1} + z_0\ddot{q}_i + (\omega_{i-1} \times z_0\dot{q}_i)] & (R) \\ A_i^{i-1}(\dot{\omega}_{i-1}) & (P) \end{cases}$$

Joint Acceleration

$$\dot{v}_i = \begin{cases} A_i^{i-1}\,\dot{v}_{i-1} + \dot{\omega}_i \times {}^{i}p_{i-1}^{i} + \omega_i \times (\omega_i \times {}^{i}p_{i-1}^{i}) & (R) \\ A_i^{i-1}(z_0\ddot{q}_i + \dot{v}_{i-1}) + \dot{\omega}_i \times {}^{i}p_{i-1}^{i} & \\ \quad + 2(\omega_i \times A_{i-1}^{i} z_0\dot{q}_i) + \omega_i \times (\omega_i \times {}^{i}p_{i}^{i-1}) & (P) \end{cases}$$

Mass Center Acceleration

$$\dot{\hat{v}}_i = (\dot{\omega}_i \times \hat{r}_i) + \omega_i \times (\omega_i \times \hat{r}_i) + \dot{v}_i$$

Inertial Force at the Mass Center

$$F_i = m_i \dot{\hat{v}}_i$$

Mass Center Inertial Moment

$$N_i = J_i\,\dot{\omega}_i + \omega_i \times (J_i\,\omega_i)$$

All vectors are referred to the frame attached to link i. The symbols (R) and (P) denote revolute and prismatic joints, respectively. The symbol z_0 denotes the z axis of frame i-1 expressed in frame i-1 which is given by the vector $[0\ 0\ 1]^T$.

link frame relative to the previous link frame is represented by a vector ${}^{k}p_{j}^{i}$. This is the vector from link j coordinate frame to link i coordinate frame represented in the coordinate frame of link k. The relation between two coordinate systems is illustrated in Figure 2.1.

There are several dynamic parameters that must be known for the pure Newton-Euler computed torque method; however estimates of the parameters will be used for the purpose of developing an adaptive control algorithm. These parameters on which the adaptation takes place are the mass of each link m_i, the vector location of the center of mass $\hat{r}_i$ referenced to coordinate system i, and the unique elements of the three by three inertia tensor J_i also referenced to the link coordinate frame.

In the backward pass, the joint reactions n_i and f_i and either the applied torque τ_i or force f_i (also illustrated in Figure 2.1) are found by starting at the farthest link and working inward to the base. The backward pass equations are given in Table 2.2.

Gravity is introduced into the calculations by giving the base, link 0, an acceleration in the opposite direction of gravity. The magnitude of the base acceleration is equal to the acceleration of gravity.

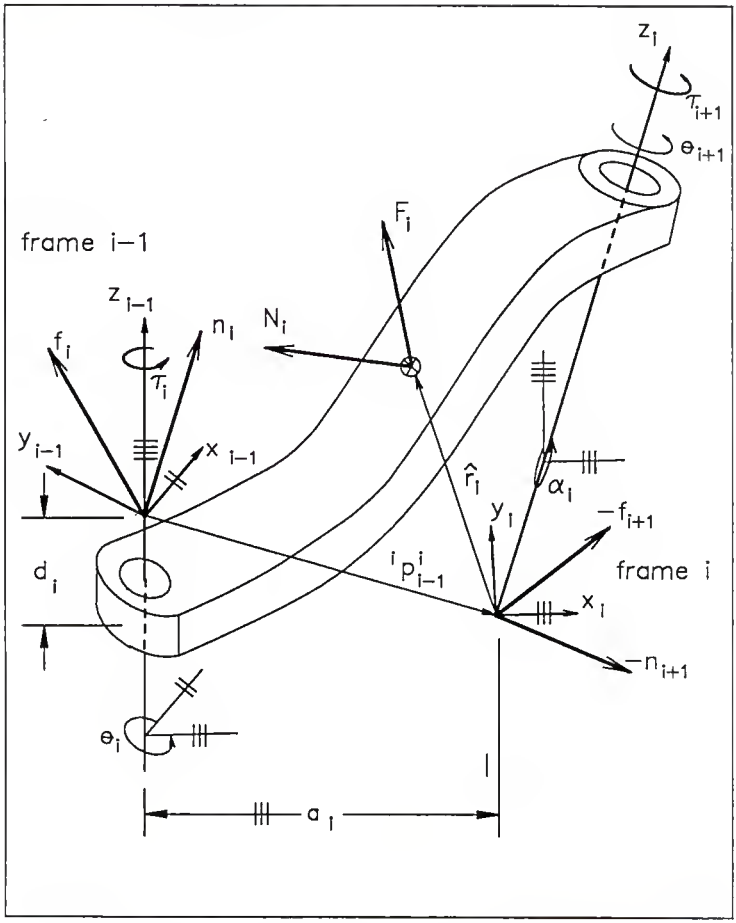


Figure 2.1 Link and Joint Parameters, Numbering, and Coordinate Frames

TABLE 2.2

RECURSIVE NEWTON-EULER BACKWARD PASS EQUATIONS

Joint Force

$$f_i = A_i^{i+1}(f_{i+1}) + F_i$$

Joint Moment

$$n_i = A_i^{i+1}[n_{i+1} + (A_{i+1}^{i}\ {}^{i}p_{i-1}^{i}) \times f_{i+1}]$$

$$+ ({}^{i}p_{i-1}^{i} + \hat{r}_i) \times F_i + N_i$$

Torque

$$\tau_i = \begin{cases} (n_i)^T (A_i^{i-1} z_0) + b_i \dot{q}_i & (R) \\ (f_i)^T (A_i^{i-1} z_0) + b_i \dot{q}_i & (P) \end{cases}$$

All vectors are referred to the frame attached to link i. The symbols (R) and (P) denote revolute and prismatic joints, respectively. The quantity b_i is the viscous friction coefficient for joint i.

Development of the Recursive Adaptive Control Equations

The control law presented by Slotine and Li [3,4,5] requires the explicit dynamic equations for each manipulator, a time consuming task in both derivation and evaluation. It would be desirable to have a recursive general implementation of the control law. Although the adaptive control law given in equations 2.2 looks very similar to the general control law given in equation 2.1, they are very different. The adaptive control law of equation 2.2 is generated using variables q, $\dot{q}$, $\dot{q}_r$, and $\ddot{q}_r$. The recursive Newton-Euler control equations produce the results of equation 2.1, but the Newton-Euler equations can not be used directly to obtain the results of equation 2.2. The Newton-Euler equations are linear with respect to the $\ddot{q}$ vector and can be replaced with the $\ddot{q}_r$ vector. The presence of both $\dot{q}$ and $\dot{q}_r$ prevents the direct substitution into the Newton-Euler recursive equations.

Craig [2] pointed out that the $C(q,\dot{q})\dot{q}$ term in equation 2.1 can be rewritten as

$$C(q,\dot{q})\dot{q} = \begin{bmatrix} \dot{q}^T C_1(q)\ \dot{q} \\ \dot{q}^T C_2(q)\ \dot{q} \\ \vdots \\ \dot{q}^T C_n(q)\ \dot{q} \end{bmatrix} \tag{2.11}$$

where each $C_i(q)$ is a symmetric matrix. From equation 2.11 it can be seen that

$$C(q,\dot{q})\dot{q}_r = C(q,\dot{q}_r)\dot{q}. \tag{2.12}$$

For a recursive method to work, it will have to provide the symmetry exhibited in equations 2.11 and 2.12.

Writing out the explicit form of Newton's equations from the Newton-Euler recursive equations for revolute joints yields insight into a solution to the symmetry problem. The explicit equation for angular velocity, ω, yields the general equation of

$$\omega_i = \sum_{j=1}^{i} (A_i^{j-1} \dot{q}_j z_0). \tag{2.13}$$

Writing out the explicit equations for angular acceleration, $\dot{\omega}$, yields the general equation of

$$\dot{\omega}_i = \sum_{j=1}^{i} A_i^{j-1} \ddot{q}_j z_0 + \sum_{j=1}^{i-1} A_i^j \left[\sum_{k=1}^{j} \dot{q}_{j+1} \dot{q}_k \left[A_j^{k-1} z_0 \times \right] \right] z_0 \tag{2.14}$$

where $\times$ is the cross product operator such that $a \times b = [a\times]b$. See Appendix 2 for details of the cross product operator $[a\times]$. Equation 2.14 can be written so that the products of $\dot{q}_{j+1}\ \dot{q}_k$ occur in symmetric pairs, yielding

$$\dot{\omega}_i = \sum_{j=1}^{i} A_i^{j-1} \ddot{q}_j z_0 + \frac{1}{2} \sum_{j=1}^{i-1} A_i^j \left[\sum_{k=1}^{j} \left(\dot{q}_{j+1} \dot{q}_k + \dot{q}_k \dot{q}_{j+1} \right) \left[A_j^{k-1} z_0 \times \right] \right] z_0. \tag{2.15}$$

The symmetry of the second term of equation 2.15 is better seen by writing it in expanded form for i having a value of three. This gives

$$\frac{1}{2}\sum_{j=1}^{2} A_3^j \left[\sum_{k=1}^{j}\left(\dot{q}_{j+1}\,\dot{q}_k + \dot{q}_k\,\dot{q}_{j+1}\right)\left[A_j^{k-1}\, z_0\times\right]\right] z_0$$

$$= \frac{1}{2}\left\{\left(\dot{q}_2\,\dot{q}_1 + \dot{q}_1\,\dot{q}_2\right)\left(A_3^0\, z_0 \times A_3^1\, z_0\right)\right.$$

$$+\left(\dot{q}_3\,\dot{q}_1 + \dot{q}_1\,\dot{q}_3\right)\left(A_3^0\, z_0 \times A_3^2\, z_0\right)$$

$$\left.+\left(\dot{q}_3\,\dot{q}_2 + \dot{q}_2\,\dot{q}_3\right)\left(A_3^1\, z_0 \times A_3^2\, z_0\right)\right\}$$

$$= \frac{1}{2}\begin{bmatrix}\dot{q}_1\\ \dot{q}_2\\ \dot{q}_3\end{bmatrix}^T\begin{bmatrix} 0 & A_3^0 z_0\times\ A_3^1 z_0 & A_3^0 z_0\times\ A_3^2 z_0\\ A_3^0 z_0\times\ A_3^1 z_0 & 0 & A_3^1 z_0\times\ A_3^2 z_0\\ A_3^0 z_0\times\ A_3^2 z_0 & A_3^1 z_0\times\ A_3^2 z_0 & 0\end{bmatrix}\begin{bmatrix}\dot{q}_1\\ \dot{q}_2\\ \dot{q}_3\end{bmatrix} \qquad (2.16)$$

where each element in the matrix is a three by one vector. The results of equation 2.16 clearly shows the same type of symmetry that equation 2.11 showed.

It can also be shown that the resulting Newton-Euler equations are symmetric in terms of the $\dot{q}_i\dot{q}_{r_j}$ products provided the inertial forces and moments are symmetric in terms of these very same products. Furthermore, it can also be shown that the inertial forces and moments are symmetric in terms of $\dot{q}_i\dot{q}_{r_j}$ products provided the kinematic expressions are symmetric in these terms. Equation 2.15 presents such a symmetric form of the kinematic equations, thus the final Newton-Euler equations will be symmetric in the terms of $\dot{q}_i\dot{q}_{r_j}$ products.

If equation 2.15 is written using the vectors $\dot{q}$, $\dot{q}_r$, and $\ddot{q}_r$ and using the symmetric $\dot{q}_i\dot{q}_{r_j}$ product terms, the resulting equation will be in terms of the variables required for the adaptive control equation 2.2. The result of rewriting equation 2.15 in terms of $\dot{q}$, $\dot{q}_r$, and $\ddot{q}$ gives

$$\dot{\omega}_{r_i} = \sum_{j=1}^{i} A_i^{j-1} \ddot{q}_{r_j} z_0 + \frac{1}{2} \sum_{j=1}^{i-1} A_i^j \left[\sum_{k=1}^{j} \left(\dot{q}_{j+1} \dot{q}_{r_k} + \dot{q}_{r_{j+1}} \dot{q}_k \right) \left[A_j^{k-1} z_0 \times \right] \right] z_0 \qquad (2.17)$$

where the subscript r on the $\dot{\omega}_r$ indicates that $\dot{q}_r$ and $\ddot{q}_r$ terms were used in the equation.

Using the $\dot{q}_r$ vector in the equation for ω from Table 2.1, a new vector, ω_r can be defined as

$$\omega_{r_i} = A_i^{i-1} \left(\omega_{r_{i-1}} + z_0 \dot{q}_{r_i} \right). \qquad (2.18)$$

The results of equation 2.17 can be simplified by substituting in the equations for ω and ω_r to give

$$\dot{\omega}_{r_i} = A_i^{i-1} \left[\dot{\omega}_{r_{i-i}} + z_0 \ddot{q}_{r_i} + \frac{1}{2} \left(\omega_{r_{i-1}} \times \dot{q}_i z_0 + \omega_{i-1} \times \dot{q}_{r_i} z_0 \right) \right]. \qquad (2.19)$$

Similar operations can be performed on the recursive Newton-Euler equations given in Table 2.1, modifying them to use the vectors $\dot{q}$, $\dot{q}_r$, and $\ddot{q}_r$. This gives the forward symmetric recursive equations as shown in Table 2.3. The values for joint force, joint moment, and driving torque are then calculated using the equations of Table 2.2.

TABLE 2.3

SYMMETRICAL RECURSIVE NEWTON-EULER
FORWARD PASS EQUATIONS

Angular Velocity

$$\omega_i = \left\{ \begin{array}{ll} A_i^{i-1}(\omega_{i-1} + z_0\dot{q}_i) & (R) \\ A_i^{i-1}\ (\omega_{i-1}) & (P) \end{array} \right\}$$

and

$$\omega_{r_i} = \left\{ \begin{array}{ll} A_i^{i-1}(\omega_{r_{i-1}} + z_0\dot{q}_r) & (R) \\ A_i^{i-1}\ (\omega_{r_{i-1}}) & (P) \end{array} \right\}$$

Angular Acceleration

$$\dot{\omega}_{r_i} = \left\{ \begin{array}{ll} A_i^{i-1}[\dot{\omega}_{r_{i-1}} + z_0\ddot{q}_{r_i} & \\ \quad + \frac{1}{2}\left[(\omega_{r_{i-1}} \times z_0\dot{q}_i) + (\omega_{i-1} \times z_0\dot{q}_{r_i})\right] & (R) \\ A_i^{i-1}(\dot{\omega}_{r_{i-1}}) & (P) \end{array} \right\}$$

Joint Acceleration

$$\dot{v}_i = \left\{ \begin{array}{ll} A_i^{i-1}\ \dot{v}_{i-1} + \dot{\omega}_{r_i} \times {}^i p_{i-1}^i + \frac{1}{2}\left[(\omega_{r_i} \times \right. & \\ \quad \left. (\omega_i \times {}^i p_{i-1}^i) + \omega_i \times (\omega_{r_i} \times {}^i p_{i-1}^i)\ \right] & (R) \\ A_i^{i-1}(z_0\ddot{q}_{r_i} + \dot{v}_{i-1}) + \dot{\omega}_{r_i} \times {}^i p_{i-1}^i + & \\ \quad (\omega_{r_i} \times A_{i-1}^i z_0\dot{q}_i) + (\omega_i \times A_{i-1}^i\ z_0\dot{q}_{r_i}) & (P) \\ \quad + \omega_{r_i} \times (\omega_i \times {}^i p_i) + \omega_i \times (\omega_{r_i} \times {}^i p_i^{i-1}) & \end{array} \right\}$$

TABLE 2.3--Continued

Mass Center Acceleration

$$\dot{\hat{v}}_i = \dot{v}_i + (\dot{\omega}_{r_i} \times \hat{r}_i) + \frac{1}{2}\left[\omega_{r_i} \times (\omega_i \times \hat{r}_i) + \omega_i \times (\omega_{r_i} \times \hat{r}_i)\right]$$

Inertial Force at Mass Center

$$F_i = m_i \dot{\hat{v}}_i$$

Mass Center Inertial Moment

$$N_i = J_i \dot{\omega}_i + \frac{1}{2}\left[\omega_{r_i} \times (J_i \omega_i) + \omega_i \times (J_i \omega_{r_i})\right]$$

All vectors are referred to the frame attached to link i. The symbols (R) and (P) denote revolute and prismatic joints, respectively. The symbol of z_0 dentes the z axis of frame i-1 expressed in frame i-1 which is the vector $[0\ 0\ 1]^T$.

Factoring the Control Equations with Respect to the Adaptive Parameters

There are ten dynamic parameters per joint that may vary significantly from manipulator to manipulator and may also change when the payload is changed. Most of these parameters are hard to determine accurately and are ideal parameters to adapt. These parameters include the six unique terms from the inertia tensor J, the location of center of mass, $\hat{r}$, and the mass, m. The ten dynamic parameters per joint are factored out of the control law equations and are placed in the $\hat{a}$ vector. This requires the torque equations to be linear in terms of all the $\hat{a}$ elements.

Writing out the computed torque equations for the joint moment given in Table 2.2, and substituting for both the link inertial force and moment in terms of the entries of Table 2.1 provides

$$
\begin{aligned}
n_i = {} & A_i^{i+1}\left[\left[\left(A_{i+1}^{i}\,{}^{i}p_{i-1}^{i}\right)\times f_{i+1}\right) + n_{i+1}\right] \\
& + \hat{r}_i \times (m_i\,\dot{v}_i) + {}^{i}p_{i-1}^{i} \times m_i\Big[\,\dot{v}_i \\
& + (\omega_i \times (\omega_i \times \hat{r}_i)) + (\dot{\omega}_i \times \hat{r}_i)\Big] \\
& + \Big\{\hat{r}_i \times m_i\Big[(\omega_i \times (\omega \times \hat{r}_i)) + (\dot{\omega}_i \times \hat{r}_i)\Big] \\
& + \omega_i \times (J_i\,\omega_i) + J_i\,\dot{\omega}_i\Big\}. \qquad (2.20)
\end{aligned}
$$

Some of the cross products of equation 2.20 result in terms containing $m_i \hat{r}_i^2$. Both the mass, m_i, and the location of the center of mass, $\hat{r}_i$, are dynamic parameters that should be included in the $\hat{a}$ vector. It can be shown that the $m_i \hat{r}_i^2$ term can be eliminated by using the unique moment of inertia tensor about the link coordinate frame rather than about the mass center.

The moment equation for a single link about a fixed axis of rotation as illustrated in Figure 2.2 is given as

$$\begin{aligned} n = \hat{r} \times m \dot{\hat{v}} + \omega \times (J\,\omega) + J\,\dot{\omega} = \\ \hat{r} \times m\,[\omega \times (\omega \times \hat{r}) + \dot{\omega} \times \hat{r}] \\ + \omega \times (J\,\omega) + J\,\dot{\omega}\ . \end{aligned} \tag{2.21}$$

The moment of inertia tensor about the mass center, J, can be transformed to the moment of inertia tensor about the link frame, $\bar{J}$, by the parallel axis theorem [13]

$$\bar{J} = J + m([\hat{r}\cdot\hat{r}][I] - \hat{r}\,\hat{r}^T). \tag{2.22}$$

Using the moment of inertia tensor about the link frame, $\bar{J}$, equation 2.21 can be written as

$$n = \omega \times (\omega\,\bar{J}) + \bar{J}\,\dot{\omega}. \tag{2.23}$$

The last two lines of equation 2.20 are equivalent to equation 2.21. Substituting equation 2.23 into equation 2.20 for the last two lines and reducing gives

$$\begin{aligned} n_i = A_i^{i+1} \left[\left(\left(A_{i+1}^{i}\ {}^{i}p_{i-1}^{i} \right) \times f_{i+1} \right) + n_{i+1} \right] \\ + (m_i \hat{r}_i) \times \dot{v}_i + {}^{i}p_{i-1}^{i} \times (m_i\,\dot{\hat{v}}_i) + M_i \end{aligned} \tag{2.24}$$

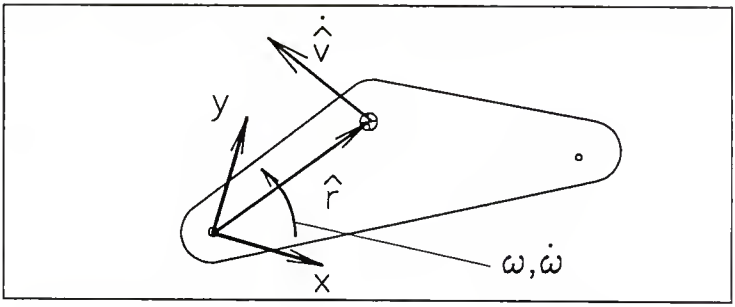


Figure 2.2 Dynamic Parameters of a Single Link

where

$$M_i = \omega_i \times (\omega_i \bar{J}_i) + \bar{J}_i \dot{\omega}_i. \tag{2.25}$$

Every occurrence of the location of the center of mass in equation 2.25 is multiplied by the mass, thus the torque is not a linear function of $\hat{r}$. If a vector $\bar{r}$ is defined to be

$$\bar{r} = (m \hat{r}) \tag{2.26}$$

and used as a new adaptive parameter to replace all occurrences of $(m \hat{r})$, Euler's equations will be linear functions of the all adaptive parameters.

The forward pass of the Newton-Euler symmetric equations is changed by eliminating the calculations of the acceleration of the center of mass and calculating the inertial force at the center of mass directly using $\bar{r}$ and m. The inertial forces at the center of mass, in terms of the adaptive parameters and acceleration of the link frame, is

$$F_i = m_i \dot{v}_i + (\dot{\omega}_{r_i} \times \bar{r}_i) + \frac{1}{2}\left[\omega_{r_i} \times (\omega_i \times \bar{r}_i) + \omega_i \times (\omega_{r_i} \times \bar{r}_i)\right]. \tag{2.27}$$

The time rate of change of angular momentum about the center of mass is no longer calculated, but the M_i term as defined by equation 2.25 is found. M_i can best be described as the time rate of change of the angular momentum about the coordinate axis with the acceleration of the coordinate axis set to zero. Rewriting equation 2.25 in symmetric form gives

$$M_i = \bar{J}_i \dot{\omega}_{r_i} + \frac{1}{2}\left[\omega_{r_i} \times (\bar{J}_i \omega_i) + \omega_i \times (\bar{J}_i \omega_{r_i}) \right]. \quad (2.28)$$

Only the joint moment equation changes in the backward pass of the Newton-Euler equations. Modifying the equation for the joint moment to account for M_i and $\hat{r}_i$ gives

$$n_i = A_i^{i+1} \left[(A_{i+1}^i \; {}^i p_{i-1}^i) \times f_{i+1} + n_{i+1} \right] + \bar{r}_i \times \dot{v}_i + {}^i p_{i-1}^i \times m_i \dot{\hat{v}}_i + M_i. \quad (2.29)$$

Additional parameters can be adapted providing that torque can be expressed as a linear function of these additional quantities. The most common additional terms are frictional parameters. The number of frictional parameters depends on the friction model being used and are normally applied to the final joint actuation calculation. The frictional model is not included within the actual dynamic model.

