Multiple Model-Based Robot Control: Development and Initial Evaluation

THESIS

Larry Don Tellman
Captain, USAF

AFIT/GE/ENG/88D-55

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

(1)

AFIT/GE/ENG/88D-55

Multiple Model-Based Robot Control: Development and Initial Evaluation

THESIS

Larry Don Tellman
Captain, USAF

AFIT/GE/ENG/88D-55

AFIT/GE/ENG/88D-55

Multiple Model-Based Robot Control: Development and Initial Evaluation

THESIS

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University
In Partial Fulfillment of the
Requirements for the Degree of
Master of Science in Electrical Engineering

Larry Don Tellman, B.S.E.E.
Captain, USAF

December, 1988

Preface

The purpose of this thesis was to develop and evaluate a new adaptive robot control technique. The approach included the use of a Multiple Model Adaptive Estimator (MMAE) to determine unknown parameters needed for robot tracking and a PD feedback loop to reject disturbances. There are presently many estimation techniques used for parameter identification in robot control. Before a preferred approach can be established, the range of possible identification schemes must be expanded and experimentally verified.

The MMAE was combined with a model-based description of the robot. Model-based control is a mature control algorithm that has been shown to produce superior tracking performance when the payload in known. The MMAE was used to provide the model-based control algorithm with an estimate of the mass of the payload. Simulation and experimentation on the *PUMA-560* clearly demonstrated the radically improved tracking performance when the MMAE is employed.

I wish to extend my deepest thanks to Capt M. B. Leahy for his many hours of assistance and constant support during this thesis. His contributions were indispensable. I would like to also express my sincere appreciation to Dr. Peter Maybeck for his additional enthusiasm and insights at strategic times during this research. A world of thanks is also owed Dr. Gary Lamont, Lt. Col Z. Lewantowicz and Mr. Dan Zambon for their invaluable services. Finally to my family, who deserved the best and had to settle for me, I want them to know that without their unselfish love and support surviving AFIT would not have been possible.

Larry Don Tellman

Table of Contents

List of Figures

Abstract

A new form of adaptive model-based robot control has been developed and experimentally evaluated. The Multiple Model Based Control (MMBC) technique utilizes knowledge of nominal manipulator dynamics and principles of Bayesian estimation to provide payload-independent trajectory tracking accuracy. The MMBC is formed by augmenting a model-based controller, which employs feedforward dynamic compensation and constant gain PD feedback, with a payload estimate provided by a Multiple Model Adaptive Estimator. Extensive simulation studies demonstrated the MMBC's ability to adapt to variations in manipulator payload quickly and accurately. Initial experimental evaluations on the first three links of a PUMA-560 validated the algorithm's potential.

Multiple Model-Based Robot Control: Development and Initial Evaluation

I. Introduction

1.1 Motivation

The ultimate goal in robotic research is to produce a robot that will emulate a human. The research at the Air Force Institute of Technology has been directed toward developing a robotic manipulator with the manual dexterity of the human arm. Human have the ability to learn and to adapt to their environment. With self adaptation mechanisms, a robot could perform a wide variety of tasks, quicker, with minimal or no human intervention. Future Air Force applications, such as telepresence, will require a robot with the capability to adapt quickly and accurately to unexpected changes in its environment while maintaining accurate high speed tracking.

A robot is defined as a machine that performs various complex acts of a human [Woo77]. Current technology can only produce robots that have the capability to replace a human for many simple repetitive tasks. The heart of the robot is the control system that guides that manipulator along a given trajectory. The equations of motion that define how the robot moves in space are a set of complex non-linear, coupled differential equations. To meet future Air Force requirements, robot control systems must address the coupled non-linear nature of the equations of motion in an uncertain environment. The model of the robot including the external payload used in the control systems must be as precise as possible to account for high speed robot dynamics. Previous research has shown that payload adaptation is crucial to high performance tracking [Lea88a].

1.2 *Objective*

The primary objective of this research effort was to develop an alternative form of adaptive model-based control that would achieve high performance trajectory tracking in the presence of uncertain payload information. The secondary purpose was to evaluate the new algorithm's potential both in simulation and on a real robot.

1.3 *Problem Statement*

The use of on-line adaptation algorithms was a new research area for the Robotics Laboratory at the Air Force Institute of Technology (AFIT). Some of the ground work had been laid for such an effort. The PUMA-560 and computer support were available at the outset of this research. Also, much of the software for the simulator and the low-level control of the robot had previously been developed.

The problem addressed in this research was how to improve high speed trajectory tracking in the presence of unknown disturbances. These disturbances arise from noise-corrupted position measurements and from incorrect models of the robot and its payload. Proper calibration of the robot provides nearly all of the data needed for accurate models. The major remaining unknown is the payload attached to the robot. Since the payload changes during normals operation, the robot control algorithm must quickly estimate the payload and adapt to any fluctuations that degrade tracking performance.

Adaptive control of robotic manipulators is an area of active research. One of the most basic forms of adaptive control is the model-based approach. Experimental evaluations of model-based techniques have demonstrated their potential for improving tracking accuracy over high speed trajectories [KK88,AAGH87,CHS87, LS88a,Lea88d,AH86,YK87].

Unfortunately, the model-based approaches patterned after the computed-

torque technique [Luh83] can only adapt to changes in manipulator joint configuration [Cra86,FGL87]. The tracking performance of those algorithms degrades noticeably in the presence of uncertain payloads [CHS87], even for robots with high torque amplification drive systems [Lea88a].

Since the model-based control algorithm provides excellent tracking performance when accurate payload information is available, one approach has been to augment that controller with a payload adaptation mechanism [MG86,CHS87, HBSP87,LS88b,SL87a]. A common theme in adaptive model-based control design has been the use of Lyapunov theory to develop the adaptation algorithms. Lyapunov theory guarantees that the controller will be stable and that the steady state errors will asymptotically approach zero. That approach is well suited to the constant acceleration trajectory tracking [CHS87] or steady state regulation [SL87a] of horizontally articulated manipulators for which experimental evaluations have been conducted. However, constant acceleration and large periods of regulation are not representative of the full range of human arm motion. Also, the horizontal manipulators were not subject to the large nonconservative forces present in vertically articulated robots. How well Lyapunov techniques control a vertically articulated manipulator, over a more complete range of motion, is still an open research issue.

Other forms of robot control include the Model Reference Adaptive Control (MRAC) and adaptive control using an autoregressive model. These methods assume a second-order model for robot dynamics is adequate, and that the coefficients of the model are estimated on-line [DD79,Ser87,LE87,KG83]. The adaptive perturbation control scheme on the other hand, linearizes the non-linear equations of motion with a feedforward element and employs a full state feedback perturbation regulator with the perturbation plant and input distribution matrices estimated on-line [LC84,dVW87]. None of these approaches attempts to model the inherent noises in the robot system and the estimators are based on Lyapunov or least-

squares techniques.

An alternative to the Lyapunov based approach is the use of stochastic estimation/adaptation techniques. In addition to providing a faster means of payload adaptation, the stochastic approach explicitly accounts for the numerous sources of noise and uncertainty in a real robotic system. Multiple Model Adaptive Estimation is a Bayesian estimation approach that employs multiple Kalman filters to estimate parameters quickly and accurately in the presence of noise and uncertainty. A Multiple Model Adaptive Estimator (MMAE) has been successfully applied to several difficult tracking problems [MZ85,MR83,Ath77,Ber83,KM87,Las87,MS85]. If that Bayesian approach could be successfully applied to the manipulator payload estimation problem, tracking realization sufficient to emulate human arm performance may be possible.

1.4 Approach

The robot control method developed in this thesis investigation was based on the model-based technique that has demonstrate good tracking performance in the presence of accurate payload information [LS88a]. The model-based control scheme is separated into two parts: a feedforward element which produces a nominal torque and a feedback element that employs a set of gains to reject any remaining disturbances.

The parameter identification technique employed was based on a Bayesian approach. The algorithm is called a Multiple Model Adaptive Estimator (MMAE) [Ath77,GW80,May82a,May82b]. The task of the MMAE was made more difficult by the closed loop formulation of the model-based technique. The parallel structure of the MMAE as shown in Chapter 3 allows for the incorporation of many different robot models into the estimation process. Each one, under different conditions, is correct. This Multiple Model-Based Control (MMBC) formulation utilizes the payload estimate from the MMAE in the feedforward element of the model-based

controller which has a heavy dependence on payload.

The testing of the MMBC technique consisted of extensive simulation and experimentation on the first three links of the PUMA-560. The PUMA-560 was selected as a case study for the Multiple Model-Based Control (MMBC) technique since it has been shown that the tracking performance of this vertically articulated manipulator is highly dependent on knowledge of the true payload [LS88a]. To perform the initial evaluation of the new control technique, the payload was assumed to be completely described by a point mass.

A sensitivity analysis of the perturbation feedback element was accomplished. The nominal torques were calculated in the feedforward element. The perturbation torques were generated in the feedback loop. The plant model, $F(a,t)$, in the Kalman filter was based on the feedback element. Where $F(a,t)$ is the linearized equations of motion and a is the unknown payload parameter. The equations for $F(a,t)$ were developed by taking a Taylor series expansion of the non-linear equations of motion about the nominal trajectory and ignoring higher order terms. An analysis of the eigenvalues of $F(a,t)$ revealed that linearized robot dynamics was a function of the trajectory and had a weak dependence on payload.

The slight $F(a,t)$ dependency on a required that the MMAE in this closed loop estimation task be set up to produce an estimate of the difference between the true payload and the assumed value in the feedforward element. The mismatch in the payloads produced a large enough disturbance in the feedback loop that the a-dependent modes in $F(a,t)$ would be excited.

To produce an MMAE, the continuous payload parameter, a, had to discretized and a Kalman filter built for each value. A single Kalman filter was built for a_1 equal to 0.0 Kg i.e. no payload. This formed the first filter in the MMAE. The filter was run in simulation with the feedforward element given the same value for the payload as the robot (0.0 Kg). The residuals of the Kalman filter were monitored as the payload on the robot was allowed to increase while holding the value

of the payload in the feedforward element at 0.0 Kg. The value of the payload, a_2, that produced filter residuals that were significantly worse than the matched payload case was used to build the next filter in the MMAE. The process was repeated using a_2 as the starting point. In this manner the entire parameter space was discretized. The upper bound on payload was assumed to be 5.0 Kg. The system and measurement noises in the Kalman filter based on each a_i were tuned to produce the smallest residuals when the difference between the assumed payload in the feedforward element and the payload on the robot equaled a_i.

The MMBC algorithm was tested in digital simulation employing several different robot arm trajectories. Each trajectory stressed a different aspect of the MMBC control scheme and in all cases the potential of using the new MMAE technique to estimate the payload was demonstrated. The results were validated by using the same algorithm to control an actual PUMA-560. No additional tuning of the filters in the MMAE was performed and the results still showed the payload estimate could radically improve tracking performance. Tuning the filters would produce a better estimate of the payload and further improve the tracking errors.

1.5 Accomplishments

A new and unique adaptive robot control algorithm has been developed and evaluated. A novel parameter estimation scheme had to be produced to operate within the closed-loop model-based control structure. The resulting control technique produced tracking errors that matched artificially informed controller (SMBC), (a controller that has been informed of the actual payload value) in both simulation and experimentation for a PUMA-560.

An analysis of the perturbation plant, $F(a,t)$, was performed. The analysis indicated that, for a given trajectory, $F(a,t)$ was not constant. This realization indicates that the often used constant $F(a,t)$ assumption is valid only for very slow trajectories. This study also revealed that the dependence on a in the perturbation

plant was reduced because of the PD feedback loop.

The end result of this research moves the Air Force one step closer in the trek to produce a robot that emulates human motion. The findings in this thesis investigation can be expanded to other areas that involve closed loop estimation of a parameter needed to improve nominal trajectory computations.

1.6 Organization

The remainder of the thesis is broken into four chapters. Chapter 2 reviews current adaptive robot control schemes. The discussions include the different system representations as well as the assorted parameter estimation algorithms employed. Chapter 3 develops the MMBC for the general case. Chapter 4 presents a case study for the PUMA-560. The results from the digital simulation and the experimental evaluation are discussed. Chapter 5 contains the concluding remarks and recommendations for future research.

II. Literature Review

2.1 Introduction

The heart of a robotic manipulator is the control scheme. The controller moves the robot along a given trajectory from one point in space to another in the performance of a predetermined task. Existing industrial designs are inadequate for high speed control of manipulators [Lea88a,LS88a]. High speed is necessary for robotic flightline maintenance and telepresence applications. These applications usually involve scenarios where the mass of the payload and the environment are not known explicitly and may be time-varying. Unknown and time-varying parameters cause uncertainties in the control system design. These uncertainties as well as other system noises must be accounted for or adapted to in order to maximize the performance of the robot.

The following review of robot control examines previously proposed centralized control techniques that do not require additional measurement data such as torque or force. Specifically, this review covers the classical approach to robot control [Luh83,Goo85] and continues with four other prominent robot control themes proposed in the current literature. They are Model Reference Adaptive Control (MRAC) [DD79,Ser87,Goo85], adaptive control using an autoregressive model [KG83], adaptive perturbation control [LC84,dVW87], and dynamics-based or model-based adaptive control [Lea88a,LS88a,CHS87,SL87a,Goo85].

2.2 Background

A robot can assume many different physical configurations depending on the particular application (see Figure 2.1). A typical industrial robot consists of mechanical links connected by rotary or sliding joints providing six degrees of freedom. The links are moved by a drive system with electric, hydraulic or

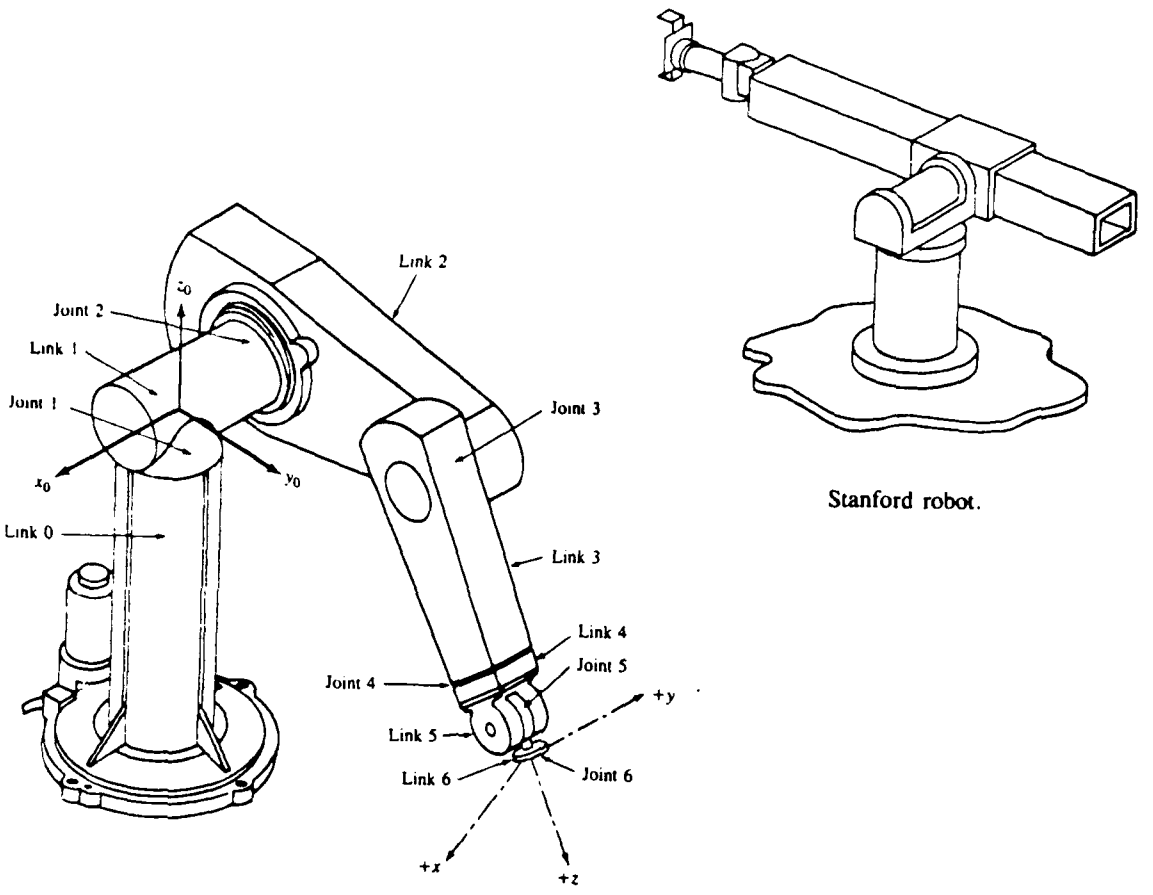

Figure 2.1. Some Typical Robot Configurations

pneumatic actuators. The equations of motion for a single link can be expressed as a linear differential equation. However, when the links are connected together they become a set of complex nonlinear, coupled differential equations [FGL87]:

$$N\Upsilon(t) = [D(q,a) + N^2 M]\ddot{q} + h(\dot{q},q,a) + N^2 B_m \dot{q} + \tau_s + g(q,a) \qquad (2.1)$$

where:

- n =the number of links in the robot

- $q, \dot{q}, \ddot{q}$ = n-vectors of joint angles, velocities, and accelerations.
- $a(t)$ = m-vector of parameters representing the unknown load as a function of time.
- $N = n \times n$ diagonal matrix of gear ratios for each joint($\frac{motor\ velocity}{link\ velocity}$).
- $D(q, a) = n \times n$ matrix of manipulator inertias which depend on the load and the position of the manipulator.
- M = diagonal $n \times n$ matrix of actuator inertia terms.
- $h(\dot{q}, q, a) = n$-vector of centrifugal and Coriolis torques.
- $\tau_s = n$-vector of static friction torques.
- $B_m = n \times n$ diagonal matrix of damping coefficients
- $g(q, a) = n$-vector of gravity loading terms.
- $\Upsilon(t) = n$-vector of joint motor torques.

2.3 Conventional Control

In conventional robot control, the complex movement of a robotic manipulator is separated into the independent control of a series of single links. Luh presents a detailed development of a transfer function for a single link for unity feedback and electrical actuators [Luh83]. For link i in the LaPlace domain, the transfer function has the form of:

$$\frac{Q_s(s)}{Q_d(s)} = \frac{NK_qK_i}{RJ_{eff}[s^2 + \frac{(RB_{eff}+K_tK_b)}{RJ_{eff}}s + \frac{NK_qK_i}{RJ_{eff}}]} \tag{2.2}$$

where:

- Q_s = commanded input position.
- Q_d = output position.

- N = gear ratio.
- K_q = encoder conversion constant in $\frac{V}{rad}$.
- K_i = torque constant of the motor in $\frac{oz \cdot in}{Amp}$.
- K_b = back EMF constant in $\frac{V \cdot s}{rad}$.
- R = resistance of the motor windings.
- J_{eff} = effective inertia ($N_i^2 M_i + D_{ii}$).
- B_{eff} = effective dampening coefficient ($N_i^2 B_{m_i}$).

In Luh's development the motor inductance was assumed to be negligible compared with the motor inertia. Equation (2.2) represents a second-order transfer function with its poles in the left half s-plane. Goor states that the motor dynamics must be included in the robot model in [Goo85] and develops a transfer function that includes motor inductance. The result is a third-order system with the driving input being motor voltages instead of torques. Goor maintains that, with motor dynamics included in the link transfer function, the speed of the robot can be increased without sacrificing performance [Goo85:page 7].

The gains of a second or third-order robot control system have upper limits determined by the resonant frequency of the structure and the desire for no overshoot. Overshoot could cause the robot to hit the environment. On the other hand, high gains are required to reject unmodeled disturbances such as changes in the payload. From Equation (2.1) it can be seen that the larger the unmodeled payload, the larger the disturbances. As shown in Equations (2.1) and (2.2), the payload contributes to the J_{eff} term and affects the natural frequency and dampening coefficient of the system. The standard industrial practice is to tune the control law for critically damped response with the assumed maximum payload [LS88a].

In order to limit the effect of disturbances, the speed of the robot must be held to a minimum. This insures stability of the robot over the entire operational

envelope. With high gear ratios, the simple second-order model can be used for point-to-point control if the gains are properly adjusted and the speed is kept within bounds. If the payload information were known *a priori*, many of the disturbances (e.g. gravity and Coriolis/centrifugal) could be compensated, and the controller gains could be adjusted to provide maximum stable performance. The desire for payload information a priori is similarly true for the third-order model.

2.4 Model Reference Adaptive Control (MRAC)

Model Reference Adaptive Control (MRAC) [DD79,Ser87,LE87] (see Figure 2.2) is a self-tuning approach based on the assumption that a second-order model is an adequate representation of the actual dynamics of the robot and that variations in the payload only affect the inertia values. The reference input is applied to the robot arm to produce positions, velocities and accelerations of the links. The same reference input is passed to the desired reference model and a desired position, velocity and acceleration. The difference between the actual and the desired is used to calculate a set of feedback gains that generate the torque required to force the robot back to the desired trajectory. The model is assumed to be a set of decoupled linear and time-invariant equations that are chosen to meet the desired performance specifications. The MRAC approach does not require *a priori* knowledge of any manipulator dynamic parameters or payload.

An attempt is made to address the unknown parameter problem employing an adaptation scheme. The adaptive algorithm is used to adjust the feedback gains and thus force the robot to perform like the assumed model.

Several different adaptation schemes have been proposed and evaluated in the current literature [DD79,Ser87,LE87]. All of those techniques are based on driving the errors between some reference input and measured values asymptotically to zero. No *a priori* knowledge of the payload or manipulator parameters is assumed. The control systems are shown to be asymptotically stable by applying Lyapunov's

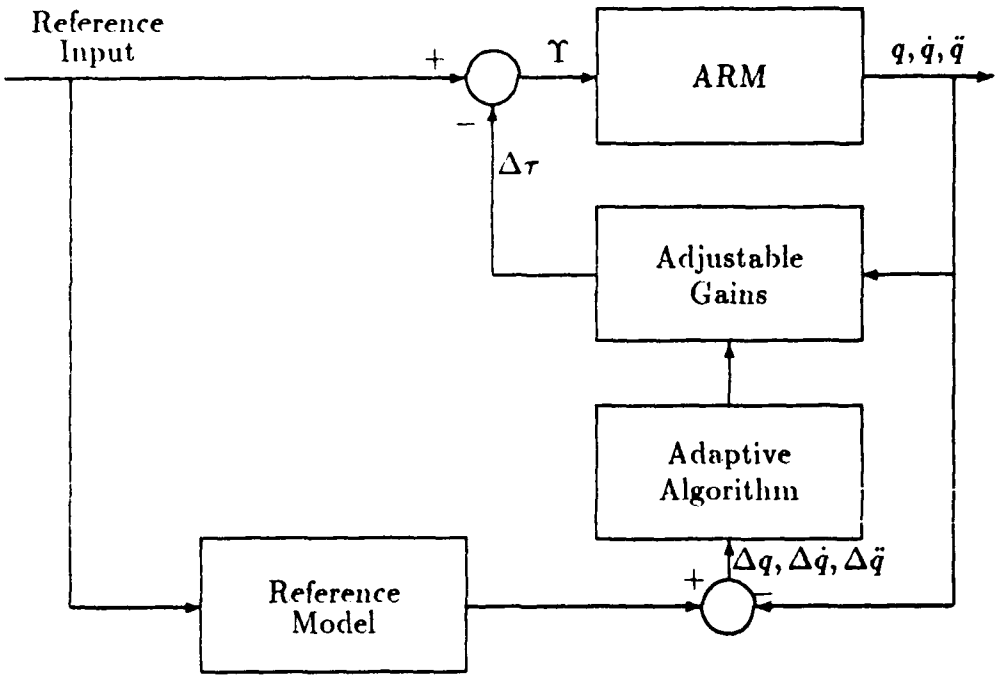

Figure 2.2. Model Reference Adaptive Control

second method [DH81]. However, that form of stability, while extremely powerful, does not guarantee that the response of the systems meets any sort of performance optimality criteria. The MRAC approach also neglects system noises and uncertainties such as encoder noise, gear backlash, initial misalignment of the links, and mismodelling. An analysis of [DD79] and [Ser87] follows.

Dubowsky and DesForges first proposed the MRAC for robot control in [DD79]. The payload and end effector were assumed to be part of the robot's last link and the load was counter-balanced so that gravity was not a factor in the system dynamics. A quadratic error function was used to minimize the difference between the reference model and the actual system dynamics. A steepest decent method is applied to the error function to find the equations defining the position and velocity feedback gain adjustments. The adaptation scheme requires position and velocity error information [DD79]. The acceleration error was set equal to zero. The stability of the control system was determined by eigenvalue placement.

Dubowsky and DesForges tested their algorithm in simulation for a three-link robot. The proposed scheme performed adequately in the setup described in [DD79]. The adaptation scheme was able to adjust the feedback gains to meet changes in the payload without any external excitation applied to the load. External excitation could be used to enhance the observability of the desired parameter. There was a short period of time that the errors were excessive when the estimator was learning. Dubowsky and Kornbluh experimentally evaluated the proposed scheme using only the second link of a Puma 560 with a step input [DK85]. The reference model that was used assumed that there was no coupling between the links, and the controller adapted only to changes in the self inertia matrix. The significant inertia coupling [LS88a] between links was ignored. Leahy, et al. have shown in [Lea87b] that the position errors committed by the MRAC approach are large and the vibration is excessive when the robot is simultaneously moving multiple links at high speeds.

Seraji proposed a MRAC approach that uses feedforward and feedback compensation and a Lyapunov adaptation algorithm [Ser87]. The feedforward compensator was based on a second-order equation designed to operate at a nominal point. The feedforward element acted as the inverse of the robot model and was used to linearize the system dynamics equations about that nominal operating point. The gains in the feedforward compensator were adjusted as the robot moved along a given trajectory [Ser87:page 194]. A Proportional-plus-Derivative (PD) feedback controller was used to improve the tracking performance of the robot, and the gains were adjusted to cope with changes in the operating point. The operating point was established by adding a third input torque to the sum of the feedforward and feedback torques. This auxiliary input was a linear function of the position and velocity errors. The coefficients of the auxiliary torque function were found experimentally and represented the relative weighting between the position and the velocity errors. The estimator was driven by the input trajectory and the error

in position and velocity. The outputs of the estimator were the PD gains, the coefficients in the feedforward element, and the auxiliary torque. No insight was provided as to how to select the initial starting values of these parameters.

The control scheme proposed by Seraji was compared in simulation against a manipulator modeled by Equation (2.1). The robot was assumed to be a two-link system with gravity acting in the plane of the links, with the payload modeled as part of the last link. The initial auxiliary and controller gains were arbitrarily set to zero. The results in [Ser87] show that the actual joint positions closely followed the commanded position.

A decentralized version of Seraji's proposed technique was experimentally evaluated on a Puma-560 [Ser88]. The trajectories used were very slow (20° per second commanded angular velocity), which eliminated any inertia coupling and viscous friction effects (see Equation (2.1)). As shown in [Lea87b], slow trajectories are not good tests of an algorithm's merits because the nonlinear effects remain negligible.

The MRAC method is based on a system model that does not include the known dynamics of the robot. This approach also requires a slow trajectory to keep the inertia coupling and viscous friction negligible.

2.5 Adaptive Control using an Autoregressive Model

Koivo and Guo have proposed an adaptive control method based on an autoregressive model [KG83]. As with the previously reviewed methods [DD79,Ser87], a second-order model was used to represent the system dynamics, and the coupling between the links was assumed to be negligible. The differences between the autoregressive and the MRAC approaches is concentrated in how the models were utilized and the approach taken to estimate the unknown parameters. Instead of a differential equation of motion as in the MRAC, Koivo and Guo proposed a second-order stochastic difference equation. The coefficients of the defining equa-

tion were determined by a least-squares error curve fit of the reference input data to the output data. The adaptation scheme does not require the controller to have *a priori* information about the payload or the configuration of the manipulator.

A recursive algorithm estimated the unknown parameters at t_i using the sampled outputs at t_{i-1}, conditioned on the measurement at t_i. To account for differences between the assumed model and the actual model, a noise component was added. The noise was assumed to have a Gaussian distribution with zero mean and a covariance of $\Re$ [KG83:page 164]. The noise addresses only model uncertainty and the other system noises were not taken into account. The assumed noise distribution was not substantiated. A more accurate model of the system dynamics and better representation of other system noises are needed to improve tracking performance.

Simulation of the proposed controller, against the same reference nonlinear equations of motion that Seraji employed [Ser88], showed that the output followed the reference closely except when the trajectory changed directions. At such times, the model output oscillated about the reference.

Koivo and Guo's autoregressive model control method was based on a system model that does not include the known dynamic structure of the robot. This method also requires the robot to move slowly to maintain the assumption that inertia coupling and viscous friction were negligible.

2.6 Adaptive Perturbation Control

The perturbation control approach linearizes the full set of nonlinear, coupled differential equations of motion about a nominal operating point [LC84,dVW87]. The nominal system dynamics equations were then used in a feedforward compensator to control the gross motion of the manipulator. A feedback compensator was also employed to compensate for small perturbations about that nominal operating point. The feedback compensator used full-state feedback and is defined by:

$$\delta\dot{x} = F\delta x + A\delta u \tag{2.3}$$

where:

- δx = the perturbations of the states.
- F = the Jacobian of the defining system dynamics equations.
- A = the Jacobian of the input distribution matrix.
- δu = the perturbation control inputs.

A least-squares or recursive least-squares estimation scheme was employed to estimate the unknown F and A matrices. The technique of linearizing the nonlinear equation about a nominal operating point and then driving the system back to the nominal is an approach often used for nonlinear control problems. In fact, this research effort used the same tact. However, the approach proposed in [LC84] and [dVW87] does not take into account any system noises and, as shown in [SLG78], a least-squares or recursive least-squares estimation scheme will produce a biased estimate of the unknown parameters if system noises are not properly modeled.

Lee and Chung utilized a Newton-Euler formulation of the manipulator dynamics in the feedforward component [LC84]. This provided for a quick and easy solution of the nominal trajectory problem. However, it did require *a priori* knowledge of a nominal payload and the configuration of the manipulator. The feedback compensator used the Lagrange-Euler formulation of the equations of motion which permitted determination of the linearized perturbation equations. A recursive least-squares estimation scheme was used to determine the parameters needed in the feedback compensator. The parameter estimates were based on the differences between the input and output positions, velocities and accelerations. The controller was then formulated as a linear quadratic control problem, optimized to drive the perturbation states to zero at each iteration [LC84:page 245].

Lee and Chung's proposed scheme was compared in simulation to the computed torque technique on a three-link robot [LC84]. The proposed approach estimated the perturbation control input coefficients which were a function of the payload. The reference controller was blinded to payload variations. The proposed approach had smaller peak tracking error, smaller end position errors and required less control energy during the trajectory, than did the uninformed reference controller. What was not indicated in [LC84] was how the nominal trajectory was computed. The common assumption is that the nominal operating condition is the unloaded manipulator. The nominal is a function of the payload and should be updated as the payload is changed, in order to keep the perturbations small.

A forgetting factor was used to deemphasize the old estimates since the model used was not accurate enough to propagate the estimates forward over multiple sample periods. The forgetting factor was determined by numerous simulation runs. No experimental evaluation of the proposed technique has been performed, perhaps because of the large computational power requirements.

deSilva and Van Winssen in [dVW87] used the same basic approach of a nominal feedforward component and a perturbation feedback component with the same linear quadratic controller as [LC84]. The difference was that deSilva and Van Winssen used a precomputed gain matrix for the feedback compensator. As the manipulator moved along its trajectory, the appropriate gains in the controller were switched in and out. Input disturbances were handled by adding a disturbance torque, and model errors were handled by adding an error to the model parameters. Both sources of error were assumed to have a zero mean and a standard deviation of 7% [dVW87:page 107], but no information was given as why 7% nor the type of distribution used.

deSilva and Van Winssen's proposed approach was simulated on a two-link robot, and the tracking errors were compared for runs with and without the feedback element in the controller. As expected, the proposed control system had diffi-

culty following the desired trajectory using only the feedforward element. Adding the feedback element to the controller significantly improved the tracking ability of the control system.

However, as was the case with [LC84], the feedforward element needs to have payload information in order to keep the perturbations small. Both [LC84] and [dVW87] need to include more information in the system models about the noise found in a real robot to reduce estimation errors. Without a more accurate system model, the actual performance of the adaptive perturbation controller cannot be fully assessed.

2.7 Model-Based Control

Model-Based control can be adaptive [CHS87,SL87a,AAH86,MG86,LS88] or non-adaptive [Lea88a,LS88a]. A more common name for the most general form of the non-adaptive model based control method is the computed torque technique [Cra86]. Both adaptive and non-adaptive approaches require control torques to be generated based on the dynamics model of the manipulator, and both adapt to changes in the configuration of the robot. The adaptive versions of the model-based method adjust to changes in the payload as well.

