Experimental Testing of a Decentralized Model Reference Adaptive Controller for a Mobile Robot

by

Donald Anderson Gardner

Thesis submitted to the Faculty of the

Virginia Polytechnic Institute and State University

In partial fulfillment of the requirements for the degree of

Masters of Science

in

Mechanical Engineering

APPROVED:

_____
Mehdi Ahmadian, Chairman

_____ _____
Don Leo Charles Reinholtz

July 13, 2001
Blacksburg, Virginia

Keywords: Model Reference Adaptive Control, Decentralized Control, Robotic Manipulators, Mobile Robots, Experimental Testing

EXPERIMENTAL TESTING OF A DECENTRALIZED MODEL REFERENCE ADAPTIVE CONTROLLER FOR A MOBILE ROBOT

by

Donald Anderson Gardner

Mehdi Ahmadian, Chairman

Mechanical Engineering

Adaptive controllers allow robots to perform a wide variety of tasks, but the extensive computations required have generated an interest in developing decentralized adaptive controllers. Horner has designed an adaptive controller for a four-degree-of-freedom mobile robot and tested it through simulations. The study described in this thesis uses the techniques described by Horner to design and test a decentralized model reference adaptive controller (DMRAC) for a physical four-degree-of-freedom mobile robot. The study revealed several difficulties in implementing this design. Most notably, the robot available for the research did not allow for the measurement of joint velocity, so it was necessary to estimate the velocity as the derivative of the position measurement. The noise created by this estimation made completion of testing impossible. Future research should be performed on a robot that provides joint velocity measurement. Alternatively, a study could include state estimation as part of the controller, thus reducing and possibly eliminating the need for velocity measurement.

Acknowledgements

I would like to thank Dr. Mehdi Ahmadian for all of his assistance while doing this research, and for his patience while I wrote this thesis. Thanks also go to Drs. Charles Reinholtz and Don Leo for being on my committee and to Derek Devnich for his assistance with proofing and editing this document. Very special thanks go to my wife, Heather. Without her support and assistance, this work may never have been completed.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

Most robots have nonlinear dynamics and are controlled by linear constant-gain controllers, such as proportional-integral-derivative (PID) controllers. Because of the nonlinear dynamics of these robots, these simple controllers only perform well with a limited range of motion or at slow speeds. The performance of the system is even further degraded if the payload that the robot must carry varies, since linear controllers are typically designed using linearized system models that assume constant payloads. These limitations may be acceptable for robots that only perform one set of tasks; however, they are unacceptable for robots that require the adaptability to perform multiple tasks, since changing conditions result in reduced performance accuracy.

These limitations on robot speed, motion, and payload have generated interest in adaptive controllers, which allow robots to have a wider range of operation. Adaptive controllers require significantly more computational power than linear constant-gain controllers. This is because adaptive controllers continually update their gains, while the gains for constant-gain controllers are calculated prior to operation and are never changed. In an effort to reduce the required computational intensity, research has been performed on the use of decentralized adaptive controllers. These controllers treat the robot as a collection of independent subsystems, each with its own controller. This treatment considerably reduces the number of computations, and the controllers for each subsystem can be run in parallel, which further increases the processing speed. Breaking the robot into independent subsystems degrades the controller's performance because the interaction between subsystems is ignored. However, when the gains are updated with a high frequency compared to the plant resonances, the performance degradation is insignificant. In the case of a robotic manipulator, motor/drive system within the manipulator is considered a subsystem.

With improvements in microprocessors and with the ability to make sensors and actuators smaller, mobile robots are now being used for a variety of applications. Many of these applications require robots to go to unreachable places, or to perform high-risk tasks where human involvement would be extremely dangerous. Robots have been designed to collect scientific data, to work in hazardous conditions, and to perform military operations, among many other useful tasks.

These types of tasks require a wide range of non-repetitive motion, making mobile robotics an excellent application for decentralized adaptive controllers. Little research has been done on the use of decentralized adaptive controllers for mobile robots, which motivated Horner to investigate their use through MATLAB simulations [1][1]. The following research has implemented and tested on a physical mobile robot the model referenced adaptive controller simulated by Horner.

1.2 Objectives

The primary objectives of this study were to:

1. Experimentally verify the analytical results obtained on decentralized adaptive controllers for mobile robots,
2. Implement a decentralized model reference adaptive controller (MRAC) on a laboratory-scale mobile robot, and
3. Highlight the difficulties associated with implementing decentralized adaptive controllers on practical systems.

1.3 Approach

First, an inexpensive articulated-arm mobile robot was modified for use in this testing. Then, a decentralized model referenced adaptive controller was implemented using the dSpace controller prototyping environment. Finally, the performance of the robot/controller system was tested and compared with the results from an earlier study by Horner, described in her thesis *Design of a Model Reference Adaptive Control for a Mobile Robot.*

[1] Numbers in brackets indicate references cited at the end of the manuscript.

1.4 Outline

This thesis is described in six chapters. The next chapter, Chapter 2, provides a background on adaptive control, model robot applications, and past studies of decentralized model reference adaptive control for a mobile robot. Chapter 3 provides a detailed description of adaptive control theory. This chapter provides the theory behind the constant gain controller, the model reference adaptive controller, the reference models, the discrete control law, and some of the methods for optimizing the parameter adjustment and decentralized model reference adaptive control methods. Chapter 4 describes the experimental setup for the tests performed. The robot used in the testing is described, and the methods used to find the describing parameters of the robot needed for the controller are described. Chapter 5 details the results obtained. Chapter 6 summarizes the results, and provides possibilities for future research.

Chapter 2

Background

This chapter gives a brief overview of previous research leading to this study and briefly covers work in adaptive control, decentralized adaptive control, mobile robotic applications, and decentralized model reference adaptive control for a mobile robot. Because of the large body of literature, only work related to robotic manipulators has been examined. Since this research is a continuation of the work done by Horner, the background information has been expanded from her research.

2.1 Adaptive Control

A number of studies have been performed to examine adaptive control for robotic manipulators [2-5]. Craig has examined an adaptive version of the computed torque method [2]. The algorithm estimates parameters such as load mass, link mass, and friction, which appear in the nonlinear dynamic model of the system, and then uses the most recent parameter estimates to calculate the required servo torque. This control method is computationally intensive because it requires a complex and accurate model of the system to calculate the necessary torques.

Dubowsky and DesForges have developed the initial application of model reference adaptive control (MRAC) to robotic manipulators [3]. They show that the MRAC algorithm originally formalized by Donalson and Leondes can perform well compared to non-adaptive controllers over a wide range of motions and payloads [4]. The advantage of this adaptive controller is that neither a detailed model of the system nor detailed information about the system parameters is needed. Sardar and Ahmadian have found the computations required for selecting a gain adjustment mechanism to be cumbersome, and have developed a more effective method for selecting the adjustment mechanism [5].

Slotine and Li have derived an adaptive robot control algorithm that can be broken down into two parts: a proportional-derivative (PD) feedback part and a feedforward compensation part [6]. The algorithm is computationally simple, but can

excite unmodeled high-frequency dynamics if the gains are not carefully selected. Mulders et al. have developed an adaptive controller that is the combination of the computed torque method and an adaptive PD controller based on the MRAC method [7].

Pourbograt has proposed an adaptive algorithm for learning control [8]. This algorithm guarantees that the error between the desired trajectory and that of the system approaches zero as the number of repetitions increase. This method does not require any knowledge of the system's dynamic parameters and can be implemented using microprocessors. However, this algorithm has limited application because the motion must be repeated for the error to decrease.

2.2 Decentralized Adaptive Control

Adaptive controllers tend to be computationally intensive, especially as the controlled system becomes more complex [16]. This is because the adaptive gains must be continually updated. The desire to reduce the required computational power for adaptive controllers has drawn interest to decentralized adaptive control for robotic manipulators. In decentralized control, a system is broken down into smaller subsystems, each with their own controller. This reduces the computational power required by each controller, and allows for parallel processing. Using decentralized controllers also reduces the size of the cable bundles and the number of wires required within a robot, which is a major issue in robotics.

A great deal of research has been performed on the development of robust decentralized adaptive controllers for robotic manipulators [9-13]. These studies have used a Lyopunov design method, which does not require detailed knowledge of the system dynamics or of the payload.

Seraji has demonstrated the feasibility of using decentralized adaptive control for robotic manipulators on a PUMA 560 arm [9]. He has found the control scheme to be failure tolerant and robust to variations in the manipulators and in the payload. While the controllers used in these studies perform well, the derivation of the Lyopunov functions used to determine the adjusting mechanism and to prove stability is a significant task. Furthermore, it is difficult to broaden the class of adaptive controllers designed by

Lyapunov methods, since only a limited number of Lyapunov function candidates are known [14].

Another technique used in designing of decentralized adaptive controllers is the model reference adaptive control method using hyperstability theory [15-18]. This method is computationally efficient and does not require a complex system model or information about the payload. Simulations have shown asymptotic trajectory tracking for a number of robot configurations implementing this theory [15-16]. Experimental studies by Tumeh, Gavel and Hsia [17-18] have shown results comparable to Seraji [9].

2.3 Mobile Robotics Applications

In recent years, mobile robots have been used in a number of different applications. The URSULA robot was designed and built for Framatome Technologies to inspect nuclear reactor vessels [25]. IS Robotics, Inc. is developing autonomous legged underwater vehicles for hunting and destroying mines close to shore [26]. The NASA Mars Pathfinder, probably the most famous example of a mobile robot, collected scientific data on the Mars surface and relayed it back to Earth [27]. These are just a few examples of a wide variety of mobile robot applications, but they illustrate that mobile robots must be able to operate in varying conditions. This need for flexibility makes mobile robots an excellent application for adaptive control.

2.4 Decentralized Model Reference Adaptive Control for a Mobile Robot

Horner designed a decentralized model reference adaptive controller for a four-degree-of-freedom mobile robot [1]. Her simulations have shown that a decentralized model reference adaptive controller is efficient, and that its performance is superior to that of a constant-gain controller when the robot performs highly nonlinear time-varying tasks.

Chapter 3

Decentralized Adaptive Control Theory

This chapter develops the adaptive controller used in this research. It is included for completeness, but does not introduce any new theory. The initial controller gains have been developed assuming a linear system. Then the adaptive portion has been developed to compensate for the system non-linearities. The reference model whose response is to be matched is described, and the continuous controller is transformed into the discrete domain. Finally, the parameters that must be defined during the controller design and the decentralizing method are discussed.

This chapter

3.1 Constant Gain Controller

A non-linear dynamic system can be described by the following equations:

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) + \mathbf{f}_{NL}(x,t), \tag{3.1}$$

$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t), \tag{3.2}$$

where $\mathbf{A}$ is the n×n plant matrix, $\mathbf{B}$ is the n×m plant input matrix, $\mathbf{x}(t)$ is the n-dimensional state vector, $\mathbf{u}(t)$ is the m-dimensional control input vector, and $\mathbf{f}_{NL}(x,t)$ represents the non-linear and time varying terms. The vector $\mathbf{y}$ and the matrix $\mathbf{C}$ are the output variable and plant output matrix, each having appropriate dimensions.

A linear system with the same number of states, inputs, and outputs is selected as a reference model. The goal of the controller is for the dynamics of the actual non-linear system to match those of the reference model. The reference model has the following form:

$$\dot{\mathbf{x}}_m(t) = \mathbf{A}_m\mathbf{x}_m(t) + \mathbf{B}_m r(t), \text{ and} \tag{3.3}$$

$$\mathbf{y}_m(t) = \mathbf{C}_m\mathbf{x}_m(t), \tag{3.4}$$

where $\mathbf{A}_m$ and $\mathbf{B}_m$ are n×n and n×m constant matrices respectively, $\mathbf{x}_m(t)$ is the model state vector, r(t) reference input, and $\mathbf{y}_m(t)$ is the reference model output. The error vector, $\mathbf{e}(t)$,

is defined as the difference between the reference model and the states of the actual system:

$$\mathbf{e}(t) = \mathbf{x}_m(t) - \mathbf{x}(t). \tag{3.5}$$

The desired result is for the controller to drive $\left(\lim_{t\to\infty} \mathbf{e}(t) \to 0\right)$. Using state feedback, the control law is:

$$\mathbf{u}(t) = -\mathbf{K}\mathbf{x}(t) + \mathbf{K}_r r(t). \tag{3.6}$$

By taking the derivative of the error equation and substituting the state equations of the system and the reference model, the error equation becomes (Appendix A.1):

$$\dot{\mathbf{e}}(t) = \mathbf{A}_m \mathbf{x}_m + \mathbf{B}_m r - \left((\mathbf{A} - \mathbf{B}\mathbf{K})\mathbf{x} + \mathbf{B}\mathbf{K}_r r\right) - \mathbf{f}_{NL}(\mathbf{x}, \mathbf{t}). \tag{3.7}$$

Then by adding and subtracting $\mathbf{A}_m\mathbf{x}(t)$, the error equation becomes:

$$\dot{\mathbf{e}}(t) = \mathbf{A}_m\left(\mathbf{x}_m(t) - \mathbf{x}(t)\right) + \left(\mathbf{A}_m - \mathbf{A} + \mathbf{B}\mathbf{K}\right)\mathbf{x}(t) + \left(\mathbf{B}_m - \mathbf{B}\mathbf{K}_r\right)r(t) - \mathbf{f}_{NL}(\mathbf{x}, \mathbf{t}). \tag{3.8}$$

If:

$$\mathbf{f}_{NL}(\mathbf{x}, t) = 0, \tag{3.9a}$$

$$\mathbf{B}_m - \mathbf{B}\mathbf{K}_r = 0, \text{ and} \tag{3.9b}$$

$$\left(\mathbf{A}_m - \mathbf{A} + \mathbf{B}\mathbf{K}\right) = 0 \tag{3.9c}$$

are true, then the error equation reduces to:

$$\dot{\mathbf{e}}(t) = \mathbf{A}_m \mathbf{e}(t). \tag{3.10}$$

This ensures that $\boldsymbol{e} \to 0$ as $t \to \infty$, as long as $\mathbf{A}_\mathrm{m}$ is stable.