Matrix Implementation of the Recursive Adaptive Control

The computer implementation of the control law is recursive and very straight forward. The procedure is broken down into three parts which are the forward pass, the backward pass, and the parameter update together with the torque calculation.

The forward pass starts at the base of the manipulator and works out link by link to find the angular velocity, angular acceleration, and joint acceleration of each link. The equations are the symmetric recursive kinematic

equations given in Table 2.3. The acceleration of the center of mass is not calculated, and no adapted dynamic parameters are used in the forward pass.

The backward pass starts at the end of the manipulator and works back to the base, calculating a matrix to find the driving torques and forces and intermediate values. The resulting matrix, the Y matrix, is used with the estimated dynamic parameters in the $\hat{a}$ vector to calculate the joint torques and forces and to perform the parameter adaptation. The format of the $\hat{a}$ vector must be chosen before the equations for the Y matrix can be written. The format for the $\hat{a}$ vector is

$$\hat{a} = \left[\cdots \bar{J}_{i_{11}} \; \bar{J}_{i_{12}} \; \bar{J}_{i_{13}} \; \bar{J}_{i_{22}} \; \bar{J}_{i_{23}} \; \bar{J}_{i_{33}} \; \bar{r}_{i_x} \; \bar{r}_{i_y} \; \bar{r}_{i_z} \; m_i \cdots\right]^T$$

so that there are 10n parameters for a n link manipulator. If any frictional parameters exist, they are then added to the end of the dynamic parameter vector. Once the format of the $\hat{a}$ vector is chosen, the adapted parameters can be factored out of the Newton-Euler equations of the backward pass and the equation put in matrix form.

The Y matrix is found from the joint reaction equations. The force reaction occurring at joint i is placed in matrix form by the operation

$$f_i = [f_i'] \hat{a} \tag{2.30}$$

where f_i' is a 3 x 10n matrix for an n link manipulator. The f_i' matrix is generated from the f_{i+1}' matrix, the coordinate

transformations, and the matrix form of the inertial forces at the mass center. The matrices $[f'_i]$ are generated recursively in the backward pass. The matrix form of the inertial forces at the mass center is found by factoring the adapted parameters from equation 2.27, giving

$$F_i = [F'_i] \left[\bar{r}_i \;\middle|\; m_i \right]^T \tag{2.31}$$

where F'_i is the 3 x 4 matrix

$$\left[F'_i\right] = \left[\frac{1}{2} \left\{ \left[\omega_{r_i}\times\right]\left[\omega_i\times\right] + \left[\omega_i\times\right]\left[\omega_{r_i}\times\right] \right\} + \left[\dot{\omega}_i\times\right] \;\middle|\; \dot{v}_i \right]. \tag{2.32}$$

The f'_i matrix is then found by

$$f'_i = A_i^{i+1} f'_{i+1} + \left[0 \;\middle|\; F'_i \;\middle|\; 0 \right] \tag{2.33}$$

where F'_i occupies columns 10(i-1)+7 to 10i. Note that equations 2.33 is similar to the first equation in Table 2.2 with the vector $\hat{a}$ factored from the expression.

To find the equations for the joint moment, the matrix form of M_i must be found. Factoring the link frame inertial tensor, $\bar{J}$, from equation 2.28 gives

$$M_i = M'_i \left[\bar{J}_{i_{11}} \; \bar{J}_{i_{12}} \; \bar{J}_{i_{13}} \; \bar{J}_{i_{22}} \; \bar{J}_{i_{23}} \; \bar{J}_{i_{33}} \right]. \tag{2.34}$$

The M'_i matrix is a 3 x 6 matrix that had to be directly evaluated instead of by matrix manipulation. The results of the evaluation gives 18 equations, one for each element of the matrix, and are presented in Appendix 3.

The joint moments are found by factoring equation 2.29 to give

$$n_i = n'_i \hat{a} \tag{2.32}$$

where

$$n_i' = \left[A_i^{i+1} \left\{ \left[\left(A_{i+1}^{i} \, {}^{i}p_{i-1}^{i} \right) \times \right] \left[f_i' \right] + n_{i+1}' \right\} \right.$$

$$+ \left[{}^{i}p_{i-1}^{i} \times \right] \left[0 \;\middle|\; F_i' \;\middle|\; 0 \right] + \left[0 \;\middle|\; M_i' \;\middle|\; 0 \right] \quad (2.35)$$

$$\left. - \left[0 \;\middle|\; \dot{v}_i \times \;\middle|\; 0 \right] \right]$$

where F_i' and $\dot{v}_i$ occur in columns $10(i-1)+7$ to $10i$ and M_i' occurs in columns $10(i-1)+1$ to $10(i-1)+6$. The negative sign on the $\dot{v}$ term stems from changing the order in the cross product $\dot{v}_i \times \bar{r}_i$.

The calculations of the n_i' and the f_i' are used to find the ith row of the Y matrix which is used to find the torque and to run the parameter adaptation. The ith row of the Y matrix is found by

$$Y_{\text{row } i} = \left\{ \begin{array}{ll} \left[A_{i-1}^{i} \, z_0 \right]^T n_i' & \text{(revolute)} \\ \\ \left[A_{i-1}^{i} \, z_0 \right]^T f_i' & \text{(prismatic)} \end{array} \right\}. \quad (2.36)$$

Terms associated with the frictional parameter will be added to the end of each row. If the frictional model is a simple viscous damper model, given by $f = b\,\dot{q}$ for each joint, then there would be n additional columns in the Y matrix for an n link manipulator. The $(10n+i)$th column in the ith row would contain $\dot{q}_i$ while all other terms in that row beyond column $10n+i$ would equal zero.

The final step is to find the torque and update the parameter estimates. The torque is found using equation 2.9. The update of the estimated parameters is found using equation 2.10 and a simple integration routine to obtain the next approximation of $\hat{a}$.

CHAPTER 3

DEVELOPMENT OF A RECURSIVE ALGORITHM FOR CLOSED KINEMATIC CHAINS

Closed Kinematic Chains

Serial links are commonly used for research purposes because of their kinematic simplicity, but closed kinematic chains are common in commercial robots. It is difficult to implement a general recursive control routine for a manipulator that has closed loop kinematics. Luh and Zheng [10] presented a method in which the basic Newton-Euler recursive method could be utilized. This method treated the closed chain as if it was cut open at one of the joints. The joint reactions were found using serial link calculations while treating each joint as if had an actuator. Lagrangian multipliers were then used to close the loop and correct the previously computed joint reactions. The Lagrangian multipliers used to close the kinematic chain have to be determined by a customized calculation for each closed chain.

The method presented in this chapter contains a similar procedure, but the customized Lagrangian multipliers are replaced by a procedure which finds the equilibrium

forces associated with the cut location. The only portion of this procedure that is customized is the kinematic equations which relate the joint position, velocity and acceleration of the dependent joints to the independent joints.

Figure 3.1 consists of a manipulator with a closed kinematic chain which contains free revolute joints and Figure 3.2 consists of a manipulator with free prismatic joints. The closed chain of Figure 3.1 or Figure 3.2 is cut at the free revolute joint T+1 to form two branches. The main branch runs from link J to link L+1 and the alternate branch runs from link J'+1 to link T. Each branch can be made up of any number of joints of any type, but there must be one free joint, K+1, between joint J and joint T+1 and one free joint, S+1, between joint J'+1 and joint T+1. The free joint K+1 may be joint J+1 and may be either revolute or prismatic. The free joint S+1 may be joint J'+1 and may be either revolute or prismatic.

The procedure for finding the driving torques starts by using the recursive algorithms for the serial links to calculate the required torque for each joint. The free joints are treated as if they are actuated and both the main and alternate branches are treated as serial links. The values of ω, $\dot{\omega}$, and $\dot{v}$ for joint J will be used to find the values of ω, $\dot{\omega}$, and $\dot{v}$ for both joints J+1 and J'+1. The

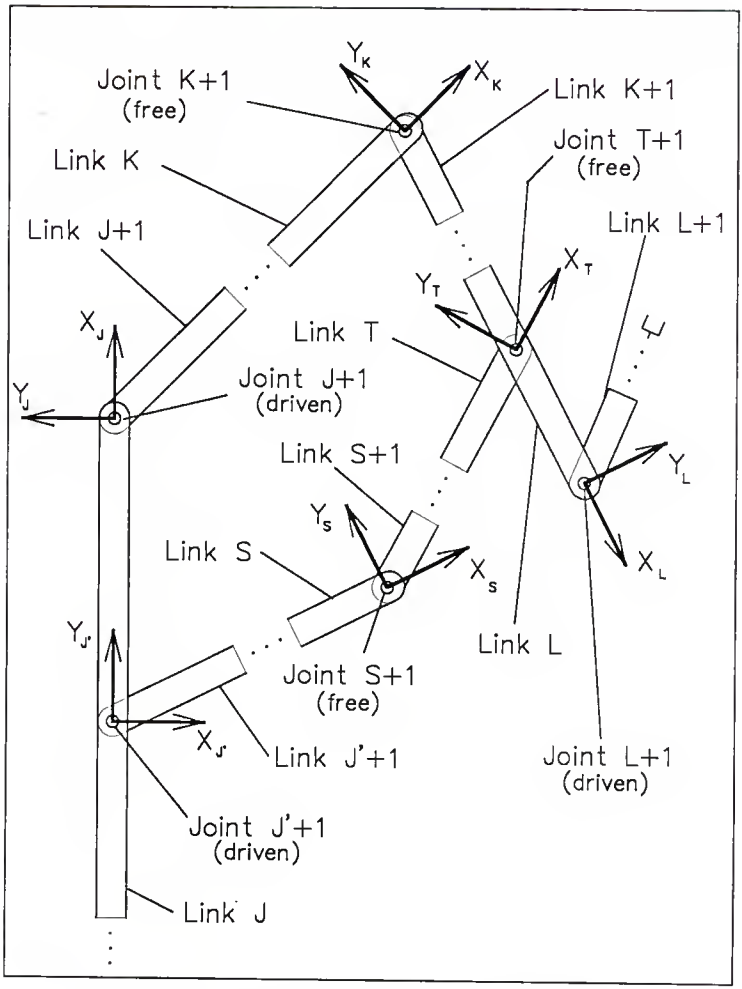


Figure 3.1 Closed Kinematic Chain
With Free Revolute Joints

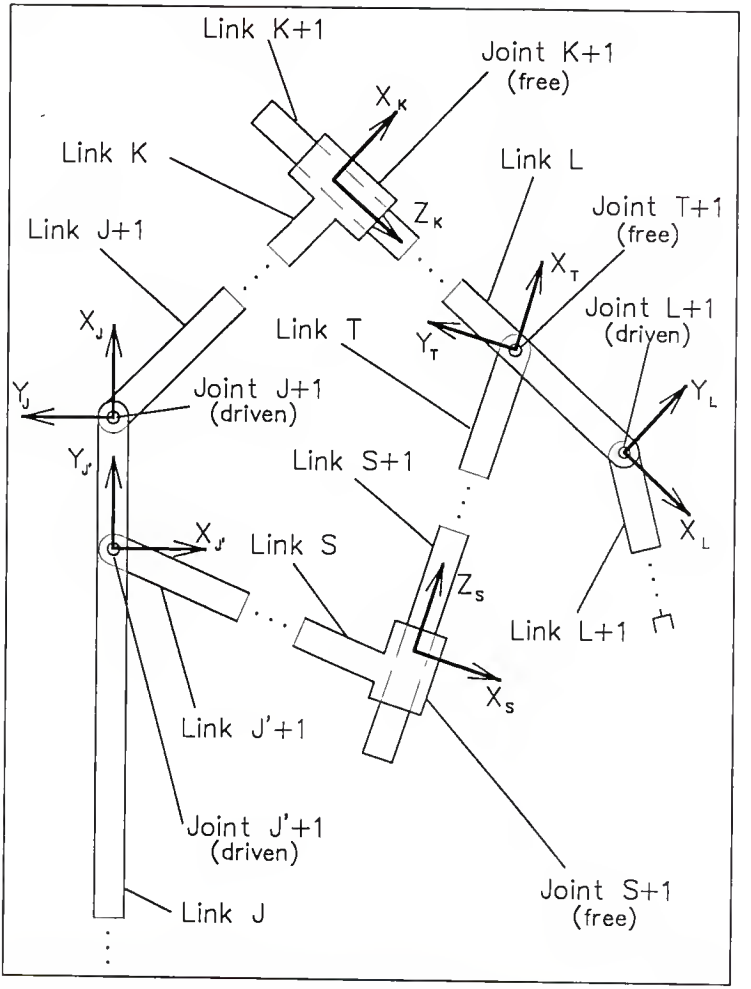


Figure 3.2 Closed Kinematic Chain
With Free Prismatic Joints

calculations of the joint reactions for joint J will include the joint forces and moments from both joint J+1 and joint J'+1.

After finding the reactions, the forces at the cut location, joint T+1 of Figure 3.1 or 3.2, are found by requiring the vanishing of the reactions parallel to the z axis of the free joints K+1 and S+1. The type of free joint, revolute or prismatic, determines the procedure to find the cut location forces.

Cut Location Force Equations

Free Revolute Joints

The only force, not included in the backward Newton-Euler pass for the open serial links, that could influence the moment about the $\hat{z}$ axis of a free revolute joints K+1 or S+1 of Figure 3.1 is the reaction at joint T+1 which occurs when the loop is closed. For a free revolute joint, the sum of the moments about the $\hat{z}$ axis of the joint must vanish. For joint K+1, this produces the relation

$$r_{o_{K+1}} + \left[{}^{K}p_{K}^{T} \times (A_{K}^{T} f_{T+1}) \right]^{T} z_0 = 0 = r_{c_{K+1}}. \qquad (3.1)$$

Likewise the sum of the moments about the $\hat{z}$ axis of S+1 must also equal zero. This fact is expressed as

$$r_{o_{S+1}} + \left[{}^{S}p_{S}^{T} \times (A_{S}^{T} (-f_{T+1})) \right]^{T} z_0 = 0 = r_{c_{S+1}}. \qquad (3.2)$$

The subscript "o" on the torque τ represents the torque with the kinematic loop open while a subscript "c" represents the torque with the kinematic loop closed. See Appendix 4 for the sign convention of the force terms.

The motion of a closed loop kinematic chain is inherently planar and requires that the $\hat{z}$ axis of joint T+1 be parallel to the $\hat{z}$ axis of joints S+1 and K+1. The parallel joints mean that any force f_{T+1_z} can not create a moment about the $\hat{z}$ axis of either joints K+1 or S+1 and therefore does not need to be found. The two forces f_{T+1_x} and f_{T+1_y} can create a moment about the $\hat{z}$ axis of joints S+1 and K+1 and must to be found. The second term of equation 3.1 can be rewritten in matrix form as

$$\left[{}^{K}p_{K}^{T} \times ({}^{T}_{K}A \, f_{T+1}) \right]^{T} z_0 = \left[z_0 \right]^{T} \left[{}^{K}p_{K}^{T} \times \right] \left[{}^{T}_{K}A \right] \left[f_{T+1} \right]$$

$$= B_K \, f_{T+1}. \qquad (3.3)$$

Since Equation 3.1 involves a scalar, the B_K term is a row vector. The second term in equation 3.2 can likewise be rewritten as

$$\left[{}^{S}p_{S}^{T} \times (A_{S}^{T} (-f_{T+1})) \right]^{T} z_{o} = \left[-z_{o} \right]^{T} \left[{}^{S}p_{S}^{T} \times \right] \left[A_{S}^{T} \right] \left[f_{T+1} \right]$$

$$= B_{S} f_{T+1}. \tag{3.4}$$

Free Prismatic Joints

The only force that has not been included in the backward Newton-Euler pass that could influence the force in the $\hat{z}$ direction of the free prismatic joints, either K+1 or S+1 of Figure 3.2, is the force of the cut location, joint T+1. The sum of forces along the $\hat{z}$ axis of either free joint K+1 or S+1 must equal zero. The sum of forces in the $\hat{z}$ direction for joint K+1 is

$$f_{o_{K+1}} + \left[A_{K}^{T} f_{T} \right]^{T} z_{o} = 0 = f_{c_{K+1}}. \tag{3.5}$$

The sum of forces in the $\hat{z}$ direction of joint S+1 gives

$$f_{o_{S+1}} + \left[A_{S}^{T} (-f_{T}) \right]^{T} z_{o} = 0 = f_{c_{S+1}}. \tag{3.6}$$

The quantity f_{o} is the driving force at the free joints before the kinematic chain is closed and f_{c} is the force at the free force with the chain closed.

Closed loops are inherently planar and the $\hat{z}$ axis of a prismatic joint is perpendicular to the $\hat{z}$ axis of the

revolute joint T+1. This means that any forces in the $\hat{z}$ direction of the revolute joint T+1 will not act in the direction of the free prismatic joint of K+1 or S+1 and does not need to be found. This leaves the two forces f_{T+1_x} and f_{T+1_y} that can act in the $\hat{z}$ direction of joint K+1 and S+1 and must to be found using the two equations 3.5 and 3.6. The second term of equation 3.5 can be rewritten in matrix form as

$$\left[A_K^T f_T \right]^T z_0 = \left[z_0 \right]^T A_K^T f_T = B_K f_T \qquad (3.7)$$

While the second term of equation 3.6 can be written in the form of

$$\left[A_S^T (-f_T) \right]^T z_0 = \left[-z_0 \right]^T A_S^T f_T = B_S f_T. \qquad (3.8)$$

The B_S term is a three element row vector.

Finding the Forces at the Cut Location

The B matrix is formed by the B_K and B_S matrices and is used to find the cut location forces by

$$\begin{bmatrix} B_K \\ B_S \end{bmatrix} \begin{bmatrix} f_{T+1_x} \\ f_{T+1_y} \\ f_{T+1_z} \end{bmatrix} = \begin{bmatrix} r_{O_{J+1}} \\ r_{O_{S+1}} \end{bmatrix}. \qquad (3.9)$$

Here and throughout the rest of the chapter, the variable τ is used to represent the driving torque for a revolute joint or driving force for a prismatic joint. The f_{T+1} term is a three by one vector and B is a two by three matrix, providing two equations and three unknowns. The component f_{T+1_z} does not need to be found since it was determined earlier that this component must vanish and can be dropped along with the third column of the B matrix. Dropping the third column of the B matrix forms the B′ matrix. The appropriate equations for the two by two B′ matrix is chosen depending upon the type of the free joint. The reaction at joint T+1 can then be found by

$$\begin{bmatrix} \tau_{O_{J+1}} \\ \tau_{O_{S+1}} \end{bmatrix} = \begin{bmatrix} B'_K \\ B'_S \end{bmatrix} \begin{bmatrix} f_{T+1_x} \\ f_{T+1_y} \end{bmatrix} \tag{3.10}$$

which can be solved by to yield

$$\begin{bmatrix} B'_J \\ B'_S \end{bmatrix}^{-1} \begin{bmatrix} \tau_{O_{K+1}} \\ \tau_{O_{S+1}} \end{bmatrix} = \begin{bmatrix} f_{T+1_x} \\ f_{T+1_y} \end{bmatrix}. \tag{3.11}$$

To complete the f_{T+1} matrix, the f_{T+1_z} term is set to zero.

Correcting the Torques to Include the Cut Location Forces

After finding the forces at the cut location, joint T+1, the effect of that force needs to be included in the

reactions of joints in the same loop below T+1. The only force that was not included in the original calculations was the force at the cut location. Taking into account the torques and forces caused by the reactions at joint T+1, the closed loop torque for a revolute joint on the main branch is

$$\tau_{c_i} = \tau_{o_i} - \left[z_0 \right]^T \left[{}^{i}p_i^T \right] \left[A_i^T \right] f_{T+1} \qquad (3.12)$$

for (J < i < L+1).

The only difference between the main branch and the alternate branch is the direction of the force at joint T+1. This gives the closed loop torque for a revolute joint on the alternate branch as

$$\tau_{c_i} = \tau_{o_i} + \left[z_0 \right]^T \left[{}^{i}p_i^T \right] \left[A_i^T \right] f_{T+1} \qquad (3.13)$$

for (J < i < T+1).

The force at the cut location, joint T+1, has to be included in the driving torque for a prismatic joint. Including the force of joint T+1, the closed loop force for a prismatic joint on the main branch is

$$\tau_{c_i} = \tau_{o_i} - \left[z_0 \right]^T \left[A_i^T \right] f_{T+1} \qquad (3.14)$$

where (J < i < L+1).

The only difference between the main branch and the alternate branch is the direction of the force at joint T+1.

This gives the closed loop force for a prismatic joint on the alternate branch as

$$\tau_{c_i} = \tau_{o_i} + \begin{bmatrix} z_0 \end{bmatrix}^T \begin{bmatrix} A_i^T \end{bmatrix} f_{T+1} \qquad (3.15)$$

where $(J < i < T+1)$.

Any joints below and including joint J and above and including L+1 are entirely correct and are not included in the correction pass. Figure 3.3 gives a view of all different situations encountered while correcting the open loop reactions to obtain the closed loop reactions.