Dynamic compensation is employed in the model-based controller to reduce the effect of the disturbances caused by differences between the modeled system and the actual system. The compensation typically takes on the form of a feedforward component that linearizes the equations of motion (see Figure 2.3) by compensating for gravity acting on the link; the coupling of the torques between the links of the robot; and the effects of the centrifugal forces generated as the robot is moved. The feedforward compensator is given the desired trajectory and produces a nominal torque. The nominal torque is applied to the robot arm and the manipulator moves along a trajectory. The difference between the desired trajectory and the actual trajectory is used by the feedback compensator to produce the torque required to

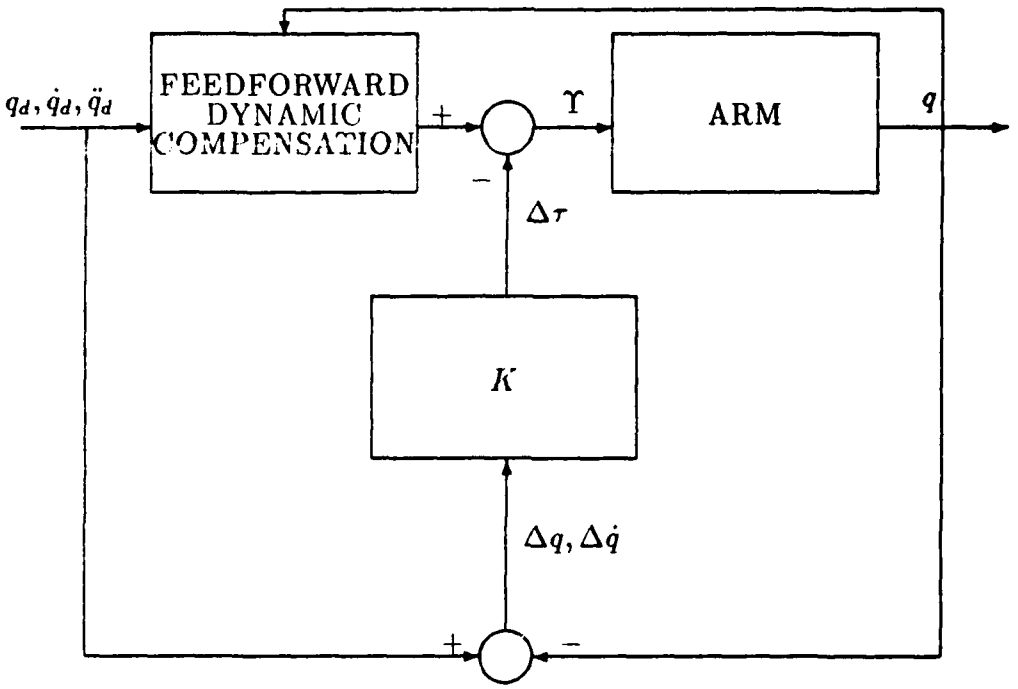

Figure 2.3. Typical Model-Based Control System

drive the error to zero. The feedforward element requires knowledge of the payload and friction in the system. It also requires that the commands to move the robot include not only the desired position but also the desired velocity and acceleration along the trajectory. Luh demonstrates in [Luh83] that proper compensation can improve the performance of the robot for a given task. The feedback element is used to reject any disturbances and reduce the tracking error of the robot.

The computed torque scheme begins with Equation (2.1) as the torque required to move the link along a given trajectory. The movement of each link is described in a coordinate frame attached to the robot defined by the Denavit-Hartenberg representation [FGL87]. The defining equations of motion of this mechanical structure is a set of nonlinear, coupled differential equations expressible in a Lagrange-Euler or a Newton-Euler formulation.

If the desired trajectory information is assumed to be known the desired control law can be written as:

$$\Upsilon_a(t) = D_a(q,a)[\ddot{q}_d + K_v\dot{e} + K_p e] + \beta_a \tag{2.4}$$

where the subscript a denotes actual and:

- $\ddot{q}_d$ = desired acceleration vector.
- e = position error vector$(q_d - q)$.
- $\dot{e}$ = velocity error vector.
- K_v = velocity error $n \times n$ diagonal gain matrix.
- K_p = position error $n \times n$ diagonal gain matrix.
- $\beta_a = h_a(\dot{q},q,a) + N^2 B_{m_a}\dot{q} + \tau_{s_a} + g_a(q,a)$ (from Equation (2.1)).

By equating Equations (2.1) and (2.4) and assuming that the modeled dynamic terms equal the real manipulator dynamics, the result is:

$$D_a(q,a)[\ddot{e} + K_v\dot{e} + K_p e] = 0 \tag{2.5}$$

In Equation (2.5) the inertia matrix is always positive definite. Therefore the bracket term must equal zero and the error states asymptotically approach zero. K_v and K_p are diagonal and the bracketed term produces a set of linear second order perturbation equations with their poles in the left-half plane. The implementation of this set of equations puts the pole placement at the discretion of the designer.

The computed torque technique was experimentally evaluated in [Lea88a] and [LS88a] for vertically articulated manipulators and in [Kho88], [AAH86] and [AAH88] for serial-link direct-drive arms. The feedback gains used were experimentally determined. The results showed in all cases, when a complete system

dynamics model which, included payload information, was used for the control of a manipulator, the tracking of the robot was superior to tracking when the compensator did not include the true payload information. Because the feedback gains were high to reject disturbances, the controller was very stiff. Most robot applications desire both tracking performance and minimum compliance. Low compliance is desirable to improve the interactions between the robot and its environment. The model-based control system would have to adapt to changes in the payload if the non-adaptive approach were to be implemented in a changing environment [Lea88a,AAH85].

One problem with the computed torque approach is the need for payload information. Leahy has shown that when the payload is known and the controller is tuned to match the payload, the performance of the robot is greatly improved [Lea88a]. A model-based controller can only adapt to changes in the configuration of the robot. In a changing environment the payload may not be known. This can be overcome by using an adaptation algorithm to estimate the payload and other required parameters and formulating the feedforward compensator to use these estimates. Figure 2.4 shows a block diagram of how a typical adaptive model-based control system might look. The addition of the estimator to the model-based control system takes the error in the position and produces an estimate of the unknown parameter. The estimate of the parameter is used by the feedforward compensator to produce a more accurate nominal torque. The feedback gains could also be recomputed using the parameter estimate. As with the MRAC approach, various parameter estimation schemes have been used to fill the estimator block. The following adaptive model-based control schemes employed the basic computed torque approach discussed above and adjusted the feedforward and/or feedback elements on-line to match the payload.

Craig, Hsu, and Sastry proposed to use tracking errors in the joint positions and velocities to drive the estimator for the mass and the feedback gains

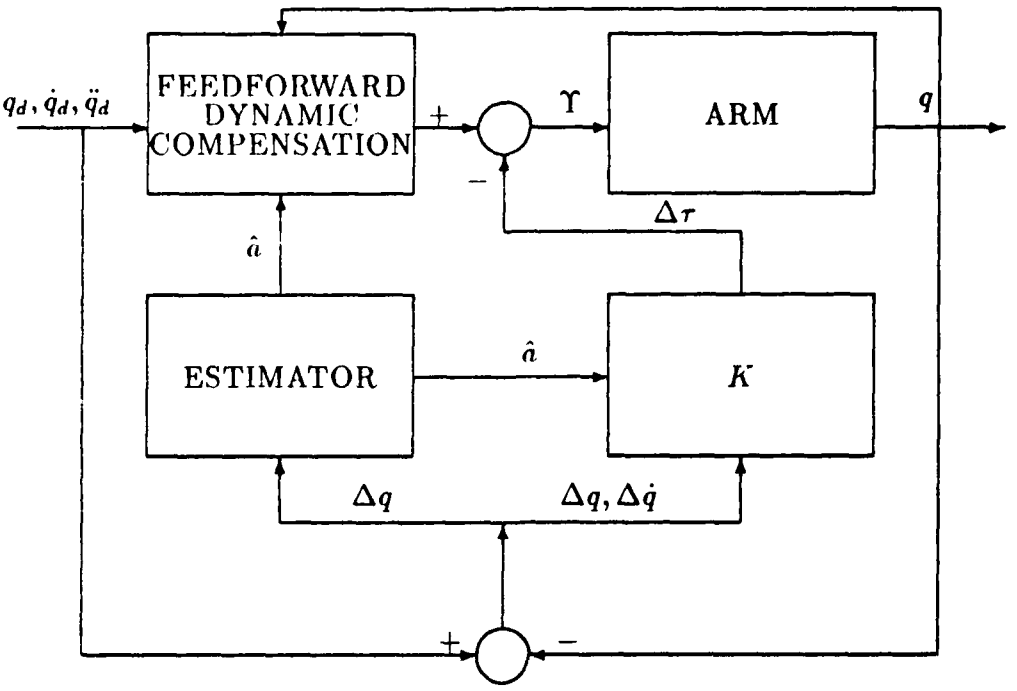

Figure 2.4. Example Adaptive Model-Based Control System

[CHS87:page 16]. The control scheme required joint accelerations, which cannot be measured; an adaptive feedforward element made up of an estimated "mass matrix" [CHS87:page 18] which must be inverted; and an adaptive feedback element. The adaptive scheme was based on a Lyapunov stability approach. The basic adaptation law was given by:

$$\dot{\hat{\mathcal{F}}} = \Gamma h^T \hat{M}^{-1} E^1 \tag{2.6}$$

where:

- $\hat{\mathcal{F}} = r$ vector of the parameters to be estimated.
- $\Gamma = r \times r$ diagonal scaling matrix.
- $h^T = r \times n$ matrix of dynamics terms and is $h(\dot{q}, q, a)$.
- $\hat{M}$ = the estimate of an $n \times n$ manipulator mass matrix.

- $E^1 = n$-vector of servo errors$(\dot{e}, e)$.

The solution of Equation (2.6) for the bounded initial condition on $\hat{\mathcal{F}}$ provides the update for the unknown parameters [CHS87:page 20]. Lyapunov theory guarantees that the controller will be stable and that the steady state errors will asymptotically approach zero. However, in most trajectories of interest, the desired acceleration is not zero and therefore, the manipulator is not operated in a steady state manner. To improve the tracking performance of the manipulator, Γ must be adjusted experimentally to produce a control system that meets the desired performance. Adjusting Γ trades peak error in tracking for speed of adaptation.

The experimental results on the first two links of an Adept One manipulator showed good performance [CHS87]. However, the trajectories used were slow and near constant acceleration. The estimates of the parameters appeared to be biased. If more of the system uncertainties were included in the system model, the estimates would be more accurate.

Middelton and Goodwin employed the same tracking error states to drive the estimator and a nearly identical Lyapunov adaptation scheme as Craig, et al., but required the inversion of the joint inertia matrix in their estimator [MG86:page 68]. This could lead to problems because of computational time and the fact that the inertia matrix can become singular due to numerical rounding in the computer. No simulation or experimental results have been presented.

Slotine and Li used only tracking errors and joint positions and velocities to drive their estimator [SL87a:page 49]. The main concern of their proposed approach was how to reduce tracking errors, and not how quickly the estimator converged. The adaptation scheme was based on Lyapunov theory and was nearly identical to the one used by Craig, et al. The exception was that the adaptation law was a function of the required joint velocities and accelerations instead of the desired quantities. The required velocities and acceleration are those values that

drive the steady state position errors toward zero. The constant Γ matrix was adjusted with different performance goals, i. e. reducing steady state position error as apposed to reducing the tracking errors.

Simulation results shows that the convergence was slow but the tracking error was small. The proposed technique was experimentally tested on a two-link semi-direct-drive robot arm [SL87b]. Gravity was not a factor due to the arm movements being restricted to the horizontal plan. The trajectories used consisted of .5 sec of movement and then .5 sec of zero velocity and acceleration. The results presented indicate that the adaptive control scheme has large tracking errors at the end of the first .5 sec interval. The errors were reduced during the next .5 sec interval but not driven to zero. Slotine and Li indicated that the remaining errors were largely due to noise corrupted velocity measurements [SL87a:page 1396]. In any case, the need to have a long stationary period at the end of the moving trajectory to reduce final positioning errors is not acceptable for telepresence applications.

Li and Slotine have also developed an estimation technique that was driven by the errors in the predicted values of the integral of the joint torque [LS88b]. Because filtered value of the torque are utilized in the estimator, joint accelerations are not needed. Four different adaptation schemes have been proposed, all based on a Lyapunov stability criteria. No simulation or experimental results of the new technique or the adaptation schemes have been presented as yet.

2.8 Summary

A review of current control schemes for robots has been presented. No single approach meets the performance needs for all applications. The model-based control scheme has shown superior results when the payload is known *a prior.*

Much has been gained in the area of robot control by current research; however, for the high performance robots needed in todays Air Force the simplifying assumptions used in the present control approaches are overly restrictive. There

is not enough information available on proposed adaptive model-based control schemes and therefore further investigations are justified. The control scheme advanced in this thesis will be an adaptive model-based approach that will attempt to account for all the system noises and uncertainties. The estimation scheme will provide an estimate of the unknown payload to the feedforward compensator. An algorithm used in difficult nonlinear estimation applications which incorporates system and measurement noises is based on a Bayesian estimation approach. The algorithm is called Multiple Model Adaptive Estimator, MMAE [DM87,KB83,LJ87a,MZ85,MR83,MS85,Net85,BG78,Aea77,Ber83,May82a, MH87, GW80,WW71]. The development of the MMAE and the resulting structure of the adaptive model-based control is presented in the following chapter.

III. Algorithm Development

3.1 Introduction

There are many control algorithms currently under study to improve the tracking capabilities of the modern robot. The previous chapter presented a review of current approaches. The model reference technique assumes a second-order model for the robot but discards any knowledge of the mechanical structure of the robot. The coefficients of the second-order model must be estimated [DD79,Ser87,Goo85]. The adaptive perturbation controller assumes a constant feedback plant and uses an on-line estimation scheme to provide the coefficients in the feedback element [LC84,dVW87]. The model-based approach uses the knowledge of the structure of the robot [LJ88b,LJS88,CHS87,SL87a]. For tracking applications, knowledge of the payload is also required and in general is not known. None of the mentioned techniques includes system and measurement uncertainties in their models of the robot system.

Chapter 2 discussed the model-based controller. It uses the nonlinear equations of motion in a feedforward element to compute the desired torque (see Equation (3.1)). Any mismatches between the model in the feedforward element and the actual robot are considered disturbances. A PD controller in a feedback loop is used to reject these disturbances. A large contribution to the disturbances in the system is the payload which consists of the mass, the center of mass, the radius of gyration and the moments of inertia (see Equation (3.1)).

$$[D(q,a) + N^2M]\ddot{q} + h(\dot{q},q,a) + N^2B_m\dot{q} + \tau_s + g(q,a) = N\Upsilon(t) \qquad (3.1)$$

- where the variables are the same as in Equation (2.1).

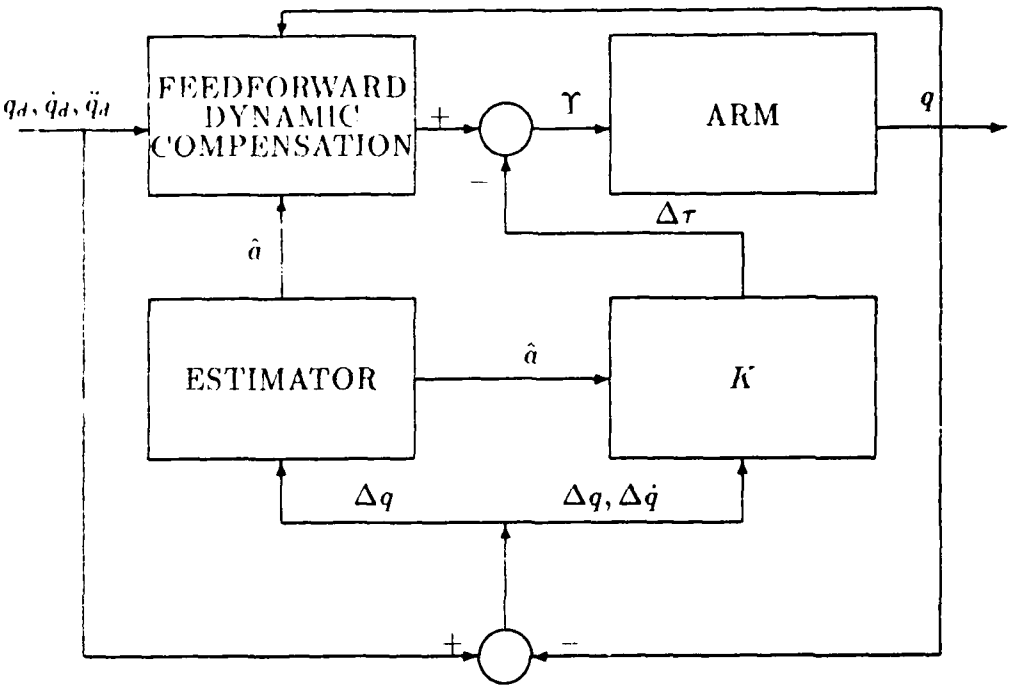

Figure 3.1. Adaptive Model-Based Controller

To reduce the disturbances in the system and to improve the overall tracking performance on-line, the parameter a must be estimated. One estimation technique used in many robot control schemes is based on Lyapunov's second method. This approach only guarantees that the system response will be asymptoticly stable. Another technique used a least-squares approach. However, without noise in the model, this estimator is biased [SLG78].

Our proposed solution is to combine the Multiple Model Adaptive Estimator (MMAE) with the model-based controller. The MMAE can provide better performance than the Lyapunov or least-squares approaches and it accounts for the noise in the robot system. The structure of the adaptive model base controller is shown in Figure 3.1. The overall control system has been called the Multiple Model-Based Control (MMBC) because the control system incorporates multiple models of the robot dynamics in the "estimator" block. The algorithm will be developed in this chapter.

3.2 *Nonlinear Estimation*

The nonlinear equations of motion (see Equation (3.1)) can be written in a more general form (see Equation (3.2)).

$$\ddot{q}(t) = f(q, \dot{q}, \Upsilon, a, z, t) \tag{3.2}$$

$$z(t) = h(q, \dot{q}, \Upsilon, a, t) \tag{3.3}$$

where:

- $z(t)$ = measurements
- $f(\bullet)$ and $h(\bullet)$ = are nonlinear functions of the arguments

As pointed out in the Chapter 2, the robot system has noise inherent in it. The sources of the noise arise from imperfect calibration of the robot, incorrectly modeled components of the robot, and imperfect measurements of the states. If the noises are assumed to be added linearly to Equations (3.2) and (3.3), the result is a stochastic nonlinear differential equation and associated measurement algebraic equation of the following form [May79]:

$$\ddot{q}(t) = f(q, \dot{q}, \Upsilon, a, z, t) + G'(t)W(t) \tag{3.4}$$

$$z(t) = h(q, \dot{q}, \Upsilon, a, t) + V(t) \tag{3.5}$$

where:

- $G'(t)$ Scaling matrix for the additive noise
- $W(t)$ =Zero mean, white Gaussian dynamics driving noise
- $V(t)$ =Zero mean, white Gaussian measurement noise

One solution to Equation (3.4) for the case where the noises are assumed to have Gaussian distributions is the Extended Kalman Filter ([May82a:pages 44-55]). This approach would require a to be included as states of the system. However, since a is slowly time varying compared to the states of the system it can be considered as a parameter and treated differently than the more rapidly varying states.

If the structure of the Multiple Model-Based Controller is used (see Figure 3.1), the control system can be separated into a feedforward compensator that produces a nominal output and a feedback element that produces a perturbation output. The motivation for this approach is to recast the problem into a perturbation regulator (see Figure 3.2). If the noise is assumed to contribute to the perturbation output, Figure 3.3 shows the structure of the resulting system. The feedforward element produces a nominal torque given the desired trajectory. The nominal torque applied to the robot generates a nominal position. Any difference between the nominal and the desired position is assumed to result from the disturbances in the system, W. The feedback gains, K attempt to drive the errors to zero. The perturbation system description, $F'(a,t)$, is the first-order result of the truncated Taylor series of $f(q,\dot{q},\Upsilon,a,t)$. The states of the controller then become the difference between the desired position and velocity and the actual position and velocity:

$$x(t) = \begin{bmatrix} \text{position error vector} \\ \text{velocity error vector} \end{bmatrix} \tag{3.6}$$

The noises directly affect $\dot{x}(t)$ and the measurements of the states. The system noises are assumed to be zero mean, white and to be pair-wise independent of each other. The measurement noises are also assumed to be zero mean, white and to be pair-wise independent, and to be independent of the system noises.

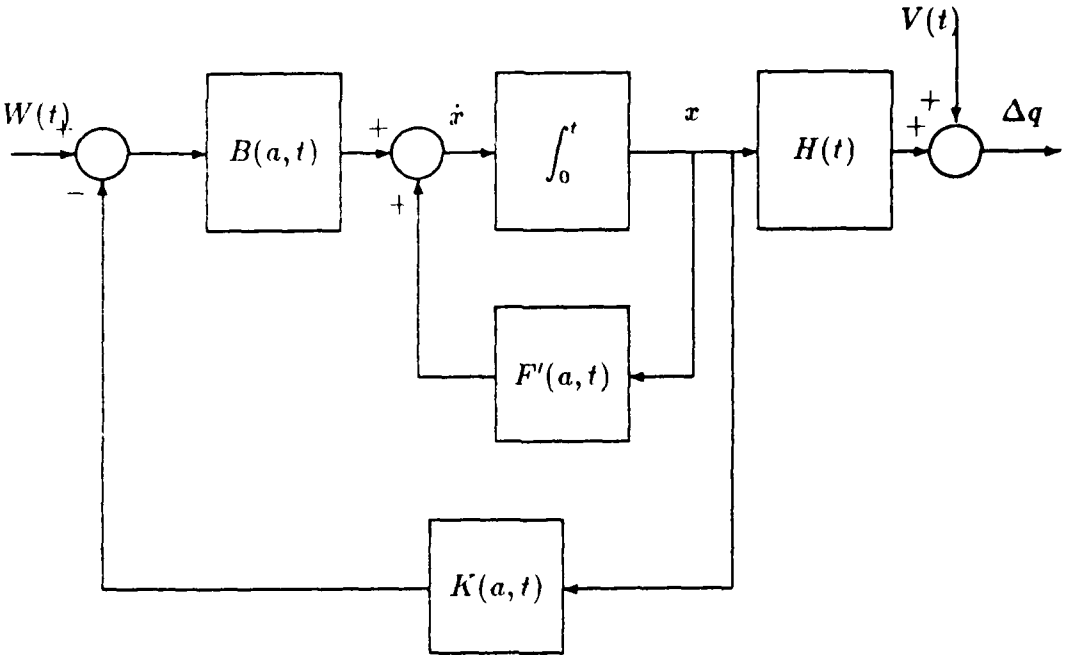

Figure 3.2. Perturbation Controller With Noise

The feedforward and feedback elements are dependant on the parameter a. Equations (3.4) and (3.5) can be written in the form:

$$\dot{x}(t) = F'(a,t)x(t) + G(t)W(t) \tag{3.7}$$

$$z(t) = H(t)x(t) + V(t) \tag{3.8}$$

where:

- $F'(a,t)$ = a nonlinear function of the payload and a linear function of the states that describes the homogeneous perturbation state dynamics characteristics.
- $z(t)$ = the noise corrupted measurements of the position error states.
- $H(t)$ = the measurement matrix that transforms the states into the measurement space.

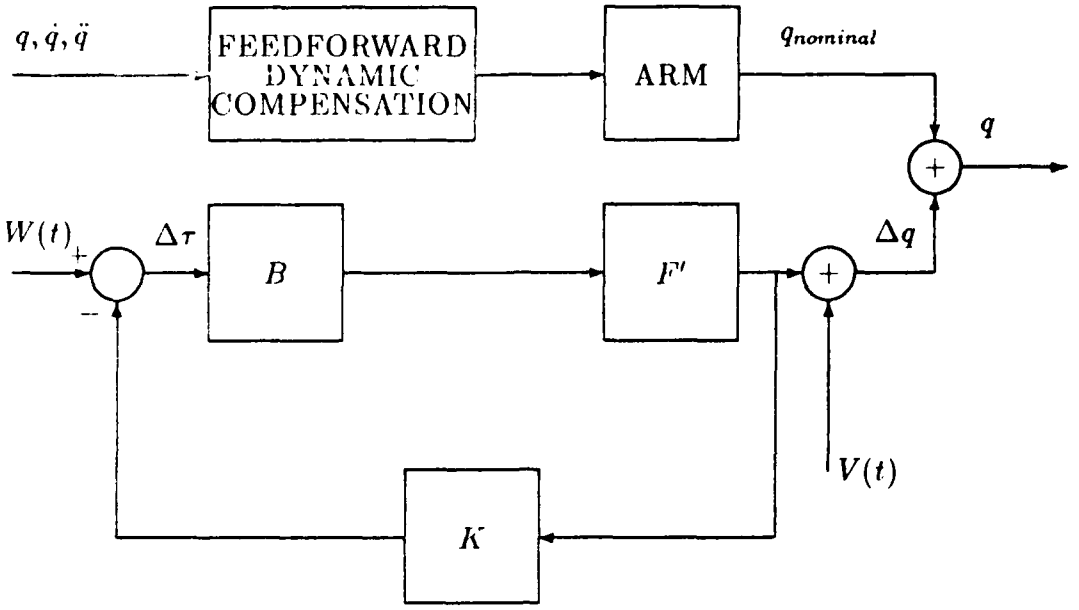

Figure 3.3. Feedforward Element with Perturbation Controller

- $V(t)$ = the measurement noise.
- $W(t)$ = the dynamics driving noise
- $G(t)$ transforms the noise into the state space.

With Equations (3.7) and (3.8), Bayesian estimation in a multiple model configuration can be used to determine the unknown parameter a [May82a:pages 129-136]. The basic premise of the Multiple Model Adaptive Estimation (MMAE) technique is that the continuous parameter a can be discretized, and thus can be assumed to be a member of the finite set of possible values, $(a_1, a_2, \ldots, a_K)$. The discretization of a must be large enough that there is a discernible difference between the models but not so large as to induce unacceptable errors in the estimate. The state estimator or Kalman filter based upon an assumed parameter value a_j and the models of (3.7) and (3.8) in a sampled data system would be:

$$\hat{x}(t_i^-) = \Phi(t_{i+1}, t_i)\hat{x}(t_i) \tag{3.9}$$

$$P(t_i^-) = \Phi(t_{i+1}, t_i)P(t_i^+)\Phi^T(t_{i+1}, t_i) + Q_d(t_i) \tag{3.10}$$

$$\hat{x}(t_i^+) = \hat{x}(t_i^-) + K(t_i)[z_i - H(t_i)\hat{x}(t_i^-)] \tag{3.11}$$

$$P(t_i^+) = P(t_i^-) - K(t_i)H(t_i)P(t_i^-) \tag{3.12}$$

$$K(t_i) = P(t_i^-)H^T[H(t_i)P(t_i^-)H^T(t_i) + R(t_i)]^{-1} \tag{3.13}$$

where:

- $\hat{x}(t_i^-)$ = the estimate of the state at time t_i just prior to the measurement being processed at t_i.
- $P(t_i^-)$ =the covariance of the state at time t_i^-.
- z_i = the noise corrupted measurement (in this case the position error state).
- $H(t_i)$ = the measurement matrix that transforms the states into the measurement space.
- $\hat{x}(t_i^+)$ = the state at time t_i after the measurement has been processed at t_i.
- $P(t_i^+)$ =the covariance of the state at time t_i^+.
- $K(t_i)$ = the Kalman filter gain at time t_i.
- $\Phi(t_{i+1}, t_i)$ = the state transition matrix associated with $F'(a,t)$ of Equation (3.7), defined as the $n \times n$ matrix that satisfies $\dot{\Phi}(t, t_i) = F'(a,t)\Phi(t, t_i)$ with $\Phi(t_i, t_i) = I$.
- $Q_d(t_i) = \int_{t_{i-1}}^{t_i} \Phi(t_{i+1}, \tau)G(\tau)Q(\tau)G^T(\tau)\Phi^T(t_{i+1}, \tau)d\tau$ and $Q(t)$ is the strength of the Gaussian noise, $W(t)$:

  $E[W(t)W^T(t+\tau)] = Q(t)\delta(\tau)$.
- $R(t_i)$ = the strength of the Gaussian noise, $V(t_i)$: $E[V(t_i)V^T(t_i)] = R(t_i)$.

If a has been discretized into K different m- vectors, the MMAE would require K such linear Kalman filters to be run in parallel. Figure 3.4 shows the structure of the algorithm. Each of the Kalman filters is presented with the same measurement $z(t_i)$ and produces a state estimate based upon its internally assumed model. Also computed as part of the estimation process are the residuals, $r(t_i)$. The residuals are passed to an executive program that computes a conditional probability, $p_k(t_i)$ (see Equation (3.15)). The smoothed state estimate, $\hat{x}(t_i)$ is the sum of the products as indicated in Figure 3.4.

The residuals, $[z_i - H(t_i)\hat{x}(t_i^-)]$, from the filter with the most correctly assumed value of a, would be the smallest relative to its internally computed residual covariance, $[HPH^T + R]$. In effect, the state estimates that were propagated forward in time using the most correct state model, most closely match the actual measurements of the states at t_i.

$$p_k(t_i) \triangleq \text{prob}\{a = a_k \mid Z(t_i) = Z_i\} \tag{3.14}$$

$$p_k(t_i) = \frac{f_{z(t_i)|a,Z(t_{i-1})}(z_i \mid a_k, Z_{i-1}) p_k(t_{i-1})}{\sum_{j=1}^{K} f_{z(t_i)|a,Z(t_{i-1})}(z_i \mid a_j, Z_{i-1}) p_j(t_{i-1})} \tag{3.15}$$

where:

- $Z(t_{i-1})$ = the measurement history up to time t_{i-1}
- $f_{z(t_i)|a,Z(t_{i-1})}(z_i \mid a_k, Z_{i-1})$ = the conditional probability that the i^{TH} filter was correct. For the assumed Gaussian distribution it has the form $\frac{1}{(2\pi)^{n/2}|A|^{1/2}} e^{(-1/2r^t A^{-1} r)}$ where $A = [HPH^T + R]$.
- the denominator scales the conditional probability such that $\sum_{k=1}^{K} p_k(t_i) = 1$

The conditional mean of the parameter a at t_i is given by:

$$\hat{a}(t_i) \triangleq \sum_{k=1}^{K} a_k p_k(t_i) \tag{3.16}$$

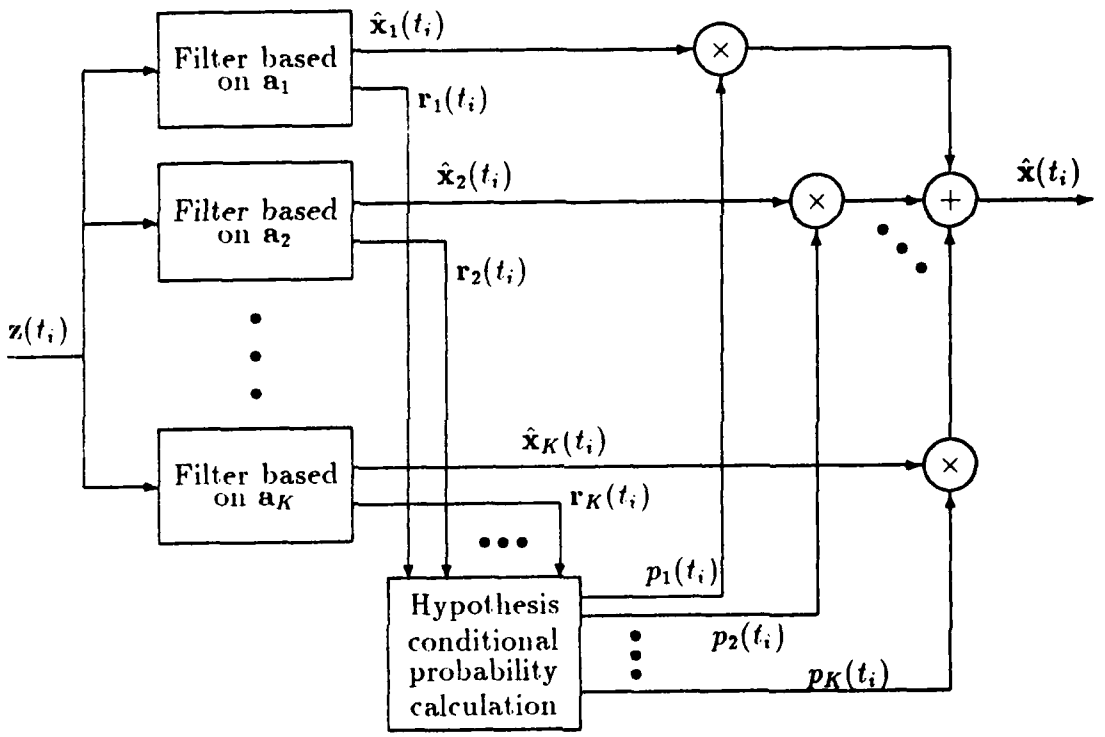

Figure 3.4. Block Diagram for the *Multiple Model Adaptive Estimation* Algorithm

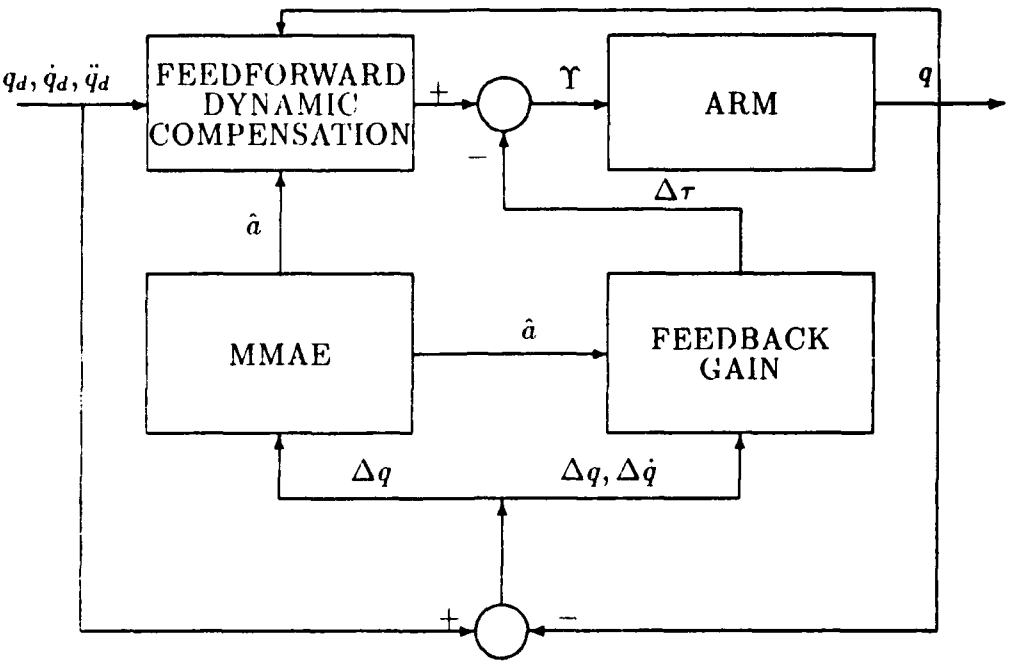

Figure 3.5. Block Diagram for the Multiple Model-Based Control (MMBC)

For a more detailed development of the MMAE algorithm see [May82a:pages 129-136].