The first condition states that the actual system is linear and time invariant. The last two conditions are used to determine the values of $\mathbf{K}$ and $\mathbf{K}_r$. To satisfy the second condition (Appendix A.2):

$$\mathbf{K}_r = \mathbf{B}^+\mathbf{B}_m. \tag{3.11}$$

where the superscript plus sign ($^+$) denotes the psuedo-inverse: $\mathbf{B}^+ = \left(\mathbf{B}^T\mathbf{B}\right)^{-1}\mathbf{B}^T$. To satisfy the third condition (Appendix A.3):

$$\mathbf{K} = \mathbf{B}^+\left(\mathbf{A} - \mathbf{A}_m\right). \tag{3.12}$$

These equations for $\mathbf{K}$ and $\mathbf{K}_r$ assume that the pseudo inverse of B has full rank.

The values of $\mathbf{K}$ and $\mathbf{K}_r$ do not guarantee Equation (3.9) unless Erzberger's conditions for perfect model following are met [16]. Erzberger's conditions are:

$$\left(\mathbf{I}_n - \mathbf{B}\mathbf{B}^+\right)\left(\mathbf{A} - \mathbf{A}_m\right) = 0, \tag{3.13a}$$

and

$$\left(\mathbf{I}_n - \mathbf{B}\mathbf{B}^+\right)\mathbf{B}_m = 0, \tag{3.13b}$$

where $\mathbf{I}_n$ is an n × n identity matrix. These conditions are derived from the same equations used to determine the controller gains, and these derivations are shown in Appendix A, Section A.4. If these conditions are met, then the gains calculated will provide the desired results. In general, however

$$\left(\mathbf{I}_n - \mathbf{B}\mathbf{B}^+\right)\mathbf{B}_m \neq 0; \tag{3.14}$$

If Erzberger's conditions are not met, then the reference model needs to be redesigned. Landau states that if both the plant and the reference model have Luenberger-type controllable canonical structures, then Erzberger's conditions will be met [14].

3.2 Model Reference Adaptive Controller

The previous section discussed how to determine appropriate values for the controller gains when assuming a linear and time invariant plant. If the plant is non-linear or is time varying, then Equation (3.10) becomes:

$$\dot{\mathbf{e}}(t) = \mathbf{A}_m\mathbf{e}(t) - \mathbf{f}_{NL}(\mathbf{x}, t). \tag{3.15}$$

To compensate for the time-varying term, the controller must also be time varying:

$$\mathbf{u}(t) = -\left(\mathbf{K} - \delta\mathbf{K}(t)\right)\mathbf{x}(t) + \left(\mathbf{K}_r + \delta\mathbf{K}_r(t)\right)r(t), \tag{3.16}$$

where $\mathbf{K}$ and $\mathbf{K}_r$ are the same as they were previously defined, and $\delta\mathbf{K}$ and $\delta\mathbf{K}_r$ are time varying gain adjustments. By substituting the adjusting controller into the error equation, the error equation becomes:

$$\dot{\mathbf{e}}(t) = \mathbf{A}_m\mathbf{e}(t) + \left(\mathbf{B}_m - \mathbf{B}\mathbf{K}_r\right)r(t) + \left(\mathbf{A}_m - \mathbf{A} + \mathbf{B}\mathbf{K}\right)\mathbf{x}(t) - \mathbf{B}\left(\delta\mathbf{K}_r r(t) + \delta\mathbf{K}\mathbf{x}(t) - \mathbf{f}_{NL}(\mathbf{x}, t)\right), \tag{3.17}$$

as is derived in Appendix A, Section A.5. Again, $\mathbf{K}$ and $\mathbf{K}_r$ are selected so that the second and third terms are canceled out. The error equation now reduces to:

$$\dot{\mathbf{e}}(t) = \mathbf{A}_m\mathbf{e}(t) - \mathbf{B}\left(\delta\mathbf{K}_r r(t) + \delta\mathbf{K}\mathbf{x}(t) - \mathbf{f}_{NL}(\mathbf{x}, t)\right). \tag{3.18}$$

The values of $\delta\mathbf{K}$ and $\delta\mathbf{K}_r$ must be selected for the nonlinear terms to be canceled, which will cause the error equation to become the same as for the linear time invariant case.

Because $\mathbf{f}_{NL}(\mathbf{x},t)$ is unknown, $\delta\boldsymbol{K}$ and $\delta\boldsymbol{K}_r$ cannot be explicitly solved for. The gain adjustments can be found using Hyperstability theory, which deals mainly with the stability of systems that can be broken into a linear time invariant block and a non-linear and/or time varying block. If the linear time invariant block is such that the system is globally stable for all non-linear blocks satisfying the Popov integral inequality [14]:

$$\int_{t_1}^{t_2} y_e{}^T(t) w_e(t) dt \geq -\gamma^2 \text{ for all } t_1 \geq t_2, \tag{3.19}$$

then the feedback system is said to be hyperstable. The linear block is therefore a hyperstable block. A block diagram of this system is shown in Figure 3.1.

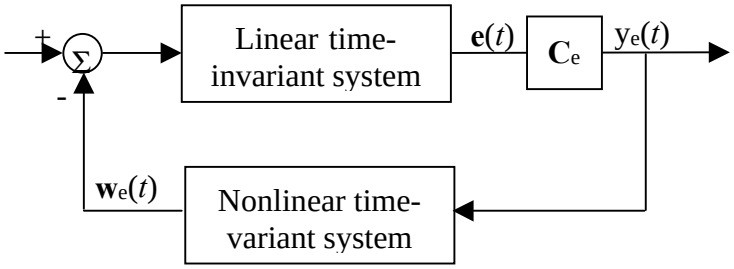

Figure 3.1: Equivalent feedback system used in hyperstability theory

For model reference adaptive control let:

$$\mathbf{w}_e(t) = \delta\mathbf{K}(t)\mathbf{x}(t) + \delta\mathbf{K}_r(t)r(t), \tag{3.20}$$

so that the dynamic error becomes:

$$\dot{\mathbf{e}}(t) = \mathbf{A}_m\mathbf{e}(t) - f_{NL}(\mathbf{x},t) - \mathbf{B}\mathbf{w}_e(t), \tag{3.21}$$

and the output error is:

$$\mathbf{y}_e(t) = \mathbf{C}_e\mathbf{e}(t). \tag{3.22}$$

One of Popov's main assertions is that if $\mathbf{A}_m$, $\mathbf{B}$, and $\mathbf{C}_e$ form a hyperstable block and the adaptive block satisfies the Popov inequality, then the error will asymptotically approach zero.

The hyperstability problem is solved though the proper selection of $\mathbf{C}_e$, according to:

$$\mathbf{C}_e = \mathbf{B}^T\mathbf{P}, \tag{3.23}$$

where the matrix $\mathbf{P}$ is found by solving the Lyapunov equation:

$$\mathbf{P}\mathbf{A}_m + \mathbf{A}_m^T\mathbf{P} = -\mathbf{Q}, \tag{3.24}$$

The coefficient matrices **P** and **Q** in (3.24) are both positive definite.

Satisfying the Popov inequality is a bit more complicated. Landau has presented a solution that yields equations for $\delta\boldsymbol{K}$ and $\delta\boldsymbol{K}_I$:

$$\delta K(t) = \int_0^t \phi_1(\tau)d\tau + \phi_2(t), \text{ and} \tag{3.25}$$

$$\delta K_I(t) = \int_0^t \psi_1(\tau)d\tau + \psi_2(t), \tag{3.26}$$

where:

$$\phi_1(\tau) = \mathbf{F}\mathbf{y}_e(\tau)\mathbf{x}^T(\tau)\mathbf{G}, \tag{3.27a}$$

$$\phi_2(\tau) = \mathbf{F}'\mathbf{y}_e(t)\mathbf{x}^T(t)\mathbf{G}, \tag{3.27b}$$

$$\psi_1(\tau) = \mathbf{M}\mathbf{y}_e(\tau)\mathbf{r}^T(\tau)\mathbf{N}, \text{ and} \tag{3.27c}$$

$$\psi_2(\tau) = \mathbf{M}'\mathbf{y}_e(t)\mathbf{r}^T(t)\mathbf{N}; \tag{3.27d}$$

The matrices **F**, **G**, **M**, and **N** are positive definite, and **F′**, **M′** are positive semi-definite. Several simplifications can be made to solve the Lyapunov equation and the equations for ϕ_1, ϕ_2 ,ψ_1, and ψ_2, such as:

$$\mathbf{Q} = diag(\eta_1, \eta_2, ...\eta_n) \text{ with } \eta_i > 0, \tag{3.28a}$$

$$\mathbf{G} = \mathbf{N} = \mathbf{I}, \tag{3.28b}$$

$$\mathbf{F} = \mathbf{M} = \alpha\mathbf{I}, \tag{3.28c}$$

and

$$\mathbf{F}' = \mathbf{M}' = \beta\mathbf{I}, \tag{3.28d}$$

where

$$\alpha > 0 \text{ and } \beta \geq 0, \tag{3.28e}$$

The matrix **I** is an n×n identity matrix. Using these simplifications, the equations for ϕ_1, ϕ_2, ψ_1, and ψ_2 become:

$$\phi_1(\tau) = \alpha\mathbf{y}_e(\tau)\mathbf{x}^T(\tau), \tag{3.29a}$$

$$\phi_2(\tau) = \beta\mathbf{y}_e(t)\mathbf{x}^T(t), \tag{3.29b}$$

$$\psi_1(\tau) = \alpha \mathbf{y}_e(\tau)\mathbf{r}^T(\tau), \text{ and} \tag{3.29c}$$

$$\psi_2(\tau) = \beta \mathbf{y}_e(t)\mathbf{r}^T(t). \tag{3.29d}$$

The constants α and β must be selected before the above equations can be solved, as will be described in more detail in Section 3.5.1.

3.3 Reference Model

To ensure the stability of the closed loop system, $\mathbf{A}_m$ must have eigenvalues in the left-hand side of the complex plane for stability. These eigenvalues should be selected to provide the desired system performance. $\mathbf{A}_m$ should also have a Luenberger-type controllable canonical structure to ensure that Erzberger's conditions are satisfied.

Once $\mathbf{A}_m$ has been selected, $\mathbf{B}_m$ must be selected to obtain the desired steady state value of the plant output. This is accomplished by assuming that the steady state value will be equal to the reference input:

$$\mathbf{y}_{ss} = \mathbf{y}_{m,ss} = \mathbf{r}, \tag{3.30}$$

and that the model has reached steady state,

$$\dot{\mathbf{x}}_{m,ss} = 0. \tag{3.31}$$

In the above equations, the subscript "ss" denotes steady state. Combining these equations with the system equation yields:

$$\mathbf{A}_m \mathbf{x}_{m,ss} + \mathbf{B}_m \mathbf{r} = 0. \tag{3.32}$$

As derived in Appendix A, Section A.6, the above equation can be rearranged to show

$$\mathbf{B}_m = -\mathbf{A}_m \mathbf{C}^+. \tag{3.33}$$

Therefore, by selecting a stable $\mathbf{A}_m$, the model will be stable, and the error will approach zero.

3.4 Discrete Control Law

Since, the controller for this system has been implemented digitally, the continuous controller that has been developed throughout this chapter must be transformed into the discrete domain. The transformation from the continuous domain to the discrete domain

is not be derived here, but is detailed by Franklin, et al. [19]. A zero-order hold assumption is used for the sampling process.

The discrete-time form of the reference model is:

$$\mathbf{x}_m(k+1) = \mathbf{\Phi}_m \mathbf{x}_m(k) + \mathbf{\Gamma}_m \mathbf{r}(t), \tag{3.34}$$

where

$$\mathbf{\Phi}_m = \exp(\Delta t \mathbf{A}_m), \tag{3.35}$$

and

$$\mathbf{\Gamma}_m = \int_0^{\Delta t} \exp(\tau \mathbf{A}_m) \mathrm{d}\tau \mathbf{B}_m . \tag{3.36}$$

The sampling period is Δt, and the discrete time variable is the integer k. The equation for the controller has no integrals or derivatives, therefore:

$$\mathbf{u}(k) = -(\mathbf{K}_d - \delta\mathbf{K}_d(k))\mathbf{x}(k) + (\mathbf{K}_{r,d} + \delta\mathbf{K}_{r,d}(k))\mathbf{r}(k). \tag{3.37}$$

The zero-order hold equivalent of the adaptive gains are:

$$\delta\mathbf{K}_d(k) = \delta\mathbf{K}_d(k-1) + \beta\mathbf{y}_e(k)\mathbf{x}^T(k) - \sigma\mathbf{y}_e(k-1)\mathbf{x}^T(k-1), \text{ and} \tag{3.38}$$

$$\delta\mathbf{K}_{r,d}(k) = \delta\mathbf{K}_{r,d}(k-1) + \beta\mathbf{y}_e(k)\mathbf{r}^T(k) - \sigma\mathbf{y}_e(k-1)\mathbf{r}^T(k-1), \text{ with} \tag{3.39}$$

$$\sigma = \beta - \alpha\Delta t , \tag{3.40}$$

$$\mathbf{y}_e(t) = \mathbf{C}_e\mathbf{e}(k), \tag{3.41}$$

where

$$\mathbf{e}(k) = \mathbf{x}_m(k) - \mathbf{x}(k). \tag{3.42}$$

The initial conditions are:

$$\delta\mathbf{K}_d(-1) = 0, \tag{3.43a}$$

$$\delta\mathbf{K}_{r,d}(-1) = 0 . \tag{3.43b}$$

The control gain $\mathbf{K}_d$, and $\mathbf{K}_{r,d}$ must be determined using the discrete equivalents of the matrices $\mathbf{A}$, $\mathbf{A}_m$, $\mathbf{B}$ and $\mathbf{B}_m$. The discrete equivalents of $\mathbf{A}_m$ and $\mathbf{B}_m$ are $\boldsymbol{\phi}_m$ and $\mathbf{\Gamma}_m$, and are described above. The discrete equivalents of $\mathbf{A}$ and $\mathbf{B}$ as $\boldsymbol{\phi}$ and $\mathbf{\Gamma}$ are found similarly. The discrete equivalents of $\mathbf{K}$ and $\mathbf{K}_r$ are:

$$\mathbf{K}_d = \mathbf{\Gamma}^+\mathbf{\Gamma}_m , \tag{3.44}$$

and

$$\mathbf{K}_{r,d} = \mathbf{\Gamma}^{+}\left(\mathbf{\Phi} - \mathbf{\Phi}_{m}\right). \tag{3.45}$$

3.5 Optimizing the Control Law

In the previous sections, several parameters have been used that must be selected as part of the controller design. These parameters include the eigenvalues of the reference model, α and β from the adaptation equations, and **Q** used to determine $\mathbf{C}_e$. In addition to selecting these parameters, the controller must be broken in subsystems for it to be decentralized. These topics are discussed in the following sections.