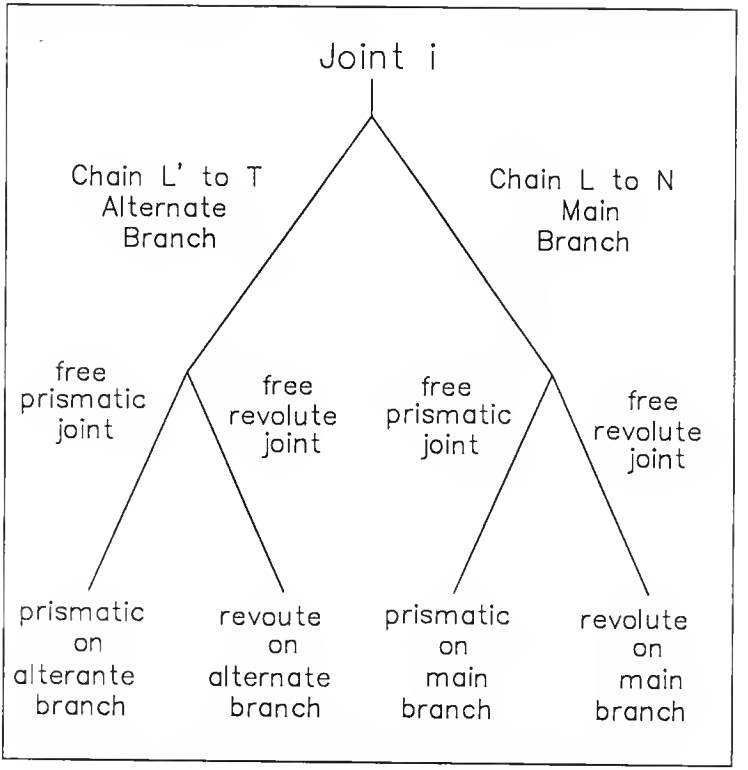


Figure 3.3 Different Conditions Encountered While Closing Kinematic Chains

CHAPTER 4

COMPUTER IMPLEMENTATION AND SIMULATION OF A RECURSIVE ADAPTIVE CONTROL ALGORITHM

The Computer Simulation Program

The simulation of the recursive adaptive control algorithm was done on an Apollo engineering workstation running code written entirely in the "C" language. The computer implementation of a recursive adaptive control algorithm that can handle closed loop kinematic chains requires an exact method of describing the manipulator. The simulation also requires a model of the manipulator that is easy to use for different arm configurations.

Manipulator Data Structures

Walker [7] used a binary tree structure to describe the manipulator where each link could reference as many as two additional links. A similar method is used here to implement the manipulator description for use in the recursive adaptive routines being developed and will be explained in later sections of this chapter.

Link Numbering Scheme

The base of the manipulator is assigned as link 0 and may have a velocity or acceleration associated with it.

Gravity is included in the manipulator dynamics by giving an upward acceleration to the base. The joints and link numbers are assigned so that the joint number at the end of the link closest to the base is the same as the link number as illustrated in Figure 4.1. The free joints in a closed kinematic loop need to be grouped separately from the driven joints. The joint numbers for the driven joints must be less than the joint numbers for the free joints. This is necessary for updating both the control and the parameters in the control algorithm. See Figure 4.1 and Table 4.1 for an example of the joint and link numbering.

Manipulator Branch Data Structure

Two data structures are used to describe the manipulator configuration and are shown in Tables 4.1 and 4.2. The first data structure is the branch data. This structure gives information about the manipulator branches that occur with closed kinematic chains. Branch 0 is defined as the main branch and will be the only branch for a serial link robot. The data structure defines the starting and ending links on the main branch. The starting link for branch 0 is normally the link that is attached to the base. The end effector is normally attached to the end link of branch 0.

Additional branch data structures must be provided for each closed loop in the kinematic chain. This data structure defines the beginning and ending links of the

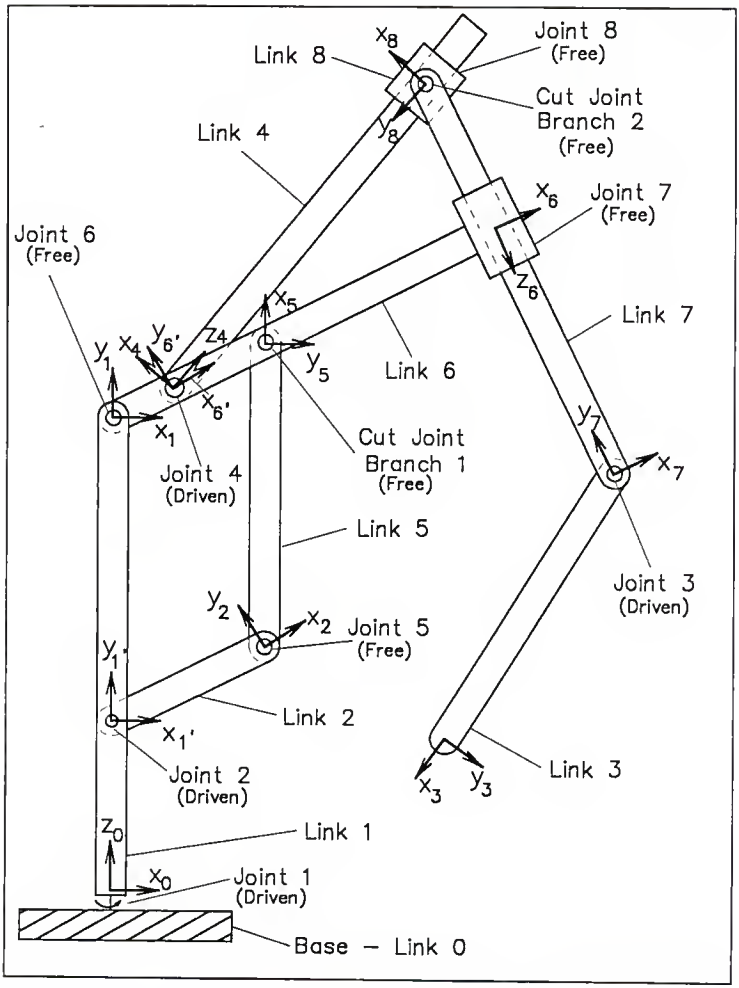


Figure 4.1 Link Numbering for Manipulator With Closed Kinematic Chains

Table 4.1

EXAMPLE OF LINK NUMBERING

LINK NUMBER	PREVIOUS LINK	NEXT LINK	TYPE	MEMBER OF BRANCH (not in data) (structure)
1	0	6	revolute/driven	0
2	1	5	revolute/driven	1
3	7	−1	revolute/driven	0
4	6	8	revolute/driven	2
5	2	6	revolute/free	1
6	2	7	revolute/free	0
7	6	3	prismatic/free	0
8	4	7	prismatic/free	2

TABLE 4.2

EXAMPLE OF BRANCH NUMBERING

BRANCH NUMBER	START LINK	END LINK	FREE JOINT MAIN	FREE JOINT BRANCH
0	1	3	——	——
1	2	5	6	5
2	4	8	7	8

alternate branch as described in Chapter 3. This structure also identifies the free joints on the main and alternate branches that are associated with the closed kinematic chain. See Table 4.2 and Figure 4.1 for an example.

The branch data structure must contain information about the location of the start of the alternate branch with respect the the main branch. This is done using the location and rotational transformations to relate the alternate branch base coordinate system L' to the L coordinate system on the main branch as shown in Figure 4.2. The L' coordinate frame is then used as the base for the alternate branch calculations. The branch start vector, $r_{start'}$ is the vector from the L coordinate frame to the L' coordinate frame given in terms of the L' coordinate system. The branch start vector for branch 1 in Figure 4.1 would be the vector from the link 1 coordinate system to the 1' coordinate system in the terms of 1' coordinate system.

The rotational transformation which maps the link L coordinate system to the L' coordinate system is found from the coordinate transformation parameters α and θ. The α term is the angle of rotation from L_z axis to L'_z axis about the L'_x axis. The θ term is the angle of rotation from the L_x axis to the L'_x axis about the $L_z$axis. See Appendix 1 for details of forming the rotational transformation from α and θ. In Figure 4.1, Table 4.1, and Table 4.2, the coordinate

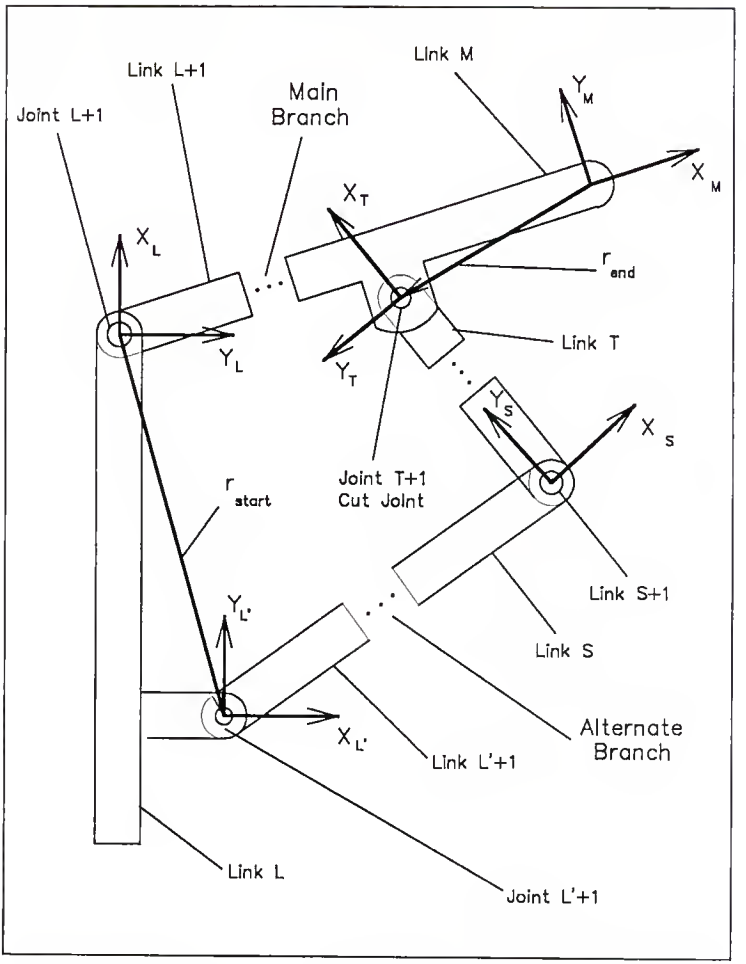


Figure 4.2 Branch Parameters for Closed Kinematic Chains

transformation to the start of branch 1 would be from the link 1 coordinate system to the 1' coordinate system such that a unit vector $\hat{p}$ in frame 1 would be mapped into frame i'. This can be shown as

$$\hat{p}_{1'} = A^{1}_{1'} \hat{p}_{1}. \tag{4.1}$$

The branch data structure must give the location of the closing joint between the alternate branch and the main branch. The position vector r_{end} locates the link T coordinate system with respect to the link M coordinate system in terms of the link M coordinate system as shown in Figure 4.2. The ending vector for branch 2 runs from the coordinate system of link 7 to the coordinate frame for link 8 in Figure 4.1.

The branch data structure also contains the rotational transformation from the link M coordinate system to the link T coordinate system. This would map a unit vector in the M coordinate system to the T coordinate system. The joint where the cut occurs is always a revolute joint and the rotational transformation between the M and T frames change as the manipulator moves which necessitates that this transformation be calculated and updated by the control routine.

For an example of part of the branch data structure, see Figure 4.1 and Table 4.2. See Table 4.3 for a complete listing of the branch data structure.

TABLE 4.3

BRANCH DATA STRUCTURE ELEMENTS

Element Name	Function and Information
start	The link number that is the first link in a branch.
end	The link number of the last link in a branch.
Note: The main branch only uses the data structure variables "start", and "end".	
r_start	User supplied position vector giving the starting location of the alternate branch from the coordinate frame of the link on the main branch which the alternate branch starts. The vector is expressed in terms of the alternate branch starting coordinate frame.
r_end	User supplied position vector describing the end location of the alternate branch with respect to the link coordinate frame of the main branch link on which the alternate branch terminates. The vector is express in terms of the coordinate frame for the link on the main branch.
theta_start	User supplied angle that expresses the rotation from the x axis of the link on the main branch from which the alternate branch starts to the x axis on the starting coordinate frame for the alternate branch; measured about the z axis of the frame on the main branch.
alpha_start	User supplied angle that expresses the rotation from the z axis of the link on the main branch from which the alternate branch starts to the z axis on the starting coordinate frame for the alternate branch; measured about the x axis on the alternate starting coordinate frame.

TABLE 4.3 -- Continued

rot_start	The computer generated rotational transformation used to rotate vectors from the coordinate system of the link from which the alternate branch starts on the main branch to the starting coordinates for the alternate branch. This transformation is built from "theta_start" and "alpha_start"
end_rot	The computer generated rotational transformation used to rotate vectors from the end of the alternate branch to the coordinate frame of the link on the main branch located at the end of the alternate branch.
main_free	The number of the joint on the main branch that is not driven and is associated with the branch.
branch_free	The number of the joint on the alternate branch that is not driven.

Manipulator Link Data Structure

The link data structure provides information about each link in the manipulator. The data structure contains two categories of information, that used by both the controller and the simulation model, and that used only by the simulation model. A data structure exists for each link in the manipulator except the base, link 0.

The information used by both the controller and the simulation model is the joint type, position in the manipulator, and the coordinate frame orientation. The joint type is either revolute or prismatic. The link position in the manipulator is obtained from the link data structures record of the previous link and the next link. The previous link is the link attached to the current link and is closer to the base of the manipulator. The next link is the link attached to the current link and is closer to the end effector. If the current link is the last link of the manipulator, the value of next link is assigned as (-1). See Figure 4.1 and Table 4.1 for an example of the link data structure.

The position and orientation of the link coordinate frame is found by the coordinate transformation data. The rotational transformation is found from the coordinate transformation values of α, and θ as described in Appendix 1. The frame for the first link in an alternate branch is

referenced to the branch base frame rather than the previous link frame as with all other links. The value of α is constant and is stored in the link data structure. The value of θ for a revolute joint is the joint variable and is obtained from the kinematic variables of the manipulator. The value of θ for a prismatic joint is constant and is stored in the link data structure. The link data also contains the position vector ${}^{i}p_{i-1}^{i}$. This is a vector from the previous link coordinate frame to the current link coordinate frame in terms of the current link coordinate frame as illustrated in Figure 2.1. The position vector for a link that is the first link in an alternate branch is from the branch base frame instead of the previous link frame. The value of the position vector is constant for revolute joints, but varies with prismatic joints. The joint variable for the prismatic joint, "d", is the distance from the origin of the previous frame to the origin of the current frame along the z axis of the previous frame. The position vector is then found using the kinematic values of "a" and θ from the data structure and the joint variable "d". See Appendix 1 for details on how to calculate the position vector.

The second part of the link data structure is the information used only by the plant model during simulation. This includes the link mass, location of the mass center, three by three inertia tensor about the mass center, and the

viscous friction coefficient required by the model. See Table 4.4 for a complete listing of the link data structure.

The Simulation Model

The simulation of a manipulator is done by solving for the joint acceleration vector, $\ddot{q}$, from the torque vector, τ, and the vectors of q and $\dot{q}$. New values for the $\dot{q}$ vector are then found be integrating the $\ddot{q}$ vector. Likewise new values for the q vector are found be integrating the $\dot{q}$ vector.

The dynamic model of a manipulator can be reduced to the simple equation of

$$\tau = H(q)\ \ddot{q} + C(q,\dot{q})\ \dot{q} + G(q). \qquad (4.2)$$

The $\ddot{q}$ vector can be found from knowing the torque vector τ and the H, C, and G matrices which are found from the q and $\dot{q}$ vectors. The equations for the matrices H, C, and G are difficult to develop and are different for each manipulator. The H, C and G matrices in equation 4.2 can be developed using the Newton-Euler recursive algorithm.

The H matrix is a n x n matrix where n is the number of joints. The H matrix is found by setting the base acceleration vector to zero, which eliminates the G(q) vector, and all $\dot{q}$ terms to zero, which eliminates the $C(q,\dot{q})\dot{q}$ term. Then the i^{th} column of the H matrix is found by setting all $\ddot{q}$ terms to zero except for $\ddot{q}_i$ which is set to unity. The torques resulting from the evaluation of the

TABLE 4.4

LINK DATA STRUCTURE ELEMENTS

Element Name	Function and Information
type	Joint type, 'r' for revolute, 'p' for prismatic.
alpha	Expresses the angle of rotation from the z axis of the previous joint coordinate system to the z axis of the present joint coordinate system about the x axis of the present joint coordinate system.
a	Expresses the displacement of the current link coordinate system from the previous link coordinate system along the x axis of the current coordinate system.
d	This variable expresses the displacement of the current link coordinate system from the previous link coordinate system along the z axis of the previous link coordinate system for a revolute joint. This variable expresses the angular rotation from x axis of the current link coordinate system to the x axis of the previous link coordinate system about the z axis of the previous coordinate system for a prismatic joint.
p	Computer generated vector that expresses the location of the current link coordinate system with respect to the previous link coordinate system. This vector is expressed in terms of the current link coordinate system.
next	The number of the next link in the chain.
prev	The number of the previous link in the chain.

TABLE 4.4 -- Continued

The following portion of the data structure is used only by the simulation model.

s	The position vector expressing the location of the center of mass with respect the the link coordinate frame.
mass	The mass of the link.
inertia	The three by three inertia tensor for the link about the center of mass for the link.
fric	The viscous friction coefficient for the link friction model.

Newton-Euler recursive algorithm become the i^{th} column of the H matrix. The entire H matrix is formed by performing this routine n times to obtain all n columns.

The C and G matrixes in equation 4.1 can be combined as

$$Q(q,\dot{q}) = C(q,\dot{q})\ \dot{q} + G(q). \qquad (4.3)$$

The Q vector is found by setting the $\ddot{q}$ vector to zero, eliminating the $H(q)\ddot{q}$ terms, and evaluating the Newton-Euler recursive algorithm with the base acceleration included and the $\dot{q}$ variables equal to their proper values. The resulting torque vector is the Q vector.

This gives the equation for the torque as

$$\tau = H(q)\ddot{q} + Q(q,\dot{q}) \qquad (4.4)$$

where the vector $\ddot{q}$ is the only unknown term.

The algorithm used in the simulation cuts open each kinematic chain and the H matrix and Q vector as if each branch was an open chain. The kinematic chains were then closed by finding the forces at the cut locations so that the free joint $\hat{z}$ axis reaction components go to zero. The reaction at the cut location was then used to modify the values of the H matrix and Q vector to reflect the changes caused by closing the kinematic loops. Zeroing the reactions parallel to the $\hat{z}$ axis of the free joints leaves the rows in the H matrix corresponding to the free joints filled with zeros, giving a singular matrix that can not be solved. The joint acceleration kinematic closure equations for the free joints were used to replace the zeroed rows of

the H matrix and Q vector. Then the vector $\ddot{q}$ can be found by solving equation 4.4. This gives the solution for $\ddot{q}$ as

$$\ddot{q} = [H(q)]^{-1} [\tau - Q(q,\dot{q}]. \qquad (4.5)$$

The joint velocity vector $\dot{q}$ and position vector q are then updated by using a fourth order, fixed step Runga-Kutta integration technique. The accumulation of roundoff and integration error in the dependent links of a closed kinematic chain may prevent kinematic closure from occurring and cause trouble for the controller. If this happens, new values for the states of the dependent variables can be found from the independent variables using the kinematic equations. When necessary, the states of the dependent variables were corrected at the same time as the new desired positions were found.

User Supplied Routines

The user of the simulation routine has to provide one routine for manipulators with only serial links, and three routines for manipulators with closed kinematic chains. The names of these routines are defined at the top of the driving routine before compiling. The routine to find the desired positions, velocity, and accelerations vectors is required for all types of manipulators. This routine is defined as POS_CALC at the top of the driving routine. A routine to find the values for the $\dot{q}_r$ and $\ddot{q}_r$ vectors for the dependent link joints from the independent link joints must

be provided. The routine "serial_p_close" is provided for serial link robots. This routine serves only as a placeholder to satisfy the compiler. This routine is defined as POS_CLOSE at the top of the driving routine. The last routine that is required is a routine to close the system model during simulation. This routine contains the equations for the acceleration of the dependent link joints in terms of the independent link joints. The routine "serial_close" is provided for serial link routines. This routine is only a placeholder so the program will compile. The routine is defined as SYSTEM_CLOSE at the top of the driving routine.

Chapter 5

RESULTS AND PERFORMANCE OF THE ADAPTIVE CONTROLLER

Verification of the Controller and the Simulation Model

The control algorithm was tested using the simulation routine developed in Chapter 4. Simulations were performed on a series of serial link manipulators and on manipulators containing closed kinematic chains.

The control program was verified by running the controller with three different manipulators using an exact model of the systems. The adapted parameters in the $\hat{a}$ matrix were set at the actual values and the adaptation was turned off by setting the values of Γ to zero. The values in the K_d and λ matrixes were set low. With the dynamic parameters in $\hat{a}$ set exactly, the output should exactly follow the desired path.