The next chapter will use the Multiple Model-Based Control scheme (MMBC) developed here. The estimator in Figure 2.4 has been replaced with the MMAE (see Figure 3.5). The MMBC algorithm was employed for a case study on the first three links of the PUMA-560. The algorithm was tested in simulation and experimentally with very promising results.

IV. Case Study

The previous chapter developed a Model-Based Controller with Multiple Model Adaptive Estimation (MMBC). This chapter addresses the details of implementing that algorithm for robot control and uses a PUMA-560 as a case study. It will also discuss the evaluation of the plant or $F(a,t)$ matrix, the PD controller section, implementation of the Kalman filter, the MMAE and the simulator used to test the algorithm. In addition, experimental results of the new control scheme for the PUMA-560 testing are presented.

4.1 Introduction

One objective of this research was to demonstrate the potential of the MMBC technique for robot payload estimation and control. The PUMA-560 was selected as the case study because it is representative of a vertically articulated manipulator needed for telepresence applications. Tracking performance of the PUMA has been experimentally determined to be greatly affected by changes in the payload.

The nonlinear equations of motion (see Equation (2.1)) were reduced in the previous chapter to a nominal part plus a linear perturbation part (see Equation (3.7)). The state estimator based on the linear stochastic differential equation (see Equation (3.7)), was given as Equations (3.9)–(3.13) (the Kalman filter equations). For application on the PUMA-560 robot, the details of those equations and the conditional probability calculations (see Equation (3.16)), must be discussed.

The first 3 links of the PUMA-560 were used in the case study since reducing the payload vector to just the mass has minimal impact on the large link tracking performance. The payload was assumed to be a point mass rigidly attached to the end of the third link, and the MMAE is to provide an estimate of the mass of the payload. The single parameter of the payload, reduces a to the scaler case and decrease the number of filters needed in the MMAE to span the parameter space.

If additional parameters of the payload are to be estimated, the size of the MMAE would be increased. Another reason for limiting the case study to the first 3 links is that the control of the last 3 links can be decoupled from the first three, because inertia coupling forces between the two sections are negligible [Lea87a].

4.2 Perturbation Equations

The process of going from Equation (3.5) to Equation (3.8) requires the partial derivative of Equation (3.5) with respect to q and $\dot{q}$, evaluated at the nominal $q, \dot{q}, \tau, a_j$:

$$F'_j(a,t) = \left[\begin{array}{cccccc} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ \frac{\partial f_1}{\partial q_1} & \frac{\partial f_1}{\partial q_2} & \frac{\partial f_1}{\partial q_3} & \frac{\partial f_1}{\partial \dot{q}_1} & \frac{\partial f_1}{\partial \dot{q}_2} & \frac{\partial f_1}{\partial \dot{q}_3} \\ \frac{\partial f_2}{\partial q_1} & \frac{\partial f_2}{\partial q_2} & \frac{\partial f_2}{\partial q_3} & \frac{\partial f_2}{\partial \dot{q}_1} & \frac{\partial f_2}{\partial \dot{q}_2} & \frac{\partial f_2}{\partial \dot{q}_3} \\ \frac{\partial f_3}{\partial q_1} & \frac{\partial f_3}{\partial q_2} & \frac{\partial f_3}{\partial q_3} & \frac{\partial f_3}{\partial \dot{q}_1} & \frac{\partial f_3}{\partial \dot{q}_2} & \frac{\partial f_3}{\partial \dot{q}_3} \end{array}\right]\Bigg|_{\text{nominal}} \tag{4.1}$$

The configuration of the control system shown in Figure 3.3 represents a full state feedback regulator. The model of the plant used by the Kalman filters should reflect the actual plant as closely as possible. Therefore the plant matrix used by the Kalman filters should include the feedback gains. This approach to the estimation task allows the feedback loop to remain unbroken and the characteristics of the original closed-loop system to be unchanged. The alternative would be to include the MMAE as part of the feedback loop. The model of the closed loop plant matrix is:

$$F_j(a,t) = (F'_j(a,t) - G(a,t)K) \tag{4.2}$$

Further definition of the j^{th} Kalman filter is required. The $G(a,t_i)$ matrix in Figure 4.1 determines how the dynamics driving noise, $W(t)$ affects the states

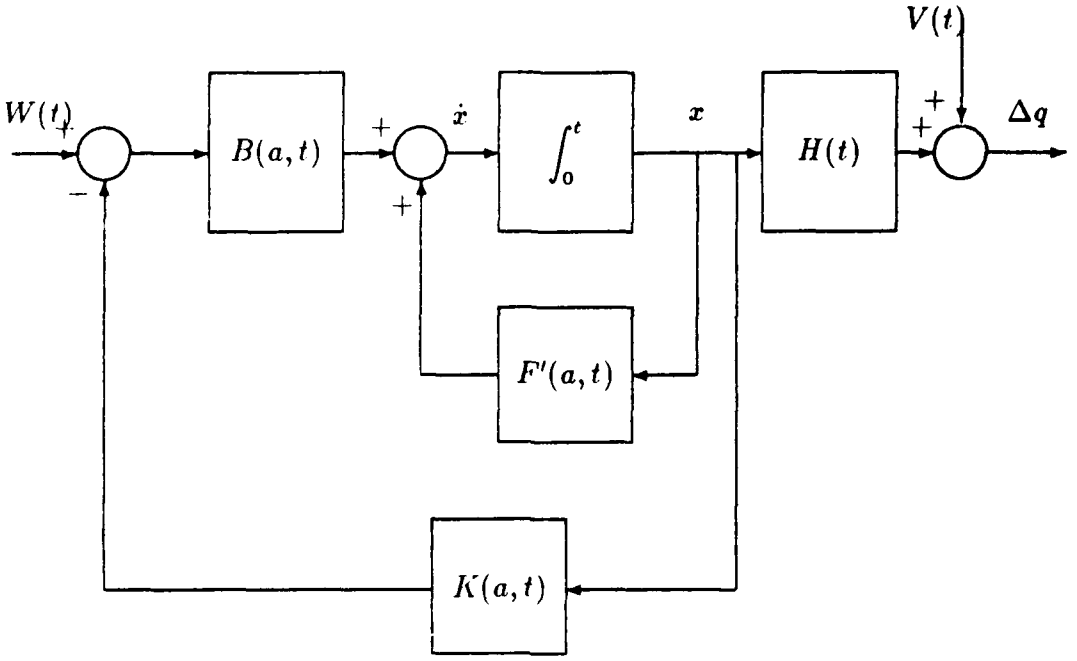

Figure 4.1. Perturbation Controller with Noise

of the controller. The $W(t)$ is assumed to be added to the torque applied to the robot. The matrix $G(a,t_i)$ transforms the torque into the state space. It has the form:

$$G(a,t_i) = \left[\begin{array}{ccc} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ D^{-1}(q,a) & & \end{array}\right] \text{ nominal} \tag{4.3}$$

where $D^{-1}(q,a)$ is the inverse of the $n \times n$ inertia matrix, $D(q,a)$ from Equation (3.1).

The only measurements available on the PUMA-560 are the actual joint positions. The position state in the Kalman filter is the difference between the desired position and the actual position (see Equation (3.6)). The $H(t_i)$ matrix

scales the state vector $x(t_i)$ to match $z(t_i)$. Since $z(t_i)$ is a linear function of the position states,

$$H(t_i) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \tag{4.4}$$

The actual calculation of Equation (4.1) is quite complex even for the first 3 links of the PUMA-560. To assist in evaluation of Equation (4.1), a commercial software package that works with symbolic equations call MACSYMA was used [Sym85]. A program using MACSYMA commands was developed to provide a symbolically reduced evaluation for Equation (4.1) (see Appendix A for program listing). The equations of motion for the first 3 links of the PUMA-560 developed by Tarn in [TB85] were fed into MACSYMA. The friction information included in the equations of motion was developed by Leahy and Saridis in [LS88a].

For the MMAE routine to provide a good estimate of a there must be a measurable difference between the system models based on different values of a. One means of assessing the differences in the plant is to examine the $F(a,t)$ matrix as a changes. An evaluation of $F(a,t)$ had not previously been presented in the literature, therefore an analysis was performed. A test trajectory that would highlight the tracking performance dependence on a was selected. The position and velocity every 7 ms along the trajectory was used to calculate the values of $F(a,t)$. The choice for 7 ms was established because of the experimental setup and will be discussed in Section 4.8. Two payload conditions, 0.0 Kg. and 5.0 Kg. were selected to provide upper and lower limits of possible payload values. The payload was added to the existing load of links 4, 5 and 6, whose total unloaded weight is 6.97 Kg. [TB85].

The real part of the eigenvalues of the $F(a,t)$ matrix were used as a measure of the differences between $F(a,t)$ for different payloads. The real part of the eigen-

values was selected as a convenient gauge of the response of the various modes of the system. The trajectory selected was fast enough that the nonlinear characteristic of the robot were excited. The trajectory is shown in Figures 4.2, 4.3 and 4.4 for each of the three links. The real parts of the eigenvalues at each point along the trajectory for the two payload cases are shown in Figure 4.5 for eigenvalues 1-2, Figure 4.6 for 3-4 and Figure 4.7 for 5-6. The numbering of the eigenvalues is arbitrary but an attempt was maintain the numbering between the different load cases.

As seen in Figures 4.5, 4.6 and 4.7, there is a slight difference between the models. It can also be seen that $F(a,t)$ is not constant in time and therefore must be re-computed along the trajectory. This thesis investigation pre-computed $F(a,t)$ at each point along the trajectory. An alternative approach was not investigated and was beyond the scope of this effort. Eigenvalue plots for a faster trajectory (Trajectory Three, see Figures B.5, B.6 and B.7 and a holding or zero trajectory (Trajectory Two, see Figure B.1) can be seen in Appendix C (Figures B.8, B.9,B.10, B.2, B.3 and B.4). It is apparent by comparing the eigenvalue plots for the three trajectories that the amount of change of $F(a,t)$ depends on the speed of the trajectory. A constant $F(a,t)$ can only be assumed if the trajectory employed is significantly slower than Trajectory One.

4.3 PD Controller

The feedback controller shown in Figure 3.2 is used to reduce the tracking error of the robot. A PD feedback loop was selected as a simple but effective controller to reject disturbances caused by errors in the model of the robot structure and unmodeled forces. As seen in Chapter 2, when the feedforward element correctly models the condition of the robot, the resulting feedback element is a second-order system:

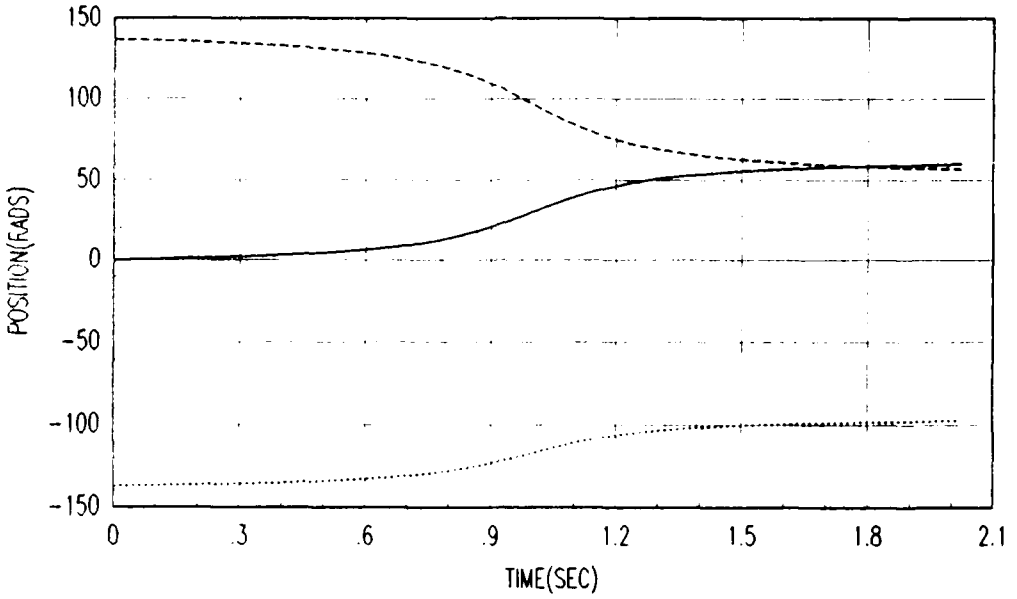

Figure 4.2. Trajectory One: Position

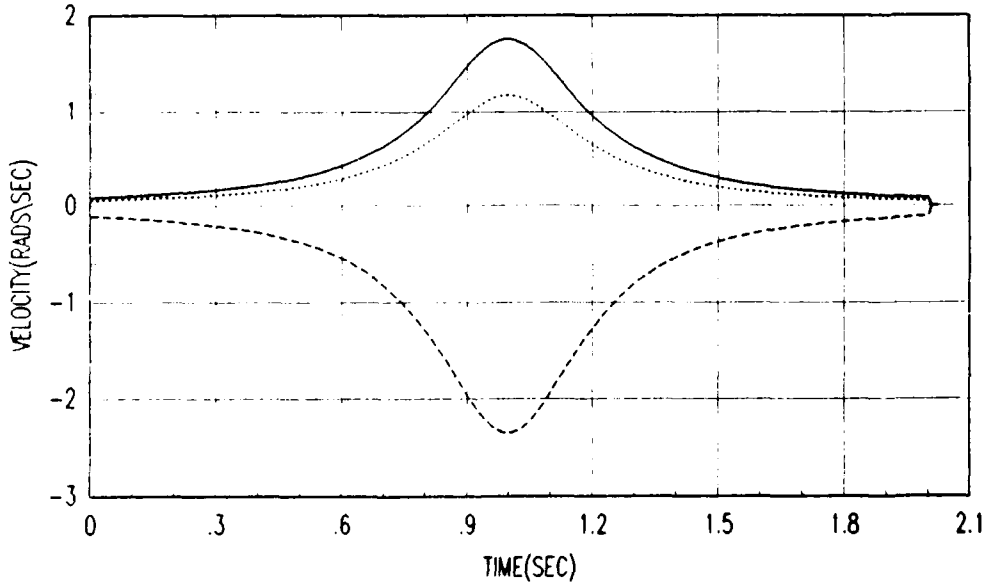

Figure 4.3. Trajectory One: Velocity

——	Link 1
· · ·	Link 2
- - -	Link 3

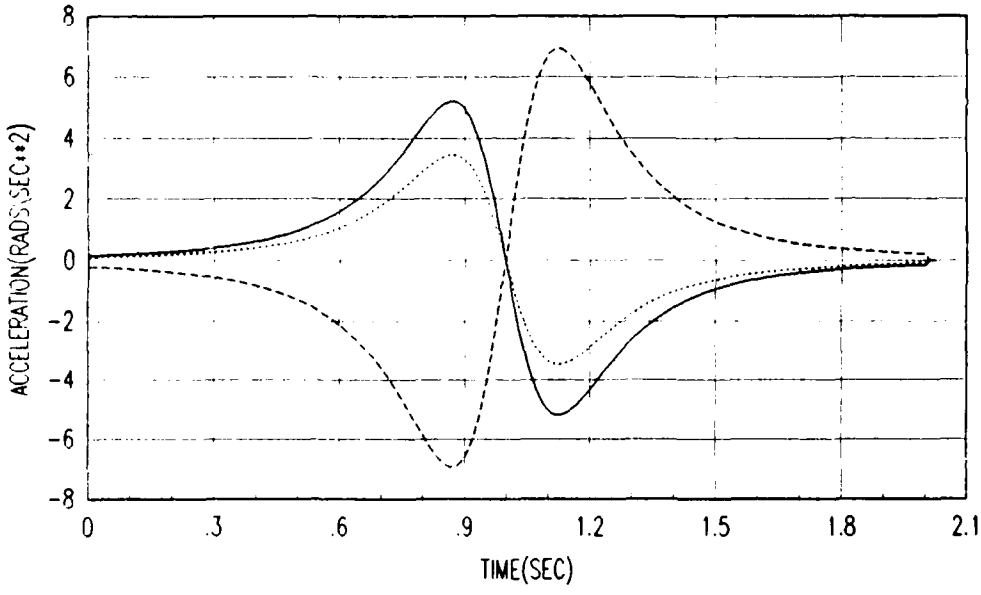

Figure 4.4. Trajectory One: Acceleration

—	Link 1
· · ·	Link 2
- - -	Link 3

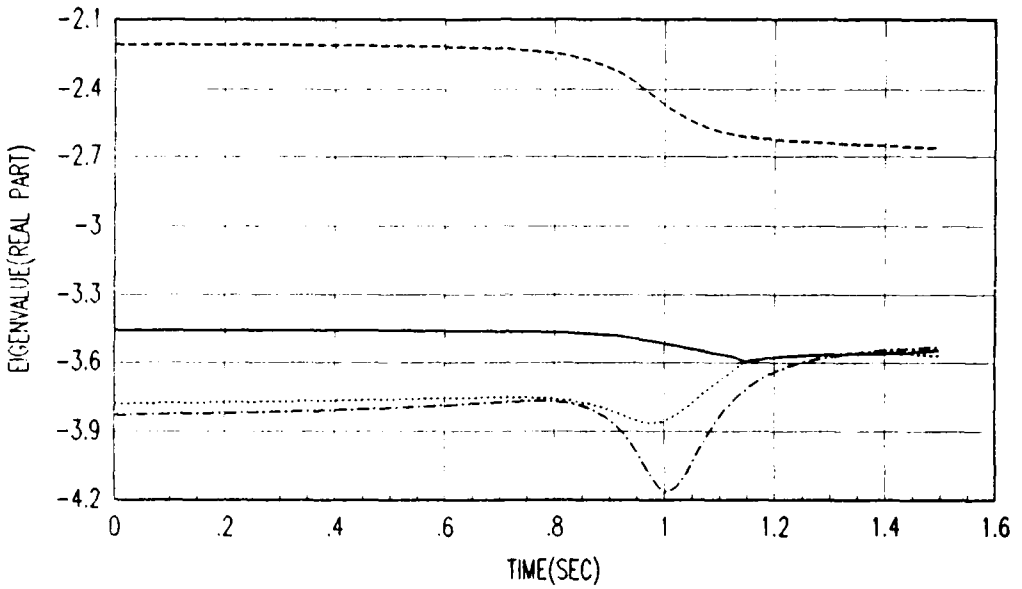

Figure 4.5. Eigenvalues of $F(a,t)$ Matrix for Trajectory One

—	0.0 Kg: Eigenvalue 1
· · ·	0.0 Kg: Eigenvalue 2
- - -	5.0 Kg: Eigenvalue 1
- · -	5.0 Kg: Eigenvalue 2

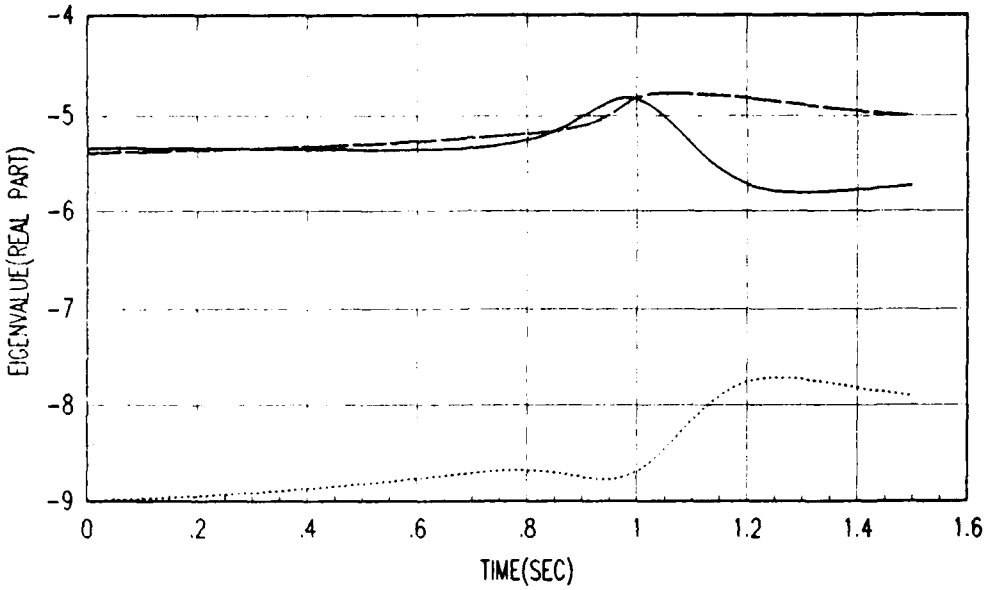

Figure 4.6. Eigenvalues of $F(a,t)$ Matrix for Trajectory One: Cont

—	0.0 Kg: Eigenvalue 3
· · ·	0.0 Kg: Eigenvalue 4
- - -	5.0 Kg: Eigenvalue 3
- · -	5.0 Kg: Eigenvalue 4

Note: The large dashed line indicates eigenvalues 3 and 4 for 5.0 Kg were a complex conjugate pair.

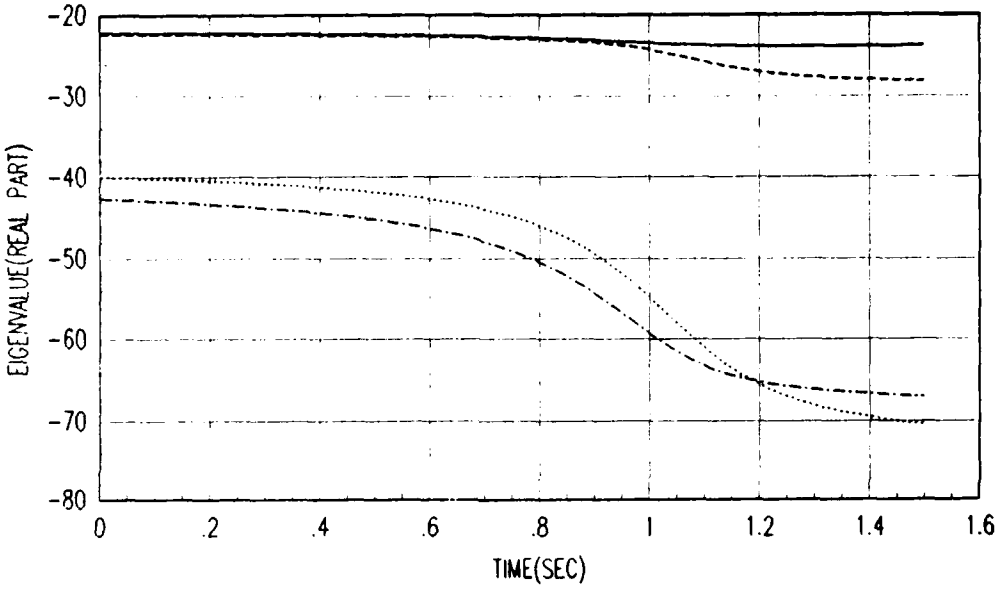

Figure 4.7. Eigenvalues of $F(a,t)$ Matrix for Trajectory One: Cont

—	0.0 Kg: Eigenvalue 5
· · ·	0.0 Kg: Eigenvalue 6
- - -	5.0 Kg: Eigenvalue 5
- · -	5.0 Kg: Eigenvalue 6

$$D(q,a)[\ddot{q} + K_v\dot{e} + K_p e] = 0 \tag{4.5}$$

The gains for the PD loop were selected to provide a critically damped response to a step input for the case of minimum inertia. This will insure that the robot will be overdamped if the inertia increases. The system poles were selected to be at $s = -10.0$. This value of frequency response was experimentally determined to provide a quick response with minimum vibration of the robot. Leahy and Saridis present in [LS88a] that PD gains for a given link can be selected by the following relationships:

$$K_v = (J_{min}2\zeta\omega_n - B_{eff})/nK_c \tag{4.6}$$

$$K_p = \omega_n^2 J_{min}/nK_c \tag{4.7}$$

where

- ζ = the link damping ratio.
- ω_n = the link natural frequency.
- J_{min} =the minimum effective inertia of the link.
- K_c = the stepping motor count to torque conversion number.
- B_{eff} = as previously defined.

For details on this development see [LS88a]. The values for K_p and K_v used in this research are tabulated in Figure 4.8.

4.4 Kalman Filter

The Kalman filter equations were developed in the previous chapter for the general case and are presented here to facilitate further discussion on the implementation of the filters in Figure 3.4:

	Position Gains	Velocity Gains
Link 1	250.0	72.0
Link 2	520.0	129.0
Link 3	95.6	24.8

Figure 4.8. PD Gains

$$\hat{x}(t_i^-) = \Phi(t_{i+1}, t_i)\hat{x}(t_i) \tag{4.8}$$

$$P(t_i^-) = \Phi(t_{i+1}, t_i)P(t_i^+)\Phi^T(t_{i+1}, t_i) + Q_d(t_i) \tag{4.9}$$

$$\hat{x}(t_i^+) = \hat{x}(t_i^-) + K(t_i)[z_i - H(t_i)\hat{x}(t_i^-)] \tag{4.10}$$

$$P(t_i^+) = P(t_i^-) - K(t_i)H(t_i)P(t_i^-) \tag{4.11}$$

$$K(t_i) = P(t_i^-)H^T[H(t_i)P(t_i^-)H^T(t_i) + R(t_i)]^{-1} \tag{4.12}$$

Some simplifying assumptions were utilized to facilitate the realization of the filters. As previously stated, the dynamics driving noises for each link were assumed pairwise independent and independent of the measurement noise. The first assumption was made to get the problem started. The second assumption is reasonable since the position encoders have very little to do with the torque applied to the robot. The measurement noise for each link was assumed to be independent of each other since there are different encoders for each link. The noises are also assumed have constant strength throughout the trajectory.

The value used for the covariance of the measurement noise, $V(t_i)$, was determined from the resolution of the encoders. The probability density function of the noise is uniform with zero mean and a standard deviation equal to 1/2 the square root of resolution of the encoder. The noise distribution was approximated by a Gaussian distribution with the same statistics.

The dynamics driving noise strength, Q, was adjusted using Trajectory One to provide the best performance of the MMAE. As a first attempt at modeling the uncertainty in the system, the noise was assumed to be not greater than 10% of the peak perturbation torque generated during the trajectory. The effects of this assumption will be addressed in the Recommendations section. Once Q was selected, the value was used for all the trajectories.

The $F(a,t)$ matrix does change during the trajectory; however, the sample period of the controller is short enough that $F(a,t)$ is assumed to be constant over the sample period. The same is true of $G(a,t)$. With $F(a,t)$ constant during the sample period and the sample period being short, the state transition matrix, $\Phi(t_i, t_{i-1})$ is approximated by:

$$\Phi(t_i, t_{i-1}) \approx I + F(t_i)\Delta t + 1/2F^2(t_i)\Delta t^2 \tag{4.13}$$

where Δt is the sample period. Similarly with Q held constant, $Q_d(t_i)$ is approximated by:

$$Q_d = \int_{t_{i-1}}^{t_i} \Phi(t_i, \xi)G(\xi)QG^T(\xi)\Phi^T(t_i, \xi)\, d\xi \approx G(t_i)QG^T(t_i)\Delta t \tag{4.14}$$

The robot was started from a known position and the error was assumed to be zero ($\hat{x}(t_0) = 0$) with probability 1 ($P(t_0) = 0$).

4.5 Parameter Discretization

The parameter a represents the external payload of the robot. The range of the payload was assumed to be continuous between 0.0 Kg. and 5.0 Kg. A procedure outlined by Maybeck [May88] discretizes the parameter space such that the different Kalman filters will be based on sufficiently different models that the MMAE can clearly separate the "good" model from the "bad". The technique

basically varies the true parameter away from the filter assumed reference point until the RMS value of the residuals, $[z_i - H(t_i)\hat{x}(t_i^-)]$ in the reference Kalman filter increases by 10% or more. An optimal technique for the discretization of a is beyond the scope of this investigation and is under investigation as a separate issue [She88]. There is a tradeoff in the number of Kalman filters needed to discretize the parameter space and the amount of on-line calculation required to process all of the filters in one sample period. Previous PUMA research suggested that three levels of discretization is reasonable. The parameters for the filters were set at 0.0, 2.5 and 5.0 Kg. This choice spans the payload possibilities of the PUMA-560 and keeps the computational time reasonable. If the upper limit of a were increased, the discretization would be different and the number of filters would also be increased.

4.6 Simulator

The Multiple Modeled-Based Control (MMBC) technique was validated and tested by digital simulation. The simulator used a fourth order Runge-Kutta routine with a 1 ms subinterval to solve Equation (3.1) [Wir87] and simulated the actual arm motion. The values for the friction coefficients were determined experimentally for the PUMA-560 by Leahy and Saridis [LS88a]. The dynamics driving noise was simulated as zero mean, white Gaussian noise of strength .01 and artificially injected into Equation (3.1) as shown in Figure 4.1. Measurement noise was simulated as zero mean, uniform noise with a variance of 1×10^{-9} and added to the position measurements. The means and covariance for the noises were selected for the same reasons as the initial values of Q and R.

Joint positions, velocities and accelerations which constituted a desired trajectory were precomputed and given to the feedforward element. The nominal torque out of the feedforward element plus the perturbation torque from the feedback element was applied to the simulator (see Figure 3.5) and the solution to Equation (3.1) was computed for the next sample period. The resulting noise cor-

rupted position was subtracted from the previous position measurement and the difference divided by the sample period to produce an approximation for the velocity (see Equation (4.15)). A single difference approximation was employed to be consistent with the previously developed model-based controller [Lea88b].

$$\dot{e} \approx \frac{position_{new} - position_{old}}{\Delta t} \tag{4.15}$$

The error states are formed and fed into the PD controller and the MMAE as measurements. The perturbation torque out of the PD controller was added to the nominal torque for the feedforward element and applied to Equation (3.1) for the next sample period. This continued until the trajectory was completed. See Figures 4.2, 4.3, 4.4 and Appendix C for plots of the trajectories used.