3.5.1 Parameter Adjustments

To complete the design of the MRAC, eigenvalues of the reference model, α and β from the adaptation equation, and **Q** used to determine $\mathbf{C}_e$, must be selected.

The eigenvalues of the reference model must be stable and should be selected so as not to exceed the capability of the system. Performance parameters such as rise time, settling time, and overshoot should be considered when selecting the eigenvalues. The eigenvalues for each subsystem should be the same so that movements can be coordinated.

The values of α and β are arbitrary and are selected through trial and error. Stoten recommends starting with $\alpha = \beta = 1$, then increasing β to reduce the settling time of the adaptation, and increasing the ratio β/α to improve damping [16]. At each iteration, the system should be simulated to examine the effect of the changes; a ten-fold increase at each step is an acceptable adjustment. The final values of α and β are not critical, but they cannot be increased indefinitely because that may magnify noise within the loop. Stoten has drawn an analogy between α and β, and the integral and proportional gains of a PI controller, respectively. Stoten has empirically found that the values of 100 for α and 10 for β worked well.

The selection of **Q** is also relatively arbitrary. **Q** is used in solving the Lyopunov equation to find **P**, which is used to calculate the weighting matrix for the error vector, $\mathbf{C}_e$. **Q** must be positive definite and symmetric about the diagonal. Choosing **Q** to be a

diagonal matrix simplifies solving the Lyopunov equation, but the effect that this has on the controller is unclear. Horner has found that increasing the values in **Q** increased the adaptation speed of the controller, but could lead to instability if the update rate was not sufficiently high [1]. This suggests that **Q** and the update rate should not be selected independently. Horner has also found that the values along the diagonal of **Q** are associated with the states of the system. This means that, if the diagonal value associated with position is larger than that with velocity, then the controller will attempt to match position more closely than velocity. The selection of **Q** is accomplished through trial and error.

3.5.2 Decentralized MRAC

Stoten originally presented the decentralized controller used in this study [16]. Breaking the overall system into several subsystems, each with an independent controller, creates a decentralized system. For a mobile robot, each joint should be an independent subsystem. If it is possible to break a system into two subsystems, then:

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix}, \tag{3.46}$$

and

$$\mathbf{B} = \begin{bmatrix} \mathbf{B}_{11} & \mathbf{B}_{12} \\ \mathbf{B}_{21} & \mathbf{B}_{22} \end{bmatrix}. \tag{3.47}$$

If the off diagonal terms are ignored, then the system equations become:

$$\dot{\mathbf{x}}_1 = \mathbf{A}_{11} + \mathbf{B}_{11} u_1 + \mathbf{f}_{NL1}(\mathbf{x}, t), \text{ and} \tag{3.48}$$

$$\dot{\mathbf{x}}_2 = \mathbf{A}_{22} + \mathbf{B}_{22} u_2 + \mathbf{f}_{NL2}(\mathbf{x}, t), \tag{3.49}$$

where $\mathbf{x}_1$ and $\mathbf{x}_2$ are the states of each subsystem. This method of decentralization ignores coupling between subsystems. By ignoring the coupling between the subsystems in the design of the controller, they effectively become a part of the non-linear term $\mathbf{f}_{NL}(\mathbf{x},t)$. If the controller is properly designed, the adjustable gains should be able to compensate. Ignoring the coupling terms has the benefits of reducing the number of computations required by the controller and of allowing for parallel processing.

Chapter 4

Experimental Setup

This chapter outlines the steps taken to prepare for testing the model reference adaptive controller (MRAC) on the mobile robot. First, a robot, shown in Figure 4.1, was purchased, and then modified to interface with the dSpace controller prototyping equipment. This modification included adding power amplifiers for the motors and designing a transducer for measuring the base position and velocity. Once the robot was assembled, specific link parameters were measured for use in determining the initial gains and the weighting constants for the adaptive controller. These parameters included the motor constants as well as the moments of inertia for each of the links. Finally, the controller was laid out in Simulink/dSpace. The dSpace module Trace was used as the user interface for the Autobox, and the dSpace module Trace was used to record data for future use. Figure 4.2 shows a block diagram of the test setup.

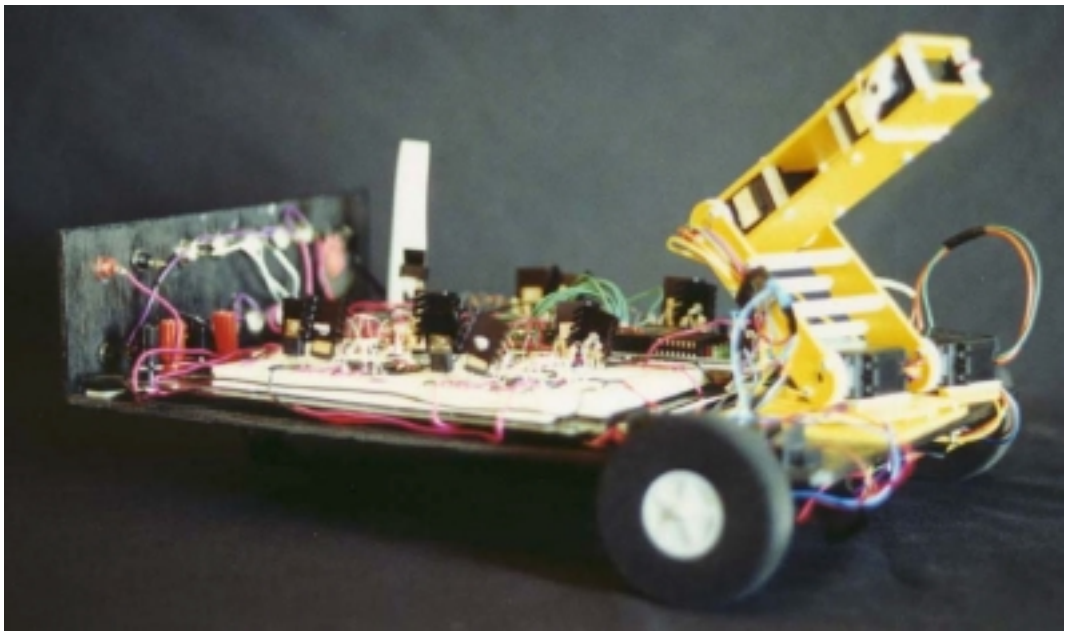

Figure 4.1: Assembled Robot

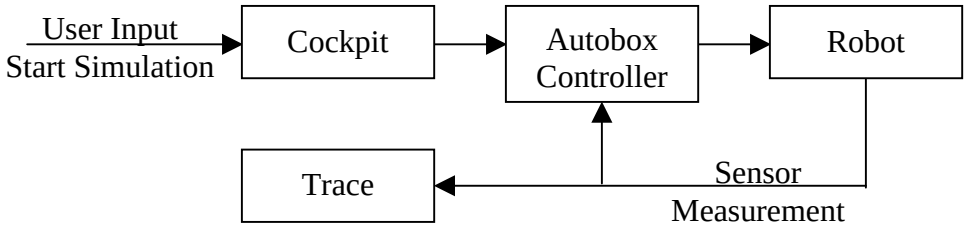

Figure 4.2: Experimental block diagram

4.1 Robot Construction

At the time of selection, the robot used in this research was the only articulated-arm mobile robot available that met the cost constraints of the research. Several commercially available mobile robots were considered, but all were beyond the cost constraints of this project. Building a robot to meet the needs of this research was briefly considered, but the design and construction of a mobile robot is a research project within itself.

A basic four-axis articulated arm mobile robot platform was purchased from Mondotronics, an Internet based hobby robotics supplier. This particular robot only provided position feedback and did not have sufficient structure on the linkages to allow for the addition of sensors for providing velocity feedback. Using the Mondotronics robot and calculate the velocity by using the derivative block in Simulink was deem to be the most effective research method.

Once the base robot was selected, it was modified to suit the research purposes. The actuators were hobby-type servomotors typically used with remote controlled vehicles. Each actuator had a potentiometer for position feedback, as well as a controller to match position reference input. The controller was removed from each servomotor and the actuators were wired to give direct access to the position feedback and to the motor input. Power amplifiers were built to increase the power of the control signal from the Autobox. A new base was constructed for the electronics could be mounted on the robot. Figure 4.1 shows the assembled robot.

Since the actuators for the mobile base of the robot were also hobby-type servomotors that allowed the wheels less than a single rotation, the potentiometers could not be used to measure the base position relative to its starting position. Therefore, the

potentiometer was removed and a different sensor was used to determine the base position. A Polaroid ultrasonic sensor and a Motorola 68HC11[22-23] based Miniboard [24] were used for measuring the base position, calculating the base velocity, and sending the output proportional signals for each through 8-bit digital-to-analog converters. The transducer had a positional range of 0.1 to 1.4 meters and a velocity range of ±2.9 meters per second. The velocity was estimated as the position change divided by the time change between pings. The transducer provided relatively clear signals that required no further filtering.

Finally, a panel was mounted on the rear of the robot, providing input/output access to the robot. BNC jacks provided access to the sensors and to the input signals. Banana plugs were used to provide ±12 volts, +5 volts, and ground. Figure 4.3 shows the arrangement of connectors on the rear panel of the robot. The ultrasonic sensor is on the left, the BNC connectors for the input and output are in the center, and the power connections are on the right.

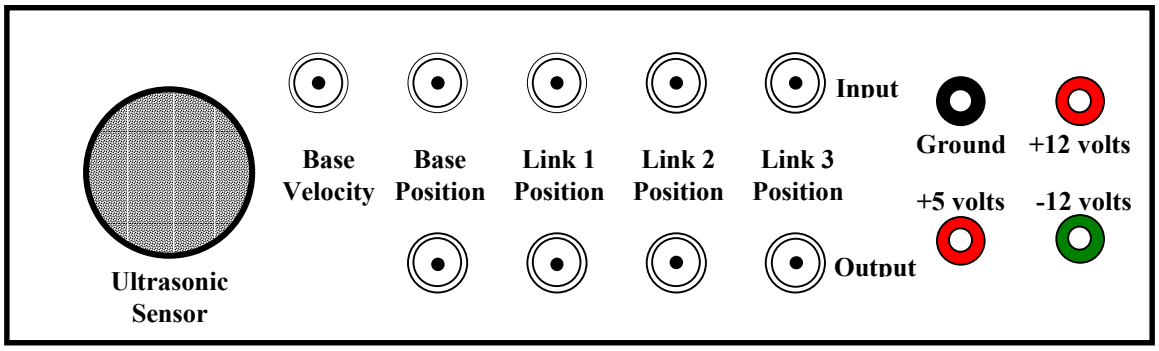

Figure 4.3: Diagram of Robot's Rear Panel

4.2 Parameter Identification

Once the robot was modified, the parameter values were determined. These parameters were necessary to calculate the initial gains and the error-scaling vector for the adaptive controller. The initial gains were calculated to give the desired response for the linearized model of the robot as shown in Section 3.1. It was desired to have the initial error to be zero, in order to make only small, incremental gain changes for maintaining the desired response. The error-scaling vector, $\boldsymbol{C}_e$, was selected so that the plant/controller system would be hyperstable, as described in Section 3.2. The

parameters that needed to be determined for each link were the mass, the length, the distance from the joint to the center of mass, and the polar and non-polar moments of inertia. In addition, the motor constant and the damping constants for each of the actuators were determined. The determination of these constants is described in the following sections.

4.2.1 Determination of Link Masses, Lengths, and Moments of Inertia

The masses and lengths for the links were easily measured. Determining the distances between the joint and the center of mass and the link moments of inertia was only slightly more difficult. The mass and dimensions of each piece of each link were measured. These pieces were simple, which made their centers of mass and moments of inertia easy to calculate. The only assumptions necessary were that the effect of the screw holes and screws could be ignored and that the mass of the motors was uniformly distributed throughout its volume. The first assumption was valid because the mass of the screws was small compared to the rest of the link. The second assumption was valid because the motors were densely packed and small relative to the link, so the motors could be modeled as uniform masses without having a significant effect on the results.