The three test problems were a double pendulum, an R-theta manipulator, and a slider crank with equal length arms. The equations for the exact models of these three manipulators are given in Appendix 5. Figure 5.1 shows the double pendulum configuration and Figure 5.2 and Table 5.1 show the results of the verification test run. Figure 5.3

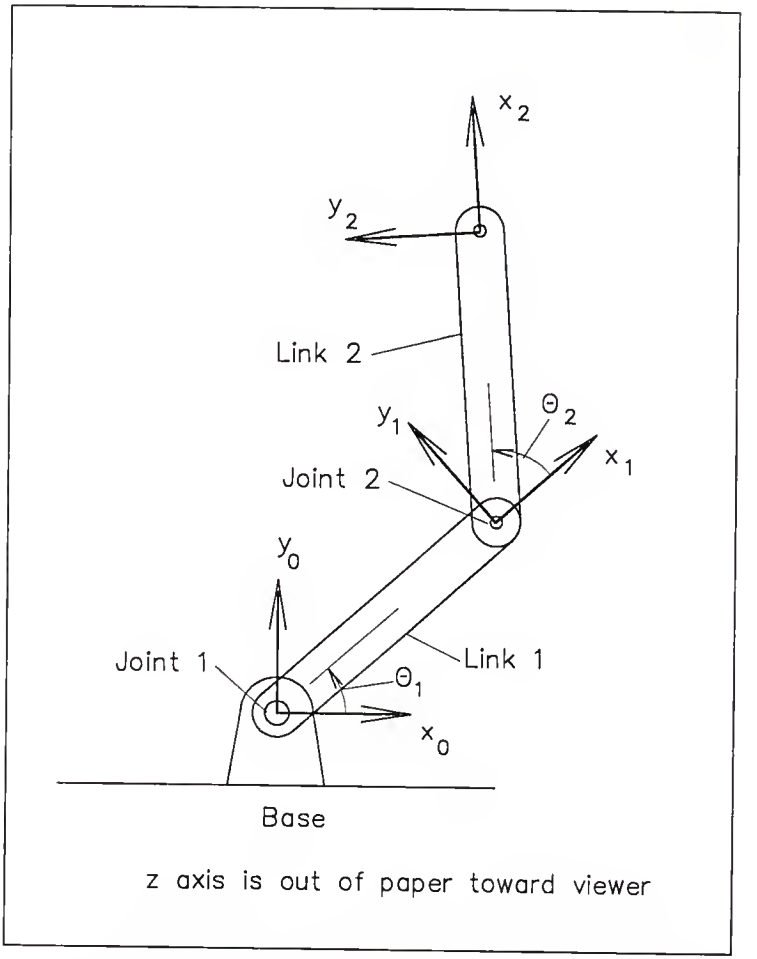


Figure 5.1 Double Pendulum

TABLE 5.1

DOUBLE PENDULUM VERIFICATION RUN INFORMATION

```
   Double Pendulum


Constant parameters for the simulation are:
 Simulation Time parameters:
    Starting time = 0.00          Ending time = 60.00
    System time step = 0.0100 seconds
    Control time step = 0.0100 seconds


 Control gain parameters lamba:
    lamba[0] = 1.00      lamba[1] = 1.00

 Control parameters Kd:
    Kd[0] = 1.00         Kd[1] = 1.00


 Adaptive gain parameters, gamma, are:

    Link 1:
           0.00      0.00      0.00      0.00      0.00      0.00
           0.00      0.00      0.00      0.00

    Link 2:
           0.00      0.00      0.00      0.00      0.00      0.00
           0.00      0.00      0.00      0.00

    Friction values:
           0.00      0.00


Link parameter information:

For Joint 1 (Revolute):
 previous link = 0 next link = 2
 alpha =  0.00000     a =  6.00000     d = 0.00000

For Joint 2 (Revolute):
 previous link = 1  last link of manipulator
 alpha =  0.00000     a =  6.00000     d = 0.00000
```

TABLE 5.1 -- Continued

```
Branch parameter information

 Branch 0:
  Starting Link = 1 Ending Link = 2


 Model Parameters:
 Joint 1:
  Mass = 1.250   Friction = 0.000
  Center of mass:   x = -3.000    y = 0.000    z = 0.000
  Moment of Inertia about Center of mass:
            0.00          0.00          0.00
            0.00          3.75          0.00
            0.00          0.00          3.75

 Joint 2:
  Mass = 1.000   Friction = 0.000
  Center of mass:   x = -3.500    y = 0.000    z = 0.000
  Moment of Inertia about Center of mass:
            0.00          0.00          0.00
            0.00          4.00          0.00
            0.00          0.00          4.00


States of the Manipulator Base:

 Angular Velocity:       0.000    0.000    0.000
 Angular Acceleration:   0.000    0.000    0.000
 Base Acceleration:      0.000    0.000    0.000
```

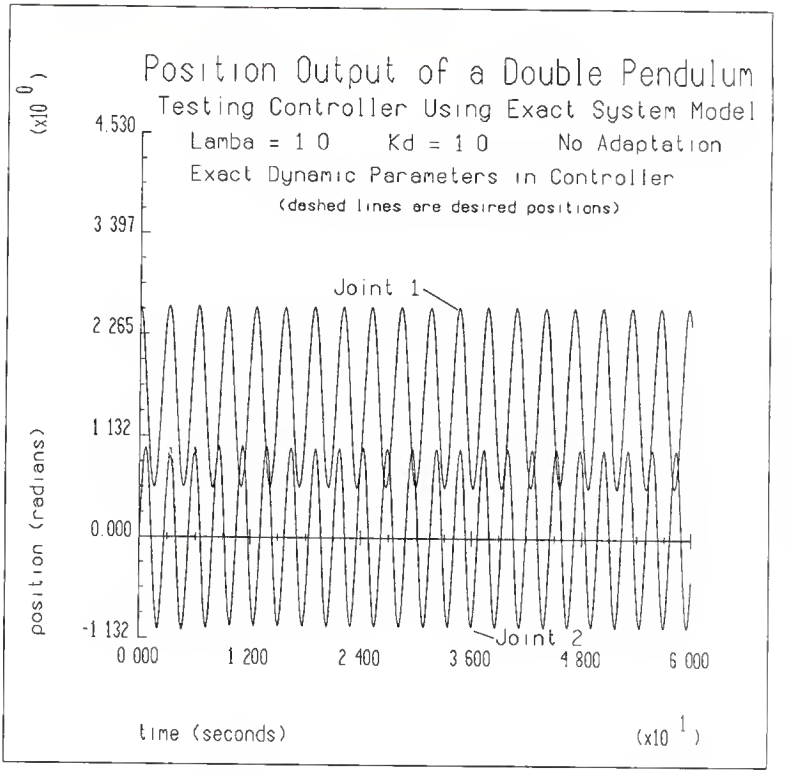


Figure 5.2 Position Output of the Double Pendulum, Verification Run

shows the configurations of the R-Theta manipulator. Figure 5.4 and Table 5.2 give the results of the verification test run. Figure 5.5 shows the configuration of the slider crank with equal length arms and Figure 5.6 and Table 5.3 give the results of the verification run.

The simulation model was then tested by running the same test but with the general simulation model. Once again the output should track the input perfectly. The customized equations relating the position, velocity and acceleration of the dependent joint to the independent joint in a closed kinematic chain were tested in the same manner.

Simulation Results for Serial Link Manipulators

Double Pendulum

The results of the double pendulum gave a lot of insight into the adaptive controller. Two different test are shown, with the only difference being in the step size used for the controller and the integration. Figure 5.1 shows the configuration of the manipulator, and Table 5.4 gives the details of the test run. Figure 5.7 shows the position of the manipulator and Figure 5.8 shows the tracking error, s. Figure 5.9 shows the response of the adapted parameters. The parameters were started at zero and did not go to the actual values, but did drive the tracking error towards zero. Figure 5.10 and Table 2.5 show the

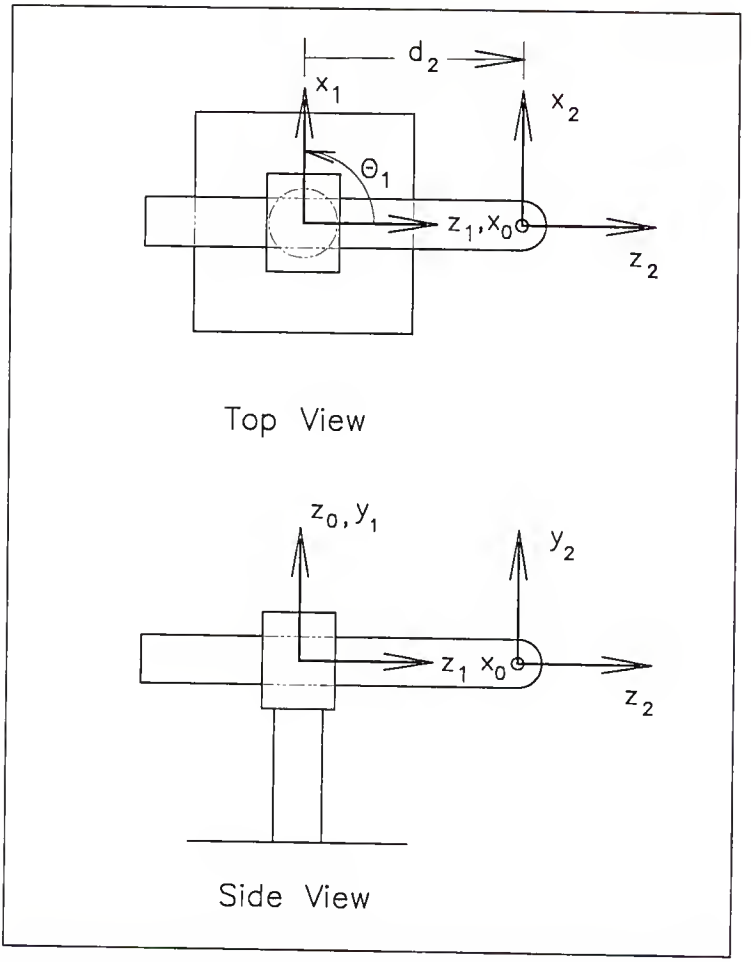


Figure 5.3 R—Theta Manipulator

TABLE 5.2

R-THETA VERIFICATION RUN INFORMATION

R-theta manipulator, Distributed Mass (General)

Constant parameters for the simulation are:

Simulation Time parameters:

Starting time = 0.00 Ending time = 60.00

System time step = 0.0100 seconds

Control time step = 0.0100 seconds

Control gain parameters lamba:

lamba[0] = 1.00 lamba[1] = 1.00

Control parameters Kd:

Kd[0] = 1.00 Kd[1] = 1.00

Adaptive gain parameters, gamma, are:

Link 1:

0.00 0.00 0.00 0.00 0.00 0.00

0.00 0.00 0.00 0.00

Link 2:

0.00 0.00 0.00 0.00 0.00 0.00

0.00 0.00 0.00 0.00

Friction values:

0.00 0.00

Link parameter information:

For Joint 1 (Revolute):

previous link = 0 next link = 2

alpha = 1.57080 a = 0.00000 d = 0.00000

For Joint 2 (Prismatic):

previous link = 1 last link of manipulator

alpha = 0.00000 a = 0.00000 theta = 0.00000

TABLE 5.2 -- Continued

```
Branch parameter information

        Branch 0:
            Starting Link = 1 Ending Link = 2


 Model Parameters:
        Joint 1:
            Mass = 0.000        Friction = 1.000
            Center of mass:   x = 0.000    y = 0.000    z = 0.000
            Moment of Inertia about Center of mass:
                      0.00          0.00          0.00
                      0.00          5.00          0.00
                      0.00          0.00          0.00

        Joint 2:
            Mass = 10.000       Friction = 1.000
            Center of mass:   x = 0.000    y = 0.000    z = -1.500
            Moment of Inertia about Center of mass:
                      7.50          0.00          0.00
                      0.00          7.50          0.00
                      0.00          0.00          0.00


States of the Manipulator Base:

        Angular Velocity:       0.000    0.000    0.000
        Angular Acceleration:   0.000    0.000    0.000
        Base Acceleration:      0.000    0.000    0.000
```

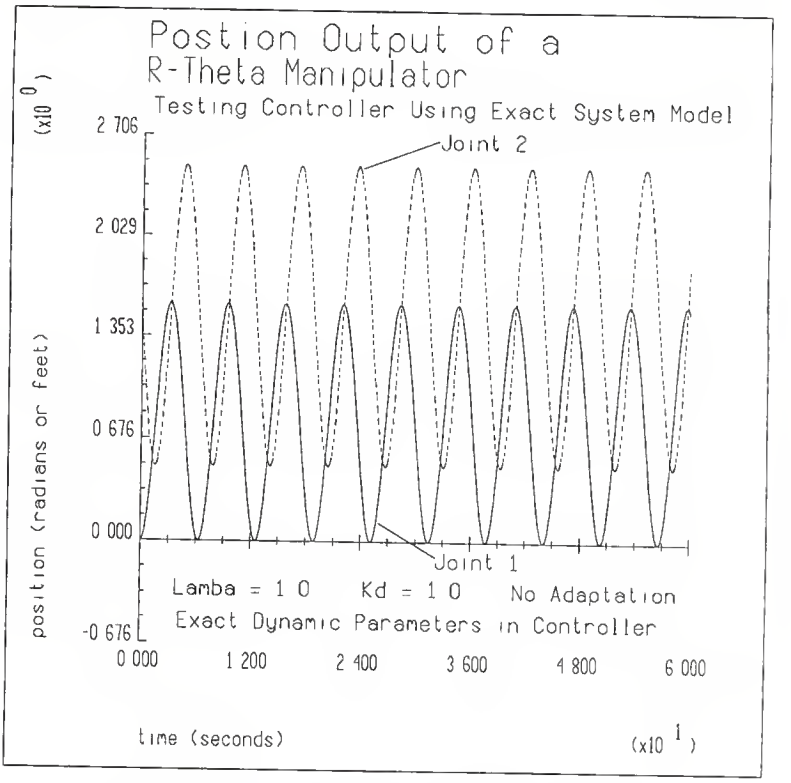


Figure 5.4 Position Output of the R-Theta Manipulator, Verification Run

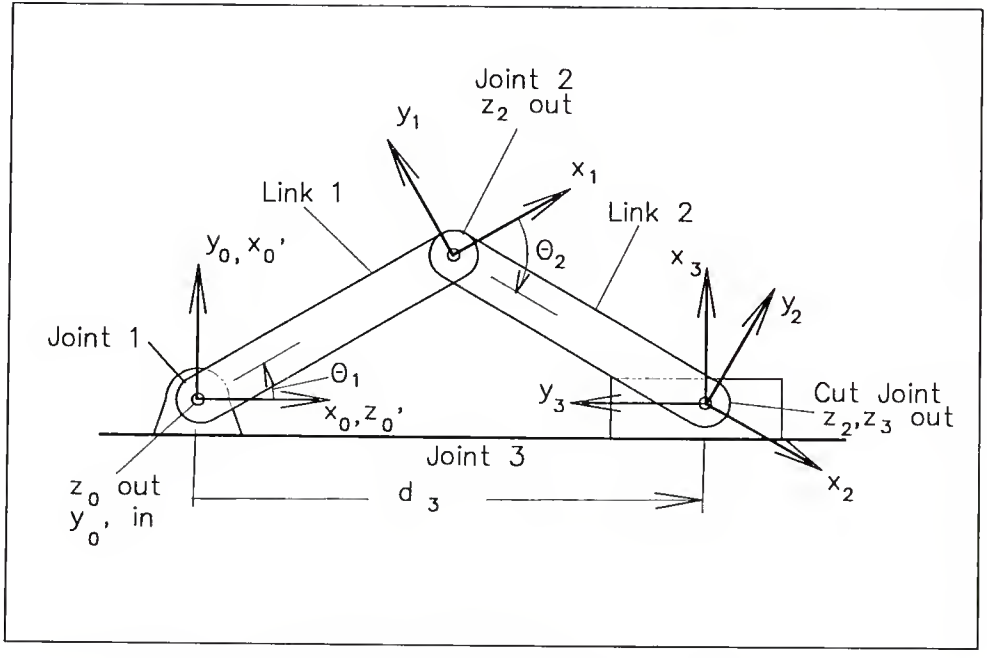


Figure 5.5 Slider Crank With Equal Length Arms

TABLE 5.3

SLIDER CRANK WITH EQUAL LENGTH ARMS
VERIFICATION RUN INFORMATION

```
   This is Slider crank Problem (equal length arms)


Constant parameters for the simulation are:
        Simulation Time parameters:
           Starting time = 0.00          Ending time = 60.00
           System time step = 0.0100 seconds
           Control time step = 0.0100 seconds


        Control gain parameters lamba:
           lamba[0] = 1.00

        Control parameters Kd:
           Kd[0] = 1.00


        Adaptive gain parameters, gamma, are:

           Link 1:
                  0.00     0.00     0.00     0.00     0.00     0.00
                  0.00     0.00     0.00     0.00

           Link 2:
                  0.00     0.00     0.00     0.00     0.00     0.00
                  0.00     0.00     0.00     0.00

           Link 3:
                  0.00     0.00     0.00     0.00     0.00     0.00
                  0.00     0.00     0.00     0.00

           Friction values:
                  0.00     0.00     0.00
```

TABLE 5.3 -- Continued

```
Link parameter information:

        For Joint 1 (Revolute):
            previous link = 0 next link = 2
            alpha =  0.00000     a =  4.00000     d = 0.00000

        For Joint 2 (Revolute):
            previous link = 1  last link of manipulator
            alpha =  0.00000     a =  4.00000     d = 0.00000

        For Joint 3 (Prismatic):
            previous link = 0 next link = 2
            alpha = -1.57080     a =  0.00000     theta = 0.00000


Branch parameter information

        Branch 0:
            Starting Link = 1 Ending Link = 2

        Branch 1:
            Starting Link = 3 Ending Link = 3
            Main branch free joint = 2            branch free joint = 3

            Branch starting theta = 1.5708
            Branch starting alpha = 1.5708

            Vector from link 0 to base of branch in branch base frame:
               x = 0.000       y = 0.000          z = 0.000
            Vector from end of branch to link 2 in link 2 frame:
               x = 0.000       y = 0.000          z = 0.000
```

TABLE 5.3 -- Continued

```
Model Parameters:
      Joint 1:
          Mass = 2.000      Friction = 0.000
          Center of mass:   x = -2.000    y = 0.000    z = 0.000
          Moment of Inertia about Center of mass:
                    0.00          0.00          0.00
                    0.00          2.67          0.00
                    0.00          0.00          2.67

      Joint 2:
          Mass = 2.000      Friction = 0.000
          Center of mass:   x = -2.000    y = 0.000    z = 0.000
          Moment of Inertia about Center of mass:
                    0.00          0.00          0.00
                    0.00          2.67          0.00
                    0.00          0.00          2.67

      Joint 3:
          Mass = 4.000      Friction = 0.000
          Center of mass:   x = 0.000    y = 0.000    z = 0.000
          Moment of Inertia about Center of mass:
                    0.00          0.00          0.00
                    0.00          0.00          0.00
                    0.00          0.00          0.00


States of the Manipulator Base:

      Angular Velocity:      0.000    0.000    0.000
      Angular Acceleration:  0.000    0.000    0.000
      Base Acceleration:     0.000    0.000    0.000
```

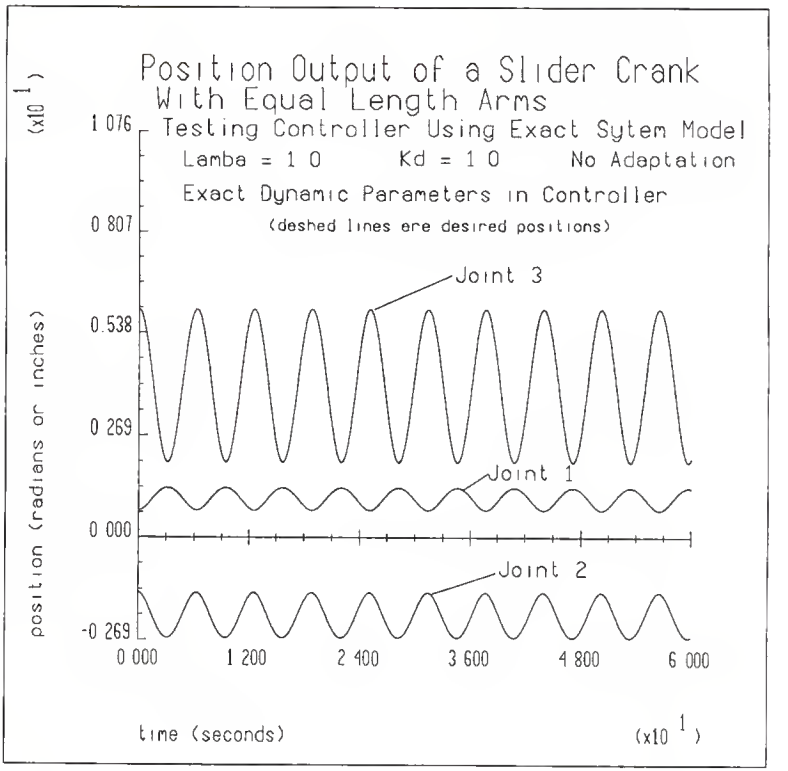


Figure 5.6 Position Output of Slider Crank With Equal Length Arms, Verification Run

TABLE 5.4

DOUBLE PENDULUM RUN INFORMATION, TEST 1

Double Pendulumn

Constant parameters for the simulation are:

Simulation Time parameters:

Starting time = 0.00 Ending time = 60.00

System time step = 0.0100 seconds

Control time step = 0.0100 seconds

Control gain parameters lamba:

lamba[0] = 1.00 lamba[1] = 1.00

Control parameters Kd:

Kd[0] = 15.00 Kd[1] = 15.00

Adaptive gain parameters, gamma, are:

Link 1:

0.10	0.10	0.10	0.10	0.10	0.10
0.01	0.01	0.01	0.01		

Link 2:

0.10	0.10	0.10	0.10	0.10	0.10
0.01	0.01	0.01	0.01		

Friction values

0.02	0.02

TABLE 5.4 -- Continued

```
Link parameter information:

        For Joint 1 (Revolute):
            previous link = 0 next link = 2
            alpha =  0.00000     a =  6.00000     d = 0.00000

        For Joint 2 (Revolute):
            previous link = 1  last link of manipulator
            alpha =  0.00000     a =  6.00000     d = 0.00000


Branch parameter information

        Branch 0:
            Starting Link = 1 Ending Link = 2


Model Parameters:
        Joint 1:
            Mass = 1.250       Friction = 0.000
            Center of mass:    x = -3.000     y = 0.000     z = 0.000
            Moment of Inertia about Center of mass:
                      0.00           0.00           0.00
                      0.00           3.75           0.00
                      0.00           0.00           3.75

        Joint 2:
            Mass = 1.000       Friction = 0.000
            Center of mass:    x = -3.500     y = 0.000     z = 0.000
            Moment of Inertia about Center of mass:
                      0.00           0.00           0.00
                      0.00           4.00           0.00
                      0.00           0.00           4.00


States of the Manipulator Base:

        Angular Velocity:       0.000    0.000    0.000
        Angular Acceleration:   0.000    0.000    0.000
        Base Acceleration:      0.000    9.810    0.000
```