4.7 Software

A large part of this research effort was devoted to the generation of the FORTRAN code necessary to implement the Model-Based Multiple Model Adaptive Estimation (MMBC) control algorithm. Appendix A has abstracts from the FORTRAN routines used. A complete listing of the source code can be found in [Tel88].

The software effort began with the development of the routine to produce $F(a,t)$ of Equation (4.2). MACSYMA was used to reduce the equations of motion and to provide the FORTRAN code. The FORTRAN routine gave the values of $F(a,t)$ at each point along the trajectory. A simple MATRIXx [Int88] routine was written to produce the real part of the eigenvalues of $F(a,t)$.

In a parallel effort, the FORTRAN code for a single Kalman filter was developed. A program to assist in a covariance analysis on the filter was also written. Once the single Kalman filter was tested, the next step was to discretize the parameter space. To do this, the MMAE had to be integrated with the simulator.

A flow chart of the simulator is shown in Figure 4.7. The basic simulator was already available, but noises had to be incorporated into the system model and the integration of the MMAE routine had to be accomplished.

The first step in the simulator program was to initialize the Kalman filters and to load the precomputed $F(a,t)$. Then the trajectories for the three links were loaded and the program entered the main loop. The loop consisted of calculating the inertia matrix, $D(q,a)$ of Equation (2.1), and finding a payload estimate from the MMAE. With the payload estimate, the program calculated the nominal and perturbation torques and applied them to the simulated robot. The arm was moved forward in time until the next sample period by the solution to Equation (2.1). The desired position was subtracted from the simulated position of the arm at the end of that sample period to form the error states. The loop continued until the trajectories were completed.

Inside the MMAE subroutine, Equations (3.13) and (3.15) were solved to produce the payload estimate. The FORTRAN code for these subroutines was also developed as part of this thesis research.

4.8 MMAE Tuning

The procedure used for tuning the filters in the MMAE was outlined by Lashlee [Las87] and Netzer [Net85]. For this case study, the MMAE consisted of filters based on a being $0.0, 2.5$ and 5.0 Kg, as previously mentioned. The goal of the tuning effort was to adjust the noise strengths Q and R to extract the best performance of the individual Kalman filters in the MMAE. Q was held constant for each filter and R was varied to achieved the smallest residuals. Then R was held at that value and Q was allowed to change until the residuals were minimized. Q was kept small to keep the model difference apparent since as Q increases, the Kalman filter places more emphasis of the incoming measurement and less on the propagated state.

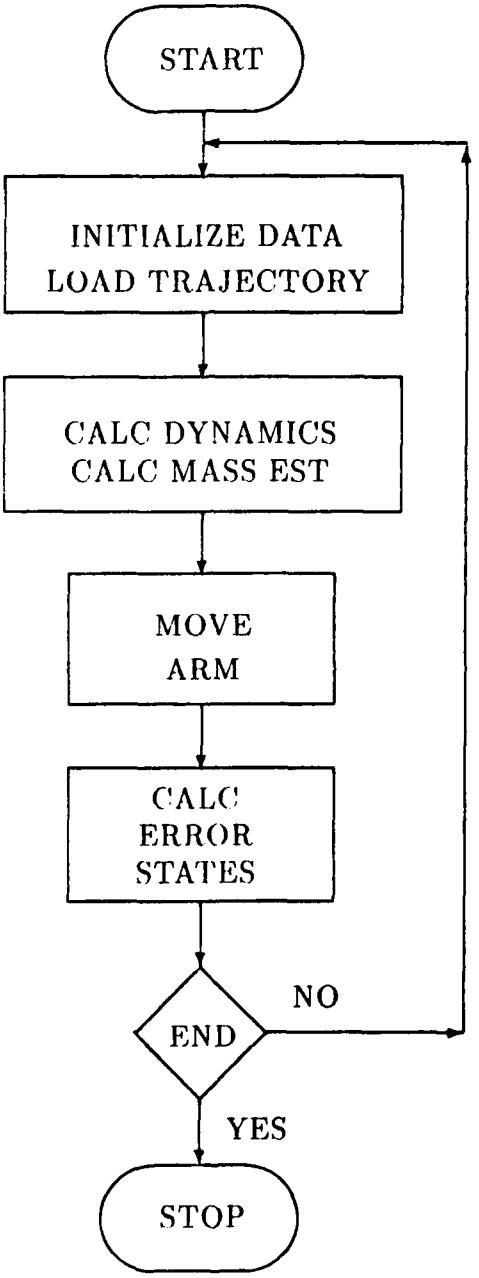

Figure 4.9. Simulator Flow Chart

The analysis of the residuals of each filter showed that, when the external load matched the controller's assumed load, the residuals based on 0 Kg. were smallest. In this situation the conditional probability calculations for $\hat{a}$ would indicate that 0 Kg. should be the estimate of the payload out of the standard MMAE. As previously indicated, when the feedforward element matches the true configuration of the robot, the result is a linear, second order system. This second-order system has very little dependence on the external payload. Hence, the MMAE cannot distinguish between the different filters.

The small changes in the feedback element caused by changes in the payload as well as other miss-modeled terms in Equation (2.1) appear as disturbances and are rejected by the PD controller. Since the $F(a,t)$ matrix was based on the closed loop system as shown in Figure 4.1, when the external payload matches the controller's value for payload, there is no significant difference between the system models in the MMAE. This was apparent by the weak dependence of $F(a,t)$ on the parameter a. Only as the external payload is allowed to change from the controller's value is the differences between the models apparent. Then the residuals in the MMAE reflect the difference in the true payload and the internally assumed value of the payload. The parameter in the MMAE is a *delta* mass instead of the actual value of the mass of the payload. This is not the previously publicized operation of the MMAE. The MMAE typically estimates the actual parameter, not the difference in the between the assumed value and the true value. The filters in the MMAE were re-tuned with the goal to minimize the residuals when the difference between the external payload and the controller's value for the payload matched the filters *delta* a_i.

The sign of the residuals is positive for the case when the external payload is larger than the controller's value for the payload. The positive sign indicates that, during a sample period the actual states of the system are propagated farther than the estimates of the states. In others words, actual errors grow larger than the

Δ Load	MMAE Output	Sign of Residuals	Curve Fit Est. (Figure 4.11)
.5	2.3	+	.87
1.5	2.8	+	1.4
2.5	3.5	+	2.7
3.5	3.9	+	3.5
4.5	4.4	+	4.4
−.5	2.3	−	.87
−1.5	2.8	−	1.4
−2.5	3.5	−	2.7
−3.5	3.9	−	3.5
−4.5	4.4	−	4.4

Figure 4.10. MMAE Performance

filters in the MMAE predict them to grow. The sign changes when the controller's value for the payload is larger than the external payload. In this situation, the estimates of the states are propagated farther than the actual states. The sign convention remained true during all three trajectories. The sign on the residuals was used to determine whether the MMAE estimate is to be added to or subtracted from to controller's present value. The output of the MMAE and the sign on the residuals is shown in Figure 4.10 for Trajectory One.

Figure 4.11 shows the same data graphically for the positive residuals case. As can be seen, the output of the MMAE is well approximated as linear except for the region where $\hat{a}_m$ is small. A least-squares curve fit to the data gave:

$$\hat{a}_f = 1.856\hat{a}_m - 3.793 \tag{4.16}$$

where

- $\hat{a}_f$ = the curve fit estimate of the *delta* mass of the payload
- $\hat{a}_m$ = the *delta* mass output of the MMAE

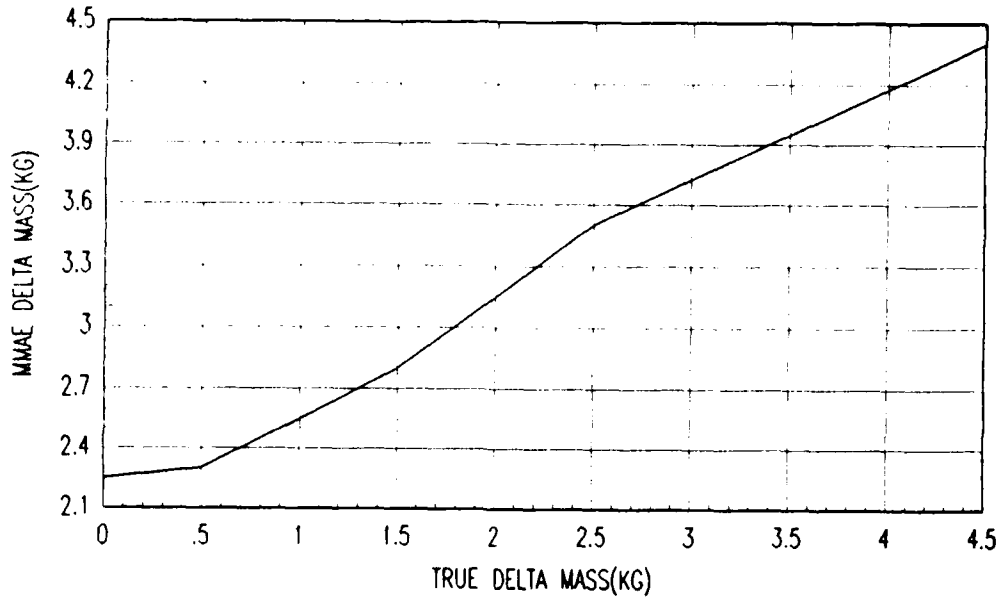

Figure 4.11. Estimated Load Verses True Load

The nonlinear region of Figure 4.11 is thought to be caused by the fact that the residuals in the MMAE become nearly equal as the *delta* mass approaches zero.

Research indicated that the sign on the residual from the link-two states in the 2.5 Kg. filter provided the best indication as to how the *delta* mass should be combined with the present value of a. The calculations using the sign from the residual and $\hat{a}_f$ can be included in the overall estimation algorithm, and the final output would be $\hat{a}$. Henceforth all reference to $\hat{a}$ will be the final output of the estimator.

The MMBC development thus far has been for the general manipulator. However, Equation (4.16) may be unique for each different class of robot and would re-evaluated when MMBC routine is implemented on different robot.

4.9 Controller Analysis

The purpose of the MMAE was to provide an estimate of the payload to the model-based controller that would reduce the tracking error. The control algorithm

consisting of the feedforward element, the feedback element and the MMAE was tested using the three trajectories discussed previously. The first case was the Trajectory Two. This trajectory held the robot stationary and should provide the least excitation to the estimator and increase the difficulty of the estimation task. Persistent excitation to the system would help to excite the parameter dependent modes thereby making the enhance the differences in the models of the MMAE. From a practicable standpoint, this external excitation could cause unnecessary vibration and reduce tracking accuracy and therefore was avoided.

The position of the arm was chosen as $0°, -135°$ and $135°$ for links one, two and three, respectively, and commanded to maintain that position. This position was selected as one that has proven very difficult for the model-based controller to handle on the PUMA-560 [Lea88c]. A payload of 4 Kg. was selected as a value that would be large enough that tracking would be difficult if the payload were not known, yet less than the upper limit of a used for the design of the MMAE.

The typical tracking errors of each link for a single run are shown in Figures 4.12, 4.13 and 4.14. Included in the plots are the tracking errors for the model-based controller with no payload information. This represents the case where the non-adaptive Single Model-based Controller (SMBC) is employed in place of the MMBC. As can be seen, the tracking error is greatly improved with the use of the MMAE. A reference plot is also included in the figures where the SMBC is artificially given the true payload. This is the best that the MMBC could hope to achieve.

The actual parameter estimate used by the controller is shown in Figure 4.15. The payload estimate has some high frequency oscillation, but is centered about the actual payload, 4 Kg. The $\hat{a}$ output of the MMAE reaches steady state very quickly. There is a small bias on $\hat{a}_f$ but the sign on the residual is dithering. This put the high frequency oscillation on $\hat{a}$. The oscillation is due to the models becoming nearly equal as the difference between the actual payload and

the controllers value of the payload approaches zero. The high frequency oscillation should not pose a problem for the PUMA since the band-pass of the robot is less than the frequency of the oscillation. If the oscillation can not be filtered out by the robot, the estimate out of the MMAE could be filtered before it is added to the controller.

Trajectory Two demonstrated the MMAE's ability to provide an estimate of the payload that will significantly reduce tracking errors with minimal movement by the robot. To test the MMAE under conditions where the nonlinear effects of the robot become significant, Trajectory One was used. The tracking errors for this trajectory are shown in Figures 4.16, 4.17 and 4.18. The results again show that the MMAE can quickly provide an estimate that will greatly reduce the tracking error. The peak and end tracking errors of the MMBC are very close to the artificially informed SMBC and much better than the uninformed SMBC. The estimated value of the payload used by the controller is shown in Figure 4.19.

The peak tracking errors for the MMBC on all three links is slightly higher that the artificially informed SMBC but much less that the uninformed SMBC. The final position errors for the MMBC are essentially equal to the artificially informed SMBC and again, much better than the uninformed SMBC.

The models in the Kalman filters in the MMAE did not include acceleration information. To improve tracking of trajectories with large jerk components, acceleration information needs to incorporated into the filters. The present system noise strength, Q, allows the filters to track profiles with mild jerk components. A third trajectory was used to test the capabilities of the MMBC with high jerk trajectories. Trajectory Three is shown in Appendix C. The tracking errors for Trajectory Three are shown in Figures C.1, C.2 and C.3. Again, tracking is greatly improved by the use of the MMAE over the uninformed SMBC but there is additional tracking performance to be gained. The payload estimate $\hat{a}$ and the value of the *delta* mass, $\hat{a}_f$ are shown in Figure C.4.

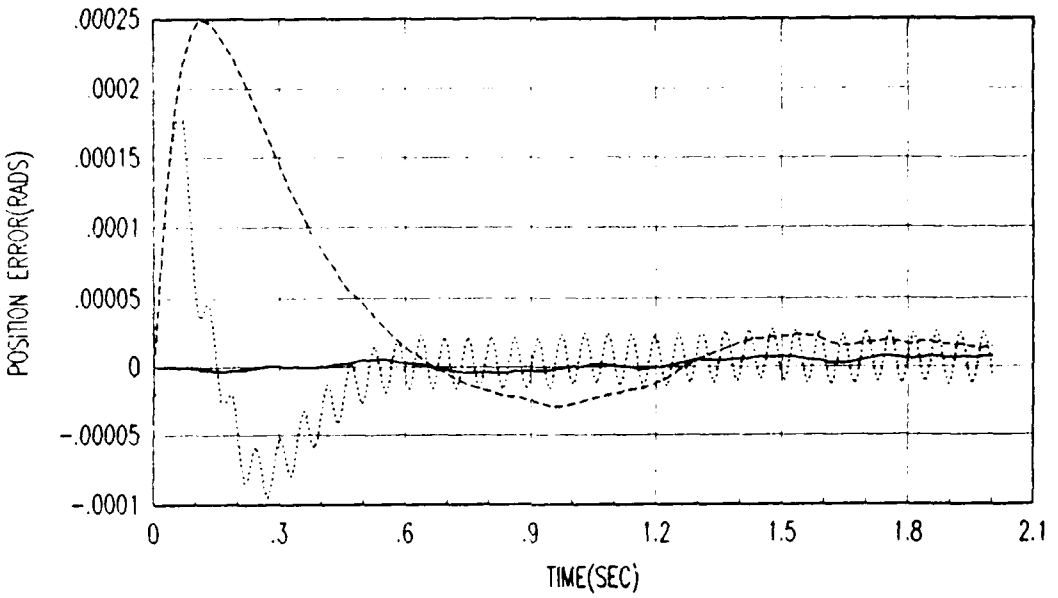

Figure 4.12. Tracking Error with Trajectory Two: Link 1

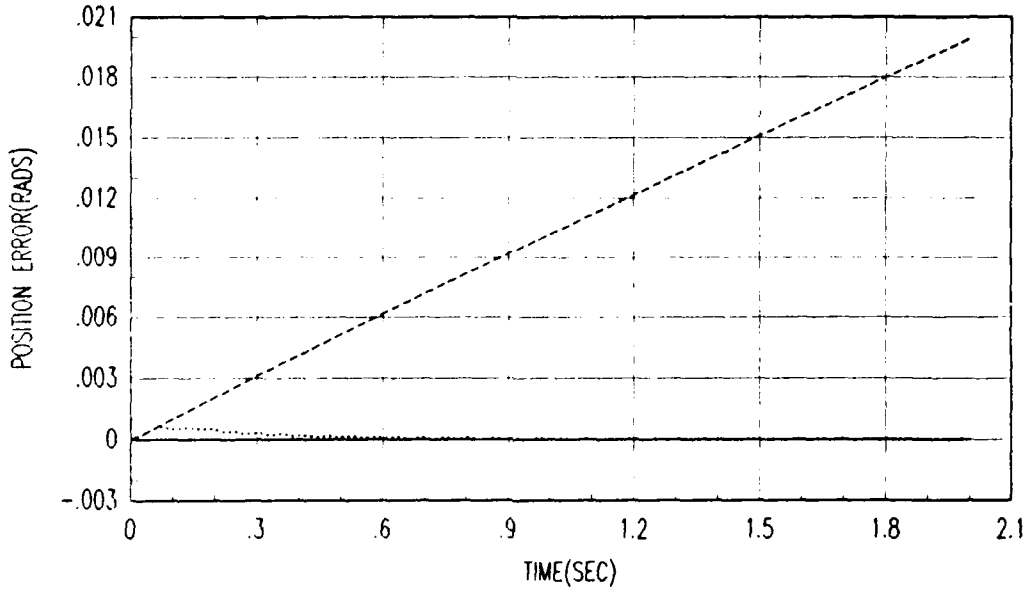

Figure 4.13. Tracking Error with Trajectory Two: Link 2

—	SMBC With Full Payload Information
· · ·	MMBC With No Payload Information
- - -	SMBC With No Payload Information

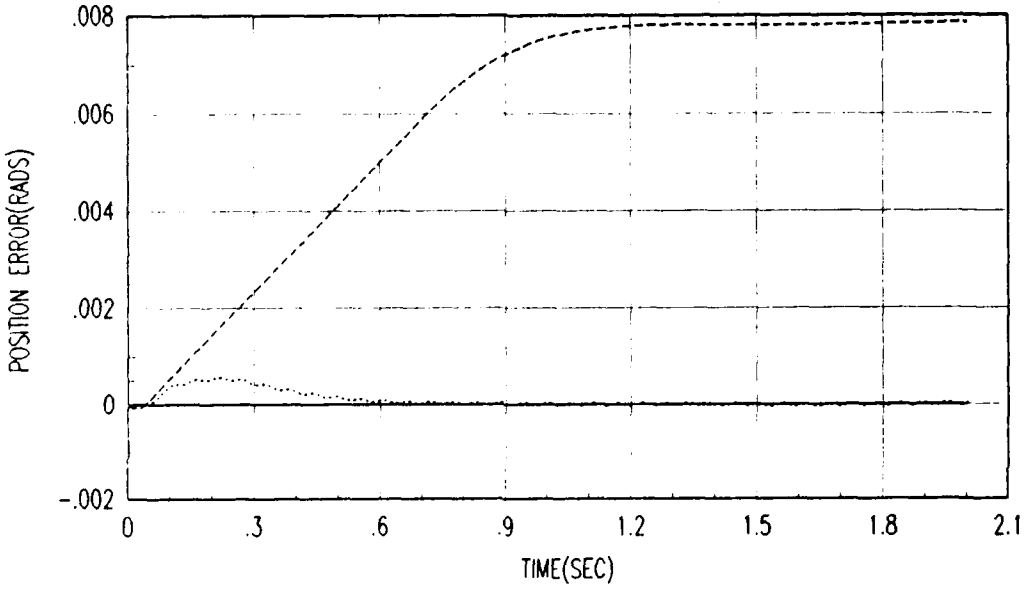

Figure 4.14. Tracking Error with Trajectory Two: Link 3

—	SMBC With Full Payload Information
· · ·	MMBC With No Payload Information
- - -	SMBC With No Payload Information

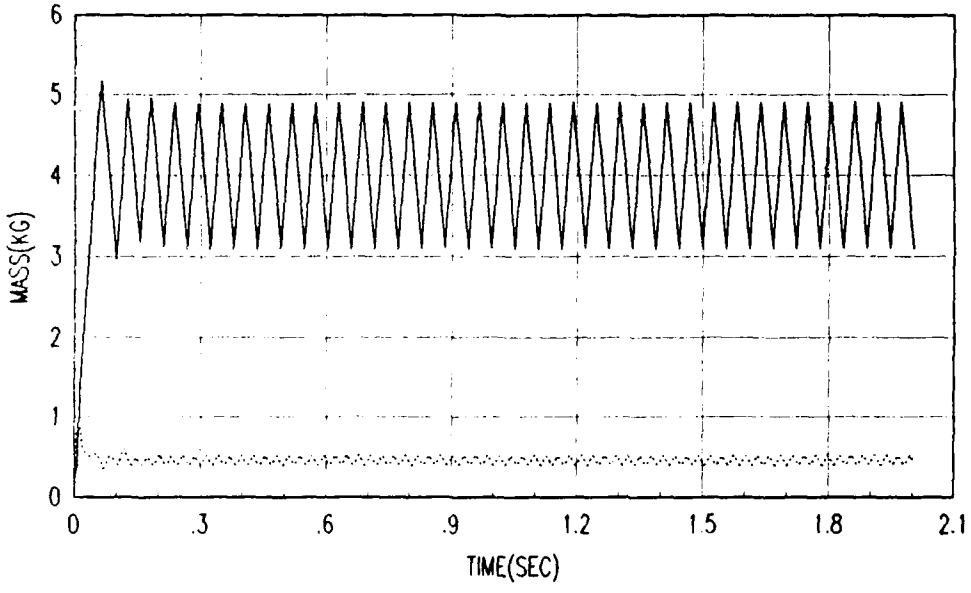

Figure 4.15. Payload Estimate for Trajectory Two

—	Payload Value Used In Feedforward Element: $\hat{a}$
· · ·	Payload Estimate Out Of MMAE: $\hat{a}_f$

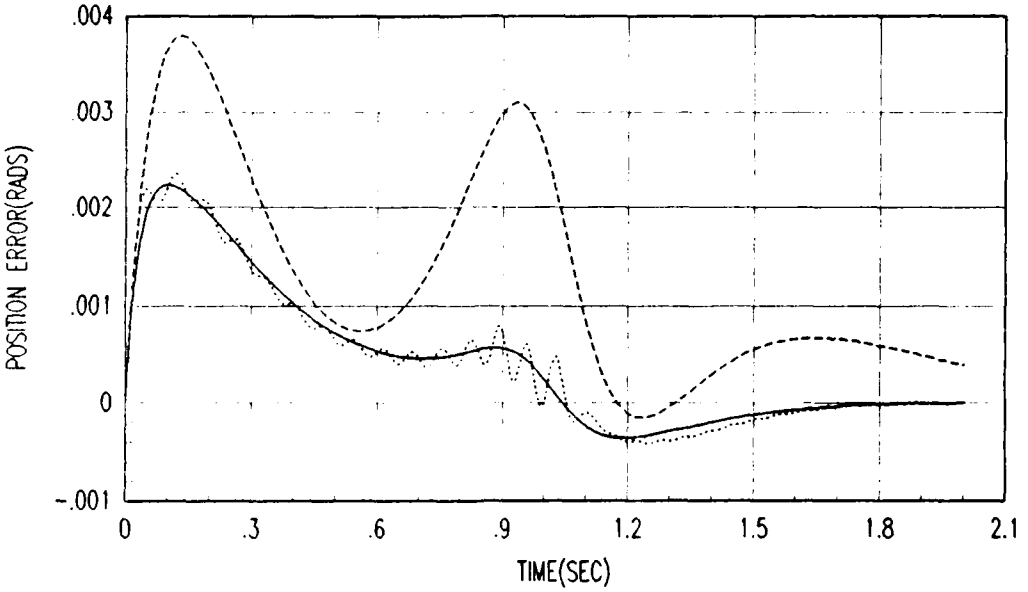

Figure 4.16. Tracking Error with Trajectory One: Link 1

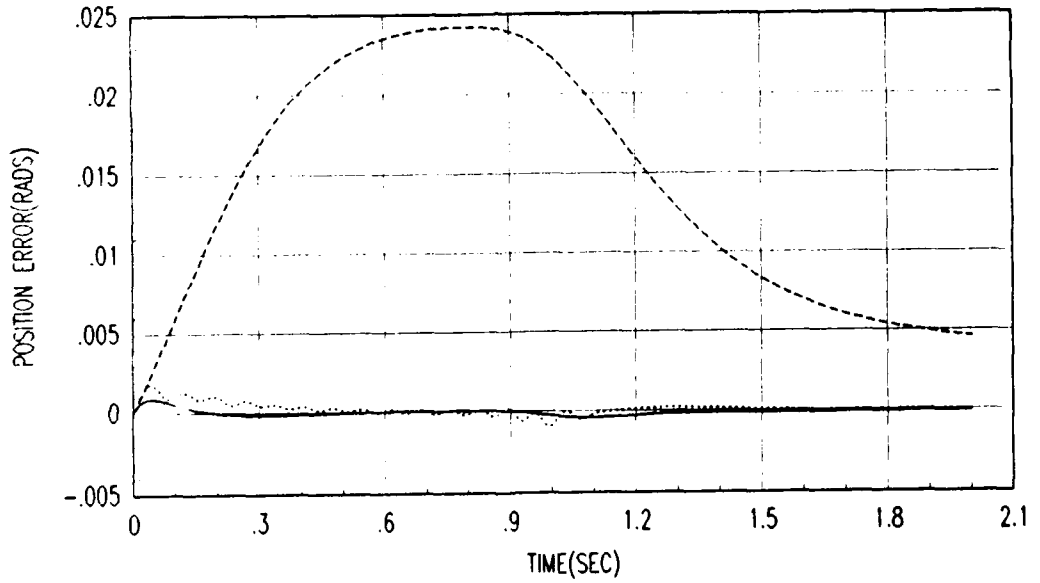

Figure 4.17. Tracking Error with Trajectory One: Link 2

—	SMBC With Full Payload Information
· · ·	MMBC With No Payload Information
- - -	SMBC With No Payload Information

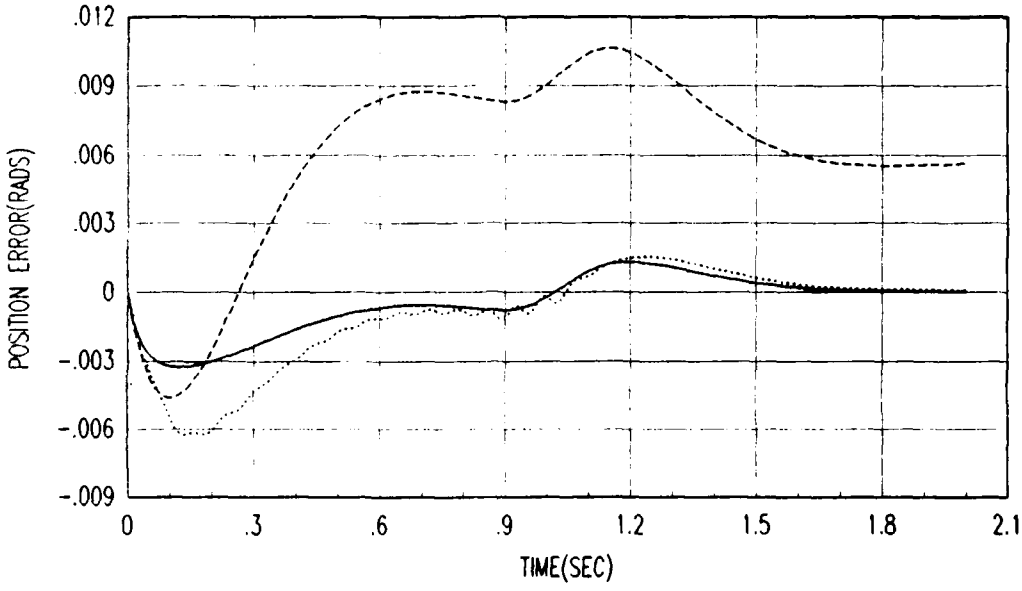

Figure 4.18. Tracking Error with Trajectory One: Link 3

—	SMBC With Full Payload Information
· · ·	MMBC With No Payload Information
- - -	SMBC With No Payload Information

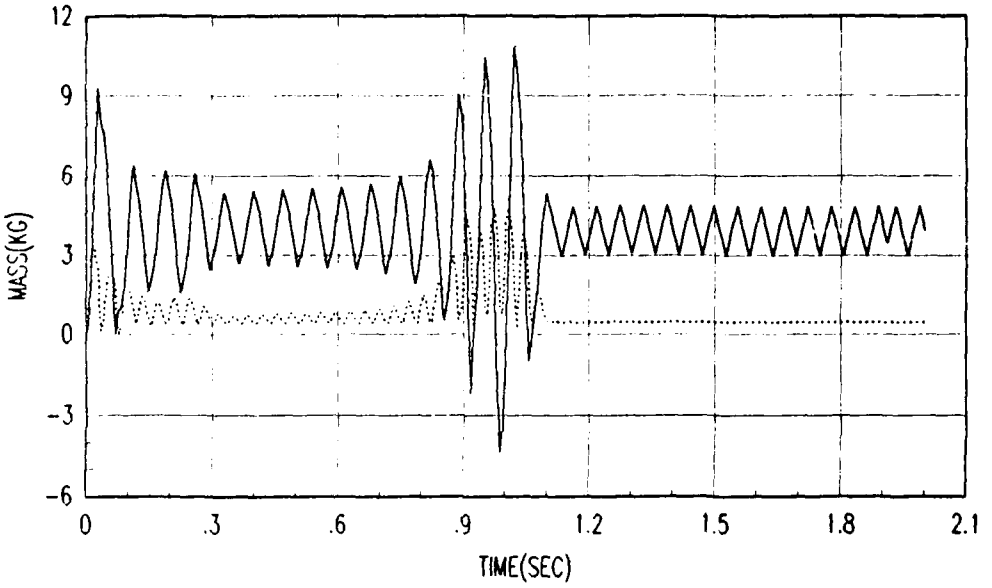

Figure 4.19. Payload Estimate for Trajectory One

—	Payload Value used In Feedforward Element: $\hat{a}$
· · ·	Payload Estimate Out Of MMAE: $\hat{a}_f$

The tracking errors are greatly improved with the MMBC over the uninformed SMBC. The payload estimate tracks well until the trajectory makes it first change in acceleration, after which the MMAE continued to track for a while and provides good estimates of the payload. Eventually the Kalman filters put out bad residuals and Figure C.4 shows the payload estimate tailing off near the end of the trajectory.

Two solutions to this problem arise. One would be to include acceleration as states of the system. The Kalman filter would carry these states around but not use them for control generation. The six-state Kalman filter would grow to nine states and thereby increasing the computational load. The other solution would be to have an executive program monitor the residuals and turn off the MMAE as long as the residuals stay small. If the residuals grow larger than some predetermined level, the MMAE could be turned on again and get a new estimate of the payload. The current value of the parameter would be used during the time the MMAE is off. This would have a minimal increase in computation time during the acquisition phase, but would reduce to total computational burden by only re-computing a new parameter estimate when necessary. Another advantage of this solution is that the oscillations in the payload value used by the controller would be reduced.

To more fully stress and better highlight the capabilities of the MMBC, a task was simulated where the robot had picked up an unknown payload and while in motion, inadvertently dropped it. The external payload was set to 4 Kg. at the start and was set to zero at 0.7 sec into the trajectory. The drop time was chosen to be after the initial acquisition period and before the peak velocity. Figures 4.20, 4.21 and 4.22 show the tracking errors when the external payload (4 Kg.) is artificially given to the SMBC only at the start of the trajectory and then the payload is dropped. Also shown is the case where the MMBC is used and the controller is initially told nothing about the external payload.

The MMAE must adapt to both the picking up and the dropping of the payload. As the figures show, the tracking errors with the MMBC are less and the payload estimate converges very quickly (see Figure 4.23). This capability has not been demonstrated with present adaptation schemes used for robot control [Ser88,SL87a,LS88b,MG86,DK85,AAH88,AAH86,dVW87,LC84,KG83,Goo85,Ser87] [DD79,SL87a,CHS87,AAH85].

In addition to examining tracking errors, a comparison of the total applied torque to the arm, Υ, for the MMBC and the true SMBC provides insight into how well the payload estimate is working. If the tracking errors are essentially the same but the difference in the applied torques is substantial, the algorithm using the least amount of torque is the preferred one. Figures 4.24, 4.25 and 4.26 show the total torque applied to the arm for the three links using the MMBC and the artificially informed SMBC. Trajectory One was used and the payload was 4 Kg. The figures show that the torques from the MMBC due oscillate. This is because of the oscillation in the payload value in the feedforward element. The peak for all the links is higher for the MMBC than the SMBC, but the area under the curves appears to be about the same. This is an indication that the amount of energy used by the MMBC and the SMBC is roughly equivalent and there is no additional cost for using the MMBC approach.