The polar moment of inertia was calculated about an axis parallel to the axis of rotation through the center of mass of the link, while the non-polar inertia was calculated about the axis that runs along the length of the link through its center of mass. Table 4.1 lists the values calculated for each of the moments of inertia, and Table 4.2 lists the masses and lengths measure for each of the links.

Table 4.1: Link Moment of Inertias

Description	Parameter	Value
Moment of Inertia for link 1 (non-polar axis)	J_1	9.00×10^{-4} kg m^2
Moment of Inertia for link 1 (polar axis)	J_{1p}	1.16×10^{-5} kg m^2
Moment of Inertia for link 2 (non-polar axis)	J_2	1.25×10^{-5} kg m^2
Moment of Inertia for link 2 (polar axis)	J_{2p}	1.45×10^{-5} kg m^2
Moment of Inertia for link 3 (polar axis)	J_3	2.06×10^{-4} kg m^2

Table 4.2: Robotic Arm Link Masses and Lengths

Description	Parameter	Value
mass of link 1	m_1	0.0167 kg
mass of link 2	m_2	0.1364 kg
mass of link 3	m_3	0.1082 kg
mass of base	m_4	0.8924 kg
Distance from Joint to end of link 1	L_1	0.0918 m
Distance from joint to center of gravity of link 1	l_1	0.0459 m
Distance from Joint to end of link 2	L_2	0.0957 m
Distance from joint to center of gravity of link 2	l_2	0.0479 m

4.2.2 Determination of Motor Constants

The torque and damping motor constants were needed to calculate the initial gains and $\mathbf{C}_e$. It was necessary to measure the motor constants because the manufacturer could not provide the information. This was not surprising since the motors were modified actuators normally used for remote controlled vehicles. Unmodified servos have internal controllers that are sufficient for their intended purpose, so the manufacturer would not expect a user to need the motor constants and therefore would not have them available.
Determining of the torque and damping motor constants proved to be considerably more difficult than determining the other robot model parameters. Three different methods were used to determine these motor constants, each method giving different results. The three methods used were frequency response matching, component parameter measurement, and step response matching. The frequency response method was used first, but the response measured did not fit the motor model, as indicated in Figures 4.4-4.7. The component parameter measurement method was used next, but one of the parameters could not be measured accurately with the available equipment. The step response matching method was used last, as shown in figures 4.8-4.11, but the motor constants varied for each moment of inertia used. Each of these testing methods should have resulted in similar values for the motor constants, with some values more accurate than others. In actuality, each test resulted in values differing from the others by several orders of magnitude. Non-linearities within motor are likely to have caused the motor to perform differently for different operating conditions. Since the operating conditions vary depending on joint position, velocity, payload, etc., the parameter values of the order

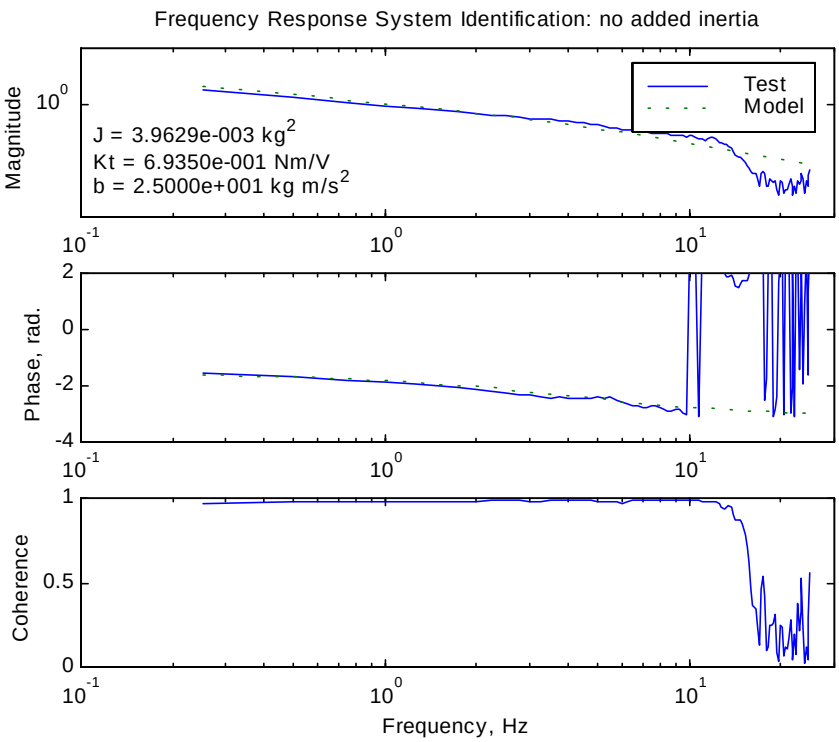

Figure 4.4: Motor Frequency Response with no added inertia

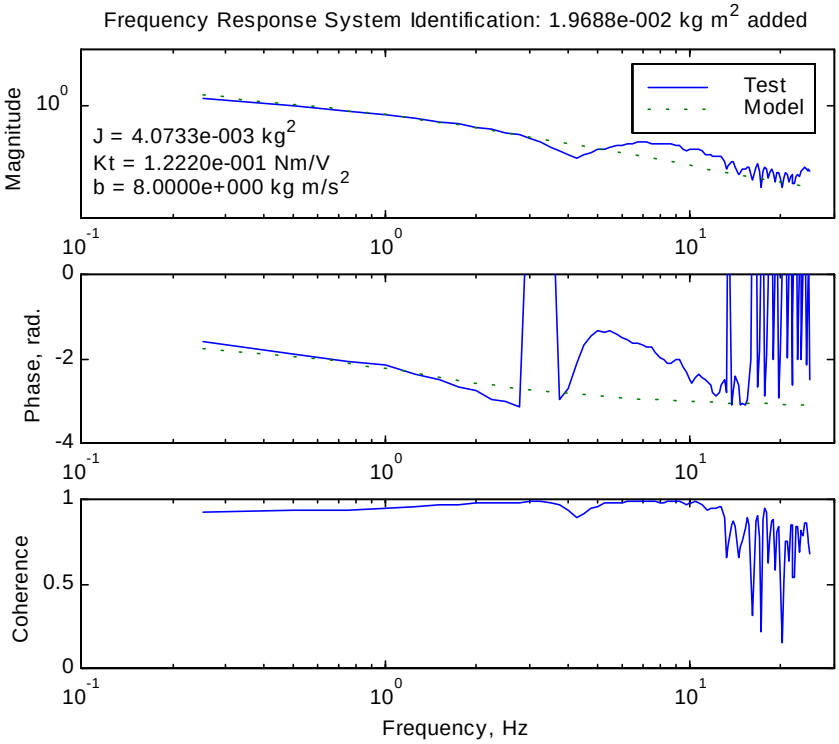

Figure 4.5: Motor Frequency Response with 0.0197 kg m2 added

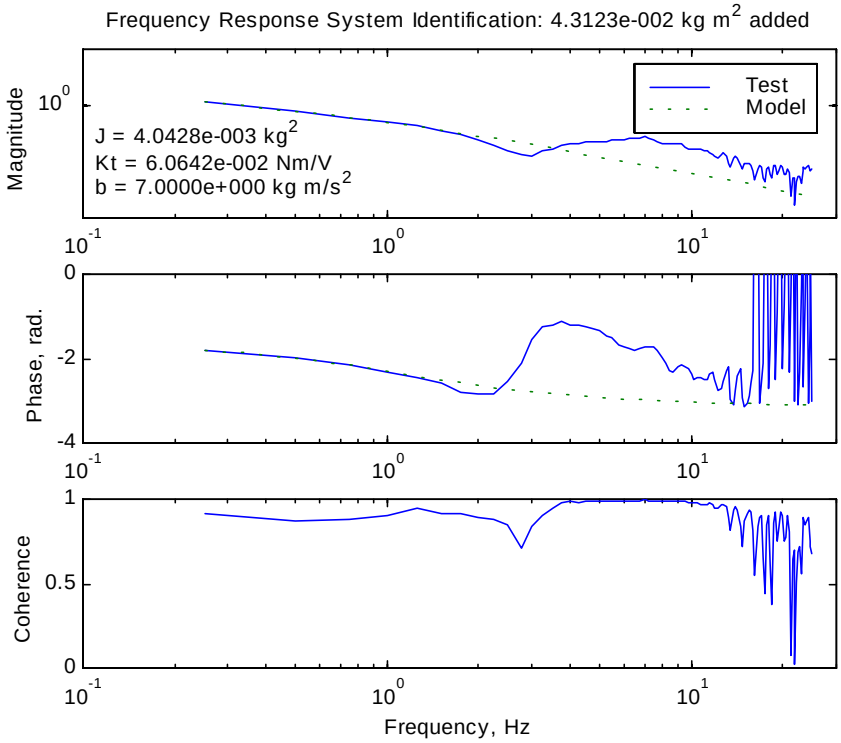

Figure 4.6: Motor Frequency Response 0.0431 kg m2 added

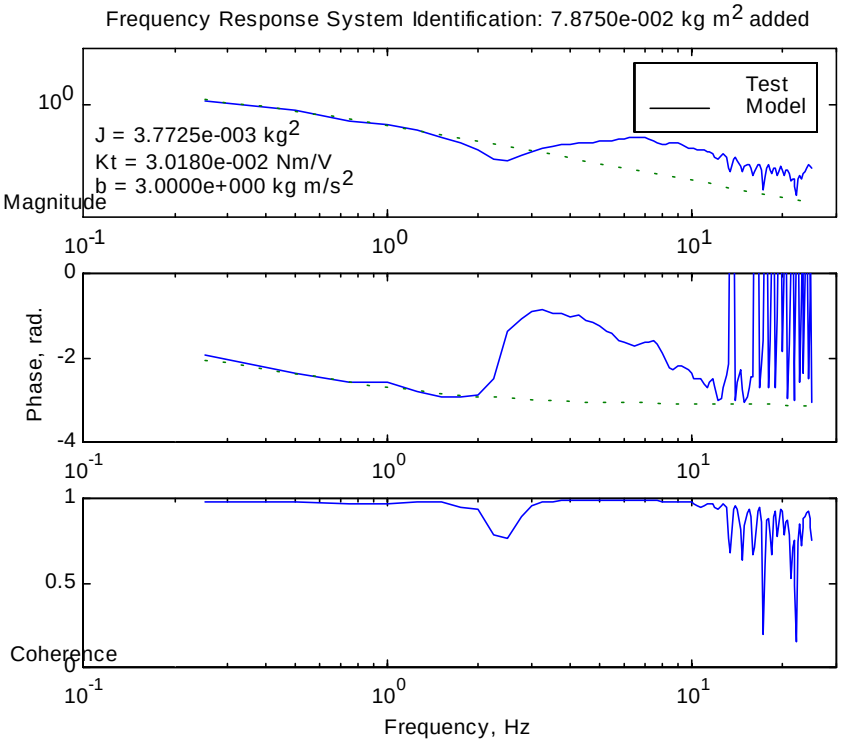

Figure 4.7: Motor Frequency Response with 0.078 kg m2 added.