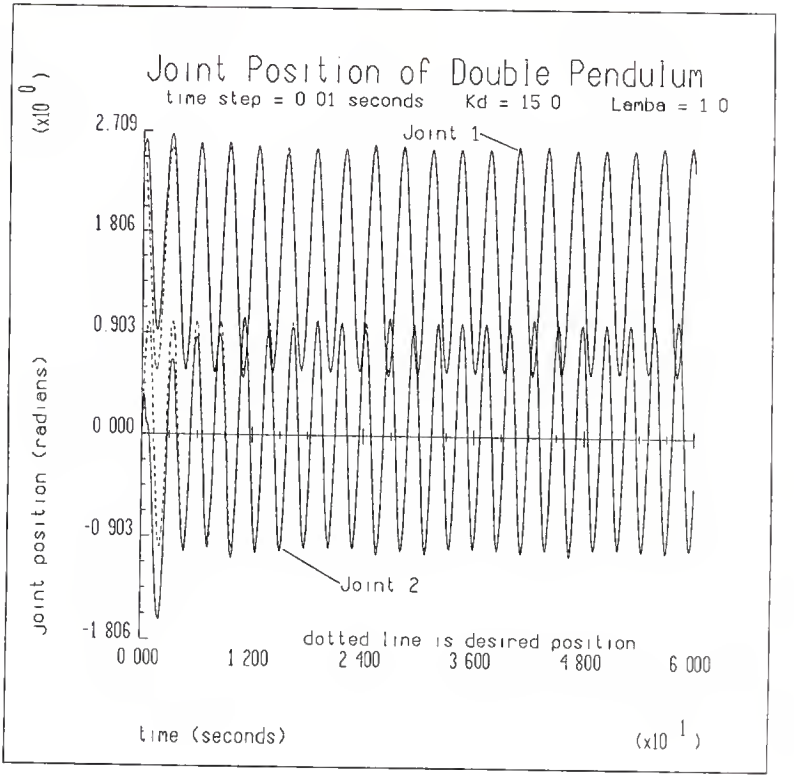


Figure 5.7 Position Output of Double Pendulum, Test1

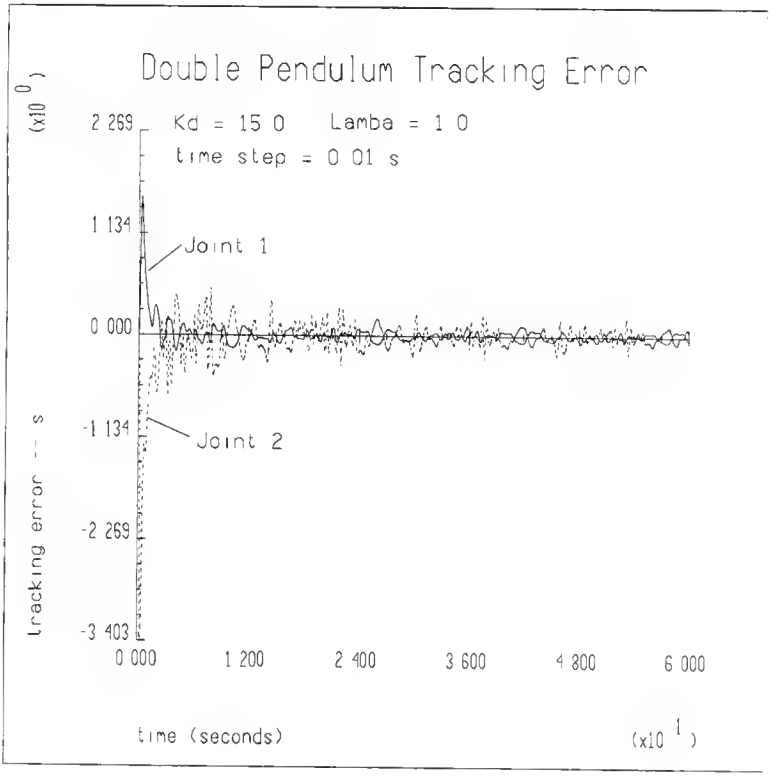


Figure 5.8 Tracking Error for Double Pendulum

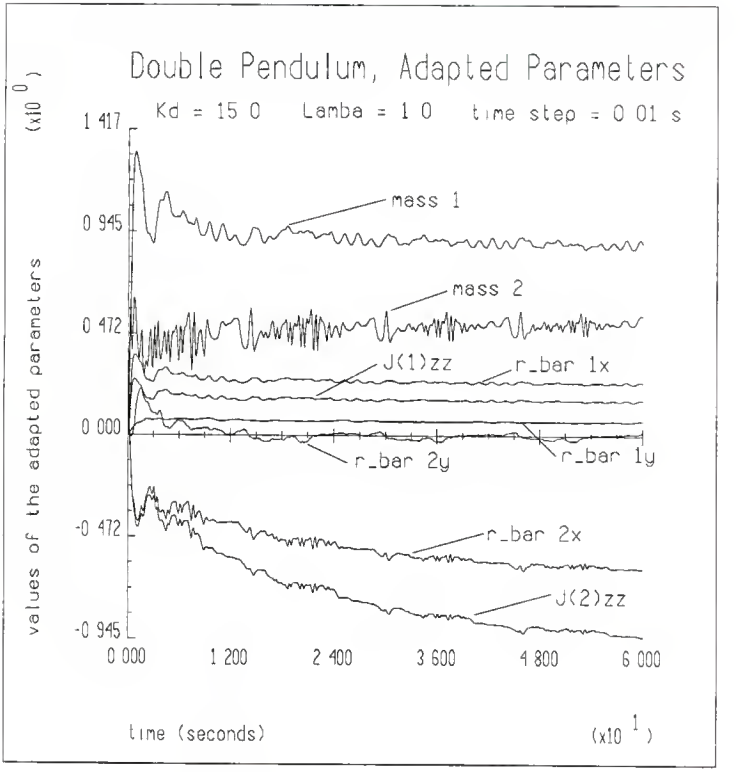


Figure 5.9 Parameter Adaptation for Double Pendulum

TABLE 5.5

DOUBLE PENDULUM RUN INFORMATION, TEST 2

Double Pendulumn

Constant parameters for the simulation are:

Simulation Time parameters:

Starting time = 0.00 Ending time = 60.00

System time step = 0.0050 seconds

Control time step = 0.0050 seconds

Control gain parameters lamba:

lamba[0] = 1.00 lamba[1] = 1.00

Control parameters Kd:

Kd[0] = 15.00 Kd[1] = 15.00

Adaptive gain parameters, gamma, are:

Link 1:

0.10	0.10	0.10	0.10	0.10	0.10
0.01	0.01	0.01	0.01		

Link 2:

0.10	0.10	0.10	0.10	0.10	0.10
0.01	0.01	0.01	0.01		

Friction values

0.02	0.02

TABLE 5.5 -- Continued

```
Link parameter information:

        For Joint 1 (Revolute):
            previous link = 0 next link = 2
            alpha =  0.00000     a =  6.00000     d = 0.00000

        For Joint 2 (Revolute):
            previous link = 1  last link of manipulator
            alpha =  0.00000     a =  6.00000     d = 0.00000


Branch parameter information

        Branch 0:
            Starting Link = 1 Ending Link = 2


 Model Parameters:
        Joint 1:
            Mass = 1.250       Friction = 0.000
            Center of mass:    x = -3.000     y = 0.000     z = 0.000
            Moment of Inertia about Center of mass:
                      0.00           0.00           0.00
                      0.00           3.75           0.00
                      0.00           0.00           3.75

        Joint 2:
            Mass = 1.000       Friction = 0.000
            Center of mass:    x = -3.500     y = 0.000     z = 0.000
            Moment of Inertia about Center of mass:
                      0.00           0.00           0.00
                      0.00           4.00           0.00
                      0.00           0.00           4.00


States of the Manipulator Base:

        Angular Velocity:       0.000     0.000     0.000
        Angular Acceleration:   0.000     0.000     0.000
        Base Acceleration:      0.000     9.810     0.000
```

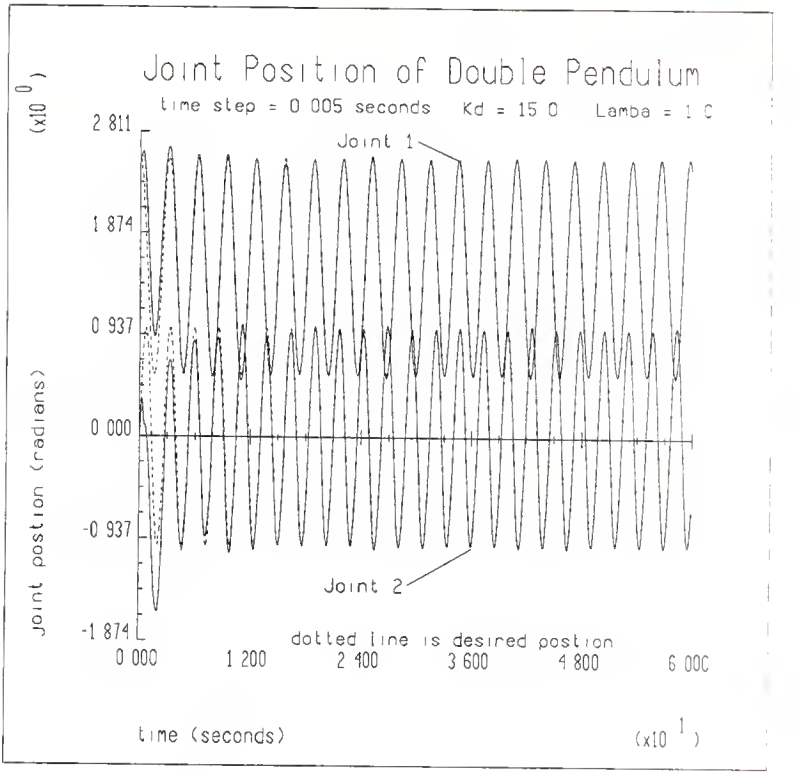


Figure 5.10 Position Output of Double Pendulum, Test 2

results of a test run done with a step size of 0.005 seconds, half of the step size of the first run. The tracking is slightly better, and pulls in slightly faster than the first run. The controller for the double pendulum was very sensitive to the values used in the K_d, Γ, and λ matrixes. The discrete control can go unstable with certain combinations of constants.

R-Theta Manipulator

The R-theta manipulator shown in Figure 5.3 was very robust and insensitive to the control constants. Table 5.6 and Figure 5.11 give the typical results of a run using the R-theta manipulator.

Three Joint Robot

The three joint robot shown in Figure 5.12 is very sensitive to the control variables used. It was not possible to get acceptable performance by starting all the adapted parameters in $\hat{a}$ at zero. Table 5.7 gives the details of a run shown in Figures 5.13 through 5.15. The position output is getting closer to the desired as time goes along, but there is still a lot of error in the system.

TABLE 5.6

R-THETA RUN INFORMATION

R-theta manipulator, Distributed Mass (General)

Constant parameters for the simulation are:

Simulation Time parameters:

Starting time = 0.00 Ending time = 60.00

System time step = 0.0100 seconds

Control time step = 0.0100 seconds

Control gain parameters lamba:

lamba[0] = 5.00 lamba[1] = 5.00

Control parameters Kd:

Kd[0] = 15.00 Kd[1] = 15.00

Adaptive gain parameters, gamma, are:

Link 1:

5.00 5.00 5.00 5.00 5.00 5.00

5.00 5.00 5.00 5.00

Link 2:

5.00 5.00 5.00 5.00 5.00 5.00

5.00 5.00 5.00 5.00

Friction values:

5.00 5.00

TABLE 5.6 -- Continued

```
Link parameter information:

        For Joint 1 (Revolute):
            previous link = 0 next link = 2
            alpha =  1.57080     a =  0.00000     d = 0.00000

        For Joint 2 (Prismatic):
            previous link = 1  last link of manipulator
            alpha =  0.00000     a =  0.00000     theta = 0.00000

Branch parameter information

        Branch 0:
            Starting Link = 1 Ending Link = 2


Model Parameters:
        Joint 1:
            Mass = 0.000       Friction = 1.000
            Center of mass:   x = 0.000     y = 0.000     z = 0.000
            Moment of Inertia about Center of mass:
                      0.00           0.00           0.00
                      0.00           0.00           0.00
                      0.00           0.00           5.00

        Joint 2:
            Mass = 10.000      Friction = 1.000
            Center of mass:   x = 0.000     y = 0.000     z = -1.500
            Moment of Inertia about Center of mass:
                      7.50           0.00           0.00
                      0.00           7.50           0.00
                      0.00           0.00           0.00

States of the Manipulator Base:

        Angular Velocity:       0.000     0.000     0.000
        Angular Acceleration:   0.000     0.000     0.000
        Base Acceleration:      0.000     0.000     0.000
```

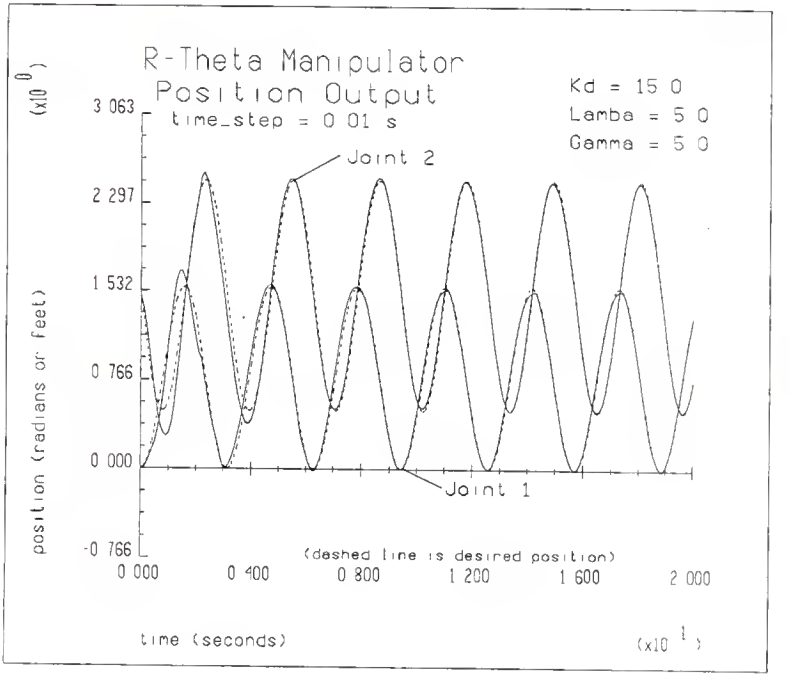


Figure 5.11 Position Output R-Theta Manipulator

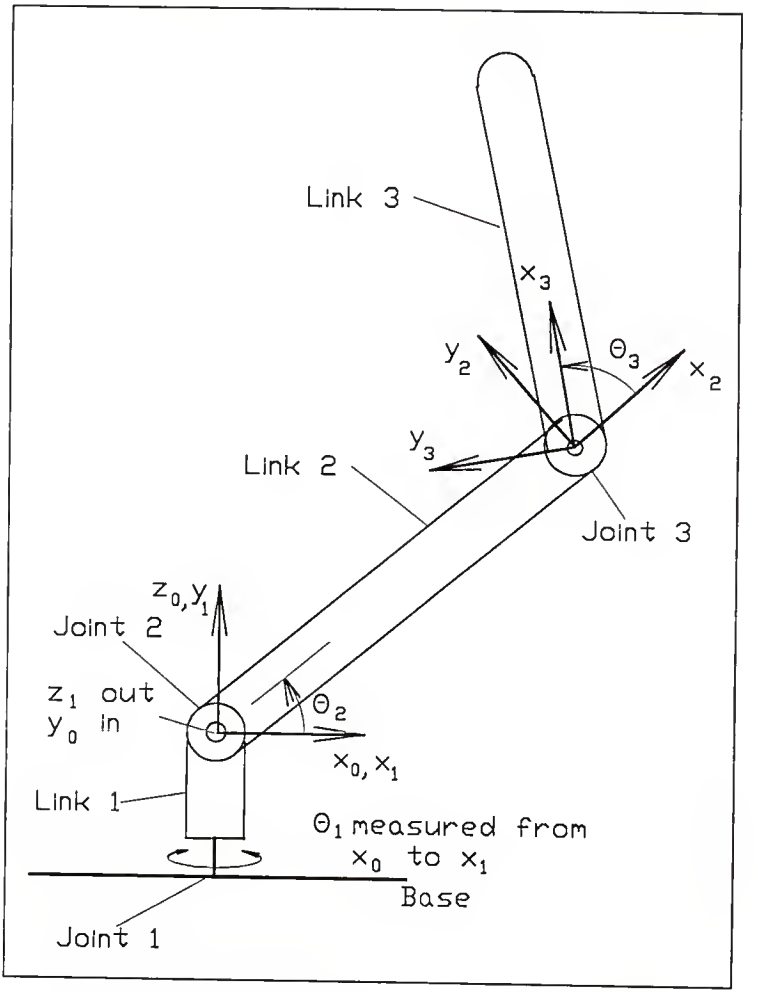


Figure 5.12 Three Joint Manipulator

TABLE 5.7

THREE JOINT MANIPULATOR RUN INFORMATION

Three Joint Serial Manipulator

Constant parameters for the simulation are:

Simulation Time parameters:

Starting time = 0.00 Ending time = 60.00

System time step = 0.0025 seconds

Control time step = 0.0025 seconds

Control gain parameters lamba:

lamba[0] = 5.00 lamba[1] = 5.00 lamba[2] = 5.00

Control parameters Kd:

Kd[0] = 15.00 Kd[1] = 15.00 Kd[2] = 15.00

Adaptive gain parameters, gamma, are:

Link 1:

0.01 0.01 0.01 0.01 0.01 0.01

0.01 0.01 0.01 0.01

Link 2:

0.01 0.01 0.01 0.01 0.01 0.01

0.01 0.01 0.01 0.01

Link 3:

0.01 0.01 0.01 0.01 0.01 0.01

0.01 0.01 0.01 0.01

Friction values:

0.01 0.01 0.01

TABLE 5.7 -- Continued

```
Link parameter information:

        For Joint 1 (Revolute):
            previous link = 0 next link = 2
            alpha =  1.57080     a =  0.00000     d = 0.00000

        For Joint 2 (Revolute):
            previous link = 1 next link = 3
            alpha =  0.00000     a =  5.00000     d = 0.00000

        For Joint 3 (Revolute):
            previous link = 2  last link of manipulator
            alpha =  0.00000     a =  0.00000     d = 0.00000


Branch parameter information

        Branch 0:
            Starting Link = 1 Ending Link = 3


Model Parameters:
        Joint 1:
            Mass = 1.000       Friction = 0.100
            Center of mass:    x = 0.000     y = 0.000     z = 0.000
            Moment of Inertia about Center of mass:
                      0.00           0.00           0.00
                      0.00           0.00           0.00
                      0.00           0.00           5.00

        Joint 2:
            Mass = 1.500       Friction = 0.150
            Center of mass:    x = -2.500     y = 0.000     z = 0.000
            Moment of Inertia about Center of mass:
                      0.00           0.00           0.00
                      0.00           3.20           0.00
                      0.00           0.00           3.20

        Joint 3:
            Mass = 2.000       Friction = 0.050
            Center of mass:    x = 4.000     y = 0.000     z = 0.000
            Moment of Inertia about Center of mass:
                      0.00           0.00           0.00
                      0.00           8.00           0.00
                      0.00           0.00           8.00
```

TABLE 5.7 -- Continued

States of the Manipulator Base:

Angular Velocity:	0.000	0.000	0.000
Angular Acceleration:	0.000	0.000	0.000
Base Acceleration:	0.000	0.000	32.200

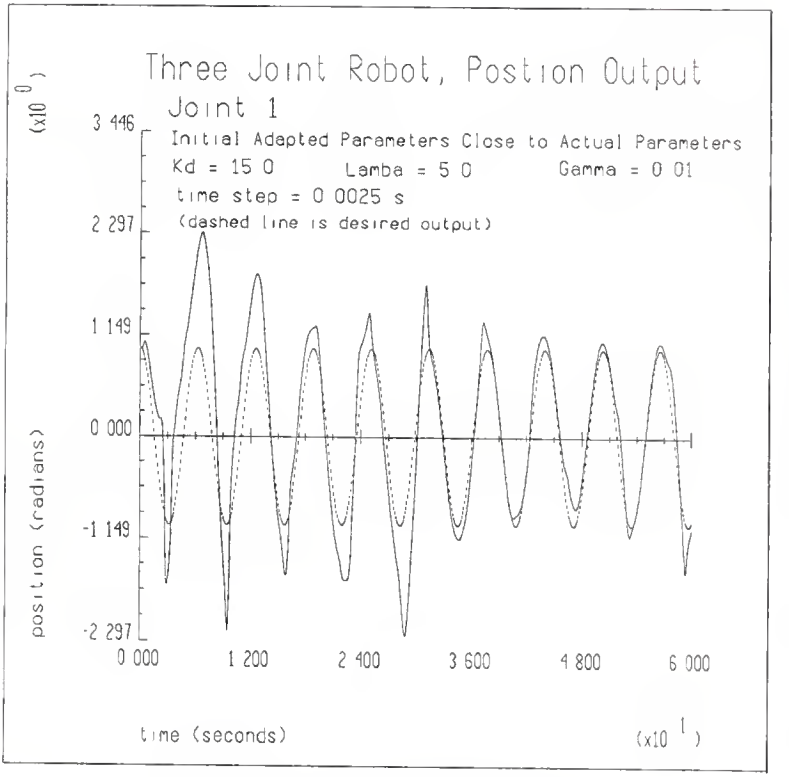


Figure 5.13 Position Output of a Three Joint Robot, Joint 1

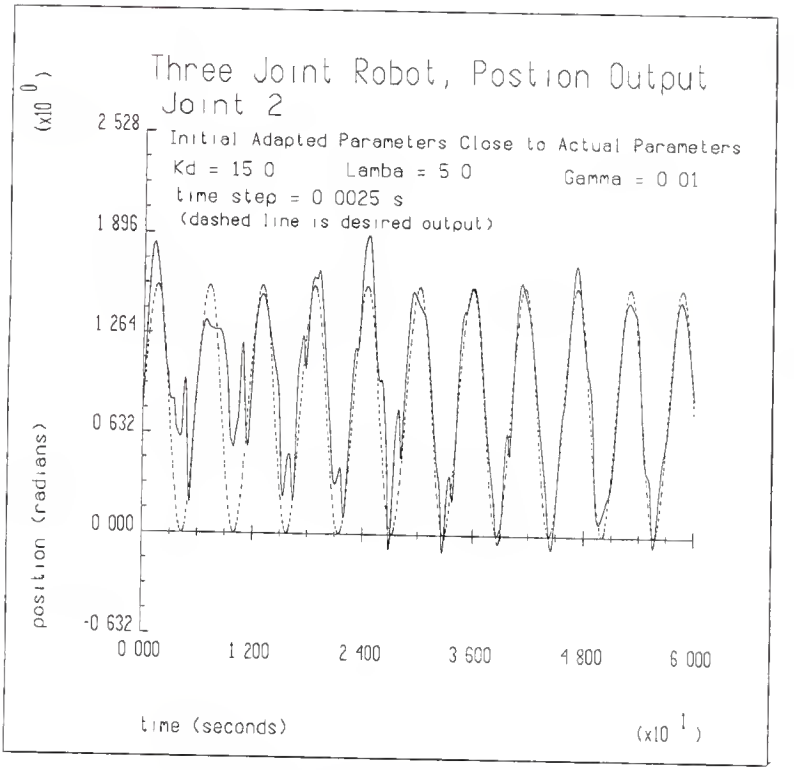


Figure 5.14 Position Output of a Three Joint Robot, Joint 2

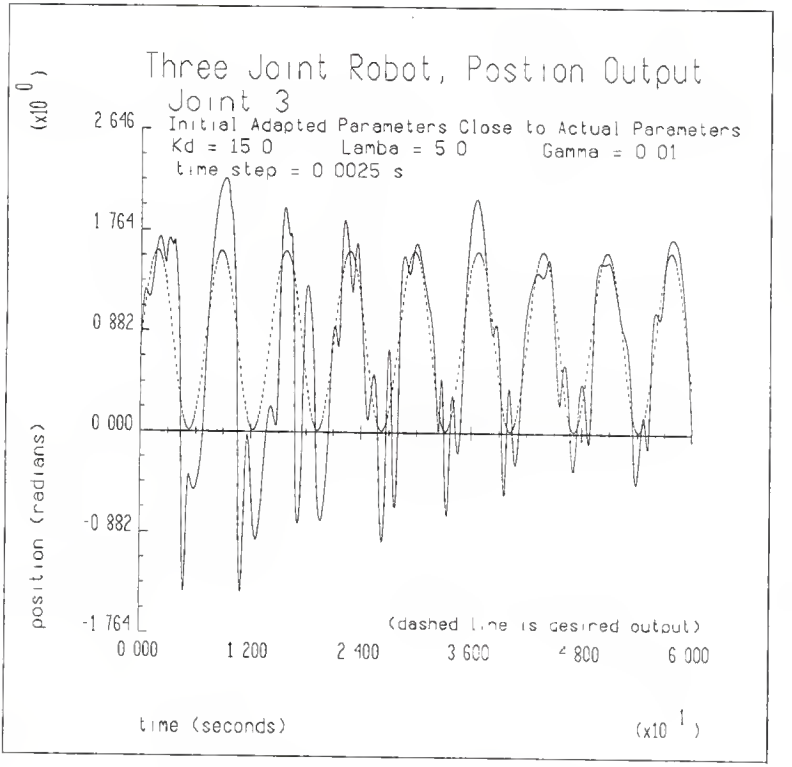


Figure 5.15 Position Output of a Three Joint Robot, Joint 3

Simulation Results of Manipulators With Closed Kinematic Chains

Slider Crank with Equal Length Arms

The slider crank with equal length arms shown in Figure 5.5 and used in the verification model was very robust and stable. The control variables could be adjusted to make the adaptation very rapid. Table 5.8 and Figures 5.16 and 5.17 show the output and tracking error of a test run that showed good tracking and rapid adaptation. Table 5.9 and Figures 5.18 and 5.19 is of a second run with higher K_d terms and it shows even better tracking and faster adaptation. The equations used to relate the joint position, velocity and acceleration for the independent joints to the dependent joints are shown in Appendix 6.