The tracking errors were greatly improved in all cases over the uninformed SMBC and were nearly equal to the artificially informed SMBC. The closed loop parameter estimation for this case study required an approach not previously taken. The MMAE was set up to provide an estimate of the difference between the assumed value of the payload and the true value. The Multiple Model-Based Control, MMBC, has shown good promise for robot control in simulation.

The following section presents the experimental results of the MMBC used to control the PUMA-560. The MMBC was used on the first three links of the PUMA-560 without any additional tuning of Q and R in the Kalman filters. Re-tuning

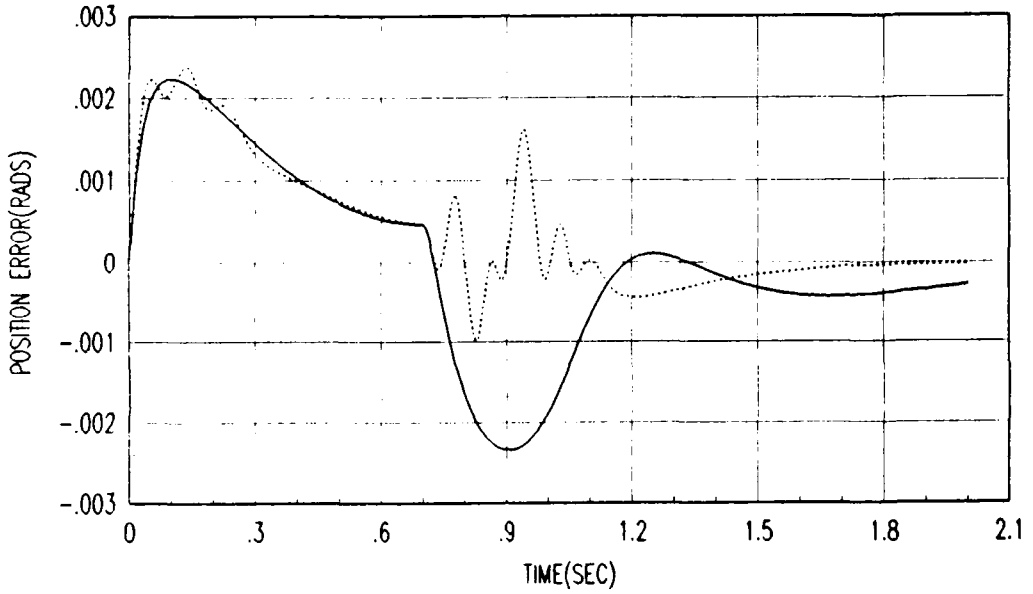

Figure 4.20. Tracking Error with Dropped Payload: Link 1

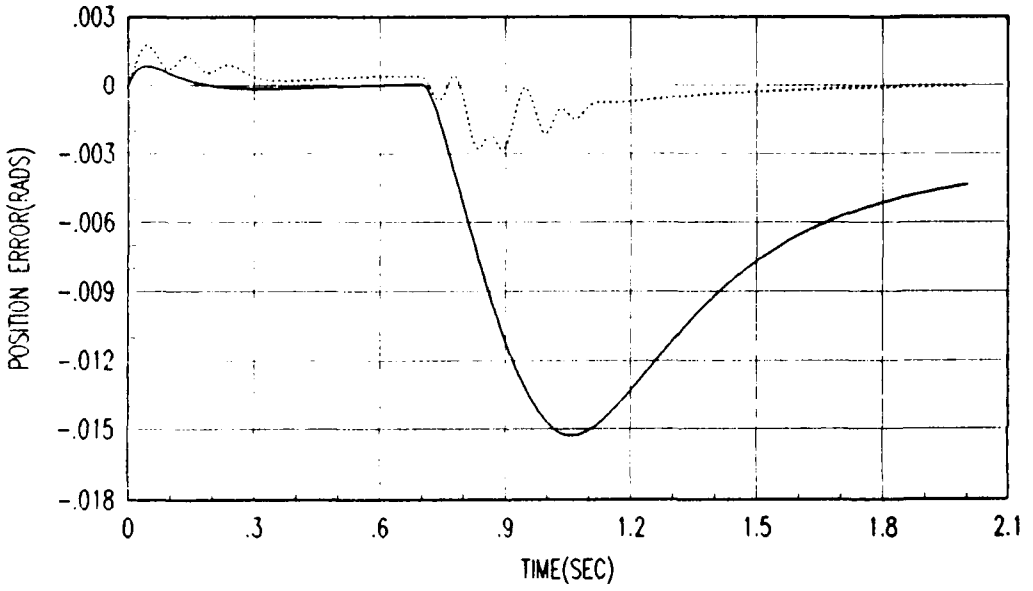

Figure 4.21. Tracking Error with Dropped Payload: Link 2

—	SMBC With Full Payload Information At Start Of Trajectory
...	MMBC With No A Priori Payload Information

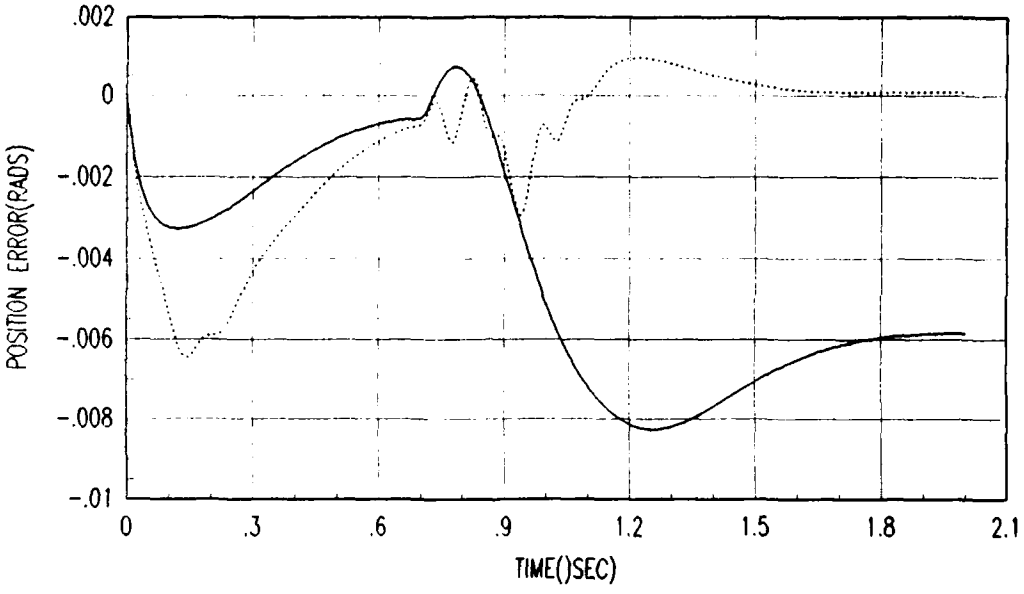

Figure 4.22. Tracking Error with Dropped Payload: Link 3

—	SMBC With Full Payload Information At Start Of Trajectory
· · ·	MMBC With No A Priori Payload Information

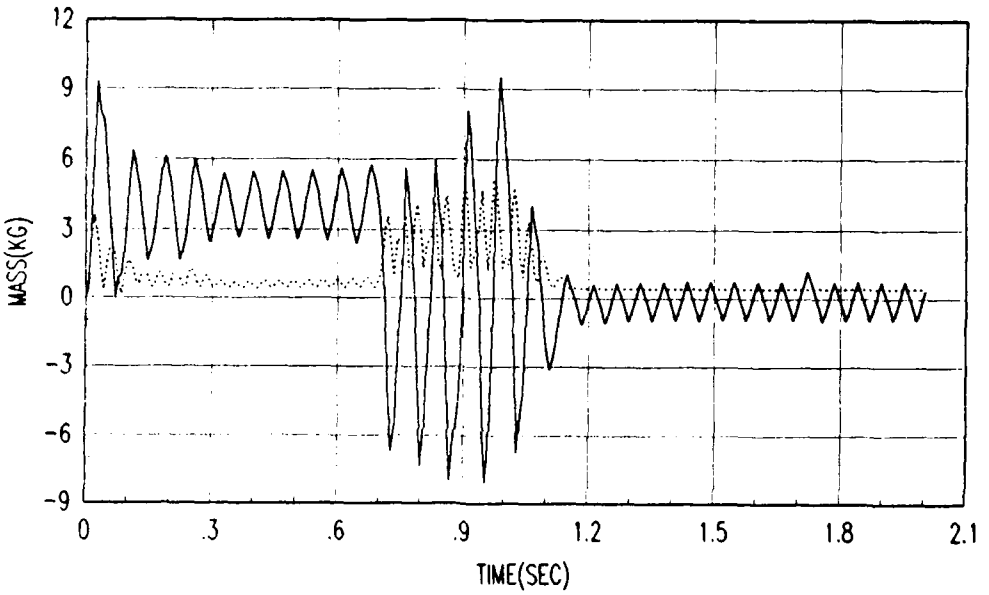

Figure 4.23. Payload Estimate with Dropped Payload

—	Payload Value In Feedforward Element: $\hat{a}$
· · ·	Payload Estimate Out Of MMAE: $\hat{a}_f$

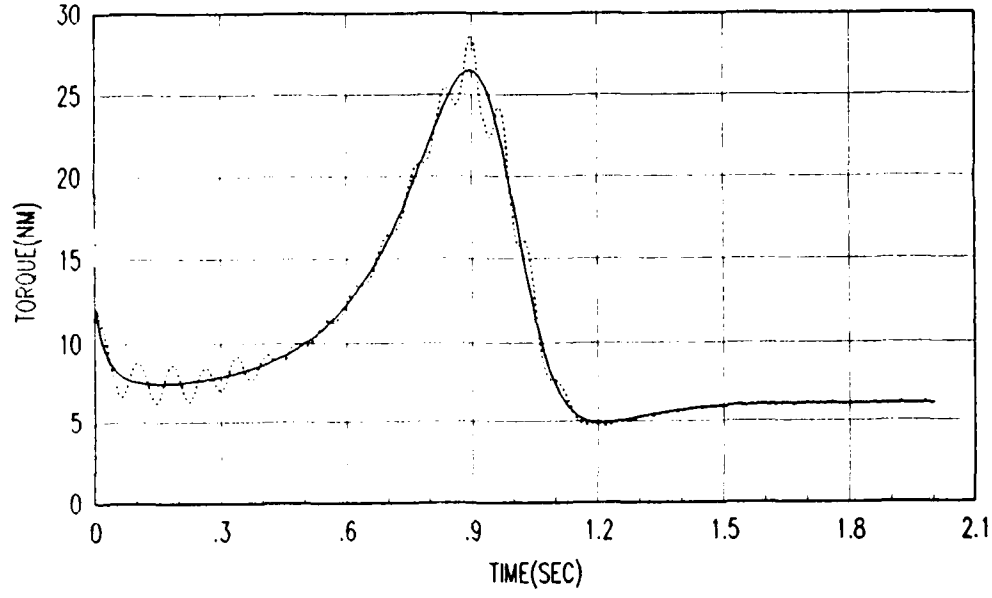

Figure 4.24. Total Applied Torque for Link One

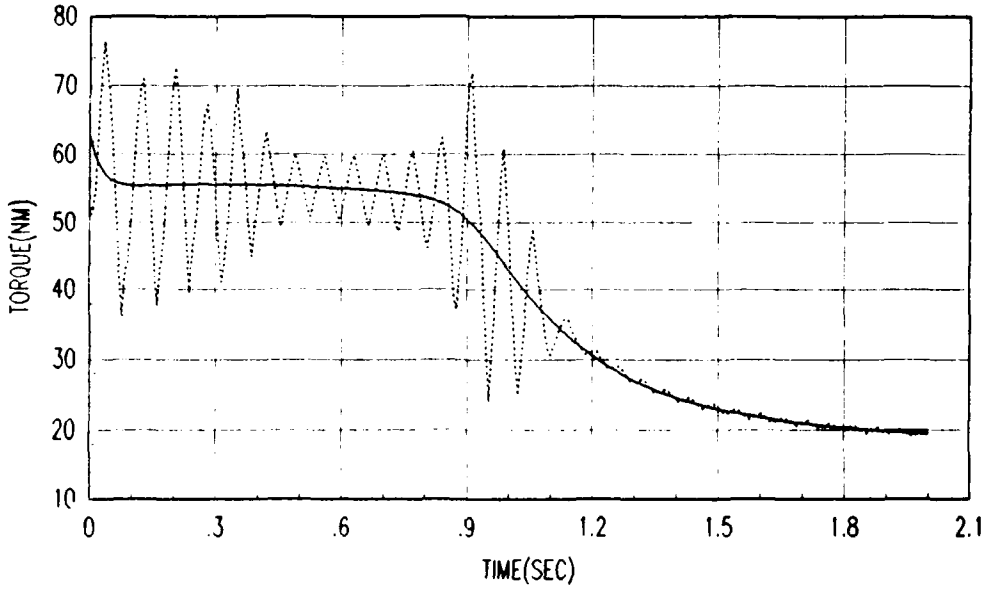

Figure 4.25. Total Applied Torque for Link Two

—	SMBC With Full Payload Information
. . .	MMBC With No Payload Information
- - -	SMBC With No Payload Information

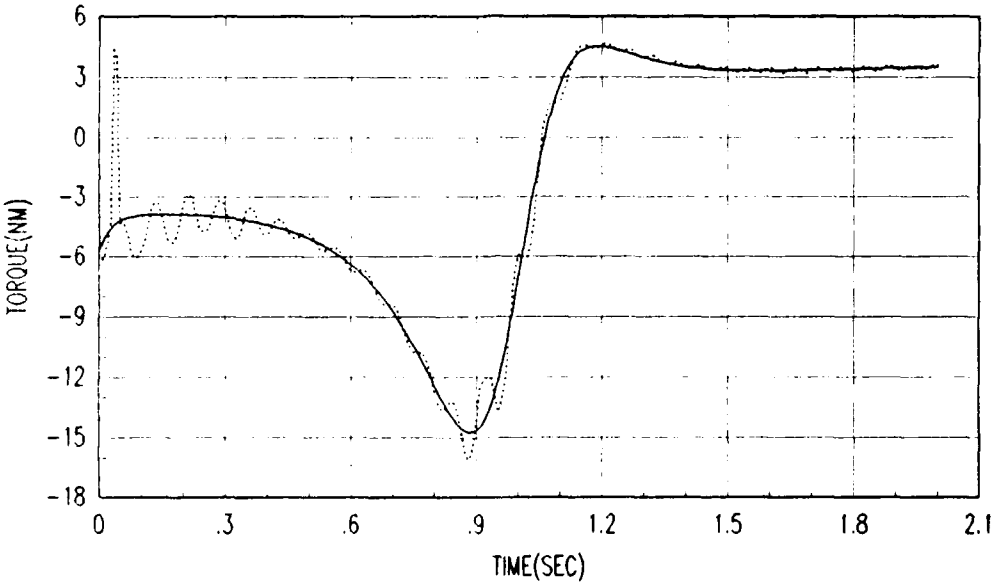

Figure 4.26. Total Applied Torque for Link Three

—	SMBC With Full Payload Information
. . .	MMBC With No Payload Information
- - -	SMBC With No Payload Information

was not required to demonstrate the potential of the MMBC technique. There is additional tracking performance to be gained by tuning the Kalman filters to match the robot.

4.10 Experimental Evaluation

The potential of any new control algorithms must by experimentally evaluated before any claims of success can be made. Leahy has developed a PUMA control environment [LS86] that the MMBC was tested under. The nature of the model-based control scheme naturally allows for a coarse parallel structure of the control algorithm. R3AGE was modified to exploit this characteristic and the feedforward and feedback calculations were put on different computers [Lea88b]. The feedforward processor was upgraded to a VAXstation III for this thesis effort.

4.10.1 Test Setup The MMBC scheme was implemented on the PUMA-560 available at AFIT. The first three links of the robot were used to demonstrate the control technique and the last three links were held stationary at 0°. The computations needed to control the robot were proportioned between two parallel processors, a PDP 11/73 and a VAXstation III. The coarse parallelism inherent in the model-based control structure permits the feedforward calculations to run at a different rate and on a different computer than the feedback calculations, without degrading tracking performance [Lea88b].

The 11/73, or Servo Processor, performed the PD loop calculations, read joint encoder values, and passed motor torques to the robot. It also established the basic timing for the overall control and communication at 7 ms. This was the timing signal available from the 11/73. The assembly language routines used in the Servo Processor to control the robot were a modified version of those originally developed at Rensselaer Polytechnic Institute by Leahy [LS86]. The modifications provided for the distribution of the nominal torque and estimation calculations to

the feedforward processor [Lea88b].

The communications between the Servo Processor and the feedforward processor was handled via a 16 bit, DRV11-J, parallel interconnect. The information passed over the buss consisted of 12 real numbers: six joint positions from the Servo Processor and six nominal motor torques to the Servo Processor. The computer system level calls for the communications were handled by a commercial software package called VAXLAB [Dig86]. The time for the 12 numbers to be transferred between the two computers was about 2.25 ms.

The time for the VAXstation to compute the nominal torques employing the payload estimate required about 19 ms. To maintain synchronization between the two computers, the timing for the nominal torque updates including the data transfer time must be a multiple of 7 ms. The VAXstation performed its calculations and waited for the Servo Processor to initiate data transfer. The Servo Processor performed the data transfer to the VAXstation at a 28 ms cycle time. The 28 : 7 split between dynamics compensation and servo loop update rates still produces good model-based control tracking results [Lea88b] when payload information is available.

4.10.2 Experimental Results The noise strengths in the MMAE filters from the simulation were used without any additional tuning. The results were very promising in spite of the lack of retuning the system noise strengths.

A payload of 2.5 Kg. was used for the experiment so as not to exceed the manufacturer's specification for maximum payload. This has been shown to cause severe performance degradation [Lea88b]. Trajectories One and Three were used and the tracking errors are shown in Figures 4.27, 4.28, 4.29 and Appendix E. The plots show the cases for the incorrect SMBC, the MMBC and the artificially informed SMBC.

As can be seen, the MMAE greatly improved the tracking performance of

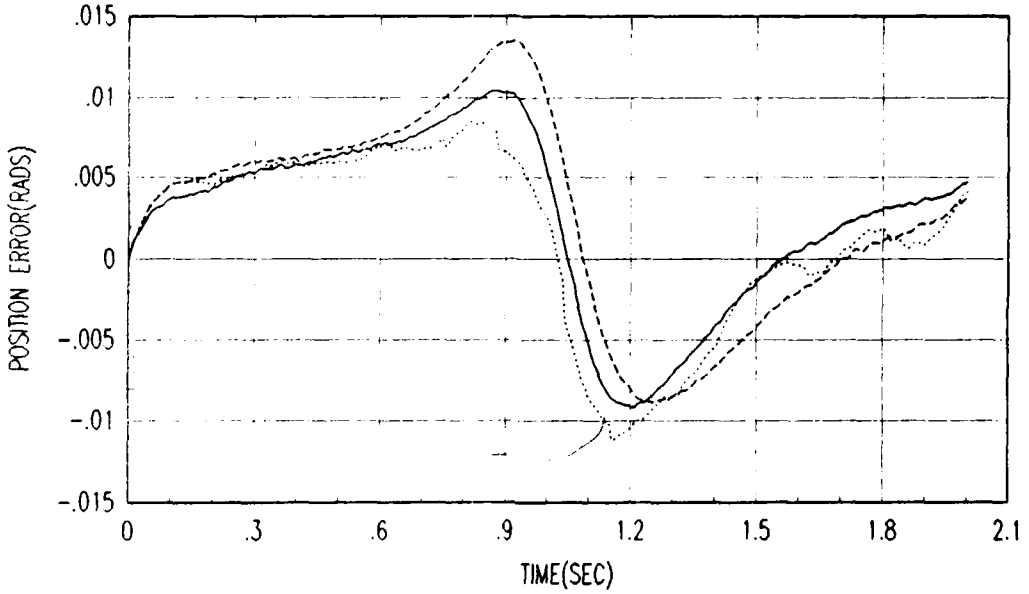

Figure 4.27. Experimental Tracking Error for Trajectory One: Link 1

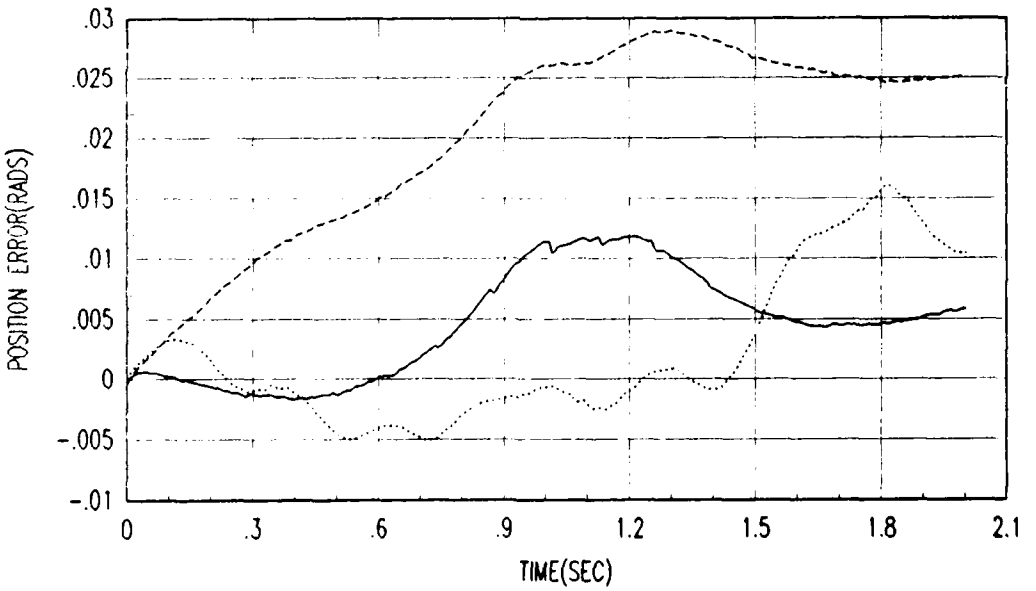

Figure 4.28. Experimental Tracking Error for Trajectory One: Link 2

—	SMBC With Full Payload Information
. . .	MMBC With No Payload Information
- - -	SMBC With No Payload Information

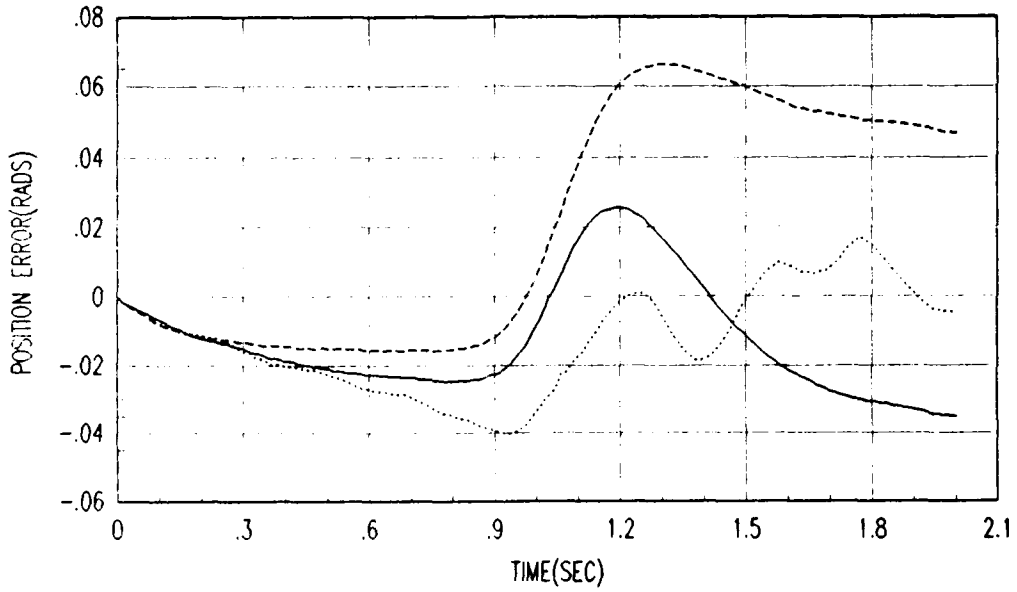

Figure 4.29. Experimental Tracking Error for Trajectory One: Link 3

—	SMBC With Full Payload Information
· · ·	MMBC With No Payload Information
- - -	SMBC With No Payload Information

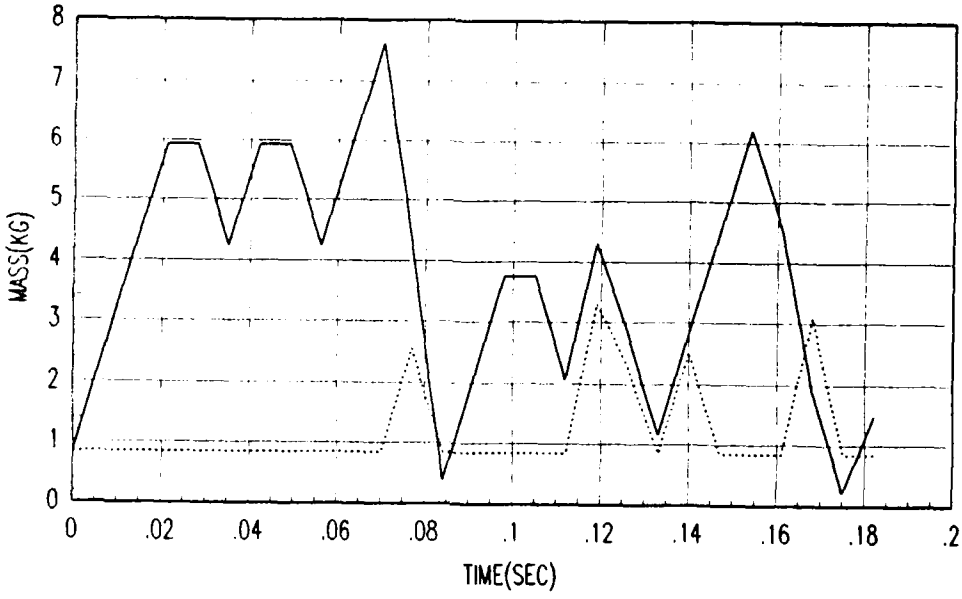

Figure 4.30. Experimental Payload Estimate

—	Payload Value Used In Feedforward Element: $\hat{a}$
· · ·	Payload Estimate Out Of MMAE: $\hat{a}_f$

the robot. Compared to the tracking performance in simulation, there is some additional accuracy to be gained by tuning the filters in the MMAE to match the robot. The noise strengths in the experiment were the same ones used in the simulation. The sign on the residuals is not always reflecting the correct situation and the estimated from the MMAE is incorrectly being combined with the controller's present value in the later part of the trajectory. Figure 4.30 indicates that the estimate from the MMAE ($\hat{a}_f$) looks the same during the trajectory but the value used by the controller ($\hat{a}$) decreases at the end of the trajectory. This indicates that the signs on the residuals are not correct. Tuning the filters should alleviate this tendency. Also the high frequency oscillation in the payload estimate seen in simulation is not present in the experimentally tracking errors.

Similar results can be seen for Trajectory Three in Appendix E. They show the same problem as in simulation. The payload value decreases during the later

part of the trajectory because of the high jerk profile. The same two solutions discussed above can be employed here.

A new robot control algorithm (MMBC) has been developed and an initial evaluation performed. The potential of using the MMBC has clearly been demonstrated. A complete evaluation of the new technique is beyond the scope of the thesis. The following section will discuss some of the remaining issues.

4.11 Discussion

The MMBC technique has been successfully demonstrated in simulation and experimentally evaluated on a PUMA-560. Some implementation issues were addressed as part of the initial evaluation. To more fully assess the potential of the MMBC, there are other issue that must be addressed.

The MMBC requires running three Kalman filters and executing the controller calculations in parallel at high speeds. This is not a trivial task. Very minimal FORTRAN code optimization has been applied to the present program. The computer used for the simulation (VAXstation III) runs at 3 MIPS and the MMAE calculation require approximately 18 to 19 ms. This could be reduced by more efficient FORTRAN coding.

A larger payoff could be realized by reducing the number of links in the models used in the Kalman filters from three to two. This would reduce the number of states from six to four. The results over Trajectory Two show that lack of motion of the links does not degrade the estimator's performance. The time to run the MMAE algorithm would be significantly reduced and the estimation routine could be run at the same sample rate as the feedback controller. Preliminary research into state reduction indicates that four states should be sufficient for the MMAE to estimate the payload.

For this investigation, the $F(a,t)$ matrix was precomputed. By reducing the number of states to four, the $F(a,t)$ matrix should be able to be computed during

the dynamic compensation interval. Having $F(a,t)$ computed on-line would make the MMBC algorithm more versatile.

The high frequency oscillation in the output of the parameter estimate could be addressed in several ways. The output could be put through a low-pass filter; however, this would reduce the convergence time of the MMAE. Another approach would be to monitor the residuals and turn the MMAE off once the controller has a good estimate of the payload. The residuals would continue to be monitored to determine when a large change had occurred. Then the MMAE would be turned on again until the controller had a new estimate of the payload. The performance of the MMBC algorithm on the PUMA-560 has shown that the tracking is not effected by the oscillation in the parameter estimate. This idea is in consonance with a number of researcher's philosophy of turning parameter identification on only periodically [May88].

As can be seen from Figure 4.10, the estimate is biased. When the bias is removed by the use of Equation (4.16), the result is a very good estimate of the *delta* payload (see Figure 4.11). The reason for the bias in the MMAE output is not exactly clear as yet. This is a nonlinear estimation problem, so a bias in the output is totally unexpected. One possible contribution could be the assumption that the dynamics driving noises are pairwise independent. Since the equations of motion are highly coupled, it is reasonable to assume that the noises would also be coupled. Since this was the first attempt at using Bayesian estimation for robot control, the noise models were kept simple in order to establish a baseline for further research.

When the output of the MMAE ($\hat{a}$) was used in the feedforward element, results have shown that tracking is enhanced and that $\hat{a}_f$ approaches zero in steady state. Figures 4.19 and C.4 indicate a bias of about 0.4 Kg. in $\hat{a}_f$. This remaining bias may be linked to the correlated noise problem just discussed. When the bias is removed from $\hat{a}_f$, Figure 4.31 shows the resulting $\hat{a}$ and the estimate of payload

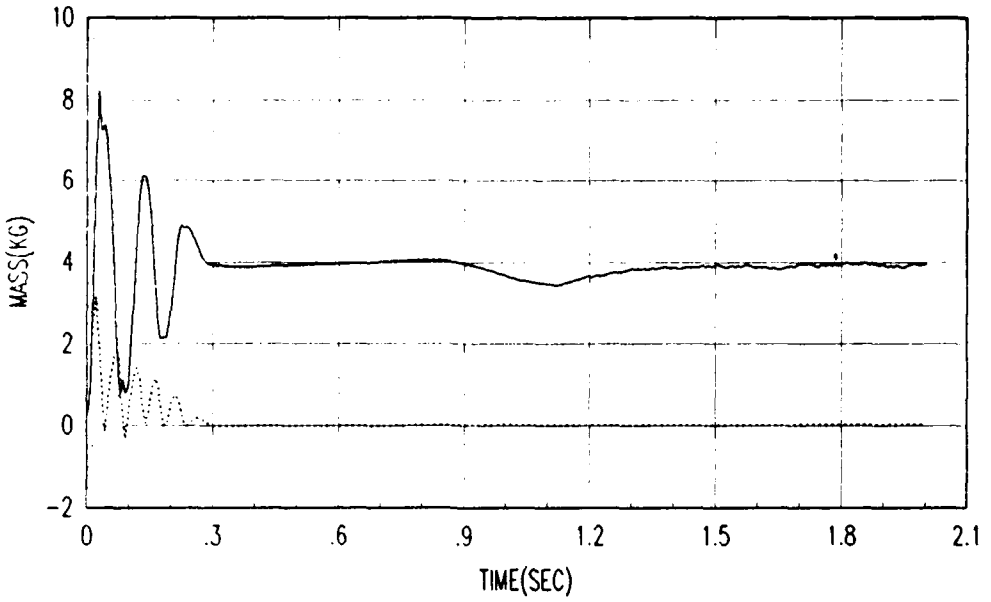

Figure 4.31. Payload Estimate With Bias Removed

—	Payload Value Used In Feedforward Element: $\hat{a}$
· · ·	Payload Estimate Out Of MMAE: $\hat{a}_f$

used in the feedforward element, $\hat{a}$. As can be seen, the tracking improvement is substantial. Figure 4.32 shows the tracking errors for all three links. The tracking performance is nearly the same as the case with the bias. Figure 4.33 shows the torques generated. The high frequency oscillation is removed for all but the acquisition phase of the scenario. When the bias was removed in the experiment, the tracking results were less impressive. Tuning the MMAE to the robot should improve this condition.