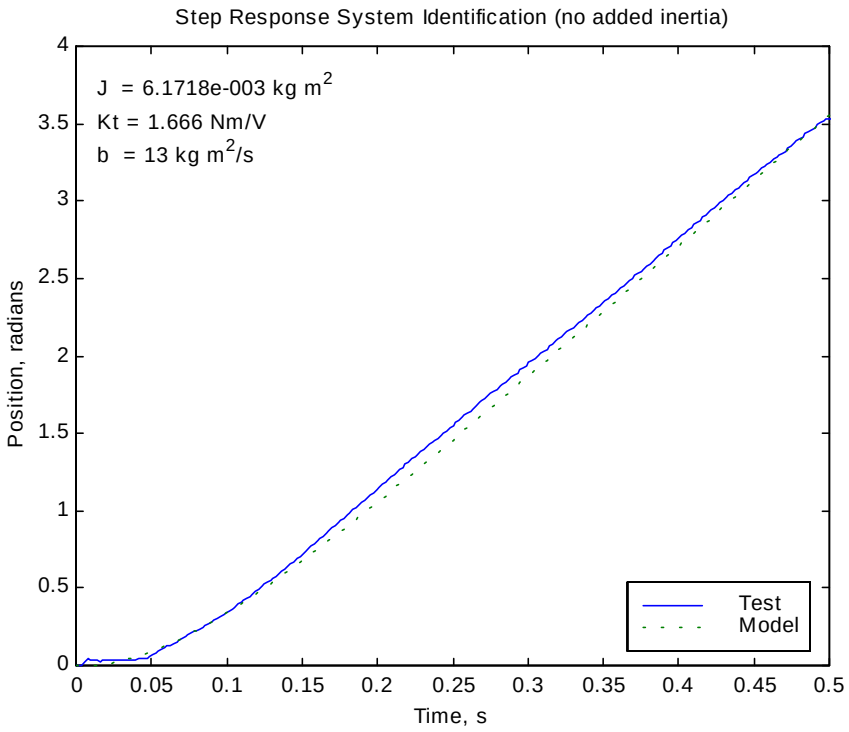

Figure 4.8: Motor Step Response with no added inertia

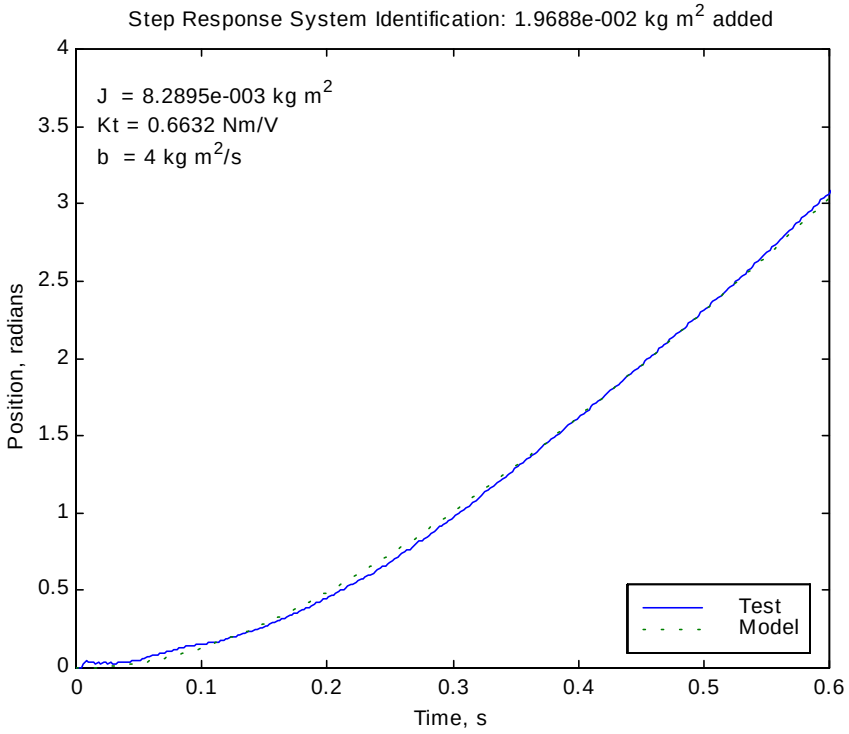

Figure 4.9: Motor Step Response with 0.0197 kg m2 added

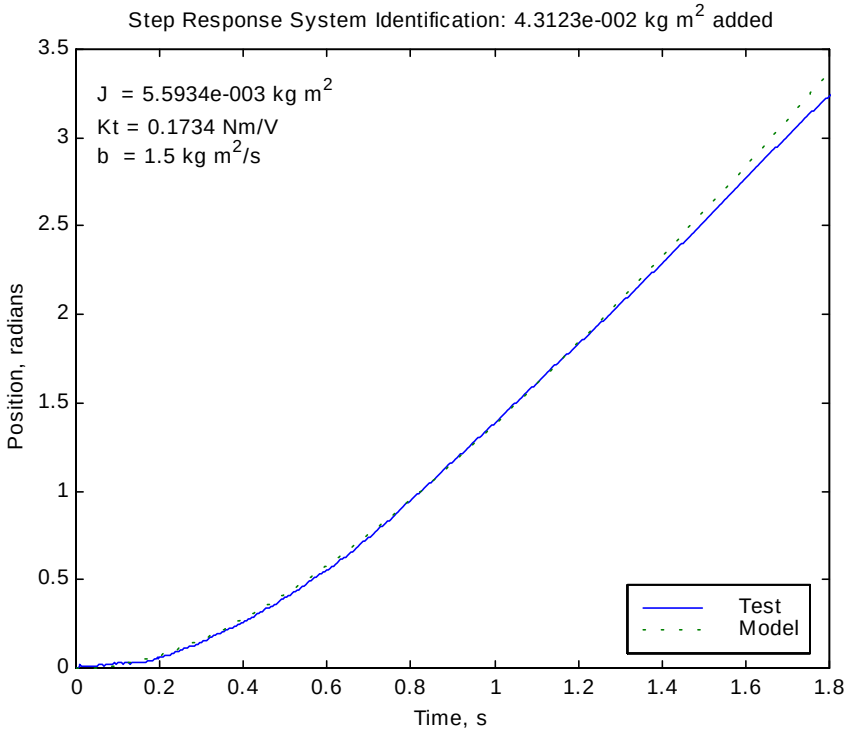

Figure 4.10: Motor Step Response with 0.0431 kg m2 added

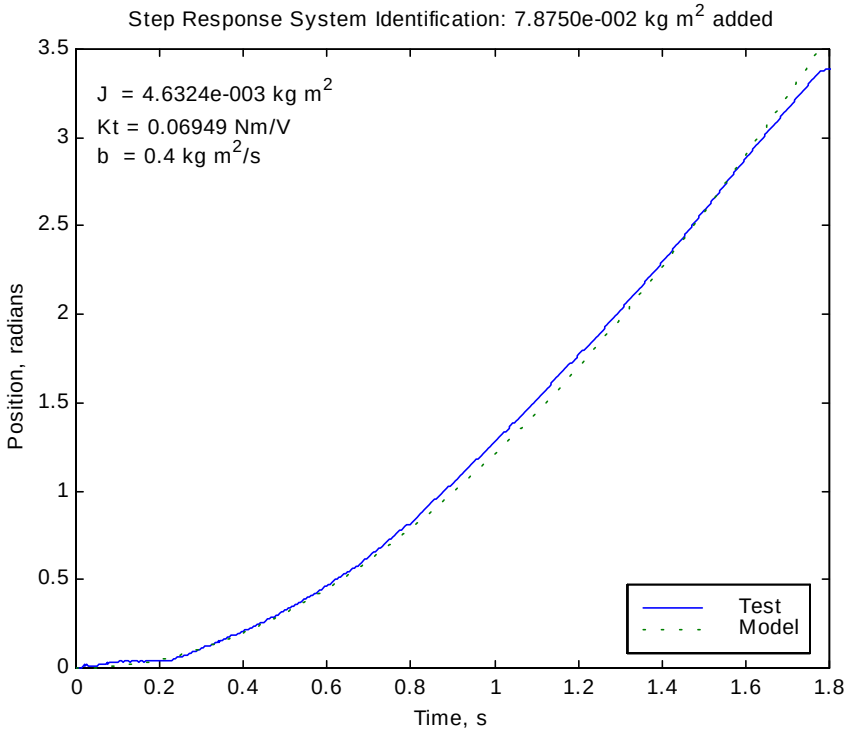

Figure 4.11: Motor Step Response 0.0788 kg m2 added

of magnitude found by two of the three tests were used. These results are detailed in the following sections.

The DC motor model used in Horner's thesis required the torque and damping constants [1]. These constants come from the transfer function of torque as a function of input voltage:

$$\frac{T}{V} = \frac{K_m}{s(s+b)}, \tag{4.1}$$

where T is torque, in N-m, V is voltage, in V, K_m is the motor's torque constant, in N-m/V, b is the motor's damping constant, in N-m/s, and s is the Laplace variable.

This transfer function could not be measured directly, so the transfer function of position, as a function of input voltage was measured instead. This transfer function for a DC motor is:

$$\frac{\theta}{V} = \frac{\frac{K_T}{R_A J}}{s\left(s + \frac{\left(R_A F_V + K_T K_B\right)}{R_A J}\right)}, \tag{4.2}$$

and is developed by Franklin and Powell in [19]. In Equation (4.2), θ is the motor's position in radians, V is the voltage applied to the motor in V, J is the inertia of the motor and attached mass in kg-m^2, K_T is the torque constant in N m/A, R_A is the armature resistance in Ω, F_V is the motor viscous damping in N m/s, and K_B is the back EMF constant in V/radian/s.

Torque is equal to the angular acceleration multiplied by the moment of inertia; therefore, the desired transfer function can be achieved by multiplying this transfer function by Js^2. Using this information, and simplifying Equation (4.2) to be in terms of K_m and b, the position transfer function becomes:

$$\frac{\theta}{V} = \frac{K}{s(s+b)}. \tag{4.3}$$

4.2.2.1 Frequency Response Matching

The first method used for determining the motor constant servomotors was matching the motor frequency response under different inertial loading conditions, to a simplified version of Equation (4.2). The frequency response of the motor/amplifier

system was collected using an HP35665A Signal Analyzer. The input signal for the frequency response tests was a periodic chirp with amplitude of 1 volt and frequencies spanning from 0 to 100 Hz. Using a Hanning window, the data were collected over a four-second sample period, and were averaged over ten tests. Tests were run with four different moment of inertia loads: no added inertia, and 0.0197, 0.0431, and 0.0788 kg-m^2 added. Once the data was collected, the simplified motor model of Equation (4.3) was matched to each data set by adjusting K and b. After adjusting the model to match the data, the inertia of the motor was calculated using the assumption that K_m is constant for different moments of inertia, and that:

$$K = \frac{K_m}{J_{total}}, \tag{4.4}$$

where J_{total} is the motor inertia plus the added inertia.

These collected frequency responses did not match what was expected from the DC motor model. Without added inertia, the frequency response of the motor gave a frequency response close to the expected model. With added inertia, the frequency responses had large, unexpected changes in both magnitude and phase, as shown in Figures 4.4–4.7. The coherence approached zero at the same frequency that magnitude and phase deviated from the expected model. The frequency at which the dip occurred decreased as the inertia increased. Further tests were run to eliminate the dips in coherence, but improving the quality of the data was not possible. Another method to determine the motor constants was needed.

4.2.2.2 Parameter Measurement

The second method used for determining the motor constants was measuring the parameters that make Equation (4.2). For this method, F_V was assumed to be much less than one, and therefore negligible. K_B was measured by driving the motor at a constant speed and measuring the voltage generated across the leads. R_A was measured as the resistance across the motor leads. J was estimated by calculating the inertia of a cylinder with the size and mass of the motor. Literature detailing the necessary test for measuring K_T was not available; therefore, K_T was estimated with a static test, measuring the necessary current for holding a constant torque stationary. The motor constants found

using this method were small, and resulted in large initial gains, which cause the system to be unstable.

4.2.2.3 Step Response Matching

The final method used for determining the motor constants was matching the motor model to the position step response of the motor. The step response of the motor was measured for input voltages of 0.25, 0.5, and 1 volt with added moments of inertia of 0.0197, 0.0431, and 0.0788 kg-m^2, as shown in Figures 4.8–4.11. By adjusting K and b, the step response of the system described by Equation (4.2) was matched to this data. The moment of inertia for the motor was then found by approximating it as a uniform cylinder, then K_m was found using Equation (4.4). For a given inertia, the values of K_m and b were similar between tests, but these constants changed significantly with inertia.

4.2.2.4 Motor Constant Results

Each testing method resulted in different values for K_m and b; however, the frequency method and the step response method gave constants with the same order of magnitude. Why the constants varied so much between tests is unknown, but the motor non-linearities are a likely explanation. Each test examined the motor's performance under different operating conditions, which may have caused the motor to perform differently. This seems likely because the motors were designed for remote control vehicles, where linearity was not a necessity. Actuator non-linearity is included in the controller design described in Section 3.2, so the controller should be able to compensate for this non-linearity. Another possibility for the variation of the motor constants between tests is that compliance in the added inertia added an additional pole to the system that was not accounted for in the system model. Adding a term for flexibility in the inertial load may have improved model matching.

Values for K_m and b were selected to be consistent with the frequency and step response methods. Selecting values for these constants made calculating initial gain possible. The initial gains are not critical to the performance of the controller because Horner showed that the controller performs well even if the initial gains are both selected to be zero.

Table 4.3 shows the ranges found for each of the testing methods [1].

Table 4.3: Damping and Motor Coefficients

Description	Frequency Response	Parameter Measurement	Step Response
Link damping (kg-m^2/s)	3 to 25	5.1 x 10^{-9}	0.4 to 13
Link motor constant (N-m/V)	0.03 to 0.7	2.4 x 10^{-5}	0.07 to 1.7

4.3 Determination of Controller Values

After all of the system parameters were defined, the values of **K**, $\mathbf{K}_r$, and $\mathbf{C}_e$ needed to be calculated. These calculations required the values of **A** and **B** for each of the subsystems, which were calculated using the following equations defined by Horner [1]:

$$\mathbf{A}_1 = \begin{bmatrix} 0 & 1 \\ 0 & -\frac{c_1}{\tilde{J}_{\theta 1}} \end{bmatrix},\ B_1 = \begin{bmatrix} 0 \\ \frac{k_1}{\tilde{J}_{\theta 1}} \end{bmatrix} \tag{4.5}$$

$$\mathbf{A}_2 = \begin{bmatrix} 0 & 1 \\ 0 & -c_2\left(\frac{1}{J_{\theta 2}} + \frac{1}{\tilde{J}_{\theta 1}}\right) \end{bmatrix},\ B_2 = \begin{bmatrix} 0 \\ k_2\left(\frac{1}{J_{\theta 2}} + \frac{1}{\tilde{J}_{\theta 1}}\right) \end{bmatrix} \tag{4.6}$$

$$\mathbf{A}_3 = \begin{bmatrix} 0 & 1 \\ 0 & -\frac{c_3}{J_3} \end{bmatrix},\ B_3 = \begin{bmatrix} 0 \\ \frac{k_3}{J_3} \end{bmatrix} \tag{4.7}$$

$$\mathbf{A}_4 = \begin{bmatrix} 0 & 1 \\ 0 & -\frac{c_4}{m_T} \end{bmatrix},\ B_4 = \begin{bmatrix} 0 \\ \frac{k_4}{m_T} \end{bmatrix}, \tag{4.8}$$

where:

$$\tilde{J}_{\theta 1} = J_{\theta 1} + m_2 L_1^2,$$

$$J_{\theta 1} = J_1 + m_1 l_1^2$$

$$J_{\theta 2} = J_2 + m_2 l_2^2, \text{ and}$$

where the above constants and parameters are as defined in Tables 4.1– 4.3.