Offset Slider Crank on a Turntable

Figure 5.20 show the configuration of a offset slider crank on a turn table. Table 5.10 and Figure 5.21 show the results of a typical calculation for this configurations. The manipulator was not as robust as the slider crank with equal length arms, but a large variation of control parameters was allowable. The simulations for this configuration takes a large amount of time and therefore is hard to try a large number of test conditions. The equations relating the kinematics of the dependent variables to the independent variables are given in Appendix 6.

TABLE 5.8

SLIDER CRANK WITH EQUAL LENGTH ARMS
RUN INFORMATION, TEST 1

```
   This is Slider crank Problem (equal length arms)


Constant parameters for the simulation are:
       Simulation Time parameters:
          Starting time = 0.00          Ending time = 60.00
          System time step = 0.0100 seconds
          Control time step = 0.0100 seconds


       Control gain parameters lamba:
          lamba[0] = 5.00

       Control parameters Kd:
          Kd[0] = 5.00


       Adaptive gain parameters, gamma, are:

          Link 1:
                  1.00     1.00     1.00     1.00     1.00     1.00
                  1.00     1.00     1.00     1.00

          Link 2:
                  1.00     1.00     1.00     1.00     1.00     1.00
                  1.00     1.00     1.00     1.00

          Link 3:
                  1.00     1.00     1.00     1.00     1.00     1.00
                  1.00     1.00     1.00     1.00

          Friction values:
                  1.00     1.00     1.00
```

TABLE 5.8 -- Continued

```
Link parameter information:

        For Joint 1 (Revolute):
            previous link = 0 next link = 2
            alpha =  0.00000     a =  4.00000     d = 0.00000

        For Joint 2 (Revolute):
            previous link = 1  last link of manipulator
            alpha =  0.00000     a =  4.00000     d = 0.00000

        For Joint 3 (Prismatic):
            previous link = 0 next link = 2
            alpha = -1.57080     a =  0.00000     theta = 0.00000


Branch parameter information

        Branch 0:
            Starting Link = 1 Ending Link = 2

        Branch 1:
            Starting Link = 3 Ending Link = 3
            Main branch free joint = 2            branch free joint = 3

            Branch starting theta = 1.5708
            Branch starting alpha = 1.5708

            Vector from link 0 to base of branch in branch base frame:
               x = 0.000       y = 0.000          z = 0.000
            Vector from end of branch to link 2 in link 2 frame:
               x = 0.000       y = 0.000          z = 0.000
```

TABLE 5.8 -- Continued

```
Model Parameters:
       Joint 1:
           Mass = 2.000       Friction = 0.750
           Center of mass:    x = -2.000     y = 0.000     z = 0.000
           Moment of Inertia about Center of mass:
                    0.00            0.00            0.00
                    0.00            2.67            0.00
                    0.00            0.00            2.67

       Joint 2:
           Mass = 2.000       Friction = 1.000
           Center of mass:    x = -2.000     y = 0.000     z = 0.000
           Moment of Inertia about Center of mass:
                    0.00            0.00            0.00
                    0.00            2.67            0.00
                    0.00            0.00            2.67

       Joint 3:
           Mass = 4.000       Friction = 0.500
           Center of mass:    x = 0.000     y = 0.000     z = 0.000
           Moment of Inertia about Center of mass:
                    0.00            0.00            0.00
                    0.00            0.00            0.00
                    0.00            0.00            0.00

States of the Manipulator Base:

       Angular Velocity:       0.000     0.000     0.000
       Angular Acceleration:   0.000     0.000     0.000
       Base Acceleration:     32.200     0.000     0.000
```

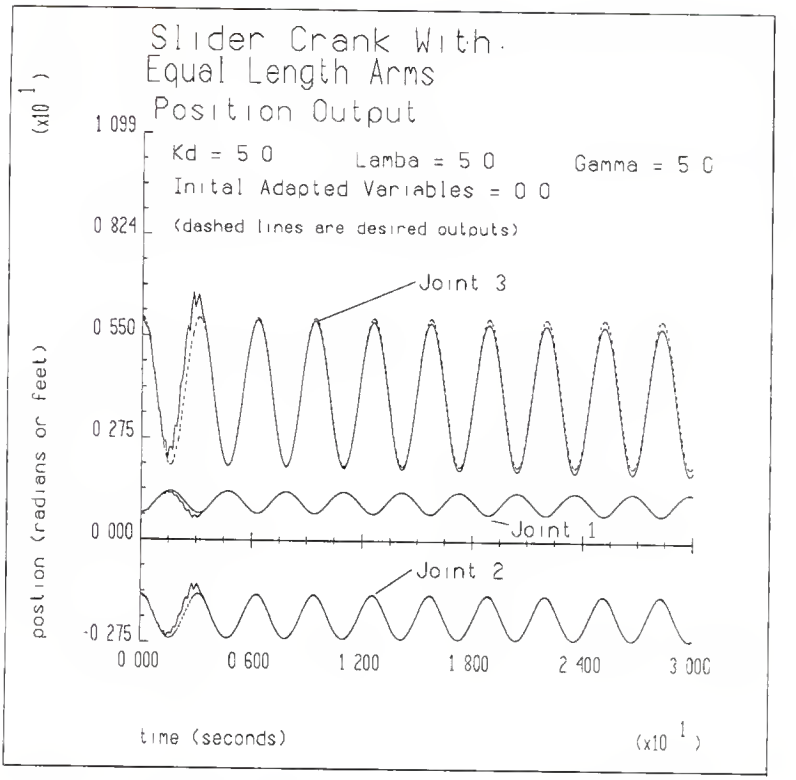


Figure 5.16 Position Output of Slider Crank With Equal Length Arms, Test 1

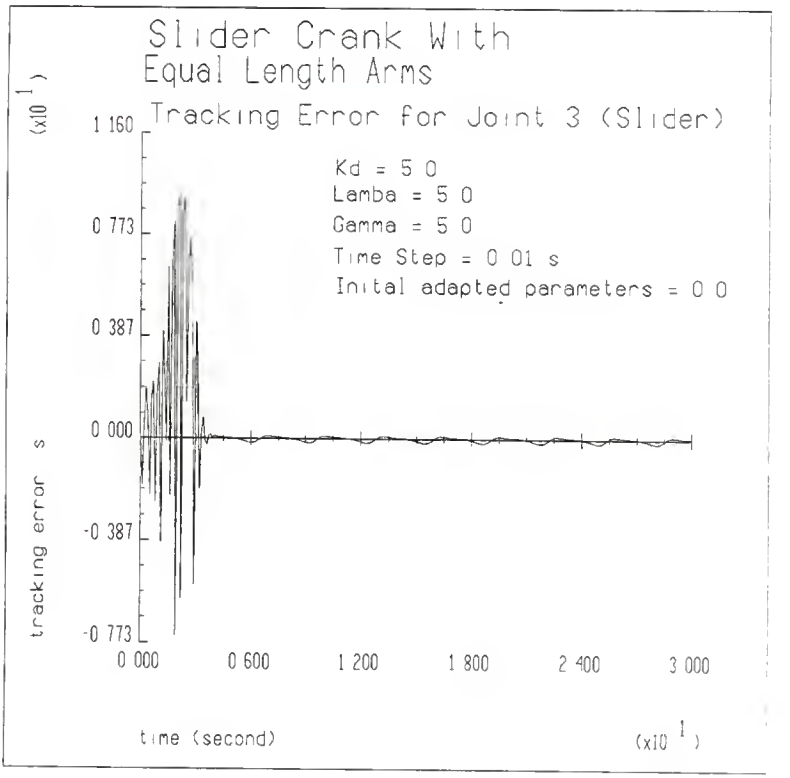


Figure 5.17 Tracking Error of Slider Crank With Equal Length Arms, Test 1

TABLE 5.9

SLIDER CRANK WITH EQUAL LENTGH ARMS
RUN INFORMATION, TEST 2

This is Slider crank Problem (equal length arms)

Constant parameters for the simulation are:

Simulation Time parameters:

Starting time = 0.00 Ending time = 60.00

System time step = 0.0050 seconds

Control time step = 0.0100 seconds

Control gain parameters lamba:

lamba[0] = 15.00

Control parameters Kd:

Kd[0] = 5.00

Adaptive gain parameters, gamma, are:

Link 1:

1.00	1.00	1.00	1.00	1.00	1.00
1.00	1.00	1.00	1.00		

Link 2:

1.00	1.00	1.00	1.00	1.00	1.00
1.00	1.00	1.00	1.00		

Link 3:

1.00	1.00	1.00	1.00	1.00	1.00
1.00	1.00	1.00	1.00		

Friction values:

1.00	1.00	1.00

TABLE 5.9 -- Continued

```
Link parameter information:

        For Joint 1 (Revolute):
            previous link = 0 next link = 2
            alpha =  0.00000     a =  4.00000     d = 0.00000

        For Joint 2 (Revolute):
            previous link = 1  last link of manipulator
            alpha =  0.00000     a =  4.00000     d = 0.00000

        For Joint 3 (Prismatic):
            previous link = 0 next link = 2
            alpha = -1.57080     a =  0.00000     theta = 0.00000


Branch parameter information

        Branch 0:
            Starting Link = 1 Ending Link = 2

        Branch 1:
            Starting Link = 3 Ending Link = 3
            Main branch free joint = 2            branch free joint = 3

            Branch starting theta = 1.5708
            Branch starting alpha = 1.5708

            Vector from link 0 to base of branch in branch base frame:
               x = 0.000       y = 0.000          z = 0.000
            Vector from end of branch to link 2 in link 2 frame:
               x = 0.000       y = 0.000          z = 0.000
```

TABLE 5.9 -- Continued

```
 Model Parameters:
        Joint 1:
            Mass = 2.000        Friction = 0.750
            Center of mass:   x = -2.000    y = 0.000    z = 0.000
            Moment of Inertia about Center of mass:
                      0.00          0.00          0.00
                      0.00          2.67          0.00
                      0.00          0.00          2.67

        Joint 2:
            Mass = 2.000        Friction = 1.000
            Center of mass:   x = -2.000    y = 0.000    z = 0.000
            Moment of Inertia about Center of mass:
                      0.00          0.00          0.00
                      0.00          2.67          0.00
                      0.00          0.00          2.67

        Joint 3:
            Mass = 4.000        Friction = 0.500
            Center of mass:   x = 0.000    y = 0.000    z = 0.000
            Moment of Inertia about Center of mass:
                      0.00          0.00          0.00
                      0.00          0.00          0.00
                      0.00          0.00          0.00

States of the Manipulator Base:

        Angular Velocity:       0.000    0.000    0.000
        Angular Acceleration:   0.000    0.000    0.000
        Base Acceleration:     32.200    0.000    0.000
```

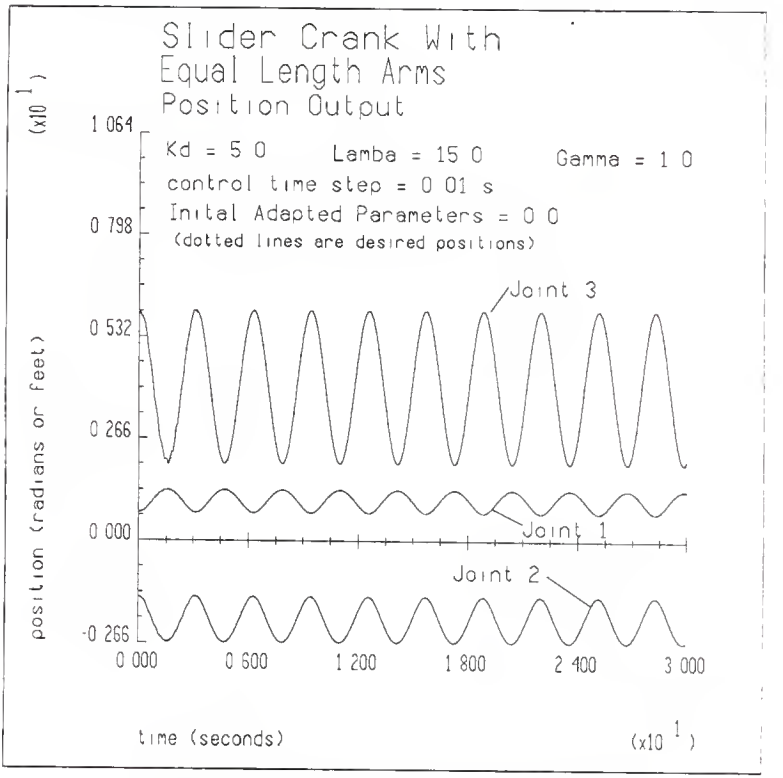


Figure 5.18 Position Output of Slider Crank With Equal Length Arms, Test 2

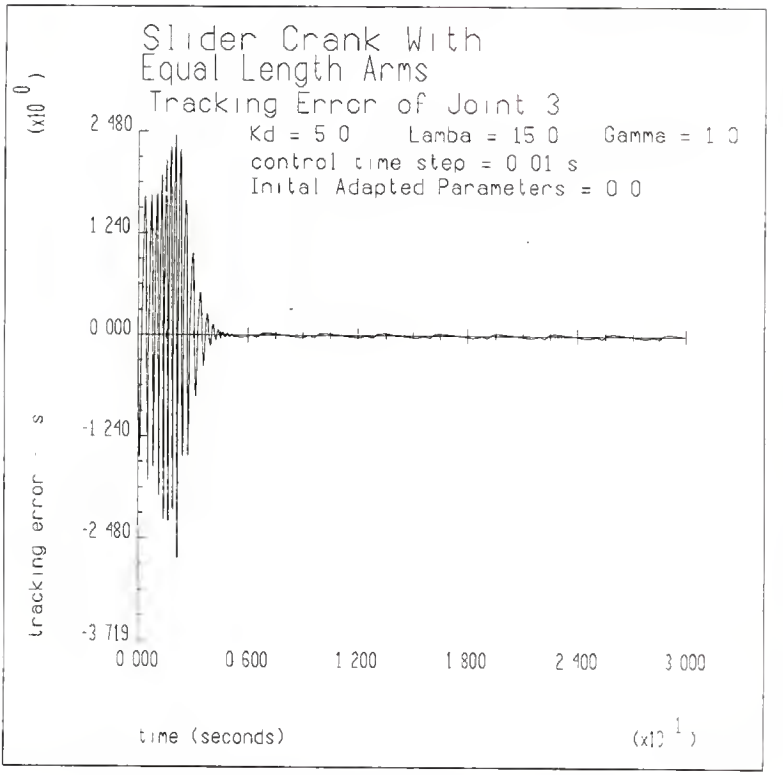


Figure 5.19 Tracking Error of Slider Crank With Equal Length Arms, Test 2

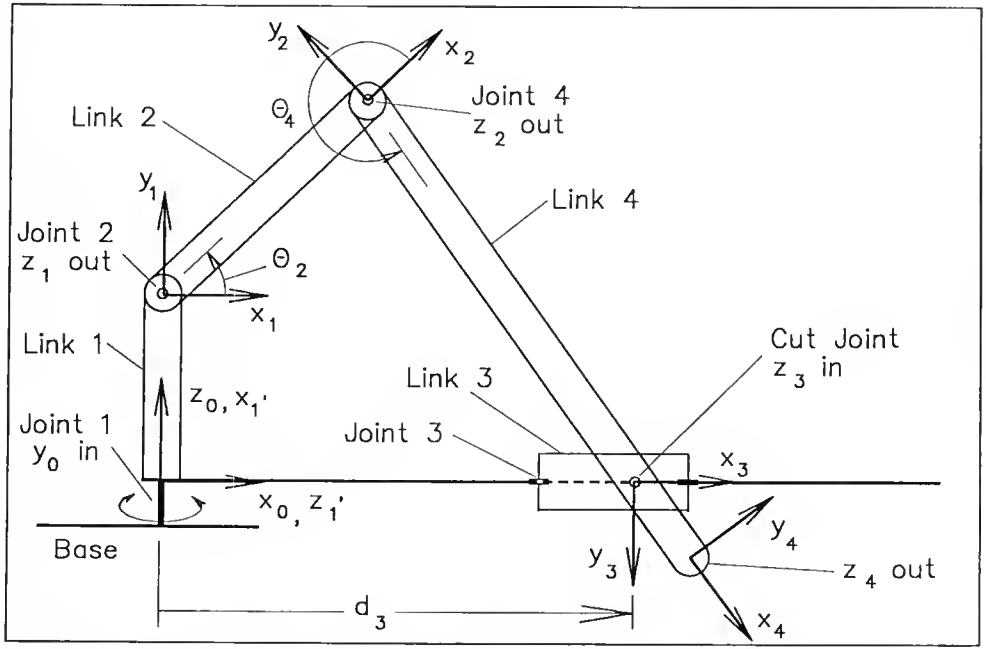


Figure 5.20 Offset Slider Crank On A Turntable

TABLE 5.10

OFFSET SLIDER CRANK ON A TURN
TABLE RUN INFORMATION

This is Slider on a rotating platform

Constant parameters for the simulation are:

Simulation Time parameters:

Starting time = 0.00 Ending time = 60.00

System time step = 0.0050 seconds

Control time step = 0.0050 seconds

Control gain parameters lamba:

lamba[0] = 5.00 lamba[1] = 5.00

Control parameters Kd:

Kd[0] = 10.00 Kd[1] = 10.00

Adaptive gain parameters, gamma, are:

Link 1:

0.50	0.50	0.50	0.50	0.50	0.50
0.50	0.50	0.50	0.50		

Link 2:

0.50	0.50	0.50	0.50	0.50	0.50
0.50	0.50	0.50	0.50		

Link 3:

0.50	0.50	0.50	0.50	0.50	0.50
0.50	0.50	0.50	0.50		

Link 4:

0.50	0.50	0.50	0.50	0.50	0.50
0.50	0.50	0.50	0.50		

Friction values:

0.50	0.50	0.50	0.50

TABLE 5.10 -- Continued

```
Link parameter information:

        For Joint 1 (Revolute):
            previous link = 0 next link = 2
            alpha =  1.57080     a =  0.00000     d = 2.00000

        For Joint 2 (Revolute):
            previous link = 1 next link = 4
            alpha =  0.00000     a =  3.00000     d = 0.00000

        For Joint 3 (Prismatic):
            previous link = 1 next link = 4
            alpha =  1.57080     a =  0.00000     theta = 1.57080

        For Joint 4 (Revolute):
            previous link = 2  last link of manipulator
            alpha =  0.00000     a =  6.00000     d = 0.00000


Branch parameter information

        Branch 0:
            Starting Link = 1 Ending Link = 4

        Branch 1:
            Starting Link = 3 Ending Link = 3
            Main branch free joint = 4            branch free joint = 3

            Branch starting theta = 1.5708
            Branch starting alpha = 1.5708

            Vector from link 1 to base of branch in branch base frame:
               x = -2.000      y = 0.000          z = 0.000
            Vector from end of branch to link 4 in link 4 frame:
               x = -1.000      y = 0.000          z = 0.000
```

TABLE 5.10 -- Continued

```
Model Parameters:
       Joint 1:
           Mass = 5.000       Friction = 1.000
           Center of mass:    x = 3.000     y = 0.000     z = 0.000
           Moment of Inertia about Center of mass:
                    1.00            0.00            0.00
                    0.00           10.00            0.00
                    0.00            0.00            1.00

       Joint 2:
           Mass = 3.000       Friction = 1.500
           Center of mass:    x = -1.500    y = 0.000     z = 0.000
           Moment of Inertia about Center of mass:
                    0.10            0.00            0.00
                    0.00            2.50            0.00
                    0.00            0.00            2.50

       Joint 3:
           Mass = 10.000      Friction = 5.000
           Center of mass:    x = 0.000     y = 1.000     z = 0.500
           Moment of Inertia about Center of mass:
                    6.00            0.00            0.00
                    0.00            2.00            0.00
                    0.00            0.00            4.00

       Joint 4:
           Mass = 4.000       Friction = 2.000
           Center of mass:    x = -3.000    y = 0.000     z = 0.000
           Moment of Inertia about Center of mass:
                    0.75            0.00            0.00
                    0.00            9.00            0.00
                    0.00            0.00            9.00


States of the Manipulator Base:

       Angular Velocity:       0.000    0.000    0.000
       Angular Acceleration:   0.000    0.000    0.000
       Base Acceleration:      0.000    0.000   32.200
```

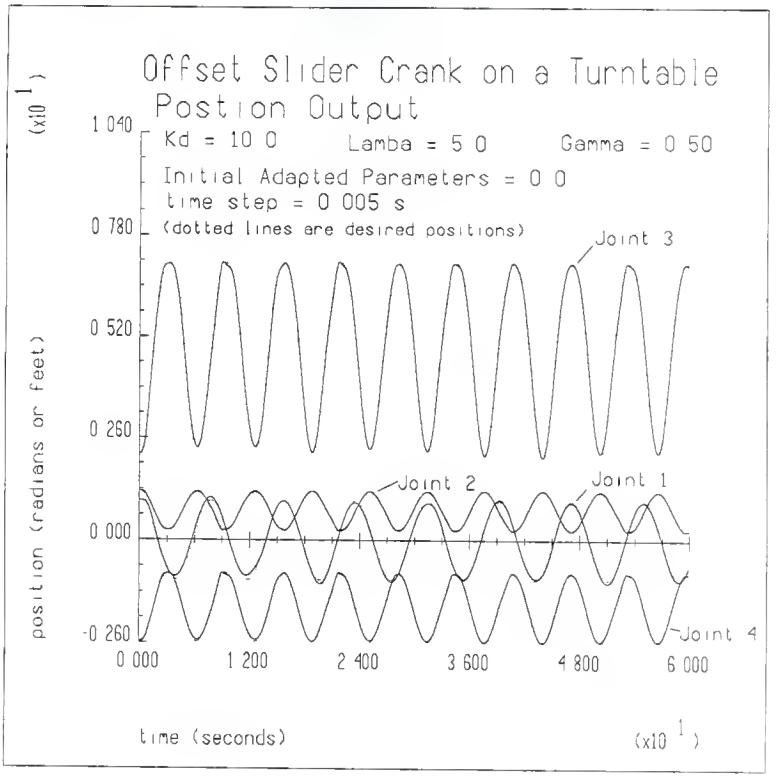


Figure 5.21 Position Output of Slider Crank On Turntable

Cincinnati Milacron T^3 776 Robot

The robot shown in Figure 5.22 is very difficult to simulate. The simulation could not be performed with the adaptive parameters in $\hat{a}$ starting at zero. The equations used to relate the dependent variables to the independent variables were valid for only a certain region of motion. If the error in the motion was great enough, the motion of the robot would fall outside the valid range for the equations. The equations are given in Appendix 6. Table 5.11 and Figure 5.23 give the results of a test run in which the variables in $\hat{a}$ were close to the actual dynamic parameters. The results are very good for this case. This configuration was hard to work with because it required significant computer time. The test shown in Figure 5.22 took approximately 30 hours of CPU time on a Apollo 3000.