Preliminary evaluation of simulation results showed that, in the conditional probability calculations (see Equation (3.15)), the value for $p_k(t_{i-1})$ had to be kept at 1/3. Without $p_k(t_{i-1})$ held constant, the output of the MMAE became erratic. The effect of keeping $p_k(t_{i-1})$ constant is to reset the conditional probability calculation each sample period and to disregard all the information that went into making the previous parameter estimate. The controller's value of the payload

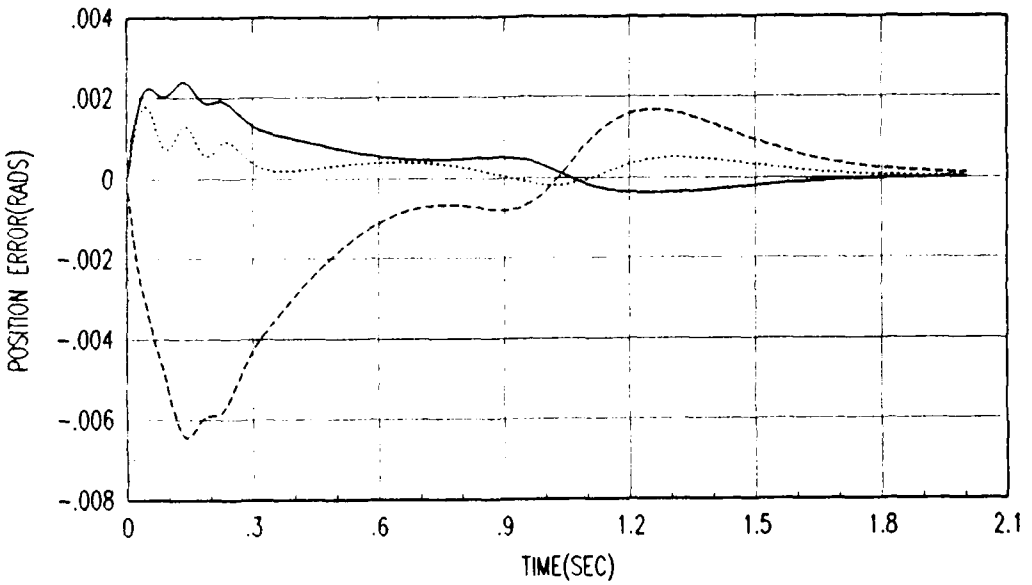

Figure 4.32. Tracking Error With Bias Removed

—	Link 1 Error
· · ·	Link 2 Error
- - -	Link 3 Error

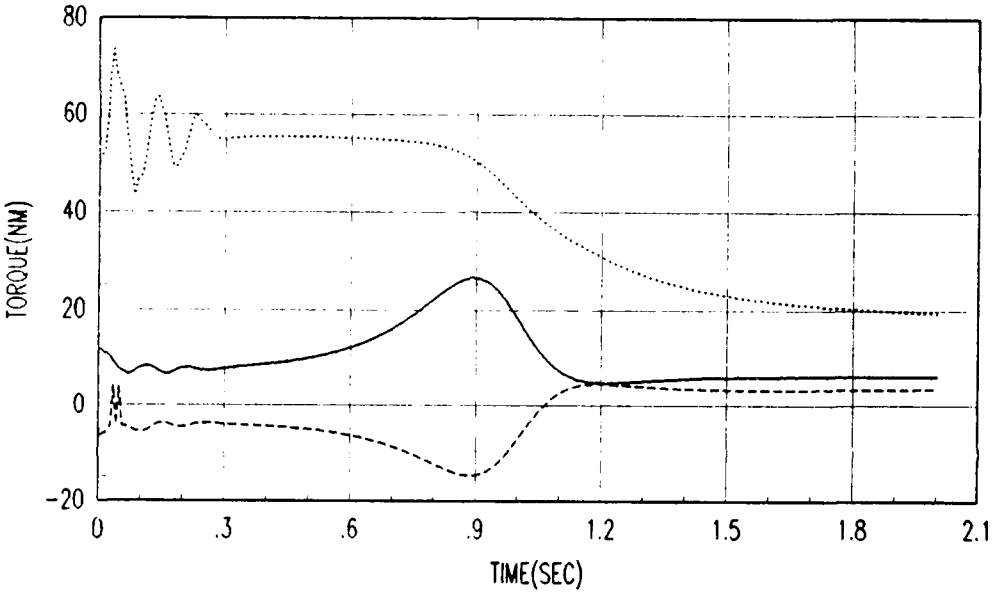

Figure 4.33. Total Generated Torque With Bias Removed

—	Link 1 Torque
. . .	Link 2 Torque
- - -	Link 3 Torque

converges very quickly to the externally applied value and tracking performance is improved when $p_k(t_{i-1}) = 1/3$.

4.12 Summary

The results of the simulated and experimental control of a PUMA-560 using a Multiple Model-Based Control (MMBC) technique have been presented. The tracking errors of the robot were greatly reduced when the MMAE was used to provide the model-based controller an estimate of the payload. In simulation the tracking performance of the controller with the MMAE was comparable to the SMBC with full payload information. The performance of the MMBC on the PUMA-560 seemed to validate the simulation results. The tracking errors were significantly reduced when compared to the uninformed SMBC and very close to the artificially informed SMBC.

Issues that warrant additional research have been highlighted. However, the results from this effort indicate that the MMAE can be used to provide a closed-loop estimate of the payload, that the MMAE can quickly adapt to changes in the payload, and that the model-based MMAE provides excellent control of the robot. In the final analysis, the MMBC has demonstrated the potential to provide a unique solution to a critical Air Force problem.

V. Conclusions and Recommendations

5.1 Conclusions

The research performed in support of this investigation met the stated objective and has proven very fruitful. The Multiple Model Adaptive Estimation (MMAE) technique has successfully been applied to the difficult problem of closed-loop payload estimation in model-based robot control. Combining the MMAE with a proven control technique has resulted in a new development that has the potential to be very useful in application where changing payloads can be expected. The estimate of the payload converges very quickly, which allows the controller to keep the peak tracking error to a minimum. The rate of convergence does not seem to depend on the trajectory used, and therefore persistent excitation appears not to be a problem for the Multiple Model-Based Controller (MMBC).

As part of the thesis effort an analysis of the perturbation plant, $F(a,t)$ was performed. The analysis showed that dependence of the payload on the perturbation plant is minimal when the feedforward element correctly models the payload condition of the robot. The investigation also showed that $F(a,t)$ can only be assumed constant for very slow trajectories. The use of the perturbation approach has been discussed in the literature and the minimal dependence of the payload has been assumed. Now that assumption has been demonstrated to be acceptable only under restrictive conditions.

A new *delta* parameter approach was taken to produce a parameter estimate because system model differences were apparent only when the controller's payload value, a, was mismatched from the true value. A new technique for estimating a based on the residuals from a bank of linear Kalman filters had to be developed. This new technique provides an estimate of the *delta* mass of the payload. The signs of the residuals indicates if the estimate of a is added to or subtracted from

the controller's present value of the payload. The result is a parameter estimate that converges very quickly. A review of current publications indicates that this approach is unique.

In simulation the MMBC tracked as well as the artificially informed model-based controller and required about the same amount of control energy. The peak and final tracking errors for the MMBC were much better than the uninformed model-based controller. A special trajectory was used to demonstrate the adaptive capability of the MMBC. The robot was commanded to move along a trajectory and the payload was dropped before the end of the run. The MMBC tracking results were much better than the non-adaptive model-based controller (SMBC) that had been given the true load initially but not told that the payload was dropped.

The simulation results were validated by implementing the MMBC on the PUMA-560. The noise strengths in the Kalman filters were not changed from the simulation values when the MMBC was run on the PUMA. Again, the tracking errors were greatly improved over the uninformed SMBC and comparable to the artificially informed SMBC.

5.2 Recommendations

The objective of this thesis was to develop and initially evaluate the potential of using a control scheme that employed the Multiple Model Adaptive Estimator (MMAE) to provide an estimate of the payload to a model-based controller. The reference used to measure the potential of the new algorithm was that the Multiple Model-Based Controller (MMBC) should track as well as the artificially informed model-based controller. The MMAE has successfully shown that it can provide payload estimates that greatly improve tracking of the robot. Some issues surfaced that were not part of this thesis effort but need further investigation.

One area that warrants additional effort would be the refinement of the

present algorithm. The FORTRAN code could be optimized and the number of states reduced. All this is in an effort to reduce the computational burden. Also there is current research at AFIT that will put the entire Kalman filter algorithm on a single integrated circuit in the 1989 time frame. This would greatly reduce computational time needed to run an MMAE scheme and may be a necessity if a is expanded to include additional parameters. It will also allow the $F(a,t)$ matrix to be computed on-line, thereby increasing the flexibility of the MMBC algorithm.

Another area should address the tuning of the Kalman filters in the MMAE. The noise levels that were used in the simulation were also employed in the experiment. The tracking of the PUMA could be improved by re-tuning the MMAE to match the robot. The system and measurement noise strengths used in this thesis were a first attempt to add noise to the model of a robot in a meaningful way.

Also the payload was assumed to be a point mass. The cases when this assumption cannot be made need to be experimentally investigated. The MMAE might have to be expanded to include parameters other than the mass of the payload. If a point mass assumption cannot be made, a could be expanded to include any of the m additional payload parameters required. This would necessitate additional Kalman filters in the MMAE.

The final area to consider would be to compare the MMAE to other techniques that have been proposed in the literature. The MMAE technique works, but it may not be the best for all robot estimation tasks. A head-to-head comparison of different techniques would help define the strong points of the MMBC approach. The *delta* estimation approach could be used in other areas as a new technique to estimate unknown parameters in a closed-loop situation. The scheme developed in this thesis provided a very quick and accurate estimate without the use of an excitation signal.

The remaining issues do not pose any real obstacles to the successful application of the Multiple Model-Base Control technique to the robot control problem.

If applied to telepresence activities, the robot employing a MMBC could operate without *a prior* payload information. The unknown payload could be estimated very quickly and be used to improve the tracking of the robot. The same estimate could be telemetered back to the remote operator to provide him/her with sensory feedback as to how heavy the load is, thereby improving the overall performance of the telepresence loop.

Appendix A. *Macros and Abstracts of FORTRAN Source Code*

This appendix provides an example for the macro used in MACSYMA. Also included are the abstracts of the FORTRAN code used in this thesis.

```
@  This macro will find the equations of motion for the
@  first 3 links of a PUMA-560.  It is based on
@  Tarn's paper


@     set up the environment


        writefile("fullrun_8.log");

        fpprec : 5;
        fpprintprec : 3;

        infeval : true;
        float2bf : true;


@     dimension the needed arrays


        array(D,3,3)$
        array(D1,3,3)$
        array(D2,3,3)$
        array(D3,3,3)$


@     initialize constants

          grav : 9.8062$

      load : 0.0;

      ai1 : .7766;
      ai2 : 2.3616;
      ai3 : .5827;
```

```
        m1 : 12.96$
        m2 : 22.37$

        s23  : sin( q2 + q3 )$
        c23  : cos( q2 + q3 )$
        c22  : cos( q2 + q2 )$

        c1 : cos(q1)$
        c2 : cos(q2)$
        c3 : cos(q3)$
        s1 : sin(q1)$
        s2 : sin(q2)$
        s3 : sin(q3)$

        a2 : 0.4318$
        a3 : -.0191$
        f3 : .1505$
        xb1 : 0.0$
        yb1 : .3088$
        zb1 : .0389$
        xb2 : -.3289$
        yb2 : .0050$
        zb2 : .2038$
        k1xs : .1816$
        k1ys : .0152$
        k1zs : .1811$
        k2xs : .0596$
        k2ys : .1930$
        k2zs : .1514$
        k3zs : .0021$
      mti : 1.0 / (load + 6.97)$

@     start calculation

        xb3 : 6.97 * .0136 * mti$
        yb3 : 6.97 * .0092 * mti$
        zb3 : (6.97 * .1522 + .48932 * load) * mti$
        k3xs : (598585344 * load + 262504960)/(2500001792 * load
               +  17424973824)$
        k3ys : k3xs$

        m3 : load + 6.97;

@     load inertia matrix

         D[1,1] : ail + m1 * k1ys + m2 * k2xs * s2^2 +
                  m2 * k2ys * c2^2 + m2 * a2^2 * c2^2
                  + 2  * m2 * a2 * xb2 * c2^2 + m3 * k3xs
```

```
                * s23^2 + m3 * k3zs * c23^2 + m3 * f3^2
                + m3 * a2^2 * c2^2 + m3 * a3^2 * c23^2
                + 2  * m3 * a2 * a3 * c2 * c23 + 2  * m3 *
                xb3 * a2 * c2 * c23 + 2  * m3 * xb3 * a3
                * c23^2 + 2  * m3 * yb3 * f3 + 2  * m3 *
                zb3 * a3 * c23 * s23 + 2  * m3* zb3 * a2 *
                c2 * s23$

        D[2,2] : m2 * ( k2zs + a2^2 + 2* a2 * xb2 ) +
                2  * m3 * a2 * ( a3 + xb3 ) * c3 +
                2  * m3* a2 * zb3 * s3 + m3 * ( k3ys +
                + a2^2 + a3^2 + 2  * a3 * xb3 )$

        D[3,3] : m3 * ( k3ys + a3^2 + 2  * a3 * xb3 )$

        D[1,2] : m2 * a2 * zb2 * s2 + m3 *( f3 * xb3 +
                a3 * yb3 + a3 * f3 )* s23 + m3 *( a2 *
                yb3 + a2 * f3 ) * s2 - m3 * f3 * zb3
                * c23$

        D[1,3] : m3 * ( xb3 * f3 + a3 * yb3 + a3 * f3 )
                * s23 - m3 * f3 * zb3 * c23$

        D[2,3] : m3 * ( a2 * xb3 + a2 * a3 ) * c3 + m3 *
                a2 * zb3 * s3 + m3 * ( 2  * a3 * xb3 +
                a3^2 + k3ys )$

        D[2,1] : D[1,2]$

        D[3,1] : D[1,3]$

        D[3,2] : D[2,3]$

@     load Corials/centrifugal matries

        D1[1,1] : 0 $

        D1[1,2] : m2 * ( k2xs - k2ys - a2^2 - 2  *
                a2 * xb2 ) * c2 * s2 - m2 * a2 * yb2
                * c22 + m3 * ( k3xs - k3zs ) * c2 *
                s2 + m3 *( k3xs - k3zs ) * c3 * s3
                + 2  * m3 *( k3zs - k3xs )* s2 * s3 *
                s23 - 2  * m3 *a2 * xb3 * c2 * s23 + 4  * m3
                * a3 * xb3 * s2 * s3 * s23 + m3 * a2 *
                xb3 * s3 - 2  * m3 * a3 *
                xb3 * c2 * s2 - 2  * m3 * a3 * xb3 * c3
                * s3 + m3 * a2* zb3 * c2 * c23 - m3 * a2*
                zb3 * s2 * s23 + 2  *m3*a3*zb3*c23^2 - m3
                * a3 * zb3 + m3 * a2 * a3 * s3 - 2  * m3
                * a2 * a3 * c2 * s23 - m3 * a2^2 * c2 *
```

```
          s2 + 2  * m3* a3^2 * s2 * s3 * s23 - m3
          * a3^2 * c2 * s2 - m3 * a3^2 * c3 * s3$

D1[1,3] : m3 *( k3xs - k3zs ) * c2 * s2 +
          m3 *(k3xs - k3zs )* c3 * s3 + 2  *
          m3 *( k3zs - k3xs ) *s2 * s3 * s23
          + 4* m3* a3* xb3* s2 * s3 * s23 - 2  * m3
          * a3 * xb3 * c2 * s2 - 2  * m3 * a3 *
          xb3 * c3 * s3 - m3 * a2 * xb3 * c2 *
          s23 + 2  * m3 * a3 * zb3 * c23^2 + m3 * a2
           * zb3 * c2 * c23 - m3 * a3 * zb3 + 2  *
          m3 * a3^2 * s2 * s3 * s23 - m3 * a2 *
          a3 * c2 *  s23 - m3 * a3^2 * c2 * s2 -
          m3 * a3^2 * c3 * s3$

D1[2,1] : D1[1,2]$

D1[2,2] : m2 * a2 * zb2 * c2 + m3 * f3 * zb3
          * s23 + m3 *( f3 * xb3 + a3 * yb3 + a3
          * f3 )$

D1[2,3] : m3 * f3 * zb3 * s23 + m3 *( f3 * xb3
          + a3 * yb3 + a3 * f3 ) * c23$

D1[3,3] : m3 * f3 * zb3 * s23 + m3 *( f3 * xb3
          + a3 * yb3 + a3 * f3 ) * c23$

D1[3,1] : D1[1,3]$

D1[3,2] : D1[2,3]$

D2[1,1] : - D1[1,2]$

D2[1,2] : 0 $

D2[2,1] : D2[1,2]$

D2[2,2] : 0 $

D2[1,3] : 0 $

D2[3,1] : 0 $

D2[2,3] : - m3 *( a2 * xb3 + a2 * a3 ) * s3 +
          m3 * a2 * zb3 * c3$

D2[3,2] : D2[2,3]$

D2[3,3] : m3 *( -a2 * xb3 - a2 * a3 ) * s3 +
          m3 * a2 * zb3 * c3$

D3[1,1] : - D1[1,3]$
```

```
D3[1,2] : - D2[1,3]$

D3[1,3] : 0 $

D3[2,1] : D3[1,2]$

D3[2,2] : - D2[2,3]$

D3[2,3] : 0 $

D3[3,1] : D3[1,3]$
D3[3,2] : D3[2,3]$

D3[3,3] : 0 $
```

@ form the gravity vector

```
G1 : 0 $

G2 : -m2 * grav *( xb2 + a2 ) * c2 + m2 * grav *
        yb2 * s2 - m3 * grav *( xb3 + a3 ) * c23 -
        m3 * grav * zb3 * s23 - m3 * grav * a2 * c2$

G3 : -m3 * grav *( xb3 + a3 ) * c23 - m3 * grav *
        zb3 * s23$
```

@ form the inertia matrix and non-linear h

```
d:matrix([d[1,1],d[1,2],d[1,3]],[d[2,1],d[2,2],d[2,3]],
   [d[3,1],d[3,2],d[3,3]])$

h1:matrax([d1[1,1],d1[1,2],d1[1,3]],[d1[2,1],
    d1[2,2],d1[2,3]],[d1[3,1],d1[3,2],d1[3,3]])$
```

@ reduce the equations

```
h1 : bfloat(h1)$
h1 : ev(h1)$
h1 : expand(h1)$
h1 : xthru(h1);

h2:matrax([d2[1,1],d2[1,2],d2[1,3]],[d2[2,1],
    d2[2,2],d2[2,3]],[d2[3,1],d2[3,2],d2[3,3]])$

h2 : bfloat(h2)$
```

```
h2 : ev(h2)$
h2 : expand(h2)$
h2 : xthru(h2);


h3:matrax([d3[1,1],d3[1,2],d3[1,3]],[d3[2,1],
    d3[2,2],d3[2,3]],[d3[3,1],d3[3,2],d3[3,3]])$

h3 : bfloat(h3)$
h3 : ev(h3)$
h3 : expand(h3)$
h3 : xthru(h3);
closefile();
quit();
```