Using these values for **A** and **B**, the equations derived in Section 3.2, the digital representation of the continuous system:

$$\mathbf{A}_m = \begin{bmatrix} 0 & 1 \\ -4 & -4 \end{bmatrix}, \mathbf{B}_m = \begin{bmatrix} 0 \\ 4 \end{bmatrix}, \tag{4.9}$$

as the reference model, and a sample time of 0.01 seconds, and

$$\mathbf{Q} = \begin{bmatrix} 2 & 0 \\ 0 & 0.2 \end{bmatrix} \tag{4.10}$$

for each link, the error weighting matrix and the initial gains for each of the joints were found to be:

$$\mathbf{K}_{rd1} = 0.71388, \tag{4.11a}$$
$$\mathbf{K}_{d1} = [0.713883 \quad -17.4297], \tag{4.11b}$$
$$\mathbf{C}_{e1} = [7.8518 \quad 2.7481], \tag{4.11c}$$

$$\mathbf{K}_{rd2} = 0.71148, \tag{4.12a}$$
$$\mathbf{K}_{d1} = [0.711483 \quad -174321], \tag{4.12b}$$
$$\mathbf{C}_{e1} = [23.2202 \quad 8.12705], \tag{4.12c}$$

$$\mathbf{K}_{rd3} = 0.71155, \tag{4.13a}$$
$$\mathbf{K}_{d1} = [0.711545 \quad -17.432], \tag{4.13b}$$
$$\mathbf{C}_{e1} = [12.9664 \quad 4.53823], \tag{4.13c}$$

$$\mathbf{K}_{rd4} = 74.3354, \tag{4.14a}$$
$$\mathbf{K}_{d1} = [74.3354 \quad 56.5625], \text{ and} \tag{4.14b}$$
$$\mathbf{C}_{e1} = [0.01325 \quad 0.0046374]. \tag{4.14c}$$

4.4 dSpace Controller Setup

As the final step in preparing for tests, the model reference adaptive controller was implemented on the Autobox in dSpace. The block diagram performed the following functions: scaled the input and output for the Autobox, filtered the input signal, converted the input signal from volts into the appropriate units, calculated the velocity for the angular position, generated the control signal, and took control signal saturation into account. The dSpace Cockpit application was used to adjust various parameters for the adaptive controller, and the Trace application was used to save data.

The highest level of the block diagram was the input and output of the link controllers, and performed the filtering and saturation for the link inputs and outputs, as

depicted in Figure 4.12. The gain blocks immediately after the input and immediately before the outputs were required so that the input and output signals recorded by the Autobox matched the actual signals. After the input voltages were properly scaled, they were converted from voltage into the units used by the controller: meters for the base position, meters per second for the base velocity, and radians for the joint positions. This linear conversion consisted of an offset and a slope variable. Next, the angular velocity of the joint was calculated using the derivative block, so that the controller could use it. Each link of the robot had its own decentralized model reference adaptive controller with inputs of position, velocity, and step trigger. The values of the step trigger reinitialized the controller and signaled a running test. When the value was on, a test was running, and when the value was zero, a test was not running. After the signal left the control block, it entered a saturation block. This block saturated the control signal at ±10 volts, matching the saturation of the Autobox.

The next lowest level of the controller was the decentralized model reference adaptive controller in Simulink, shown in Figure 4.13. The link position, velocity, and step trigger were input, while the control signal was output from the link controller. The controller was the block diagram implementation of the digital adaptive controller described in Section 3.4. The response of the link was supposed to match the response of the reference model mentioned in the previous section.

The block values highlighted in blue are the adaptive controller variables that were adjusted using dSpace Cockpit. The on/off block was used to isolate joints while testing. When the block was set to off, the control signal from the block was set to zero.

dSpace modules, Cockpit and Trace, simplified the adjustment of the adaptive controller as well as recorded variables from within the controller. The variables that could be adjusted for each link were: $\mathbf{C}_e$, $\mathbf{K}$, $\mathbf{K}_r$, β, σ, the reference angle, and the calibration variables. The Cockpit window also allowed each link and the step-input to be turned on and off.

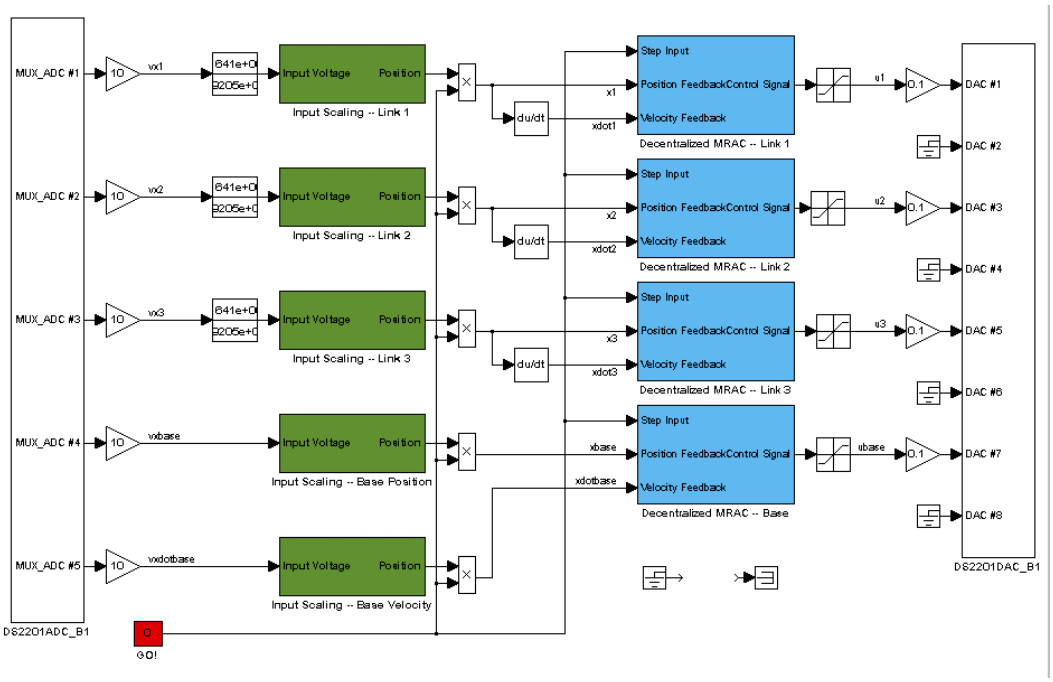

Figure 4.12: Top Level Controller Block Diagram in dSpace/Simulink

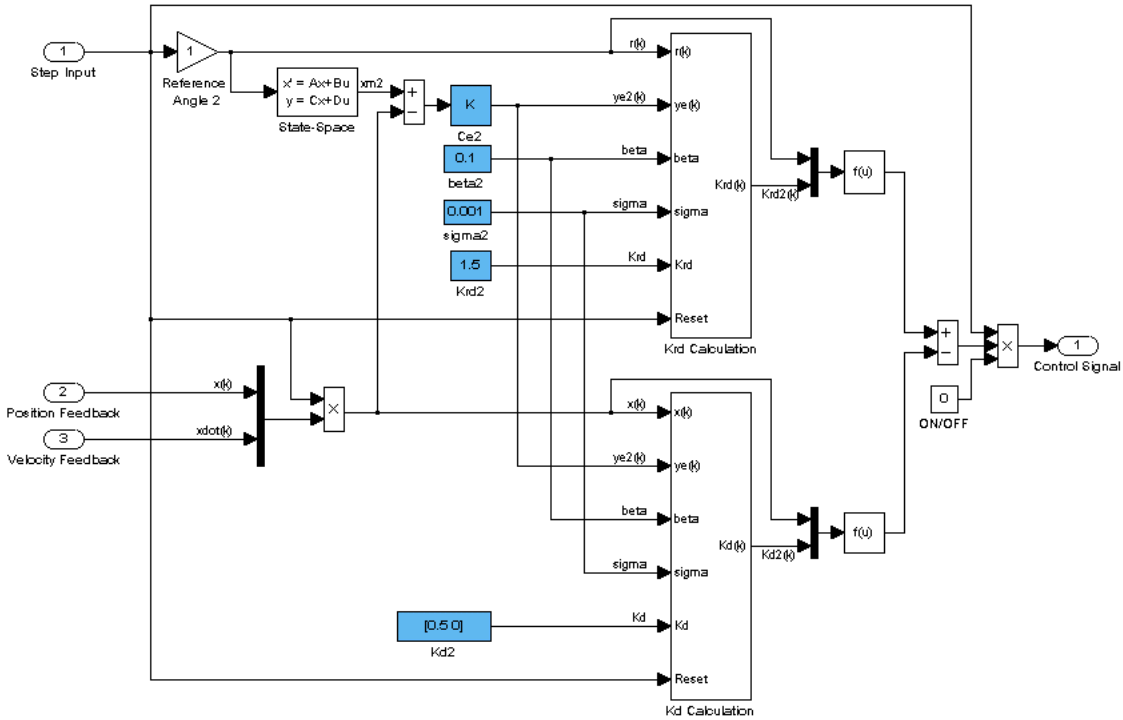

Figure 4.13: Simulink Block Diagram of Model Reference Adaptive Controller

Chapter 5

Experimental Results

This chapter outlines the tests that were performed to evaluate the performance of a decentralized model reference adaptive controller on a mobile robot. The response was unstable when the robot was first run with the state space controller using the gains that were calculated using the theory described in Section 3.1. Because it was unclear at the time why the system was unstable, tests were run beginning with the least complex system, then increasing the complexity of the system.

5.1 Constant Gain Controller without Robot in the Loop

The first tests were performed to determine stable ranges of gains for the state space controller. These tests had only the state-space controller in the loop, and they helped to resolve issues about how the Autobox scaled the input and output signals.

For these tests, the Autobox outputs were connected directly to its inputs, the learning constants were set to 0 (zero) so that the controller was constant-gain, and the step response was examined. These tests confirmed that the Autobox was setup correctly and that the constant-gain controller was working properly. These tests also showed that the Autobox scales its inputs and output signals by a factor of 10. Scaling blocks were included to the block diagram in Figure 4.12 so the actual output matched the expected output.

Figure 5.1 shows the position response of the Autobox as the position gain increased. As the position gain increased, the magnitude of the oscillations increased, and when the gain is increased beyond 1.0, then the response becomes unstable. Figure 5.2 shows the position response of the Autobox as the delayed position gain increased. Again, the magnitude of the response increased as the gain increased, and when the gain approached 1.0, then the response becomes unstable. These responses make sense because the input for the next iteration is the same as the output of the current iteration, and if at each step the output is increased, then the response will become unstable.

Figure 5.2 shows the position response, as the delayed position gain was made more negative. This added damping to the system; reducing the overshoot, and increasing the rise time. However, if the gain is increased beyond -0.7, then the response becomes unstable.

These tests showed that the Autobox, and the constant-gain controller were working properly, that the position gain for the Autobox only system must be between 0.0 and 1.0, and that the delayed position gain must be between –0.7 and 1.0.

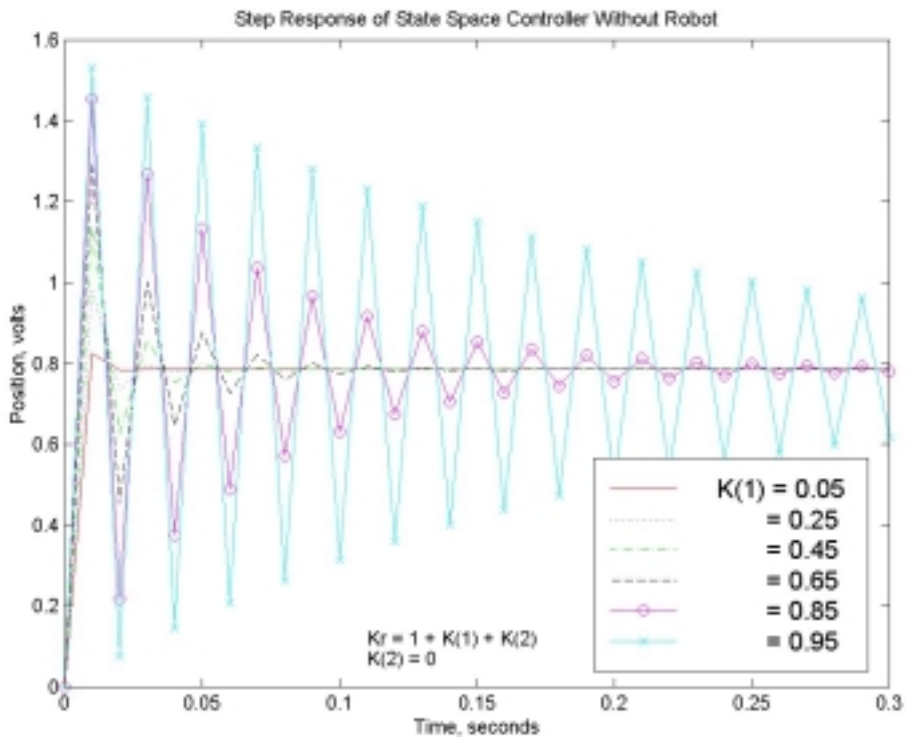

Figure 5.1: Linear Constant Gain Controller Tests Without Robot, Varying $\mathbf{K}_1$