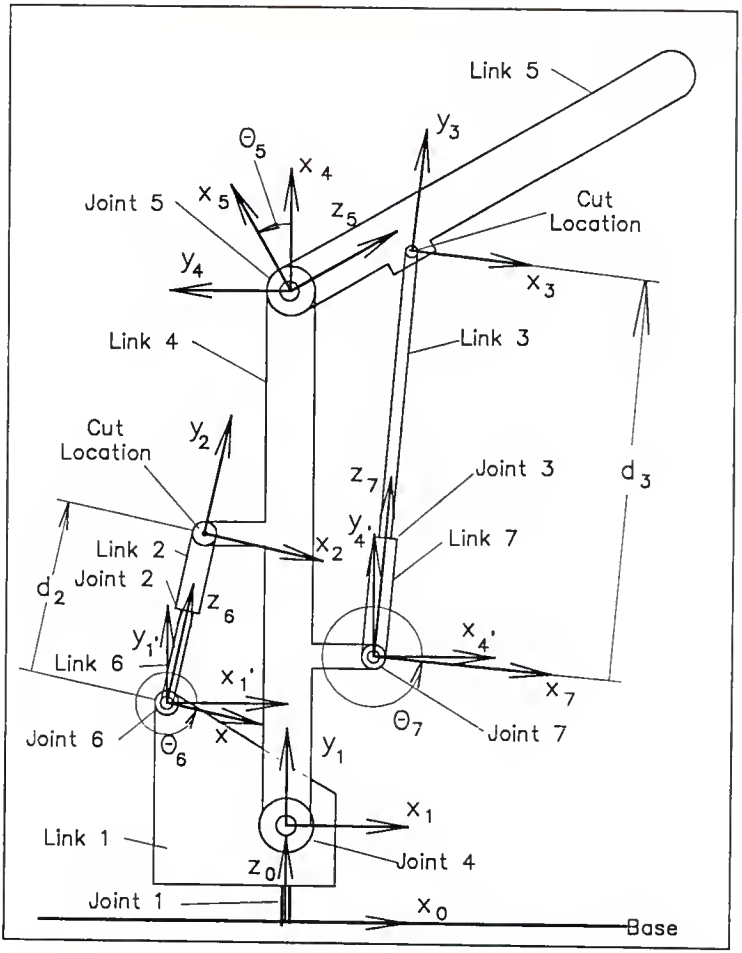


Figure 5.22 Cincinnati Milacron
T(3) 776 Robot

TABLE 5.11

CINCINNATI MILACRON T(3) 776 ROBOT
RUN INFORMATION

This is Model of the T3 robot

Constant parameters for the simulation are:

Simulation Time parameters:

Starting time = 0.00 Ending time = 60.00

System time step = 0.0020 seconds

Control time step = 0.0020 seconds

Control gain parameters lamba:

lamba[0] = 10.00 lamba[1] = 10.00 lamba[2] = 10.00

Control parameters Kd:

Kd[0] = 25.00 Kd[1] = 25.00 Kd[2] = 25.00

Adaptive gain parameters, gamma, are:

Link 1:

0.10	0.10	0.10	0.10	0.10	0.10
0.01	0.01	0.01	0.01		

Link 2:

0.10	0.10	0.10	0.10	0.10	0.10
0.01	0.01	0.02	0.01		

Link 3:

0.10	0.10	0.10	0.10	0.10	0.10
0.01	0.01	0.01	0.01		

Link 4:

0.10	0.10	0.10	0.10	0.10	0.10
0.01	0.01	0.01	0.01		

Link 5:

0.10	0.10	0.10	0.10	0.10	0.10
0.01	0.01	0.01	0.01		

TABLE 5.11 -- Continued

```
    Link 6:
           0.10     0.10     0.10     0.10     0.10     0.10
           0.01     0.01     0.01     0.01

    Link 7:
           0.10     0.10     0.10     0.10     0.10     0.10
           0.01     0.01     0.01     0.01

    Friction values:
           0.01     0.01     0.01     0.01     0.01     0.01
           0.01

Link parameter information:

    For Joint 1 (Revolute):
        previous link = 0 next link = 4
        alpha =  1.57080    a =  0.00000    d = 10.00000

    For Joint 2 (Prismatic):
        previous link = 6 next link = 4
        alpha =  1.57080    a =  0.00000    theta = 0.00000

    For Joint 3 (Prismatic):
        previous link = 7 next link = 5
        alpha =  1.57080    a =  0.00000    theta = 0.00000

    For Joint 4 (Revolute):
        previous link = 1 next link = 5
        alpha =  0.00000    a = 44.00000    d = 0.00000

    For Joint 5 (Revolute):
        previous link = 4  last link of manipulator
        alpha =  1.57080    a =  0.00000    d = 0.00000

    For Joint 6 (Revolute):
        previous link = 1 next link = 2
        alpha = -1.57080    a =  0.00000    d = 0.00000

    For Joint 7 (Revolute):
        previous link = 4 next link = 3
        alpha = -1.57080    a =  0.00000    d = 0.00000
```

TABLE 5.11 -- Continued

```
Branch parameter information

        Branch 0:
            Starting Link = 1 Ending Link = 5

        Branch 1:
            Starting Link = 6 Ending Link = 2
            Main branch free joint = 4          branch free joint = 6

            Branch starting theta = 0.0000
            Branch starting alpha = 0.0000

            Vector from link 1 to base of branch in branch base frame:
               x = -10.000    y = 10.000        z = 0.000
            Vector from end of branch to link 4 in link 4 frame:
               x = -20.000    y = 7.000         z = 0.000

        Branch 2:
            Starting Link = 7 Ending Link = 3
            Main branch free joint = 5          branch free joint = 7

            Branch starting theta = -1.5708
            Branch starting alpha = 0.0000

            Vector from link 4 to base of branch in branch base frame:
               x = 7.000      y = -30.000       z = 0.000
            Vector from end of branch to link 5 in link 5 frame:
               x = -2.000     y = 0.000         z = 10.000


Model Parameters:
       Joint 1:
            Mass = 15.540      Friction = 2.000
            Center of mass:   x = -10.000    y = 5.000    z = 0.000
            Moment of Inertia about Center of mass:
                    404.05          0.00          0.00
                      0.00       1213.71          0.00
                      0.00          0.00        388.51

       Joint 2:
            Mass = 9.324       Friction = 10.000
            Center of mass:   x = 5.000    y = -10.000    z = 5.000
            Moment of Inertia about Center of mass:
                    233.11          0.00          0.00
                      0.00        115.00          0.00
                      0.00          0.00        239.32
```

TABLE 5.11 -- Continued

```
Joint 3:
    Mass = 2.331      Friction = 10.000
    Center of mass:   x = 0.000    y = -15.000    z = 0.000
    Moment of Inertia about Center of mass:
            177.16          0.00          0.00
              0.00          4.66          0.00
              0.00          0.00        177.16

Joint 4:
    Mass = 27.973     Friction = 2.000
    Center of mass:   x = -22.000    y = 0.000    z = 0.000
    Moment of Inertia about Center of mass:
           2797.29          0.00          0.00
              0.00       7310.25          0.00
              0.00          0.00       4512.96

Joint 5:
    Mass = 32.635     Friction = 2.000
    Center of mass:   x = 5.000    y = 0.000    z = 10.000
    Moment of Inertia about Center of mass:
          15540.50          0.00          0.00
              0.00      15540.50          0.00
              0.00          0.00        404.05

Joint 6:
    Mass = 3.108      Friction = 1.000
    Center of mass:   x = 0.000    y = 0.000    z = -5.000
    Moment of Inertia about Center of mass:
             25.64          0.00          0.00
              0.00         25.64          0.00
              0.00          0.00          6.22

Joint 7:
    Mass = 10.101     Friction = 1.000
    Center of mass:   x = -3.000    y = -3.000    z = 3.000
    Moment of Inertia about Center of mass:
             99.46          0.00          0.00
              0.00         99.46          0.00
              0.00          0.00         49.73

States of the Manipulator Base:

    Angular Velocity:       0.000    0.000    0.000
    Angular Acceleration:   0.000    0.000    0.000
    Base Acceleration:      0.000    0.000   32.200
```

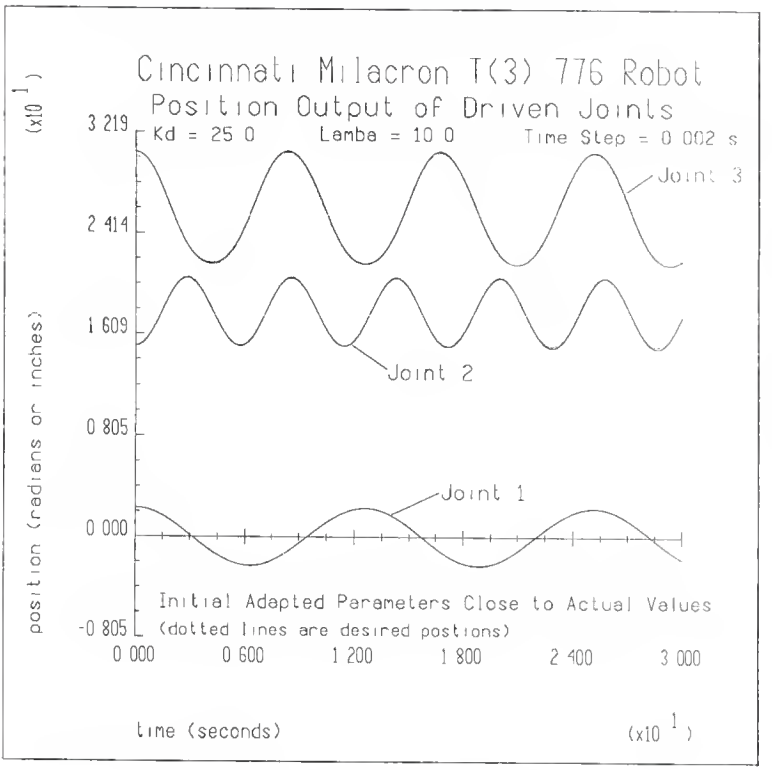


Figure 5.23 Position Output for the Cincinnati Milacron T(3) 776 Robot

CHAPTER 6

CONCLUSIONS AND RECOMMENDATIONS

Conclusions of the Investigation

The purpose of this investigation was to develop a recursive adaptive control algorithm for manipulators with both open and closed kinematic chains. The recursive adaptive controller was simulated with both open and closed kinematic chains. The open link, serial manipulators included a double pendulum, an R-theta manipulator and a simple three axis robot. The manipulators containing closed kinematic chains included a slider crank with equal length arms, an offset slider crank on a turntable, and a Cincinnati Milacron T^3 776 robot.

Slotine and Li [3,4,5] used a Lyapunov stability approach to show that the continuous time controller is asymptotically stable. The simulations indicated that this stability can not be guaranteed in a discrete time system. The robustness of the controller was very dependent on the type of system it was controlling. If the link mass is located a large distance away from the driven joints, the controller becomes sensitive to the controller constants. The adaptation can occur too rapidly and overshoot the

stable value, causing the controller to overcorrect and generating an error of the opposite sign, leading to a potentially unstable system. This was very obvious for the double pendulum test problem. A small change in the estimation of the mass of link 2 could greatly change the control output being calculated for joint 1. The system was made more robust by slowing the rate of adaptation by using either small values for the diagonal elements of the Γ matrix or by decreasing the step size of the controller, making the control appear to be almost continuous. The slider crank problems adapted very fast and showed the potential of the technique.

The adaptation was always trying to reduce the tracking error. The results of this adaptation was to drive the adapted parameters to some stable situation, but were not normally the actual values.

The recursive technique is easy to implement for a manipulator, but it exacts a high price in computer power to perform the calculations. The controller requires the calculation of a large number of terms, thus a very large computational power will have to be available for a real time implementation.

Recommendations for Further Study

The entire adaptation scheme may be able to be improved by determining additional methods to help drive the

adaptation. Slotine and Niemeyer [9] discussed the used of a parameter estimator working with the adaptation used in the development of the recursive algorithm. The controller might be made more robust by placing a limit on the maximum adaptation rate of a single variable, or by making the terms associated with Γ a time variable function.

The controller will be able to be implemented in a real time situation as the speed of control computers keep increasing.

APPENDIX 1

Coordinate Transformation Equations and Link Coordinate Parameters

The link coordinate systems for a robotic manipulator are described using the Denavit-Hartenberg [10] representation. The relationship between the coordinate systems of two adjacent links can be described using four link coordinate parameters as described by Paul, Shimano, and Mayer [12] and Lee [14]. See Figure A1.1 for a view of the parameters.

The θ_i parameter is the angle of rotation from the x_{i-1} axis to the x_i axis measured in the right-handed sense about the z_{i-1} axis. This parameter is the joint variable for a revolute joint. The α_i parameter is the angle of rotation from the z_{i-1} axis to the z_i axis measured in the right-handed sense about the x_i axis. The d_i parameter is the distance along the z_{i-1} axis from the $i\text{-}1^{th}$ coordinate system to the intersection of the z_{i-1} axis and the x_i axis. This is the joint variable for a prismatic joint. The a_i parameter is the distance along the x_i axis from its intersection with the z_{i-1} axis to the origin of the i^{th} coordinate system.

The rotational coordinate transformation is obtained from the link coordinate parameters. The matrix A_i^{i-1} is the

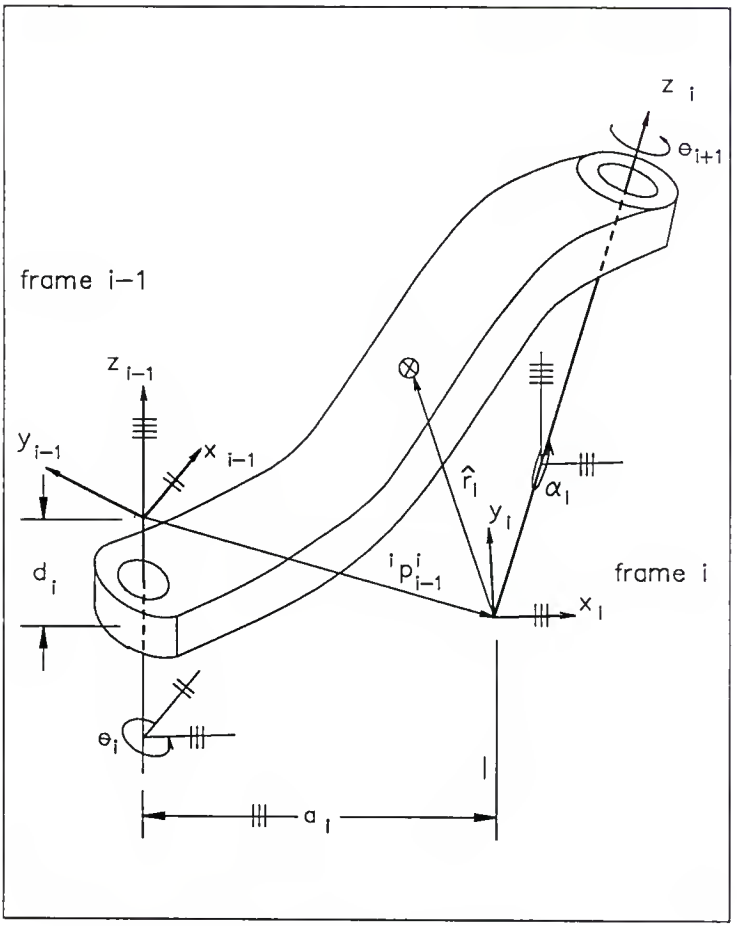


Figure A1.1 Link Parameters and Coordinate Frames

rotational coordinate transformation from the coordinate system of link i-1 to the coordinate system of link i such that

$$\hat{p}_i = A_i^{i-1} \hat{p}_{i-1}. \tag{A1.1}$$

The transformation between the coordinate systems are given by both Paul, Shimano and Mayer [12] and Lee [14]. The coordinate transformations is achieved by

$$A_i^{i-1} = \begin{bmatrix} c\theta & s\theta & 0 \\ -c\alpha s\theta & c\alpha c\theta & s\alpha \\ s\alpha s\theta & -s\alpha c\theta & c\alpha \end{bmatrix} \tag{A1.2}$$

where $c\theta = \cos(\theta_i)$, $s\theta = \sin(\theta_i)$, $c\alpha = \cos(\alpha_i)$, and $s\alpha = \sin(\alpha_i)$. Equation A1.2 describes only the change in orientation.

There are several rules that can be applied to rotational transformations to make the calculations easier. The rotational transformation A_{i-1}^{i} is given by

$$A_{i-1}^{i} = \left[A_i^{i-1} \right]^{-1} = \left[A_i^{i-1} \right]^{T}. \tag{A1.3}$$

Rotational transformations can be combined to form a transformation along several links. The transformation from link i to link i+3 would be given by

$$A_{i+3}^{i} = A_{i+3}^{i+2} A_{i+2}^{i+1} A_{i+1}^{i}. \tag{A1.4}$$

The position vector ${}^{i}p_{i-1}^{i}$ is the vector from the i-1th coordinate frame to the ith coordinate frame given in terms of the ith coordinate frame. The position vector is found from the variables a, d, and θ and is given as

$$ {}^{i}p_{i-1}^{i} = A_i^{i-1} \begin{bmatrix} a_i \, c\theta \\ a_i \, s\theta \\ d_i \end{bmatrix}. \qquad \text{(A1.5)} $$

The position vector is constant for revolute joints and the rotational transformation matrix is constant for prismatic joints.

APPENDIX 2

Cross Product Operator

The cross product of two vectors can be reduced to a matrix multiplication by using the cross product matrix operator. The cross product of two vectors is usually denoted as

$$A \times B = C. \tag{A2.1}$$

However, the corss product could also be written as

$$[A\times][B] = C. \tag{A2.2}$$

If the vector A is given as $[\ a_x \quad a_y \quad a_z\ ]^T$, then $[A\times]$ is given by

$$[A\times] = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix}. \tag{A2.3}$$

The $[A\times]$ can be though of as a vector operator. Any legal matrix operation can be performed on the matrix of equation A2.3.