% %

S I M U L A T I O N   R O U T I N E S

% %

```
C      % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
C
C      SIMFLECT3:  Simulate Full Lagrange-Euler Computed Tourque

C                  for the MMAE scenerio.
C
C      Abstract:  This program attempts to control the PUMA-600
C        using the computed torque formulation with LE equations
C         of motion for a six degree of freedom robot arm. C
 PUMACLE3 is used to generate
C         the dynamics and the full interial matrix and gravity
C         vector are used.  Nonlinear terms are ignored. Desired
C         positions, velocities accelerations and initial C
positions are input from the TRAJSETUP subroutine.  The C
PUMA-600 is simulated by the SIM6DOF subroutine.
C         Error data is formated and stored by the SEOUT C
subroutine.  The GRDATA subroutine allows the user to C
```

```
select the operational environment.
C
C      VERSION 3      by      LARRY TELLMAN      28 JUL 88
C
C
C      % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
C      % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
C
      SUBROUTINE RBTFLE3(OPT,Q,QD,I6,RB6,F6M,D,P,GG)
C
C      Abstract:  This subroutine allows the user to obtain C
   several formulations of the Lagrange-Euler dynamics C
for a 3 link PUMA-600 robot arm.  The user must select C
which option and also provide position, velocity, C
acceleration and joint 6 load information.
C         Any load is assumed to be rigidly attached to joint 6.
C         All user supplied joint 6 values must represent the C
    link and load modeled as one entity.  Actuator inertia C
   values are summed with the diagonal inertia terms using C
   Tarn's values.  The reduced MACSYMA LE equations are C
used.
C
C
C      VERSION 2.0    by          LARRY TELLMAN          13 JUL 88
C
C      Inputs:
C
C         OPT:  An integer variable with selects the dynamics C
            formulation desired.
C
C         Q:  A (6x1) real vector of joint angles in radians.
C
C         QD:  A (6x1) real vector of joint velocities in C
     radians.
C
C         I6:  A (3x3) matrix of joint 6 interia terms.
C
C         RB6:  A (3x1) real position vector of the center of C
            mass of joint 6 with respect to itself as ( C
    x,y,z) vector.
C
C         F6M:  A real variable representing the external mass of
C               joint 6.
C
C      Outputs:
C
C         D:  A (6x6) real matrix of interial terms.
```

```
C
C        P:  A (6x1) real vector of coriolis and centrifugal C
           forces
C
C        GG: A (6X1) GRAVITY VECTOR
C
C      SUBROUTINE OPTIONS:
C
C      OPT=1:  The D matrix is assumed to be diagonal and C
           Coriolis and centrifugal terms are ignored.
C
C      OPT=2:  The full D matrix is calculated but the the C
           Coriolis and centrifugal terms are ignored.
C
C
C      OPT=3:  The full Lagrange-Euler dynamics are calculated.
C
C      TOURQUE CALCULATION:
C                                          T
C      T = (D * QDD) + P          P =  (QD) * H * QD + G
C
C      QDD:  A (6x1) vector of joint accelerations.
C
C      % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
```

```
C      % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
```

```
       SUBROUTINE SROBOT(Q,QD,QDD,TIN,I6,RB6,F6M,DELT,
      #                  NINT,ENOISE,SNOISE)

C      Abstract:  This subroutine simulates the motion of a 6 C
      DOF robot arm.  Manipulator dynamics are calculated C
 using the full
C        Lagrange-Euler formulation.  A 4th order Runga-Kutta C
      integration technique is employed to compute the C
position, velocity and acceleration of the six joints C
that result from an applied torque.  The user can C
specify the total simulation time, size of the C
integration interval and joint 6 loading.
C
C    VERSION 2          MICHAEL B. LEAHY JR.              15 SEP 85
C
C      REVISION 1:   Incorporate viscous and static friction C
                 models
C      17 Jul 87
C
```

```
C     REVISION 2:   Incorparate encoder and system noises
C     28 JUL 88
C
C     Inputs:
C
C        Q:  A (6x1) real vector of joint angles in radians.
C
C        QD:  A (6x1) real vector of joint velocities rad/sec.
C
C        I6:  A (3x3) matrix of joint 6 inertia terms.
C
C         RB6:  A (3x1) real position vector of the center of C
C            mass of joint 6 with respect to itself.
C
C         F6M:  A real variable representing the mass of joint 6
C
C         TIN:  A (6x1) real vector of applied torques.
C
C          DELT:  A real variable representing the total C
C        simulation period.
C
C         NINT:  An integer variable representing the number of
C                integration intervals.
C
C     ENOISE: STRENGTH OF THE UNIFORM ENCODER NOISE
C
C     SNOISE: STRENGTH OF THE GAUSSIAN SYSTEM NOISE
C
C     Outputs:
C
C        Q,QD:  Final values of these vectors.
C
C         QDD:  A (6x1) real vector of final joint acceleration
C               values in rad/sec% 2.
C
C     NOTE:  DELT/NINT  SHOULD ALWAYS BE LESS THAN 1 ms.
C
C     % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
```

```
C     % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
C
      SUBROUTINE SLCTIC3(NIC)

C     Abstract:  This subroutine allows the user to select the
C        manipulator initial condition for control algorithm C
    evaluation under R3AGE.  The IC may be one of three C
predefined conditions, input by the user or remain C
```

```
unchanged.  User input conditions are automatically C
checked against the specific manipulator range limits C
by the RCHK subroutine.  IC values are stored in
C       COMMON vectors in degrees and radians.
C
C      VERSION 1.0          MICHAEL B. LEAHY JR.          7 DEC 85
C
C      REVISION 1:  Incorporates TMODE into MTYPE common and C
                corrects
C       27 FEB 86   error of missing T6D matrix in TRAJ common.
C
C      REVISION 2:  CHANGED TO RUN ON THE MICROVAX FILE C
           STRUCTURE
C     14 JUL 88      LARRY TELLMAN
C
C      Output:
C
C         Q0:  A (6x1) COMMON vector of initial joint angles in
C              degrees.
C
C         Q0R:  A (6x1) COMMON vector of initial joint angles in
C               radians.
C
C          NIC:  An integer representing IC option number.  When
C               the IC's remain unchanged so does this value.
C
C      % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
```

```
C      % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
C
       SUBROUTINE SLCTLD3
C
C      Abstract:  This subroutine determines the manipulator C
     load configuration used by R3AGE for a 3 link PUMA.
C          The user may select the default load model or input C
     his/her own inertial and center of mass
C          values.  Total link and load mass is checked against C
      manipulator limits.
C
C      VERSION 1.0          MICHAEL B. LEAHY JR.          7 DEC 85

C
C           REVISED  BY   LARRY TELLMAN                28 JUL 88
C
C
C      REVISION 1:  Incorporates TMODE into MTYPE common.
C       27 FEB 86
```

```
C
C     REVISION 2:  Incorporates changes to default loading to
C      26 AUG 87    to account for joint 6 w/o a gripper as per
C                   Tarn's dynamics.
C
C     REVISION 3:  CHANGED TO REPRESENT A 3 LINK PUMA ARM
C      28 JUL 88
C
C
C     Inputs:
C
C        RTYPE:  A character*2 COMMON variable that containing
C                the selected manipulator code.
C
C     Outputs:
C
C        H6:  A (3x1) COMMON vector of load/link inertia about
C             the center of mass.
C
C        R6B:  A (3x1) COMMON vector of load/link center of C
C        mass.
C
C        F6M:  A COMMON real variable of load/link total mass.
C
C     % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
C
C
C
C
C     % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
C
C
      SUBROUTINE SLCTTJ3(NIC,PNIC,NSPI,ND)
C
C     Abstract:  This subroutine allows the user to select the
C        manipulator joint space position, velocity and C
acceleration trajectories for control algorithm C
evaluation under R3AGE.  A zero, slow and fast set of C
base trajectories are predefined.  The user may C       specify
his/her own base trajectories contained in
C        a set of three files.  Actual trajectories stored in C
     COMMON arrays are determined from the base trajectory C
   and input sample rate.  Position trajectories are C
formed by addition of the initial conditions selected C
by the SLCTIC subroutine and actual trajectory data, C
and are checked against specific manipulator
C        range limits by the RCHK subroutine.  Trajectories C
   starting from IC option 2 are reversed.  The option to C
  leave existing actual trajectory data unaltered is also C
```

```
  available.
C
C
C      VERSION 1.0          MICHAEL B. LEAHY JR.          7 DEC 85
C
C       REVISION 1:  Incorporates the changes necessary so that
C                    IC2
C      30 JAN 86    initial condition selection is correctly C
             handled when an unchanged trajectory is C
        selected.
C
C      REVISION 2:  Incorporates TMODE into MTYPE common.
C       26 FEB 86
C
C      REVISION 3:  Incorporates changes to allows generation of

C       27 MAR 86    zero trajectory for any 7ms multiple.
C
C       REVISION 4:  Corrects errors in trajectory file C
          specification
C        8 Aug 86    read statements.
C
C       REVISION 5:  Change default fast trajectory to spline C
                  one.
C        22 FEB 88
C
C       REVISION 6:  CHANGED TO MATCH THE FILE STRUCTURE ON THE
C                    MICROVAX
C     14 JUL 88 LARRY TELLMAN
C
C      Input:
C
C         Q0R:  A (6x1) COMMON vector of initial joint angles in
C               radians.
C
C        NSPI:  An integer representing sampling rate speed.
C
C        NIC:  An integer representing initial condition number.

C
C         PNIC:  An integer representing the previous initial C
            condition number.
C
C      Output:
C
C           ND:  An integer representing the number of sampling
points.
C
C           QDSI:  A (6,ND) COMMON matrix of incremental joint
positions.
C
C        QDST: A (6,ND) COMMON matrix of joint velocities.
```

```
C
C       QDSTT: A (6,ND) COMMON matrix of joint accelerations.
C
C     % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %



C     % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %

      SUBROUTINE DINV(Q, IFILT, DET, A)
C
C
C
C     THIS ROUTINE WILL CONPUT THE INVERSE INERTIA MATIRX FOR A
C     THREE LINK PUMA ARM.  THE EQUATIONS ARE BASED ON TARN'S
C     PAPER AND HAVE BEEN REDUCED BY MACSYMA.  THE MATRIX NUMBER
C     DIRECTS THIS ROUTINE TO CALCULATE THE INERTIA FOR AN
C     ASSUMED LOAD.
C
C     VERSION 1:          BY   LARRY TELLMAN       2 AUG 88
C
C
C     INPUTS:
C
C
C        Q:      POSITION VECTOR
C
C
C        IFILT:  THE INERTIA MATRIX NUMBER
C     OUTPUTS:
C
C        A:      THE ADJOINT OF THE INVERSE INERTIA MATRIX
C
C        DET:    THE DETERMINT OF THE INVERSE INERTIA MATRIX
C
C
C     % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %



 % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %

                E X P E R I M E N T A L   R O U N T I N E S
```

```
% % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %

C      % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
C
      PROGRAM  FFMBC
C
C       THIS MAIN PROGRAM  WILL RUN THE MMBC ALGORITHM ON THE
PUMA-560.
C     THE FIRST THREE LINKS OF THE ROBOT ARE USED TO DEMONSTRATE
THE  CONCEPT. IT NEEDED TO BE LINKED TO THE MMAELIB LIBRARY TO
PICK  UP  THE  SUBROUTINES  CALLED.   THIS  ROUTINE  IS  RUN  IN
CONJUNCTION WITH THE ARCADE PROGRAM.  THIS ROUTINE WILL MAKE
THE CALLS TO
C     READ AND WRITE THE DATA TO THE SERVO PROCESSOR AND THE
ARCADE
C     PROGRAM HANDELS THE CONTROL OF THE ROBOT.  THE MMAE
ESTIMATE
C     USED BY THIS ROUTINE CAN BE TURNED ON AND OFF BUT THE
C     CALCULATIONS FOR THE ESTIMATE ARE DONE EACH SAMPLE PERIOD.
THE
C     TIME REQUIRED FOR THE MMAE CALCULATIONS IS ON THE ORDER OF
18ms.
C     THE DATA TRANSFER TAKERS AN ADDITIONAL 3 ms.  THEREFORE
THE
C     FEEDFORWRAD SAMPLE PERIOD NEEDED BY THIS ROUTINE IS 28ms.

C     THE ACTUATOR INERTIAS ARE INCLUDED IN THE INERTIA MATRIX

C     RETURNED FROM THE DYNAMICS SUBROUTINE.
C
C
C     VERSION: 1     BY     Larry Tellman      2 Oct 1988
C
C
C
C
C      % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
% % %


C      % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
% % %
      SUBROUTINE MMAEINI(INSP,ND)
C
C       THIS ROUTINE WILL LOAD THE INITIAL DATA NEEDED FOR THE
MMAE TO
C       RUN ON THE PUMA.  IT IS CALLED BY THE FFMMBC ROUTINE.
C
C
```

```
C     DATE:  26 SEPT 88                    Larry Tellman
C
C      % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
% % %



C      % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
% % %
C
      SUBROUTINE PFDYN3(Q,QD)
C
C        Abstract:  This subroutine allows the user to obtain
several
C           formulations of the Lagrange-Euler dynamics for a 3
link PUMA-600
C         robot arm.  The user must select which option and also
provide
C            position, velocity, acceleration and joint 6 load
information.
C         Any load is assumed to be rigidly attached to joint 6.
All
C          user supplied joint 6 values must represent the link
and load
C           modeled as one entity.  Actuator inertia values are
summed with
C          the diagonal inertia terms using Tarn's values.  The
reduced
C          MACSYMA LE equations are used.  This routine is used
by FFMMBC.
C
C
C      VERSION 2.0    by          LARRY TELLMAN          26 SEPT
88
C
C     Inputs:
C
C         OPT:  An integer variable with selects the dynamics
formulation
C             desired.
C
C        Q:  A (6x1) real vector of joint angles in radians.
C
C           QD:   A (6x1) real vector of joint velocities in
radians.
C
C        I6:  A (3x3) matrix of joint 6 interia terms.
C
C         RB6:  A (3x1) real position vector of the center of
mass of
C                joint 6 with respect to itself as ( x,y,z )
```

```
vector.
C
C       F6M:  A real variable representing the external mass of
joint 6.
C
C     Outputs:
C
C       D:  A (6x6) real matrix of interial terms.
C
C         P:  A (6x1) real vector of coriolis and centrifugal
forces
C
C       GG: A (6X1) GRAVITY VECTOR
C
C     SUBROUTINE OPTIONS:
C
C        OPT-1:   The D matrix is assumed to be diagonal and
coriolis and
C              centrifugal terms are ignored.
C
C       OPT-2:   The full D matrix is calculated but the the
coriolis and
C              centrifugal terms are ignored.
C
C
C      OPT-3:  The full Lagrange-Euler dynamics are calculated.

C
C     TOURQUE CALCULATION:
C                                                T
C     T = (D * QDD) + P        P =  (QD) * H * QD + G
C
C     QDD:  A (6x1) vector of joint accelerations.
C
C     % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
% % %
```

```
 % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
% % % % %
```

```
    M M A E    A N D    K A L M A N   F I L T E R   R O U N
T I N E S
```

```
 % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
% % % % %
```

```
C     % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
% %
      PROGRAM KALTST
C
C     THIS  PROGRAM  CALCULATES  AND  STORES  THE  PROPAGATED
COVARIANCE
C     MATRIX AND TEH MEASUREMENT UPDATE COVARIANCE MATRIX.
C
C
C     VERSION 1:     BY    LARRY TELLMAN   31 JUL 88
C
C
C     % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
%



C     % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
% % %
      SUBROUTINE KGAIN(IFILT,GAIN)
C
C
C     THIS  SUBROUTINE  WILL  COMPUTE  THE  KALMAN  FILTER  GAIN  FOR
THE
C     FIRST  THREE  LINKS  OF  THE  PUMA  ARM.   THE  CODE  HAS  BEEN
REDUCED
C     AND GENERATED BY MACSYMA.  FOR DETAILS ON THE NOTATION SEE

C     DR. MAYBACK'S BOOK vol. 1.
C
C
C     VERSION 1     BY    LARRY TELLMAN      27 JUL 88
C
C
C     INPUTS:
C
C        IFILT:  THE FILTER NUMBER
C
C     COMMON DATA NEEDED:
C
C       PTM: 6x6 MATRIX OF P(ti-)
C
C       R:  3x3 MATRIX OF THE MEASUREMENT NOISE
C
C     OUTPUTS
C
C       GAIN: 6x3 KALMAN FILTER GAIN MATRIX
C
```

```
C     % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
%



C     % % % % % % % % % % % % % % % % % % % % % % % % % % % % %

      SUBROUTINE MMAE(ND,E,POSD,IPROP,ELOAD)
C
C
C     THIS ROUTINE WILL COMPUTE AN ESTIMATE OF THE LOAD
C     USING THE MULTPLE MODEL ADAPTIVE ESTIMATION SCHEME.
C     SEE DR MAYBECK'S BOOK vol. 2 FOR MORE DETAILS ON THE
C     ALGORITHM AND THE NOTATION.  THE ROBOT IS ASSUME TO BE A
C     THREE LINK PUMA MANIPULATOR.
C
C
C     VERSION 1:     BY   LARRY TELLMAN  2 AUG 88
C
C     INPUTS:
C
C       ND:     THE NUMBER OF DATA POINTS
C
C       Z: THE MEASURED OF THE POSITION ERROR IN THE LINKS
C
C       POSD:   THE DESIRED POSITIONS
C
C       DEL:    THE TIME BETWEEN MEASUREMENTS
C
C       IPORP: THE NUMBER OF ITERATIONS TO PROPAGATE OVER
C
C
C
C     OUTPUT
C
C       ELOAD:  THE ESTIMATE OF THE LOAD
C
C
C     % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
% %



C     % % % % % % % % % % % % % % % % % % % % % % % % % % % % %

      SUBROUTINE PROBEST(ELOAD)

C
C     THIS ROUTINE WILL CALCULATE THE CONDITIONAL PROBABLITY
C     DESITY NEEDED FOR THE MULTIPLE MODEL ADAPTIVE ESTIMATOR
C     ALGORITHM.  FOR MORE DETAILS SEE DR. MAYBECK'S BOOK vol.
```

```
1.
C     H - I FOR THESE CALCULATIONS.  THE PREVIOUS VALUE OF THE

C     CONTITIONAL PROBALITY IS ASSUNED TO BE 1/3 FOR EACH
FILTER.
C     THE CALULATIONS ARE BASED ON LINKS 1 AND 2 ONLY.  THIS
SEEMED
C     TO PROVIDE THE BEST ESTIMATE OF THE LOAD.  THE INITIAL
LOAD
C     ESTIMATE HAS A BIAS TO IT AND MUST BE RESCALED.
C
C
C
C                  1                                T   -1
C     P = ---------------- EXP[ -.5 * RES * A  * RES ]
C                 3/2     1/2
C           (2 PI)      |A|
C
C                                T
C     WHERE    A = H * PTM * H + R
C
C
C
C
C     VERSION 1:           BY    LARRY TELLMAN        3 AUG 88
C
C
C     INPUTS:
C
C       RES:     THE RESIDUES FROM THE KALMAN FILTER
C
C       PTM:     THE  COVARIANCE  MATRIX  AT  THE  END  OF  THE
PROPAGATION
C
C       R:       THE MEASURMENT NOISE MATRIX
C
C     OUTPUT:
C
C       ELOAD: THE ESTIMATE OF THE LOAD
C
C
C     % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
% %



C     % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
%
        SUBROUTINE PTPLUS(IF)
C
C
```

```
C     THIS ROUTINE WILL COMPUTE THE COVARIANCE MATRIX AFTER THE
C     MEASUREMENT UPDATE.  THE CODE WAS REDUCED AND GENERATED BY
MACSYMA.
C      FOR MORE DETAIL ON THE NOTATION SEE DR. MAYBECK'S BOOK
vol. 1.
C
C     VERSION  1          BY          LARRY TELLMAN          3
SEPT 88
C
C     INPUT
C
C     IF:  THE FILTER NUMBER
C
C
C     COMMON DATA NEEDED:
C
C
C       R: THE MEASUREMENT NOISE MATRIX (3x3)
C
C        PTM: THE COVARIANCE MATRIX AFTER THE PROPAGATION CYCLE
(6x6)
C
C     OUTPUT:
C
C       PTP: THE COVARIANCE MATRIX AFTER THE MEASUREMENT UPDATE
(6x6)
C
C
C
C     % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
% %
C


      SUBROUTINE PHIMAT(A,DEL,PHI)
C
C     % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
%
C
C     THIS ROUTINE CONPUTES THE STATE TRANSITION MATRIX WITH THE
C     ASSUMPTION THAT THE F MATRIX IS CHANGING SLOWLY
C     COMPARED TO THE SYSTEM DYNMAICS.   A TRUNCATED TAYLOR
SERIES
C     IS USED TO APPROXIMATE THE exp(Ft) EXPRESSION.  REFERENCE
C     DR. MAYBECK'S BOOK  vol. 1.
C
C
C     VERSION 1        BY      LARRY TELLMAN     27 JUL 88
C
```

```
C
C     INPUT:
C
C        IF:  THE FILTER NUMBER
C
C        A: 6x6 PERTURBATION MATRIX
C
C        DEL:  TIME BETWEEN SAMPLES
C
C     OUTPUT
C
C        PHI: THE STATE TRANSITION MATRIX  6x6
C
C
C     % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %



      SUBROUTINE PTMINUS(IF,POSD,PHI)
C
C     % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
%
C
C     THIS ROUTINE WILL FIND THE COVARIANCE MATRIX AT THE END OF

C     THE PROPAGATION CYCLE.  AN FIRST ORDER APPROXIMATION IS
MADE
C     TO MAKE THE INTEGRATION OF THE (PHI G Q G' PHI') TERM
POSSIBLE.
C     PHI HAS A SECOND ORDER APPROXIMATION IN IT.  THE INERTIA
MATRIX
C     IS ASSUMED TO BE CONSTANT OVER THE PROPAGATION PERIOD.
C
C
C    VERSION 1:       BY   LARRY TELLMAN  31 JUL 88
C
C     UPDATE 1:  2 SEPT 88
C
C                ALLOW FOR THE PROPAGATION OVER MULTIPLE CYCLES
C
C
C
C     INPUTS:
C
C        IF:  THE FILTER THAT SHOULD BE PROPAGATED
C
C        POSD:  DESIRED POSITION
C
C        PHI:   THE STATE TRASITION MATRIX
C
```

```
C     COMMON DATA NEEDED:
C
C        PTP:  THE (6x6) COVARIANCE MATRIX AFTER THE LAST
MEASUREMENT
C
C        QNOISE:  THE (3x3) DYNAMICS DRIVING NOISE. ASSUMED TO
BE DIAGONAL
C
C        DEL:  THE PROPAGATION TIME
C
C     COMMON DATA UPDATED:
C
C        PTM:     THE COVARIANCE MATRIX AT THE END OF THE
PROPAGATION CYCLE
C
C
C     % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
% % % % %
C


      SUBROUTINE XTMINUS(IF,IPROP,PHI)
C
C     % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
%
C
C     THIS ROUTINE WILL COMPUTE THE STATE ESTIMATE AT THE END OF

C     A PROPAGATION CYCLE FOR A THREE LINK PUMA ARM.
C
C
C    VERSION 1:       BY   LARRY TELLMAN       2 AUG 88
C
C     UPDATE 1:  2 SEPT 88
C                TO ALLOW THE PROPAGATION OVER MULTIPLE SAMPLE
C                PERIODS
C
C
C     INPUTS:
C
C        IF:  THE FILTER NUMBER
C
C        IPROP:  THE NUMBER OF CYCLES TO PROPAGATE OVER
C
C
C     COMMON DATA NEEEDED:
C
C        XTP:   THE STATE ESTIMATE AT THE END OF THE PREVIOUS
C             MEASUREMENT UPDATE
C
```

```
C              F:      SYSTEM DISCRIPTION MATRIX
C
C        DEL:     THE TIME BETWEEN SAMPLE
C
C
C
C     OUTPUT
C
C         XTM:    THE STATE ESTIMATE AFTER THE PROPAGATION CYCLE
C
C         PHI:    THE  STATE  TRANSITION  MATRIX.    NEEDED  IN  THE
COVARIANCE
C                ROUTINE
C
C     % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
%



C     % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %

      SUBROUTINE XTPLUS(IF)
C
C
C     THIS ROUTINE WILL CONPUTE THE STATE ESTIMATE OF THE KALMAN

C     FILTER EQUATIONS.
C
C     VERSION 1:      BY    LARRY TELLMAN        3 SEPT 88
C
C
C     INPUTS:
C
C         IF:  THE FILTER NUMBER
C
C     COMMON DATA NEEDED:
C
C        PTM:    COVARIANCE  MATRIX  AT  THE  END  OF  THE  PROAGATION
CYCLE
C
C        R:       MEASUREMENT NOISE MATRIX
C
C        RES:     THE RESIDULES OF THE STATES
C
C
C     OUTPUT
C
C        XTP:     THE NEW STATE ESTIMATE AFTER THE NEW MEASUREMENT
C
C
C
```

```
C     % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %

 % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %

                    O T H E R   R O U T I N E S   U S E D

 % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %


C     % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
C
      PROGRAM FMAT
C
C     THIS PROGRAM WILL COMPUT THE F MATRIX USED IN PURTABATION
C     CONTROL
C     OF A 3 LINK PUMA 560 WITH NO LOAD.  THE EQUATIONS ARE C
DERIVED
C     FROM THE SIMBOLICLLY RTEDUCED EQUATIONS OF MOTION.  FOR A
C     DIFFERENT
C     LOAD THE EOM MUST BE RERUN USING MACSYMA TO GENERATE THE
C     FORTRAN
C     CODE.  THERE ARE THREE DIFFERENT VERSIONS OF THIS PROGRAM.
C     ONE FOR EACH OF THE LOADS IN THE MMAE.
C
C     SUBROUTINES CALL
C          SLCTTJ3 : COMPUTES THE TRAJECTORY
C          PDGCST3 : COMPUTS THE TORQUE FOR 3DOF PUMA
C     INPUTS
C          NONE
C
C     OUTPUTS
C          F MATRIX OF NUMBERS FOR EACH POINT ALONG THE C
 TRAJECTORY
C
C     VERSION 1.0          LARRY TELLMAN       12 JUL 88
C
C     % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %


C     % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
C
      PROGRAM VECMAX
C
C
```

```
C     THIS PROGRAM TAKES DATA FROM A 6 X 6 vector DATA FILE
C     AND CONVERTS IT INTO THE MATRIXX FORMAT FOR USE IN
C     MATRIXX.
C
C          WRITTEN BY: CAPT LARRY TELLMAN
C               14 JUL 88
C          NO RIGHTS RESERVED
C
C     % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
```

```
C     % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
C
      PROGRAM MATMAX
C
C     THIS PROGRAM TAKES DATA FROM A 6 X 6 MATRIX DATA FILE
C     AND CONVERTS IT INTO THE MATRIXX FORMAT FOR USE IN C
C     MATRIXX.
C
C          WRITTEN BY: CAPT LARRY TELLMAN
C               14 JUL 88
C          NO RIGHTS RESERVED
C
C      % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
```

Appendix B. *Trajectory Profiles*

This appendix contains plots of Trajectories Two and Three. It also has plots of the eigenvalues of the $F(a,t)$ matrices.

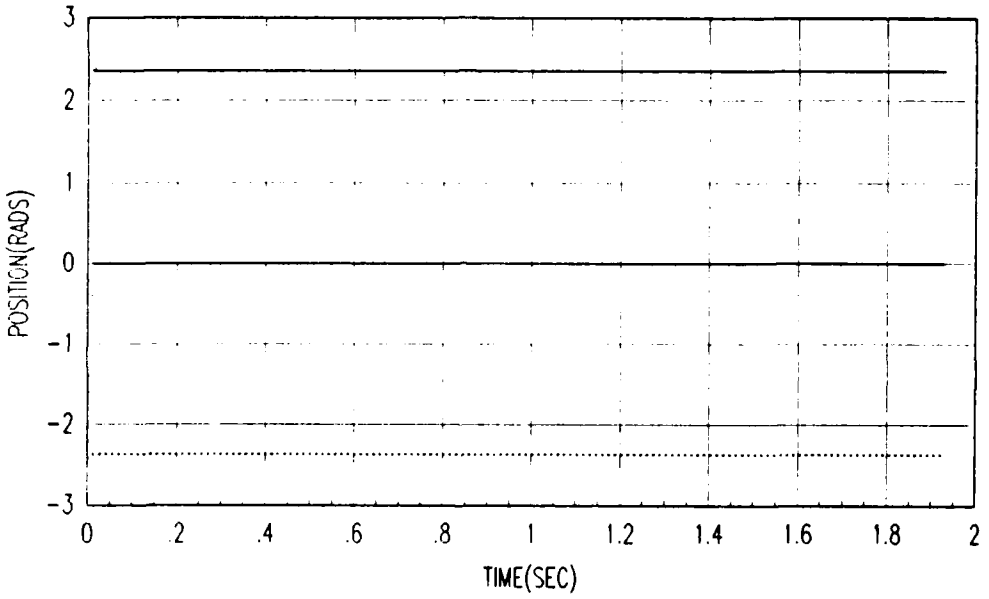

Figure B.1. Trajectory Two: Position

—	Link 1
· · ·	Link 2
- - -	Link 3

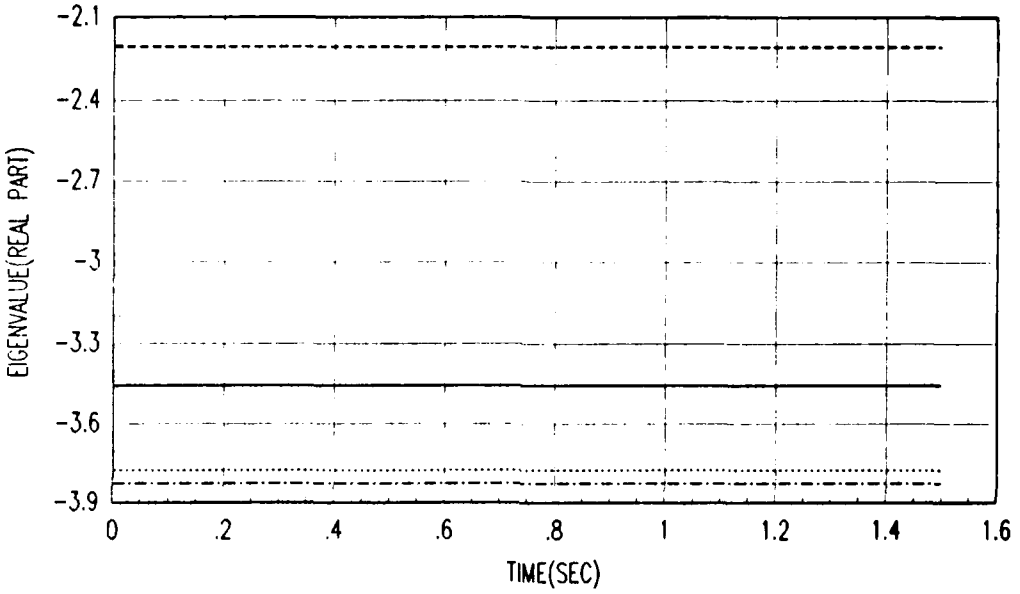

Figure B.2. Eigenvalues for Trajectory Two

—	0.0 Kg: Eigenvalue 1
· · ·	0.0 Kg: Eigenvalue 2
- - -	5.0 Kg: Eigenvalue 1
- · -	5.0 Kg: Eigenvalue 2

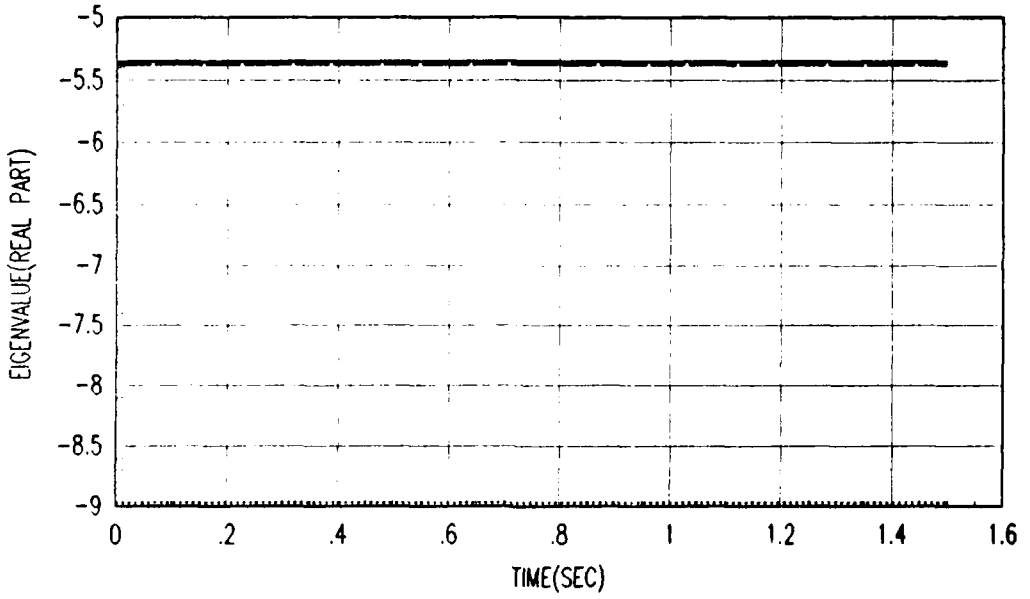

Figure B.3. Eigenvalues for Trajectory Two: Cont

—	0.0 Kg: Eigenvalue 3
· · ·	0.0 Kg: Eigenvalue 4
- - -	5.0 Kg: Eigenvalue 3
- · -	5.0 Kg: Eigenvalue 4

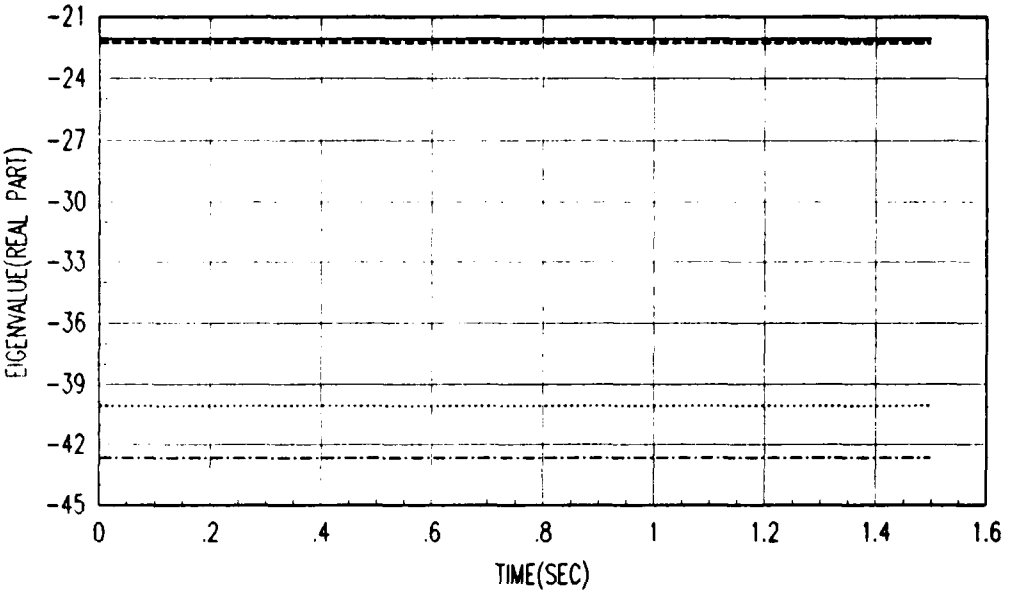

Figure B.4. Eigenvalues for Trajectory Two: Cont

—	0.0 Kg: Eigenvalue 5
· · ·	0.0 Kg: Eigenvalue 6
- - -	5.0 Kg: Eigenvalue 5
- · -	5.0 Kg: Eigenvalue 6

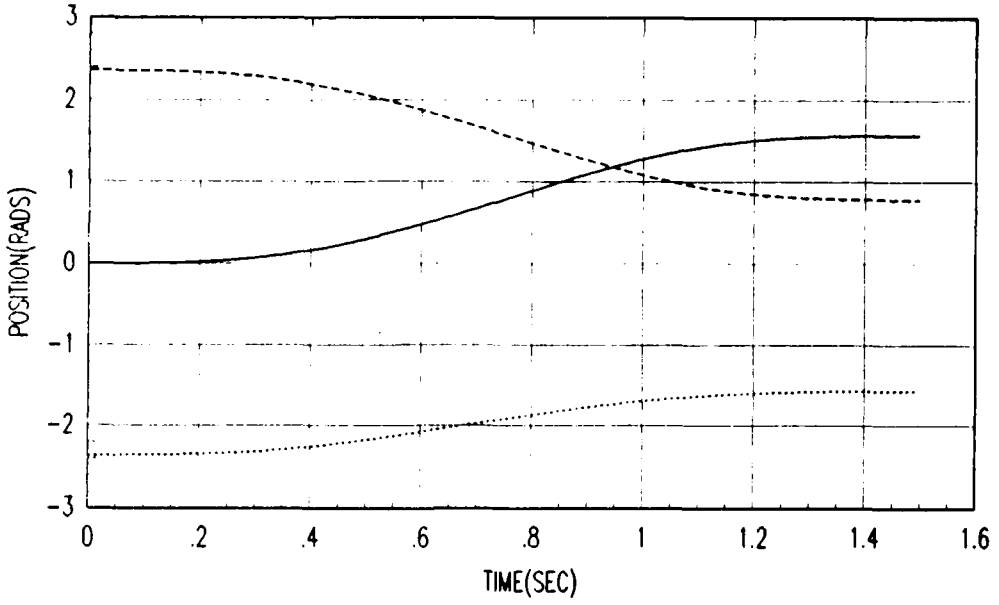

Figure B.5. Trajectory Three: Position

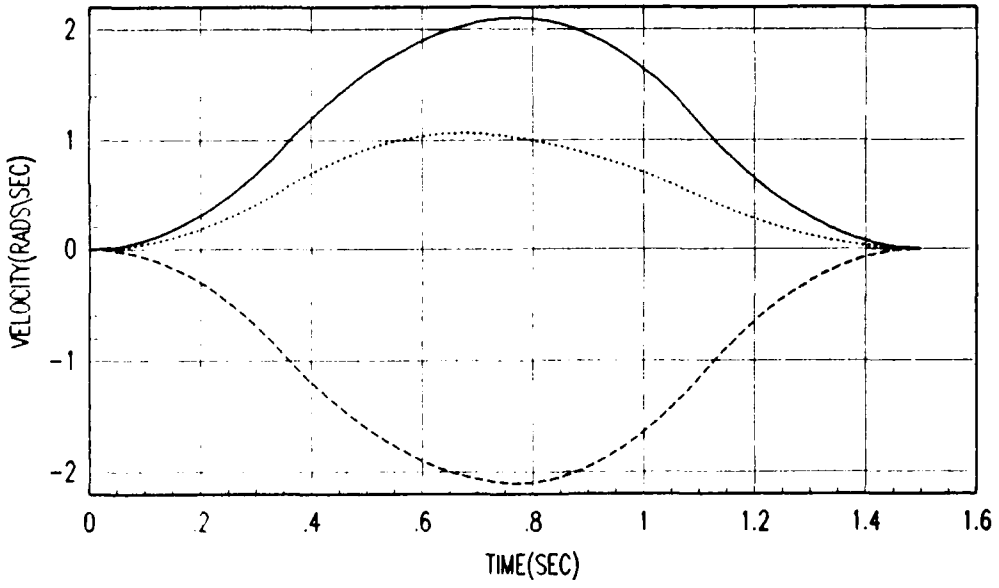

Figure B.6. Trajectory Three: Velocity

—	Link1
· · ·	Link 2
- - -	Link 3

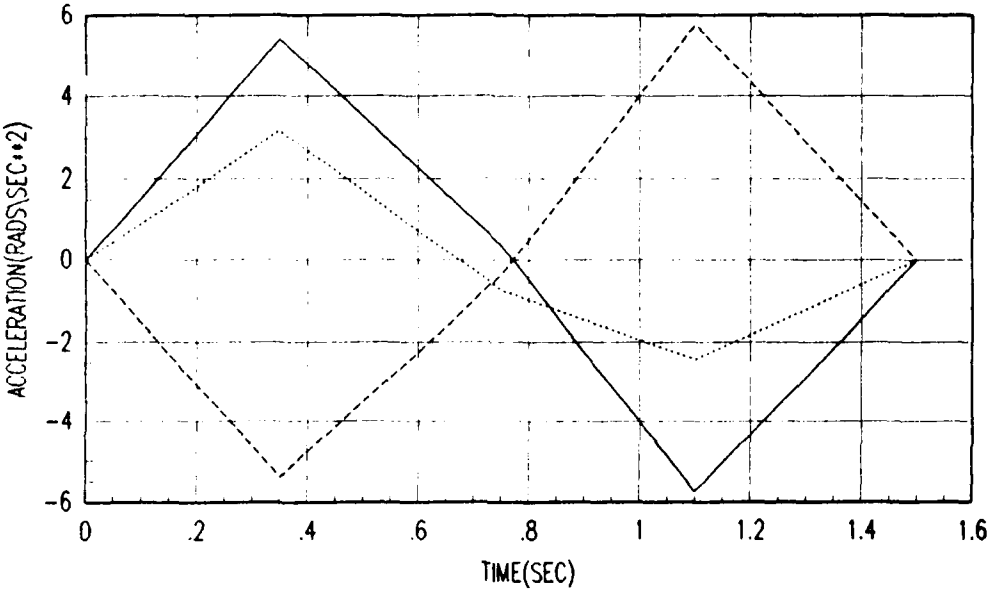

Figure B.7. Trajectory Three: Acceleration

—	Link 1
· · ·	Link 2
- - -	Link 3

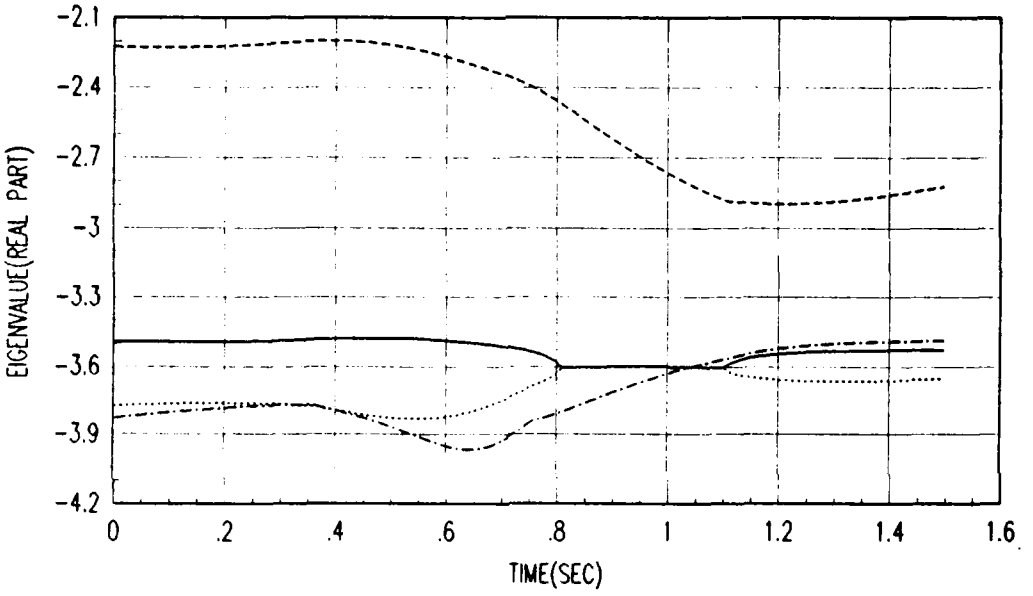

Figure B.8. Eigenvalues for Trajectory Three

—	0.0 Kg: Eigenvalue 1
· · ·	0.0 Kg: Eigenvalue 2
- - -	5.0 Kg: Eigenvalue 1
- · -	5.0 Kg: Eigenvalue 2

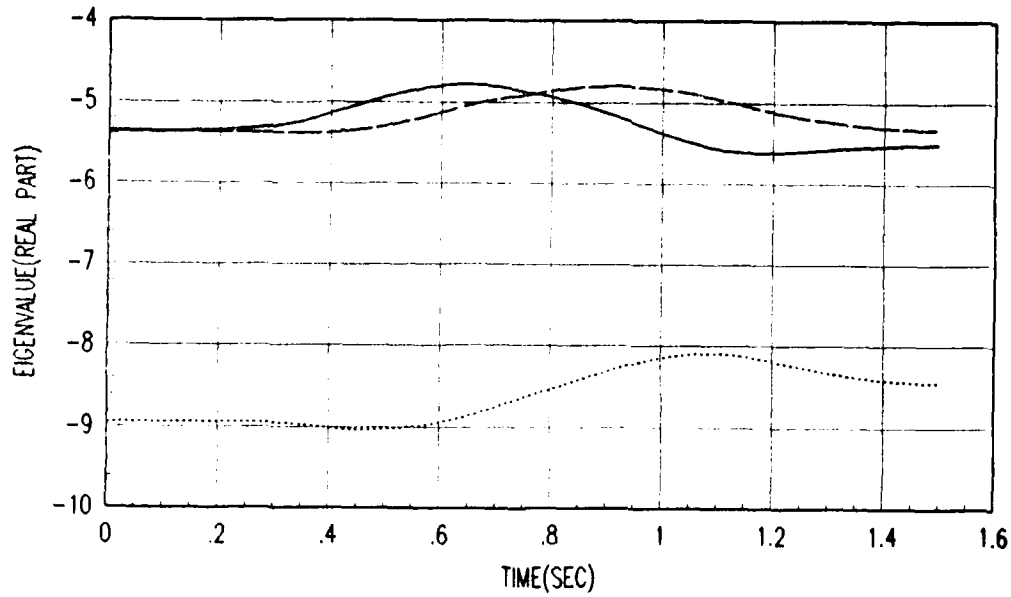

Figure B.9. Eigenvalues for Trajectory Three: Cont

—	0.0 Kg: Eigenvalue 3
· · ·	0.0 Kg: Eigenvalue 4
- - -	5.0 Kg: Eigenvalue 3
- · -	5.0 Kg: Eigenvalue 4

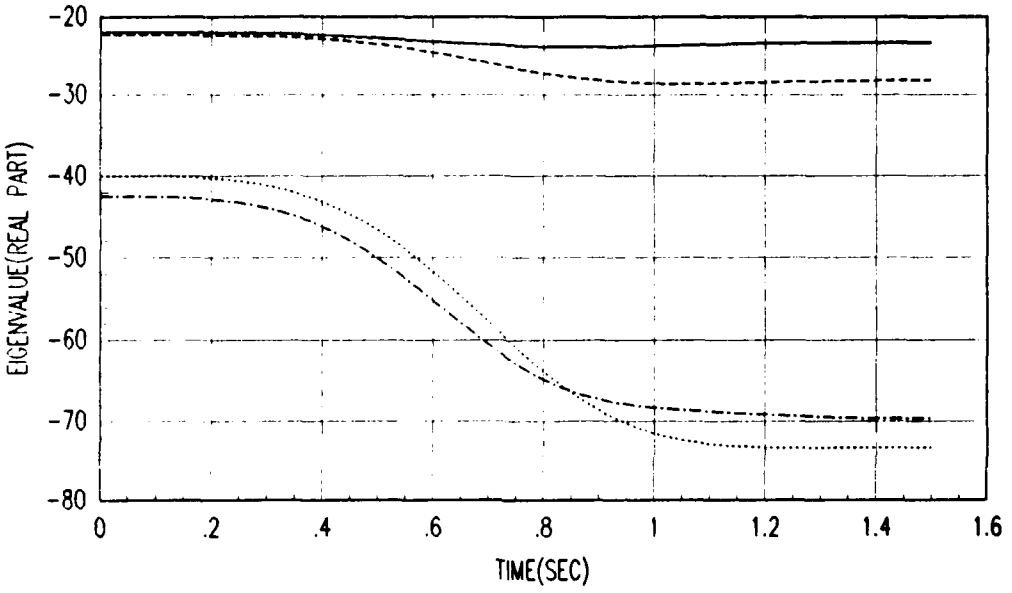

Figure B.10. Eigenvalues for Trajectory Three: Cont

—	0.0 Kg: Eigenvalue 5
· · ·	0.0 Kg: Eigenvalue 6
- - -	5.0 Kg: Eigenvalue 5
- · -	5.0 Kg: Eigenvalue 6

Appendix C. *Error Tracking Profiles for Trajectory Three*

This appendix contains plots of the tracking errors for Trajectory Three. It also has the plot of the payload estimate for Trajectory Three.