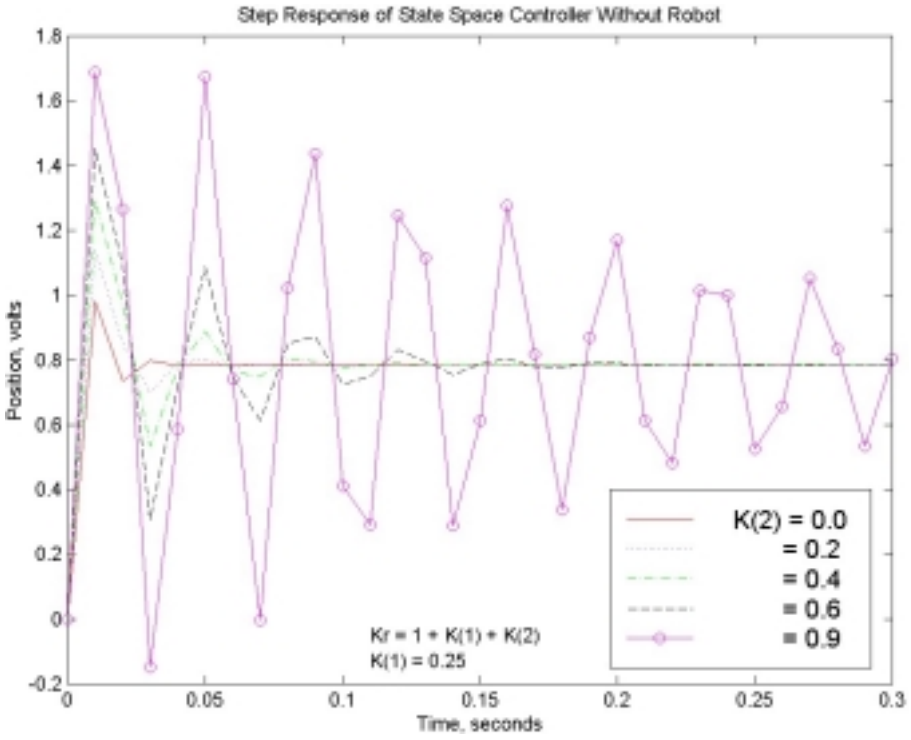

Figure 5.2: Linear Constant Gain Controller Tests Without Robot, Varying $\mathbf{K}_2$

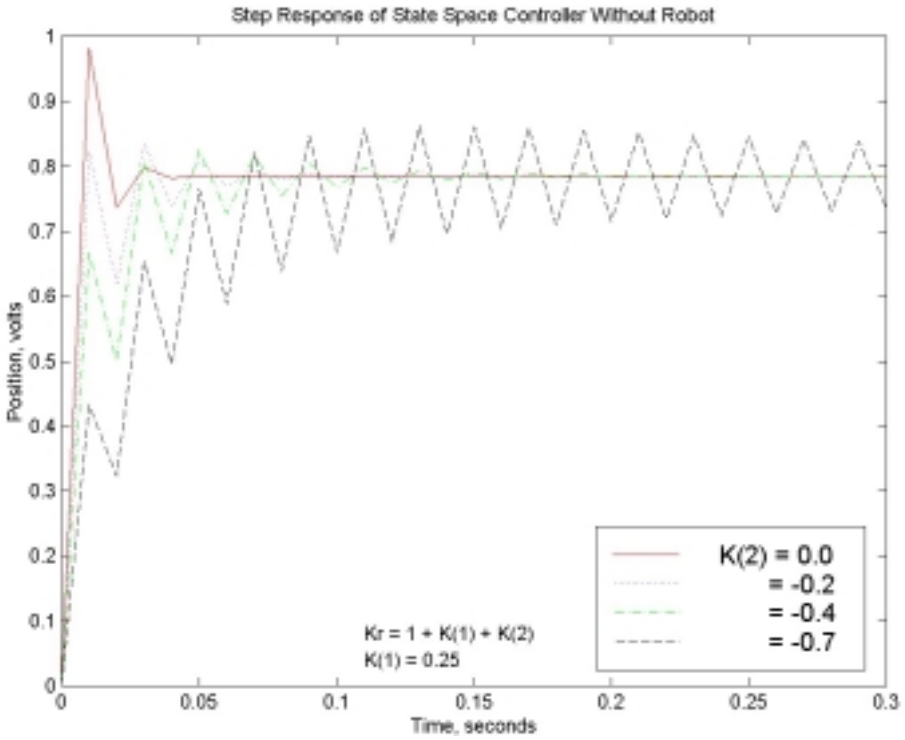

Figure 5.3: Linear Constant Gain Controller Tests Without Robot, Varying $\mathbf{K}_1$ Negative

5.2 Constant Gain Controller with Robot in the Loop

A second set of tests was run using the state-space controller on a single joint of the robot. These tests were performed to calibrate the sensors, to verify that the robot was operational, and to determine stable initial gains for the adaptive controller.

To calibrate the joint position sensors, each link was attached to the Autobox and moved through its entire range of motion. The voltage range was recorded, and an appropriate scaling equation was found for each joint.

The initial gains calculated in Section 4.3 were tested and found to make the system unstable, causing significant damage to the robot. After the robot was repaired, various values for $\mathbf{K}$ were tested. At first, the velocity gain, $\mathbf{K}_2$, was set to zero, and the position gain, $\mathbf{K}_1$, was increased from zero until the response became unstable. The results of these tests can be seen in Figures 5.4 – 5.6. The position gain was then set to 0.25, and the velocity gain was increased from zero until the response became unstable. The results from these tests can be seen in Figures 5.7-5.9.

These tests demonstrated the correct operation of the adaptive controller, and found that the initial gains calculated in Section 4.3 were unstable. A suitable stable gain of:

$$\mathbf{K} = \begin{bmatrix} 0.25 & 0 \end{bmatrix}$$

was selected as the initial gain for the adaptive controller.

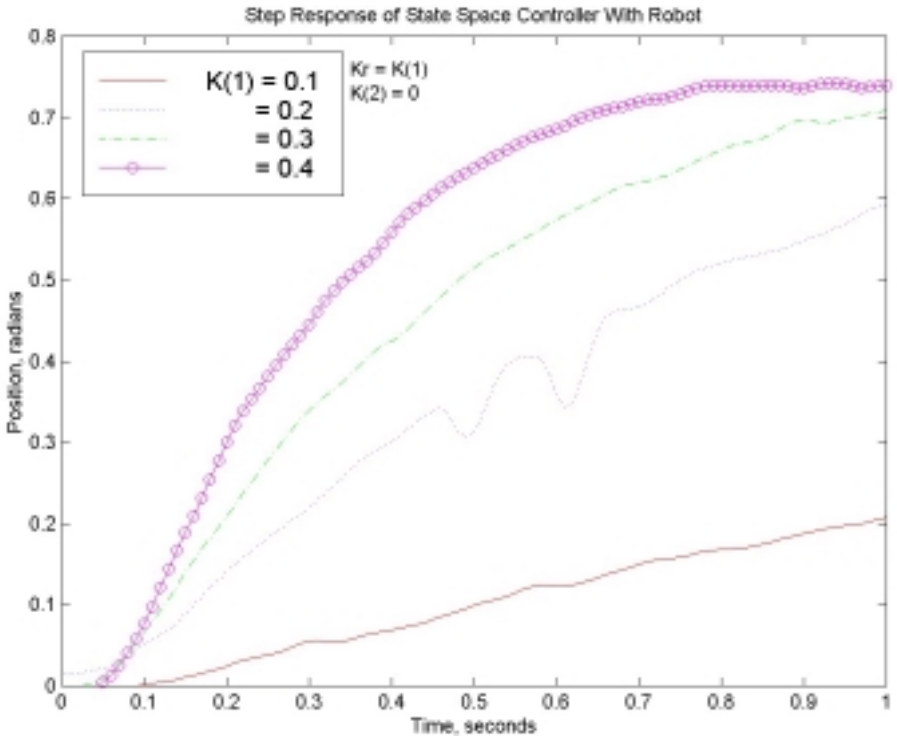

Figure 5.4: Linear Constant Gain Controller Tests With Robot, Varying $\mathbf{K}_1$: Position

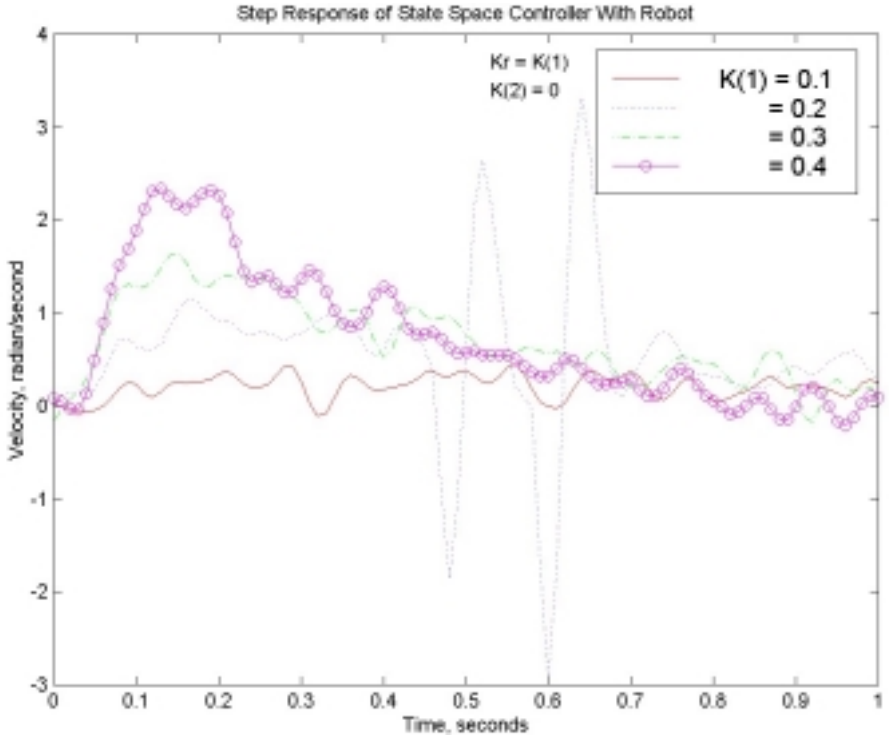

Figure 5.5: Linear Constant Gain Controller Tests With Robot, Varying $\mathbf{K}_1$: Velocity

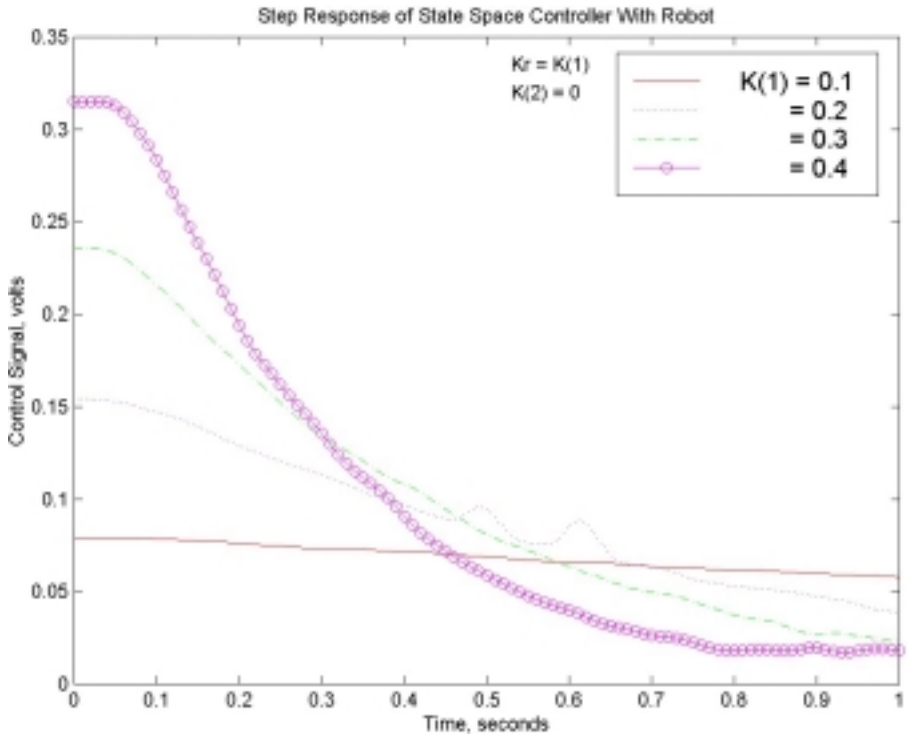

Figure 5.6: Linear Constant Gain Controller Tests With Robot, Varying $\mathbf{K}_1$: Control Signal

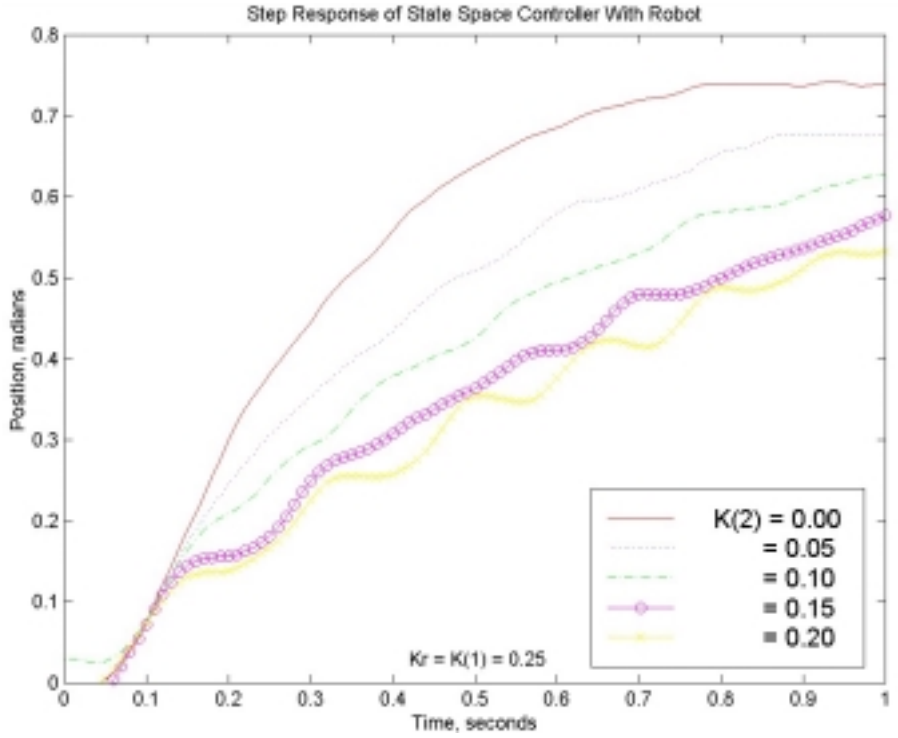

Figure 5.7: Linear Constant Gain Controller Tests With Robot, Varying $\mathbf{K}_2$: Position