APPENDIX 3

The Equations for M′

The M′ matrix is a three by six matrix used in the calculation of the Y matrix in the recursive adaptive routine. The terms in the M′ matrix cannot be reduced as needed in matrix form so they have to be written out explicitly. The equations for M′ for link i are:

$$m'_{i_{11}} = \dot{\omega}_{r_{i_x}} \tag{A3.1}$$

$$m'_{i_{12}} = \dot{\omega}_{r_{i_y}} - \frac{1}{2}\left({}^{\omega}r_{i_z}\,{}^{\omega}i_x + {}^{\omega}i_z\,{}^{\omega}r_{i_x} \right) \tag{A3.2}$$

$$m'_{i_{13}} = \dot{\omega}_{r_{i_z}} + \frac{1}{2}\left({}^{\omega}r_{i_y}\,{}^{\omega}i_x + {}^{\omega}i_y\,{}^{\omega}r_{i_x} \right) \tag{A3.3}$$

$$m'_{i_{14}} = -\frac{1}{2}\left({}^{\omega}r_{i_z}\,{}^{\omega}i_y + {}^{\omega}i_z\,{}^{\omega}r_{i_y} \right) \tag{A3.4}$$

$$m'_{i_{15}} = {}^{\omega}r_{i_y}\,{}^{\omega}i_y - {}^{\omega}i_z\,{}^{\omega}r_{i_z} \tag{A3.5}$$

$$m'_{i_{16}} = -\,m'_{i_{14}} \tag{A3.6}$$

$$m'_{i_{21}} = \frac{1}{2}\left({}^{\omega}r_{i_z}\,{}^{\omega}i_x + {}^{\omega}i_z\,{}^{\omega}r_{i_x} \right) \tag{A3.7}$$

$$m'_{i_{22}} = \dot{\omega}_{r_{i_x}} + \frac{1}{2}\left(\omega_{r_{i_z}} \omega_{i_y} + \omega_{i_z} \omega_{r_{i_y}} \right) \tag{A3.8}$$

$$m'_{i_{23}} = \omega_{r_{i_z}} \omega_{i_z} - \omega_{i_x} \omega_{r_{i_x}} \tag{A3.9}$$

$$m'_{i_{24}} = \dot{\omega}_{r_{i_y}} \tag{A3.10}$$

$$m'_{i_{25}} = \dot{\omega}_{r_{i_z}} - \frac{1}{2}\left(\omega_{r_{i_x}} \omega_{i_y} + \omega_{i_x} \omega_{r_{i_y}} \right) \tag{A3.11}$$

$$m'_{i_{26}} = - m'_{i_{21}} \tag{A3.12}$$

$$m'_{i_{31}} = - \frac{1}{2}\left(\omega_{r_{i_x}} \omega_{i_y} + \omega_{i_x} \omega_{r_{i_y}} \right) \tag{A3.13}$$

$$m'_{i_{32}} = \omega_{r_{i_x}} \omega_{i_x} - \omega_{i_y} \omega_{r_{i_y}} \tag{A3.14}$$

$$m'_{i_{33}} = \dot{\omega}_{r_{i_x}} - \frac{1}{2}\left(\omega_{r_{i_z}} \omega_{i_y} + \omega_{i_z} \omega_{r_{i_y}} \right) \tag{A3.15}$$

$$m'_{i_{34}} = - m'_{i_{31}} \tag{A3.16}$$

$$m'_{i_{35}} = \dot{\omega}_{r_{i_y}} + \frac{1}{2}\left(\omega_{r_{i_z}} \omega_{i_x} + \omega_{i_z} \omega_{r_{i_x}} \right) \tag{A3.17}$$

$$m'_{i_{36}} = \dot{\omega}_{r_{i_z}} \tag{A3.18}$$

where

$$\omega_i = \begin{bmatrix} \omega_{i_x} & \omega_{i_y} & \omega_{i_z} \end{bmatrix}^T \tag{A3.19}$$

and

$$\omega_{r_i} = \begin{bmatrix} \omega_{r_{i_x}} & \omega_{r_{i_y}} & \omega_{r_{i_z}} \end{bmatrix}^T. \tag{A3.20}$$

APPENDIX 4

The Force Notation at the Cut Joint in a Closed Kinematic Chain

A simple linkage is shown in Figure A4.1. The free body diagram of this linkage is shown in Figure A4.2. Joint 2' is serving as the cut location as defined in chapter 3 and torque $\tau_{2'}$ is zero. The sum of moments for link 2 about the $\hat{z}_1$ for the open loop is given as

$$\Sigma M_{z_1} = 0 = \tau_{2_o} + p_2 \times F \qquad \text{(A4.1)}$$

and for the closed loop the sum of moments is given by

$$\Sigma M_{z_1} = 0 = \tau_{2_c} + r_2 \times f_{2'} + p_2 \times F. \qquad \text{(A4.2)}$$

Where the subscript "o" is open loop torque and "c" is closed loop torque. Setting equations A4.1 and A4.2 equal to each other and simplifying gives

$$\tau_{2_o} = \tau_{2_c} + r_2 \times f_{2'}. \qquad \text{(A4.3)}$$

Equation A4.4 can be solved for the open loop torque, τ_{2_c} as

$$\tau_{2_c} = \tau_{2_o} - r_2 \times f_{2'}. \qquad \text{(A4.4)}$$

The sum of moments for link 3 about the $\hat{z}_{1'}$ axis for the open loop is given as

$$\Sigma M_{z_{1'}} = \tau_{3_o} = 0 \qquad \text{(A4.5)}$$

and for the closed loop the sum of moment is given by

$$\Sigma M_{z_{1'}} = \tau_{3_c} + r_3 \times (-f_{2'}) = 0. \qquad \text{(A4.6)}$$

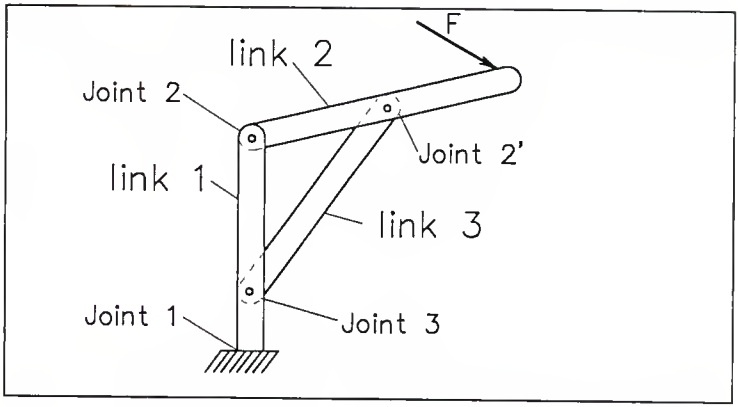


Figure A4.1 Simple Linkage

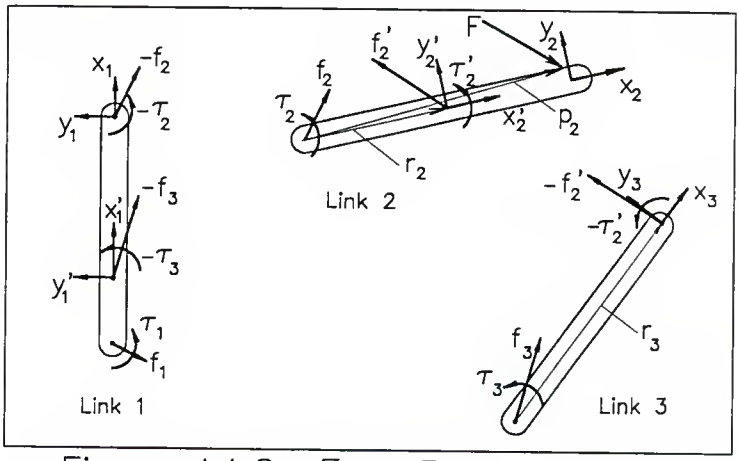


Figure A4.2 Free Body Diagram of a Simple Linkage

Equations A4.5 and A4.6 can be set equal to each other to give

$$r_{3_o} = r_{3_c} + r_3 \times (-f_{2'}). \qquad \text{(A4.7)}$$

Solving equation A4.7 for r_{3_c} gives

$$T_{3_c} = r_{2_o} + r_3 \times f_{2'}. \qquad \text{(A4.8)}$$

APPENDIX 5

EXACT DYNAMIC MODEL EQUATIONS

The state equations for the exact models used for verifications are given.

R-Theta Manipulator

See Figure 5.3 for definitions of the links and coordinate frames. The state equations are given as:

$$x[0] = \theta_1 \tag{A5.1}$$

$$x[1] = \dot{\theta}_1 \tag{A5.2}$$

$$x[2] = d_2 \tag{A5.3}$$

$$x[3] = \dot{d}_2 \tag{A5.4}$$

$$dx[0] = x[1] \tag{A5.5}$$

$$dx[1] = (\tau_1 - \zeta_1\, x[1]) - 2\, m_2 + (x[2] + \hat{r}_1)\, x[3]\, x[1]) \tag{A5.6}$$

$$/\ (m_2\ (x[2] + \hat{r}_1)^2 + J_{1_{yy}} + J_{2_{xx}})$$

$$dx[2] = (\tau_2 - \zeta_2\, x[3]) \ /\ (m_2 + x[2]\, x[1]^2 \tag{A5.7}$$

where τ is the driving torque or force, ζ is the viscous friction coefficient, m is the mass, $\hat{r}$ is the distance from link coordinate frame to the center of mass, and J is the inertia tensor about the center of mass.

Double Pendulum

See Figure 5.1 for definitions of the links and coordinate frames. The state equations are determined using equation 2.1. The equations were obtained from a book by Asada and Slotine [15] and are given as:

$$\tau_1 = H_{11}\ddot{\theta}_1 + H_{12}\ddot{\theta}_2 + G_1 \tag{A5.8}$$

$$\tau_2 = H_{21}\ddot{\theta}_1 + H_{22}\ddot{\theta}_2 + G_2 \tag{A5.9}$$

$$H_{11} = m_1(\ell_1 + \hat{r}_1)^2 + J_{1_{ZZ}} + m_2[\ell_1^2 + (\ell_2 + \hat{r}_2)^2 + 2\,\ell_1(\ell_2 + \hat{r}_2)\cos(\theta_2)] + J_{2_{ZZ}} \tag{A5.10}$$

$$H_{12} = H_{21} = m_2\,\ell_2(\ell_2 + \hat{r}_2)\cos(\theta_2) + m_2\,(\ell_2 + \hat{r}_2)^2 + J_{2_{ZZ}} \tag{A5.11}$$

$$H_{22} = m_2\,(\ell_2 + \hat{r}_2)^2 + J_{2_{ZZ}} \tag{A5.12}$$

$$G_1 = -[m_2\ \ell_1(\ell_2 + \hat{r}_2)\ \sin(\theta_2)]\ (\dot{\theta}_2^2 - 2\ \dot{\theta}_1\ \dot{\theta}_2)$$
$$+ g\{m_1(\ell_1 + \hat{r}_1)\ \cos(\theta_1) \qquad \text{(A5.13)}$$
$$+ m_2[(\ell_2 + \hat{r}_2)\ \cos(\theta_1+\theta_2) + \ell_1\ \cos(\theta_1)]\}$$

$$G_2 = m_2(\ell_2 + \hat{r}_2)\ \cos(\theta_1+\theta_2) \qquad \text{(A5.14)}$$

where g is the gravity constant, and ℓ is the link length. The joint accelerations were then found by

$$\ddot{\theta} = H^{-1}[\tau - G] \qquad \text{(A5.15)}$$

Slider Crank with Equal Length Arms

See Figure 5.5 for details of the coordinate systems and link definitions. The state equations are given as:

$$x[0] = \theta_1 \qquad \text{(A5.16)}$$

$$x[1] = \dot{\theta}_1 \qquad \text{(A5.17)}$$

$$x[2] = \theta_2 \qquad \text{(A5.18)}$$

$$x[3] = \dot{\theta}_2 \qquad \text{(A5.19)}$$

$$x[4] = d_3 \qquad \text{(A5.20)}$$

$$x[5] = \dot{d}_3 \qquad \text{(A5.21)}$$

$$dx[0] = x[1] \qquad \text{(A5.22)}$$

$$dx[1] = (\tau_1 - (4\ m_3 + 2\ m_1)\ \sin(\theta_1)\ \cos(\theta_1)\ (\ell_1 + \hat{r}_1)x[1]^2) \ /\ ((2/3)\ m_1\ (\ell_1 + \hat{r}_1) + (4\ m_3 + 2\ m_1) * \sin(\theta_1)^2\ (\ell_1 + \hat{r}_1)) \quad \text{(A5.23)}$$

$$dx[2] = x[3] \quad \text{(A5.24)}$$

$$dx[3] = -2\ dx[1] \quad \text{(A5.25)}$$

$$dx[4] = x[4] \quad \text{(A5.26)}$$

$$dx[5] = -2\ \ell_1(\sin(\theta_1)\ dx[1] + \cos(\theta_1)\ x[1]^2 \quad \text{(A5.27)}$$

APPENDIX 6

Kinematic Closure Equations

The following equations relate the dependent joint kinematic variables to the independent joint kinematic variables. The equations of a variation of them were used in the programs to close the kinematics of the closed kinematic chains. The equations were used using desired joint variables to obtain the desired joint position, velocity and acceleration.

The joint velocity and acceleration equations were then used to obtain the value of $\ddot{q}_r$ and $\dot{q}_r$ for the dependent joints by replacing the independent joint variables $\ddot{q}$ and $\dot{q}$ with the independent joint variables $\ddot{q}_r$ and $\dot{q}_r$.

The joint acceleration equations were placed in matrix form to provide the additional equations needed while obtaining new values of joint acceleration during the model simulation.

Kinematic Closure Equations for Slider Crank with Equal Length Arms

Figure 5.5 shows the configuration and identifies the links for the slider crank. Joint 1 is the independent

joint and joints 2 and 3 are the dependent joints. Joint 2 is a revolute joint and joint 3 is a prismatic joint.

$$\theta_2 = -2\ \theta_1 \tag{A6.1}$$

$$\dot{\theta}_2 = -2\ \dot{\theta}_1 \tag{A6.2}$$

$$\ddot{\theta}_1 = -2\ \ddot{\theta}_2 \tag{A6.3}$$

$$d_3 = 2\ \ell_1 \cos(\theta_1) \tag{A6.4}$$

$$\dot{d}_3 = -2\ \ell_1 \sin(\theta_1)\dot{\theta}_1 \tag{A6.5}$$

$$\ddot{d}_3 = -2\ \ell_1 \sin(\theta_1)\ddot{\theta}_1 - 2\ \ell_1 \cos(\theta_1)\ \dot{\theta}_1^2 \tag{A6.6}$$

The d terms are joint variable terms for the slider, joint 1. The θ terms are the joint variables for the revolute joint 2. The term ℓ is the length of the link.

Kinematic Closer Equations for an Offset Slider Crank on a Turntable

This manipulator has one closed kinematic loop. The dependent joints are 3 and 4 which are are dependent upon joint 2. Link 3 is a prismatic joint and link 4 is a revolute joint. See Figure 5.20 for details on the link configurations and and numbering.

$$\theta_4 = -\sin^{-1}((\ell_1 + \ell_2 \sin(\theta_2)) / \ell_4) - \theta_2 \tag{A6.7}$$

$$d_3 = \ell_2 \cos(\theta_2) + \ell_4 \cos(\theta_2 + \theta_4) \tag{A6.8}$$

$$\dot{\theta}_4 = -\ell_2 \cos(\theta_2)\dot{\theta}_2 / (\ell_4 \cos(\theta_2 + \theta_4)) - \dot{\theta}_2 \tag{A6.9}$$

$$\dot{d}_3 = -\ell_2 \sin(\theta_2)\ \dot{\theta}_2 - \ell_4 \sin(\theta_2 + \theta_4)\ (\theta_2 + \theta_4) \tag{A6.10}$$

$$\ddot{\theta}_4 = ((-\ell_2\cos(\theta_2) - \ell_4\cos(\theta_2+\theta_4))\ \ddot{\theta}_2 + \ell_2 \sin(\theta_2)\ \dot{\theta}_2^2 + \ell_4 \sin(\theta_2+\theta_4)\ (\theta_2+\theta_4)^2) / (\ell_4\cos(\theta_2+\theta_4)) \quad \text{(A6.11)}$$

$$\ddot{d}_3 = -\ell_2 \cos(\theta_2)\ \dot{\theta}_2^2 - \ell_4 \cos(\theta_2+\theta_4)(\theta_2+\theta_4)^2 - \ell_2 \sin(\theta_2)\ \ddot{\theta}_2 - \ell_4 \sin(\theta_2+\theta_4)(\ddot{\theta}_2 + \ddot{\theta}_4) \quad \text{(A6.12)}$$

Kinematic Closure Equations for the Cincinnati Milacron T³ 776 Robot

This robot contains two closed loop chains. See Figure 5.22 and Table 5.11 for details of the configuration. The dependent joints for which the following equations are for are revolute joints 4 through 7.

$$\theta_6 = \cos^{-1}\left[\frac{d_2^2 - 425}{28.2842714\ d_2}\right] - 2.35619449 \quad \text{(A6.13)}$$

$$\theta_4 = 2.072400382 - \cos^{-1}\left[\frac{825 - d_2^2}{707.106781}\right] \quad \text{(A6.14)}$$

$$\theta_5 = \cos^{-1}\left[\frac{1053 - d_3^2}{628.3183908}\right] - 1.144168834 \quad \text{(A6.15)}$$

$$\theta_7 = 0.229271933 - \cos^{-1}\left[\frac{845 - d_3^2}{61.611872\ d_3}\right] \quad \text{(A6.16)}$$

$$\dot{\theta}_6 = (-\dot{d}_2 \cos(\theta_4 - \theta_6 - 1.287002218)) / (d_2 \sin(\theta_4 - \theta_6 - 1.287002218) \quad \text{(A6.17)}$$

$$\dot{\theta}_4 = (\dot{d}_2 \sin(\pi/2 + \theta_6) + \dot{\theta}_6\ \dot{d}_2 \cos(\pi/2 + \theta_6)) / (25 \cos(\theta_4 + 0.2837941092)) \quad \text{(A6.18)}$$

$$\dot{\theta}_7 = -(\dot{d}_3 \cos(\theta_5 - \theta_7 - 1.768191887)) / (d_3 \sin(\theta_5 - \theta_7 - 1.768191887)) \quad \text{(A6.19)}$$

$$\dot{\theta}_5 = (\dot{\theta}_7 d_3 \sin(\pi/2 + \theta_7) - \dot{d}_3 \cos(\pi/2 + \theta_7)) / (10.19803903 \sin(\theta_5 - 0.19739556)) \quad \text{(A6.20)}$$

$$\ddot{\theta}_6 = (d_2 \theta_6^2 \cos(\theta_4 - \theta_6 - 1.287002218) - 25 \theta_4^2 - \ddot{d}_2 \cos(\theta_4 - \theta_6 - 1.287002218) - 2 \dot{d}_2 \dot{\theta}_6 \sin(\theta_4 - \theta_6 - 1.287002218)) / (d_2 \sin(\theta_4 - \theta_6 - 1.287002218)) \quad \text{(A6.21)}$$

$$\ddot{\theta}_4 = (25 \sin(\theta_4 + 0.2837941092) \dot{\theta}_4^2 + \ddot{d}_2 \sin(\pi/2 + \theta_6) + 2 \dot{d}_2 \dot{\theta}_6 \cos(\pi/2 + \theta_6) + d_2 (\ddot{\theta}_6 \cos(\pi/2 + \theta_6) - \sin(\pi/2 + \theta_6) \theta_6^2)) / (25 \cos(\theta_4 + 0.2837941092)) \quad \text{(A6.22)}$$

$$\ddot{\theta}_7 = (d_3 \cos(\theta_5 - \theta_7 - 1.768191887) \dot{\theta}_7^2 - 10.19803903 \dot{\theta}_5^2 - \ddot{d}_3 \cos(\theta_5 - \theta_7 - 1.768191887) - 2 \dot{\theta}_3 \dot{\theta}_7 \sin(\theta_5 - \theta_7 - 1.768191887)) / (d_3 \sin(\theta_5 - \theta_7 - 1.768191887)) \quad \text{(A6.23)}$$

$$\ddot{\theta}_5 = (10.19803903 \sin(\theta_5 - 0.19739556) \dot{\theta}_5^2 + \ddot{d}_3 \sin(\pi/2 + \theta_7) + 2 \dot{\theta}_3 \dot{\theta}_7 \cos(\pi/2 + \theta_7) + d_3 (\ddot{\theta}_7 \cos(\pi/2 + \theta_7) - \dot{\theta}_7^2 \sin(\pi/2 + \theta_7)) / (10.19803903 \cos(\theta_5 - 0.19739556)) \quad \text{(A6.24)}$$

REFERENCES

1. Luh, J.Y.S., M.W. Walker, and R.P. Paul. 1980. On-Line Computational Scheme for Mechanical Manipulators. Transactions of ASME Journal of Dynamic Systems, Measurement, and Control 102 (June): 120-127.

2. Craig, John J. 1988. Adaptive Control of Mechanical Manipulators. Reading, Massachusetts: Addison-Wesley Publishing Company, Inc.

3. Slotine, J.J.E. and W. Li. 1987. On the Adaptive Control of Robot Manipulators. The International Journal of Robotics Research 6 (Fall): 49-59.

4. Slotine, J.J.E. and W. Li. 1987. Parameter Estimation Strategies for Robotic Applications. ASME Winter Annual Meeting in Boston, Massachusetts, December xx-xx, 1987 by The American Society of Mechanical Engineers, 213-218.

5. Slotine, J.J.E. and W. Li. 1987. Adaptive Robot Control, A Case Study. IEEE International Conference on Robotics and Automation in Raleigh, North Carolina, March 30-April 3, 1987 by The Institute of Electrical and Electronics Engineers, 3: 1392-1400.

6. Snyder, W.E. 1985. Industrial Robots: Computer Interfacing and Control. Englewood Cliffs, New Jersey: Prentice-Hall, Inc. 208-209.

7. Walker, M.W. 1988. An Efficient Algorithm for the Adaptive Control of a Manipulator. IEEE International Conference on Robotics and Automation in Philadelphia, Pennsylvania, April 24-29, 1988 by The Institute of Electrical and Electronics Engineers, 2: 682-690.

8. Featherstone, R. 1983. The Calculation of Robot Dynamics Using Articulated-Body Inertias. The International Journal of Robotic Research 2 (Spring): 13-30.

9. Slotine, J.J. and G. Niemeyer. 1988. Performance in Adaptive Manipulator Control," Proceedings of the 27th Conference on Decision and Control in Austin, Texas, December 1988 by The Institute of Electrical and Electronics Engineers, 1585-1591.

10. Denavit, J and R.S. Hartenberg. 1955. A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices. Journal of Applied Mechanics : 215-221

11. Luh, J.Y.S. and Y.F. Zhenng. 1985. Computation of Input Generalized Forces for Robots with Closed Kinematic Chain Mechanisms. IEEE Journal of Robotics and Automation RA-1 (June): 95-103.

12. Paul, R.P., B. Shimano and G.E. Mayer. 1981. Kinematic Control Equations for Simple Manipulators. IEEE Transactions on Systems, Man and Cybernetics SMC-11 (June 1981): 80-86.

13. Greenwood, Donald T. 1965. Principles of Dynamics. Englewood Cliff, New Jersey: Prentice-Hall, Inc.

14. Lee, C.S. George. 1983. Robot Arm Kinematics. Chapter In Tutorial on Robotics, C.S.G. Lee, R.C. Gonzalez, and K.S. Fu, North-Holland: Elsevier Science Publishers B.V..

15. Asada, H. and J.J. Slotine. 1986. Robot Analysis and Control. New York: John Wiley and Sons.

RECURSIVE ADAPTIVE CONTROL OF OPEN AND CLOSED KINEMATIC CHAINS

by

TRAVIS EUGENE BARNES

B.S., Kansas State University, 1985

AN ABSTRACT OF A MASTER'S THESIS

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

MECHANICAL ENGINEERING

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1989

ABSTRACT

This thesis describes an investigation into the development of a recursive adaptive control law implementation for controlling robotic manipulators with open and closed kinematic chains. The control is done by a recursive computed torque controller along with a PD compensator. The tracking error is used to modify the estimates of the dynamic parameters used in the computed torque controller. The uniqueness of the approach is demonstrated by comparing this work to that of other investigators.

A recursive adaptive controller and a simulated model are developed for both open and closed kinematic chains. The controller and simulation model require the kinematic closure equations for closed kinematic chains.

The controller was simulated for serial link arms and manipulators containing closed kinematic chains. The digital controller stability was found to be sensitive to control constant magnitudes and the step size. The adaptive controller was able to successfully control each manipulators and modify the dynamic parameters to reduce the tracking error.