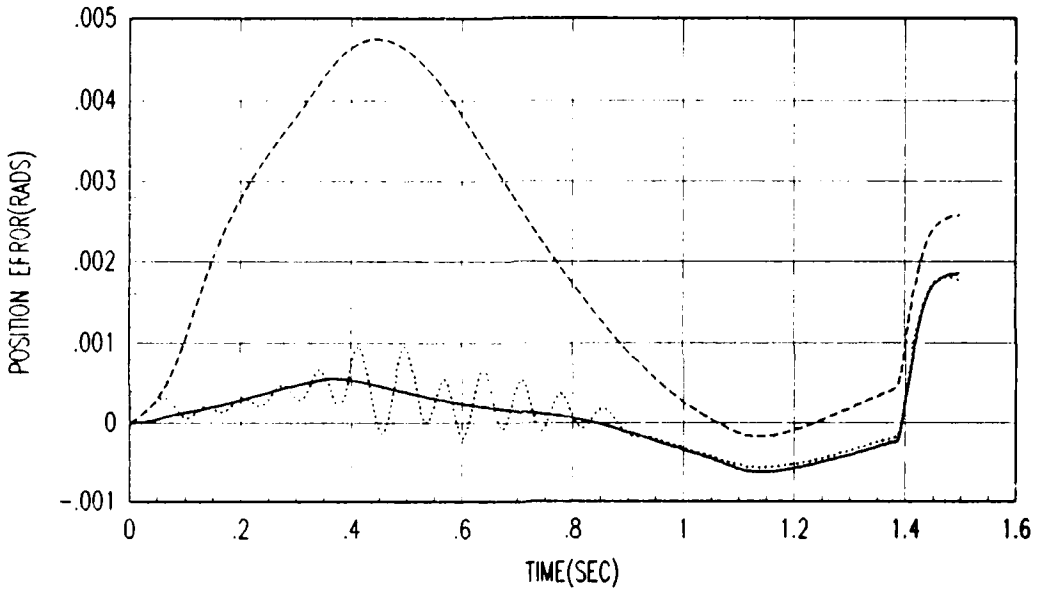

Figure C.1. Tracking Error for the Trajectory Three: Link 1

—	SMBC With Full Payload Information
· · ·	MMBC No Payload Information
- - -	SMBC With No Payload Information

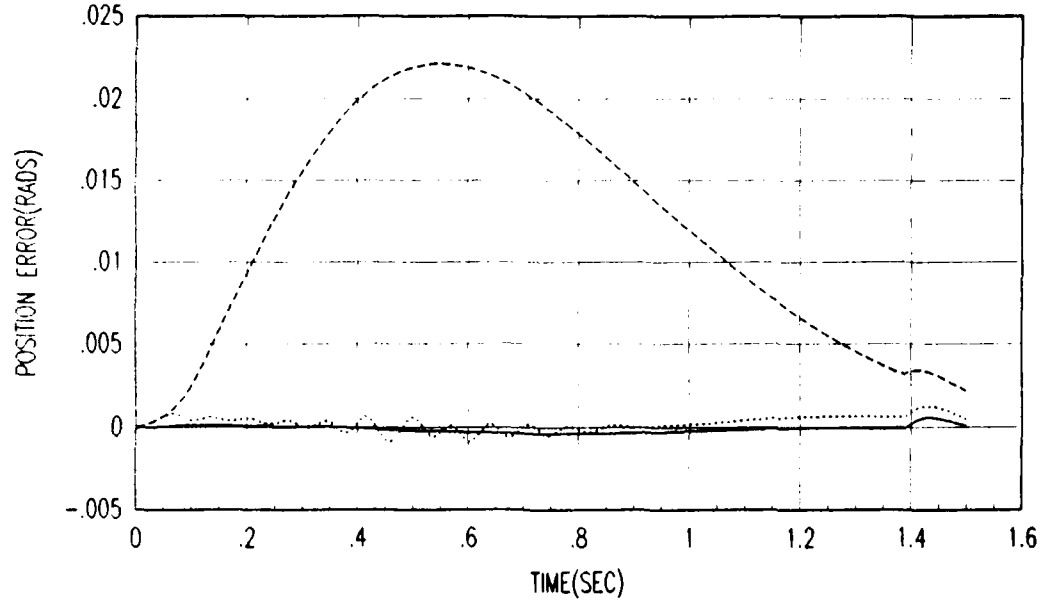

Figure C.2. Tracking Error for the Trajectory Three: Link 2

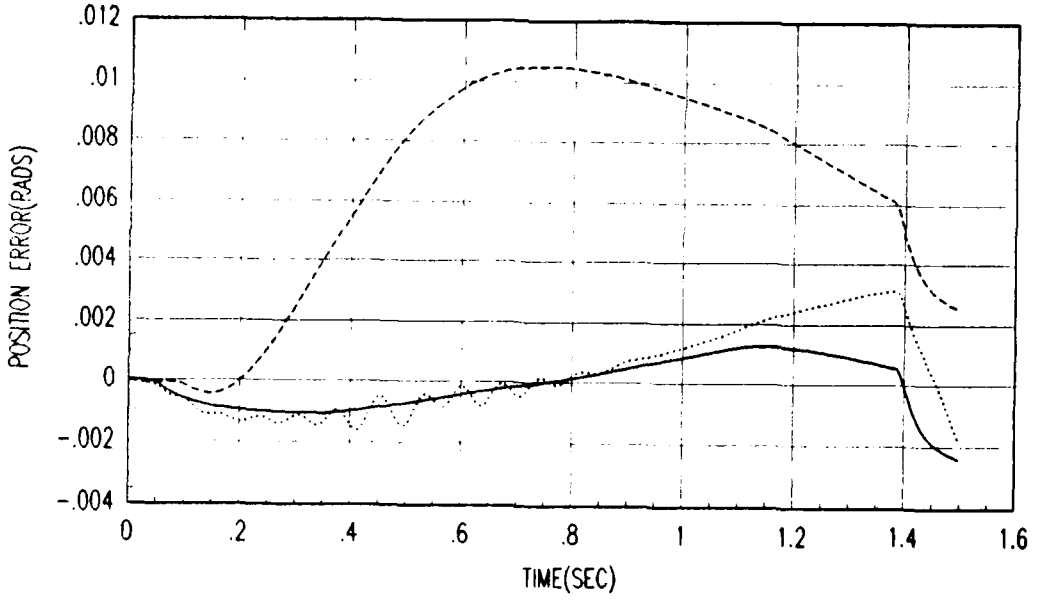

Figure C.3. Tracking Error for the Trajectory Three: Link 3

—	SMBC With Full Payload Information
· · ·	MMBC No Payload Information
- - -	SMBC With No Payload Information

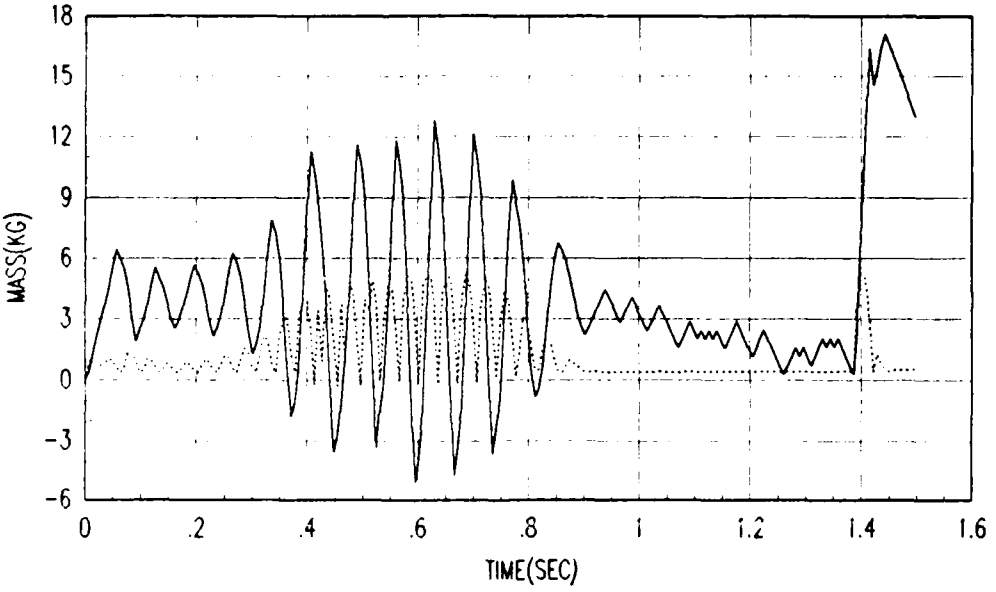

Figure C.4. Payload Estimate for the Trajectory Three

—	Payload Value Used In The Feedforward Element: $\hat{a}$
· · ·	Payload Estimate Out Of MMAE: $\hat{a}_f$

Appendix D. *Experimental Results*

This appendix contains the experimental results for Trajectory Three. The first set of plot are of the tracking errors for each of the links. The plots show the cases where the incorrect SMBC, true SMBC, and the MMBC. The final plot is of the payload value used by the controller and the payload estimate out of the MMAE.

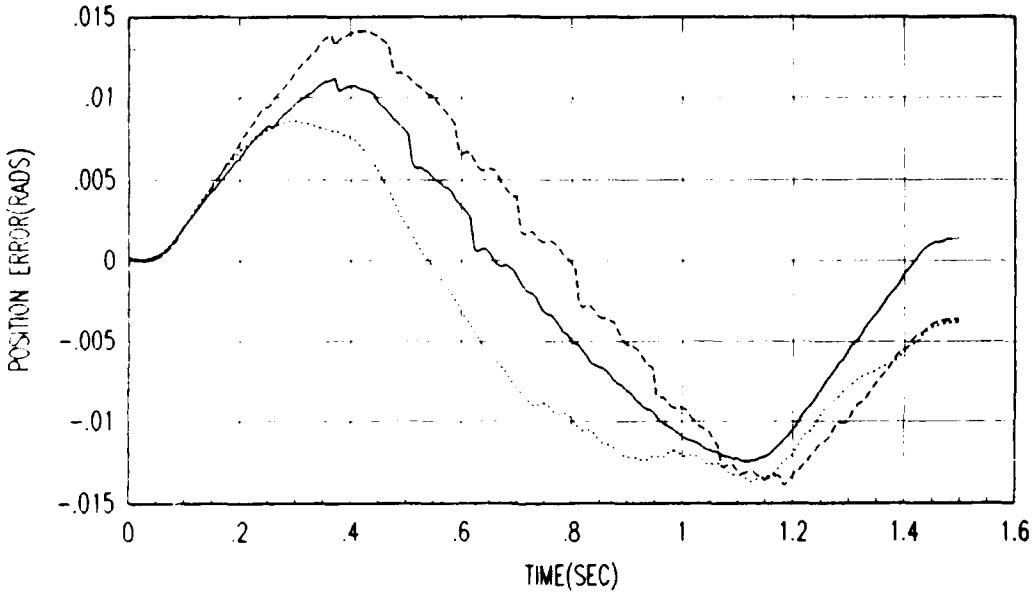

Figure D.1. Experimental Tracking Error for Trajectory Three: Link 1

—	SMBC With Full Payload Information
· · ·	MMBC No Payload Information
- - -	SMBC With No Payload Information

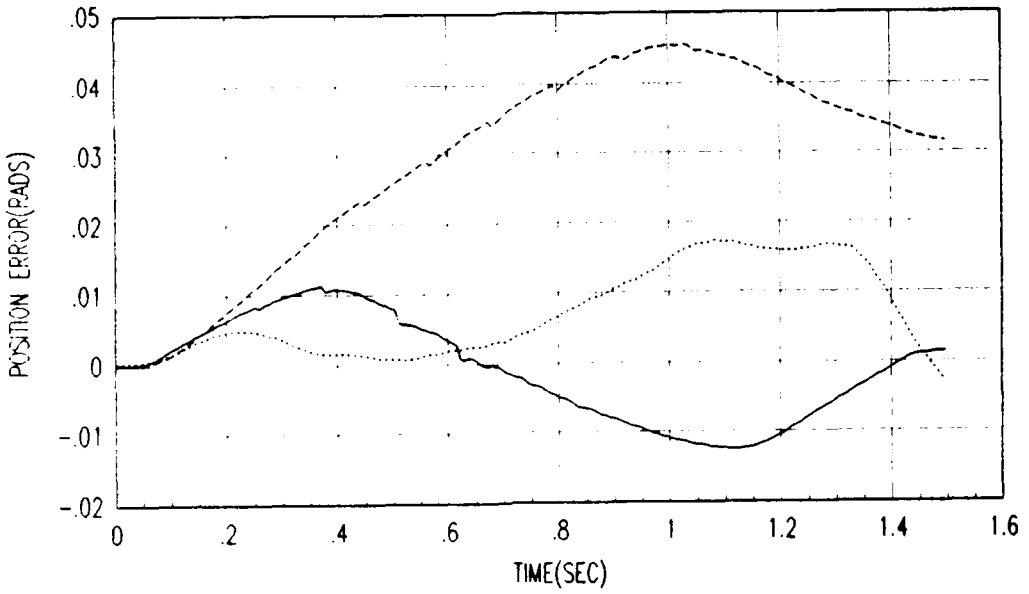

Figure D.2. Experimental Tracking Error for Trajectory Three: Link 2

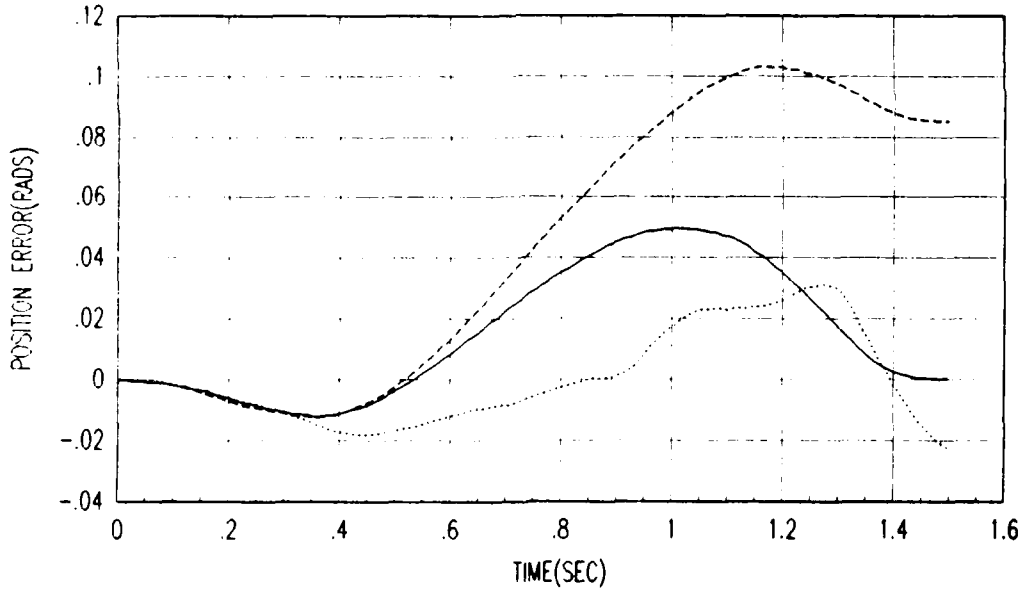

Figure D.3. Experimental Tracking Error for Trajectory Three: Link 3

——	SMBC With Full Payload Information
· · ·	MMBC No Payload Information
- - -	SMBC With No Payload Information

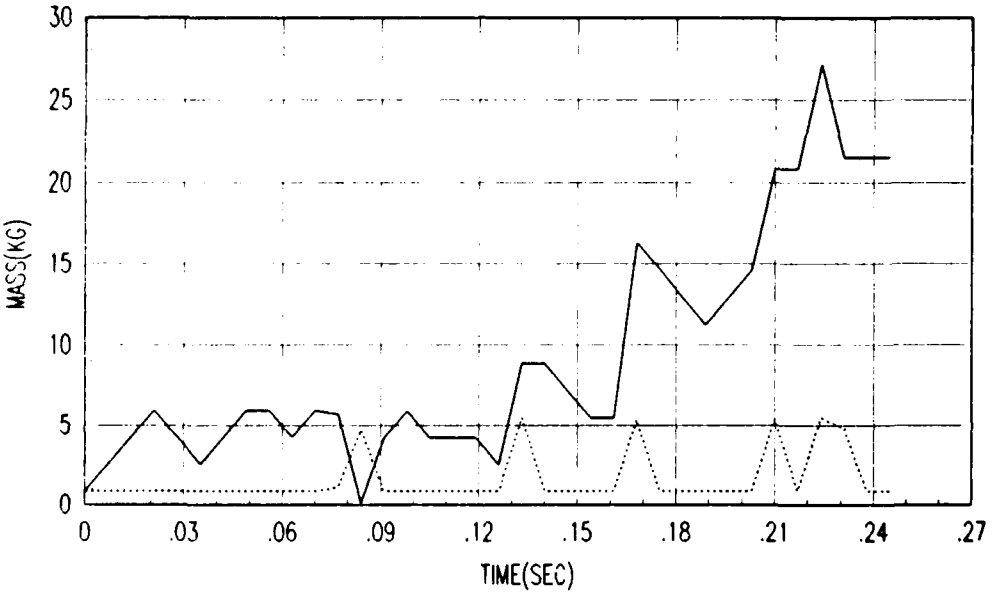

Figure D.4. Experimental Load Estimate for Trajectory Three

—	Payload Value Used In the Feedforward Element: $\hat{a}$
· · ·	Payload Estimate Out Of MMAE: $\hat{a}_f$

Bibliography

AAGH87. C. H. An, C. G. Atkeson, J. D. Griffiths, and J. M. Hollerbach. Experimental evaluation of feedforward and computed torque control. *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 165–168, March-April 1987.

AAH85. C. G. Atkeson, C. H. An, and J. M. Hollerbach. Rigid body load identification for manipulators. *Proc. of 24th IEEE Conf. on Decision and Control*, 996–1002, December 1985.

AAH86. C. H. An, C. G. Atkeson, and J. M. Hollerbach. Experimental determination of the effects of feedfoward control on trajectory tracking errors. *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 55–60, 1986.

AAH88. C. H. An, C. G. Atkeson, and J. M. Hollerbach. Model-based control of a direct drive arm, part ii: control. *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 1386–1391, 1988.

AH86. H. Asada and K. Hara. Load sensitivity analysis and adaptive control of a direct drive arm. *Proc. of the ACC*, 799–805, 1986.

Ath77. M. Athans et al. The stochastic control of the F-8C aircraft using a multiple model adaptive control (MMAC) method - part 1: equilibrium flight. *IEEE Trans. on Automatic Control*, AC-22(5):768–780, October 1977.

Ber83. R. F. Berg. Estimation and prediction for maneuvering target trajectories. *IEEE Trans. on Automatic Control*, AC-28(3):294–304, March 1983.

CHS87. J. J. Craig, P. Hsu, and S. S. Sastry. Adaptive control of mechanical manipulators. *International Journal of Robotics Research*, 6(2):16–28, Summer 1987.

Cra86. J.J. Craig. *Introduction to Robotics, Mechanics and Control.* Addison-Wesley, 1986.

DD79. S. Dubowsky and D. T. DesForges. The application of model-reference adaptive control to robotic manipulators. *Trans. of the Journal of Dynamic Systems, Measurements, and Control*, 101:225–232, September 1979.

DH81. J. J. D'Azzo and C. H. Houpis. *Linear Control Systems Analysis and Design.* McGraw-Hill Book Company, New York, 1981.

Dig86. Digital, Inc. *VAX Realtime User's Guide.* November 1986.

DK85. S. Dubowsky and R. Kornbluh. On the development of high performance adaptive control algorithms for robotic manipulators. *Robotics Research: The 2nd Int. Symposium*, 1985.

dVW87. C. W. deSilva and J. Van Winssen. Least squares adaptive control for trajectory following robots. *Trans. of the ASME*, 109:104–110, June 1987.

FGL87. K.S. Fu, R.C. Gonzalez, and C.S.G. Lee. *Robotics: Control, Sensing, Vision and Intelligence.* McGraw-Hill Book Company, New York, 1987.

Goo85. R. M. Goor. A new approach to minimum time robot control. *Proc. of Winter Annual Meeting of ASME, Miami Beach, Florida*, November 1985.

GW80. C. S. Greene and A. S. Willski. An analysis of the multiple model adaptive control algorithm. *Proc. of the 19th IEEE Conf. on Decision and Control*, 1142–1145, December 1980.

HBSP87. P. Hsu, M. Bodson, S. Sastry, and B. Paden. Adaptive identification and control of manipulators without using joint acceleration. *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 1210–1215, December 1987.

Int88. Integrated Systems, Inc. *MATRIXx Reference Manual.* February 1988.

KG83. A. J. Koivo and T-H Guo. Adaptive linear controller for robotic manipulators. *IEEE Trans. on Automatic Control*, AC–28(2):162–170, February 1983.

Kho88. P. K. Khosla. Some experimental results on model-based control schemes. *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 1380–1385, 1988.

KK88. P. K. Khosla and T. Kanade. Experimental evaluation of nonlinear feedback and feedforward control schemes for manipulators. *Int. Journal of Robotics Research*, 7(1), February 1988.

KM87. D. A. Karnick and P. S. Maybeck. Moving bank multiple model adaptive estimation applied to flexible spacestructure control. *Proc. of the 26th IEEE Conf. on Decision and Control*, 2:1249–1257, December 1987.

Las87. Robert W. Lashlee Jr. *Moving Bank Multiple Model Adaptive Estimation Applied to Flexible Spacestructure Control.* Master's thesis, Air Force Instutitute of Technology, Air University, December 1987.

LC84. C.S.G. Lee and M.J. Chung. An adaptive control strategy for mechanical manipulators. *IEEE Trans. Automatic Control*, AC-29,(9):837–840, 1984.

LE87. K. Y. Lim and M. Eslami. Robust adaptive controller designs for robot manipulator systems. *IEEE Journal of Robotics and Automation*, RA-3(3):54–66, February 1987.

Lea87a. M.B. Leahy Jr. Class handouts from eeng 540, robotic fundimentals. January 1987. School of Engineering, Air Force Instute of Technology (AU), Wright-Patterson AFB OH.

Lea87b. M. B. Leahy Jr. et al. Experimental evaluation of a puma manipulator model-referenced adaptive controller. *Proc. of the 26th IEEE Conf. on Decision and Control*, 1987.

Lea88a. M.B. Leahy Jr. Dynamics based control of vertically articulated manipulators with variable payloads. *Accepted for publication in Int. Journal of Robotics Research*, 1988.

Lea88b. M.B. Leahy Jr. Industrial manipulator control with feedforward dynamic compensation. *Accepted for publication in the Proc. of the 27th IEEE Conf. on Decision and Control*, 1988.

Lea88c. M.B. Leahy Jr. July-September 1988. Discussions concerning model based control of the PUMA-560.

Lea88d. M.B. Leahy Jr. Dynamics based control of vertically articulated manipulators. *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 1046-1056, April 1988.

LS86. M.B. Leahy Jr. and G. N. Saridis. The ral hierarchical control system. *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 407–411, April 1986.

LS88a. M.B. Leahy Jr. and G.N. Saridis. Compensation of industrial manipulator dynamics. *Accepted for publication in Int. Journal of Robotics Research*, 1988.

LS88b. W. Li and J-J. Slotine. Indirect adaptive robot control. *Submitted to the IEEE Trans. on Automatic Control*, April 1988.

Luh83. J.Y.S. Luh. Conventional control design for industrial robots: a tutorial. *IEEE Trans. on Systems, Man, and Cybernetics*, SMC-13(3):210–228, May/June 1983.

May79. P. S. Maybeck. *Stochastic Models, Estimation and Control.* Volume 1, Academic Press, Inc., New York, 1979.

May82a. P. S. Maybeck. *Stochastic Models, Estimation and Control.* Volume 2, Academic Press, Inc., New York, 1982.

May82b. P. S. Maybeck. *Stochastic Models, Estimation and Control.* Volume 3, Academic Press, Inc., New York, 1982.

May88. Peter S. Maybeck. July-September 1988. Discussions concerning Multiple Model Apadtive Estimation.

MG86. R. H. Middelton and G. C. Goodwin. Adaptive computed torque control for rigid link manipulators. *Proc. of the 25th IEEE Conf. on Decision and Control*, 1:68–73, 1986.

MR83. P. S. Maybeck and S. K. Rogers. Adaptive tracking of multiple hot spot target ir images. *IEEE Trans. on Automatic Control*, AC 28(10):937–943, October 1983.

MS85. P. S. Maybeck and R. L. Suizu. Adaptive tracker field of view variation via multiple model filtering. *IEEE Trans. on Aerospace and Electronic Systems*, AES-21(4):529–539, July 1985.

MZ85. P. S. Maybeck and W. L. Zicker. MMAE-based control with space-time point process observations. *IEEE Trans. on Aerospace and Electronic Systems*, AES 21(3):292–300, May 1985.

Net85. Allan S. Netzer. *Characteristics of Bayesian Multiple Model Adaptive Estimation for Tracking Airborne Targets.* Master's thesis, Air Force Instutitute of Technology, Air University, December 1985.

Ser87. H. Seraji. A new approach to adaptive control of manipulators. *Trans. of The Journal of Dynamic Systems, Measurement, and Control*, 109:193–202, September 1987.

Ser88. H. Seraji. Adaptive independent joint control of manipulators: theory and experiment. *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 854–861, 1988.

She88. Stuart N. Sheldon. Dissertation prospectus: an optimal discretization scheme for multiple model adaptive estimation and control with applications in robotics. April 1988. School of Engineering, Air Force Instute of Technology (AU), Wright-Patterson AFB OH.

SL87a. J-J. E. Slotine and W. Li. On the adaptive control of robot manipulators. *Int. Journal of Robotics Research*, 6(3):49–49, Fall 1987.

SL87b. J-J. E. Slotine and W. Li. Adaptive manipulator control: a case study. *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 1392–1400, 1987.

SLG78. T. Söderström, L. Ljung, and I. Gustavsson. A theoretical analysis of recursive identification methods. *Automatica*, 14:231–244, 1978.

Sym85. Symbolics, Inc. *VAX UNIX MACSYMA Reference Manual.* November 1985.

TB85. T.J. Tarn and A.K. Bejczy. *Dynamic Equations for PUMA-560 Robot Arm.* Dept. of Systems Science and Mathematics, Washington University, St. Louis, MO, July 1985.

Tel88. Larry D. Tellman. *Inernal Report ARSL 88-2 on Multiple Model Based Robot Control: Development and Initial Evaluation.* Air Force Instutitute of Technology, Air University, December 1988.

Wir87. Peter M. Van Wirt. *Development Of A Hybrid Simulator For Robotic Manipulators.* Master's thesis, Air Force Instutitute of Technology, Air University, December 1987.

Woo77. Henry B. Woolf et al., editors. *Webster's New Collagiate Dictinary.* G. and C. Merriam Co., 1977.

YK87. K. Youcef-Toumi and A. T. Y. Kuo. High speed trajectory control of a direct-drive manipulator. *Proc. of the 26th IEEE Conf. on Decision and Control*, 2209–2214, December 1987.

Vita

Captain Larry D. Tellman [redacted] He enlisted in the USAF in December, 1975. He graduated from Arizona State University in 1980 and was commissioned in April, 1981. Capt. Tellman was stationed at Holloman AFB, NM as a Guidance Test Engineer for the MX and Trident missile systems. He was then assigned to Headquarters, US Air Forces Europe in the Directorate of Collection. Following his tour in Germany, he entered the Air Force Institute of Technology in the Guidance and Control sequence under the Electrical Engineering Department in June, 1987.

[redacted]

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION: Unclassified	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION / AVAILABILITY OF REPORT: Approved for public release; distribution unlimited
2b. DECLASSIFICATION / DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION: AFIT; 6b. OFFICE SYMBOL (If applicable): ENG	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS (City, State, and ZIP Code): Air Force Institute of Technology, Wright-Patterson AFB, OH 45433	7b. ADDRESS (City, State, and ZIP Code)
8a. NAME OF FUNDING / SPONSORING ORGANIZATION: AAMRL; 8b. OFFICE SYMBOL (If applicable): BBA	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code): Wright-Patterson AFB, OH 45433	10. SOURCE OF FUNDING NUMBERS: PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification): see box 19

12. PERSONAL AUTHOR(S): Larry D. Tellman, M.S., Capt, USAF

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
M S Thesis	FROM _____ TO _____	1988 December	

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Robot Control, Parameter Estimation, Multiple Model Adaptive Estimation, Closed Loop Parameter Estimation
12	09		
17	07	03	

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

11) Title: Multiple Model-Based Robot Control: Development and Initial Evaluation

Thesis Advisor: Michael B. Leahy Jr, Capt, USAF
Associate Professor of Electrical Engineering

12 Jan 1989

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT: ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION: unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL: Michael B. Leahy Jr, Capt, USAF	22b. TELEPHONE (Include Area Code): (513) 255-3517 / 22c. OFFICE SYMBOL: ENG

DD Form 1473, JUN 86 — Previous editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE
unclassified

ABSTRACT

A new form of adaptive model-based robot control has been developed and experimentally evaluated. The Multiple Model Based Control (MMBC) technique utilizes knowledge of nominal manipulator dynamics and principles of Bayesian estimation to provide payload-independent trajectory tracking accuracy. The MMBC is formed by augmenting a model-based controller, which employs feedforward dynamic compensation and constant gain PD feedback, with a payload estimate provided by a Multiple Model Adaptive Estimator. Extensive simulation studies demonstrated the MMBC's ability to adapt to variations in manipulator payload quickly and accurately. Initial experimental evaluations on the first three links of a PUMA-560 validated the algorithm's potential.