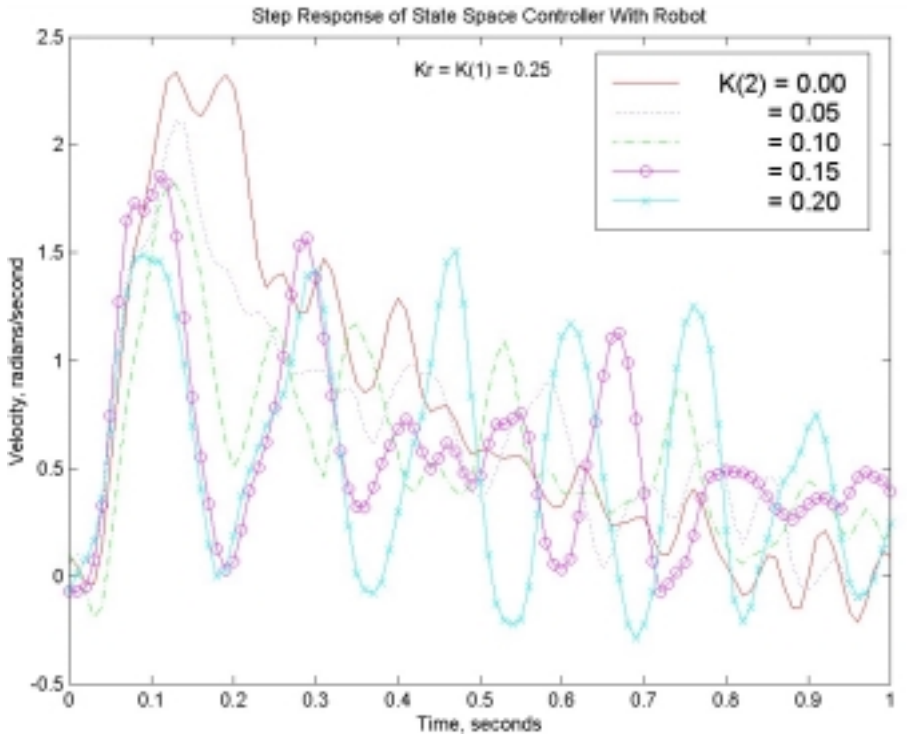

Figure 5.8: Linear Constant Gain Controller Tests With Robot, Varying $\mathbf{K}_2$: Velocity

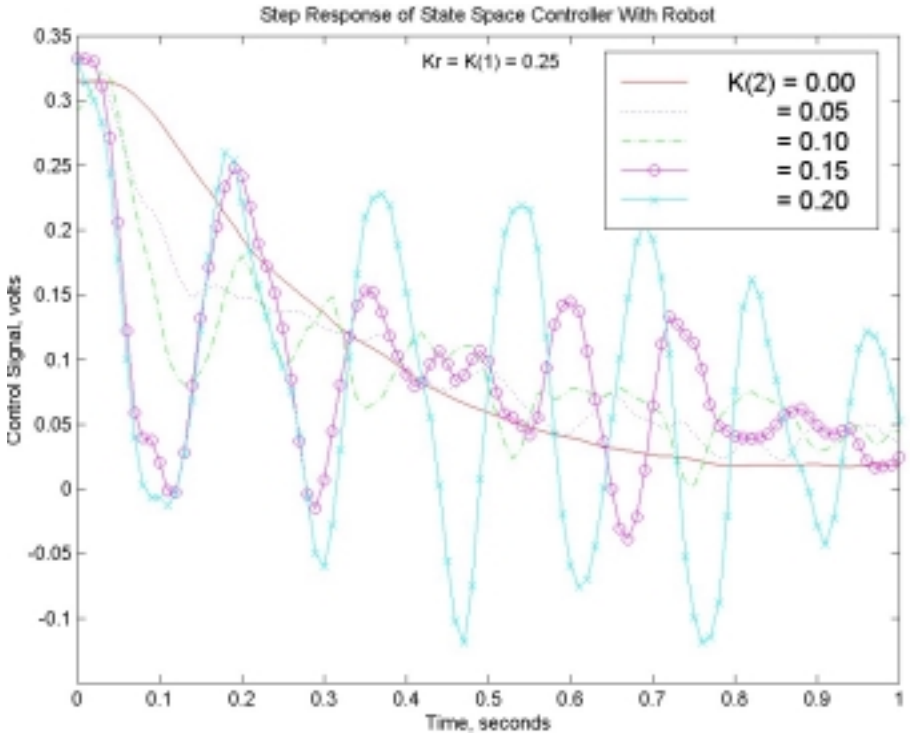

Figure 5.9: Linear Constant Gain Controller Tests With Robot, Varying $\mathbf{K}_2$: Control Signal

5.3 Decentralized MRAC without Robot in the Loop

The next set of tests was designed to verify that the model reference adaptive controller was operating correctly. Again, the controller was tested without the robot in the loop, which simplified the process of debugging the adaptive controller.

The correct operation of the adaptive controller was validated by comparing the response of the Autobox controller to the response of a Simulink simulation of the test, and to a MATLAB M-File of the test.

The simulation of the decentralized model reference adaptive controller system almost exactly matched the response of the actual system, as shown in Figures 5.10-5.12. Figures 5.13-5.17 show a β/σ of 100 does not provide good model following early in the step response if the initial gains are non-zero; however, if the initial gains are set to zero, then a β/σ of 100 works well as long as β is sufficiently large, as can be seen in Figures 5.18-5.22. It should also be noted that by making β too large, the system will become unstable.

Finally, the effect of varying $\mathbf{C}_e$ was examined, as shown in Figures 5.23-5.32. First $\mathbf{C}_{e1}$ was varied while $\mathbf{C}_{e2}$ was constant at 0.5, shown in Figures 5.23-5.27, then $\mathbf{C}_{e2}$ was varied while $\mathbf{C}_{e1}$ was held constant at 1.5, as shown in Figures 5.28-5.32. Increasing $\mathbf{C}_{e1}$ improved the ability of the system to match the model, but no values of $\mathbf{C}_{e1}$ were able to make the system follow the model through the transient part of the response. Increasing $\mathbf{C}_{e2}$ improved the ability of the system to match the model slightly, but quickly led to oscillations in the response and instability.

These tests confirmed that the adaptive controller was working properly, allowed the selection of a β/σ of 100, showed that initial gains of zeros sometimes provide better performance, and gave some indication of how $\mathbf{C}_e$ affects system performance.

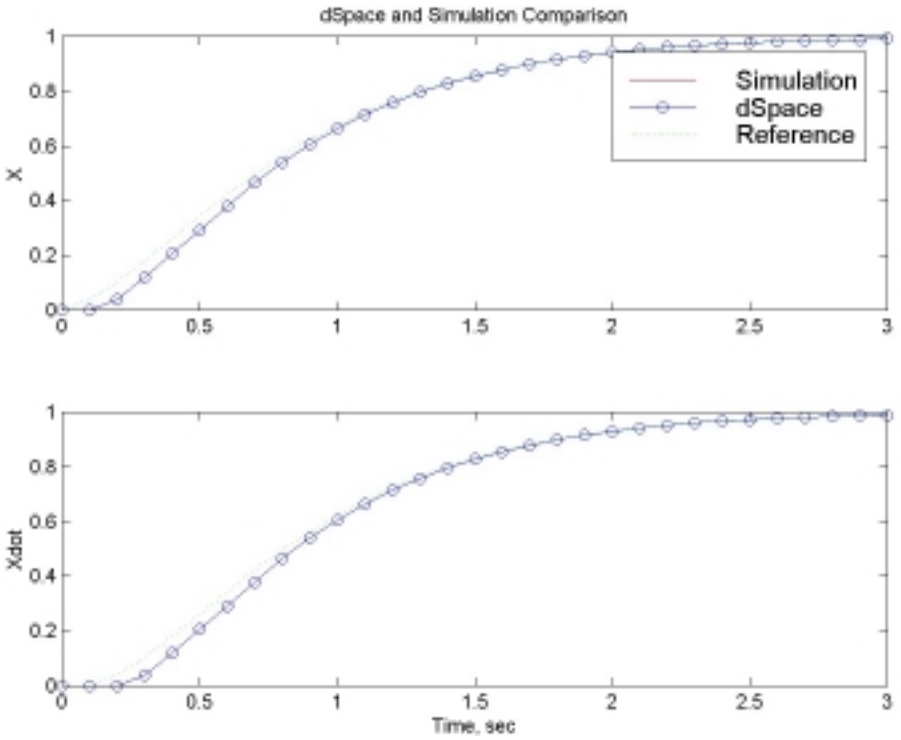

Figure 5.10: Sample comparison of Autobox Only DMRAC response and MATLAB Simulation: Position

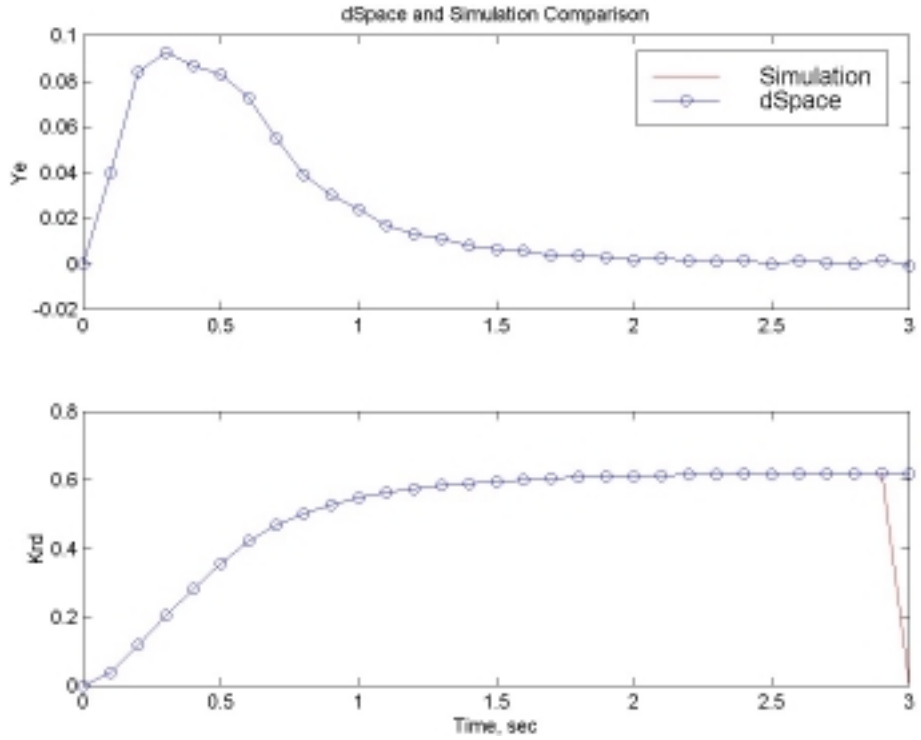

Figure 5.11: Sample comparison of Autobox Only DMRAC response and MATLAB Simulation: Scaled Error

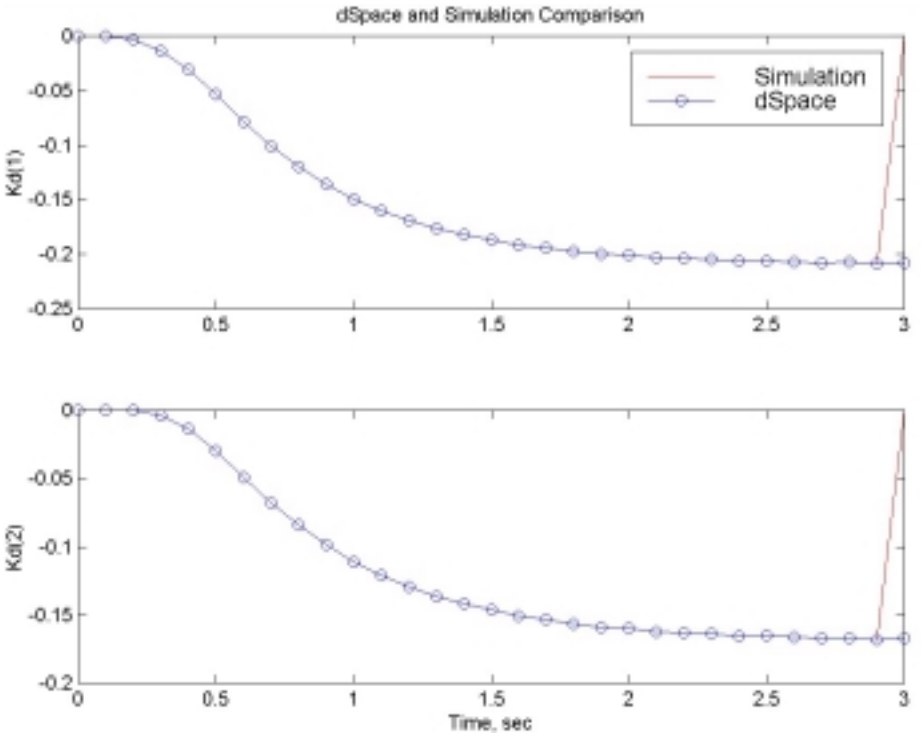

Figure 5.12: Sample comparison of Autobox Only DMRAC response and MATLAB Simulation: **K**

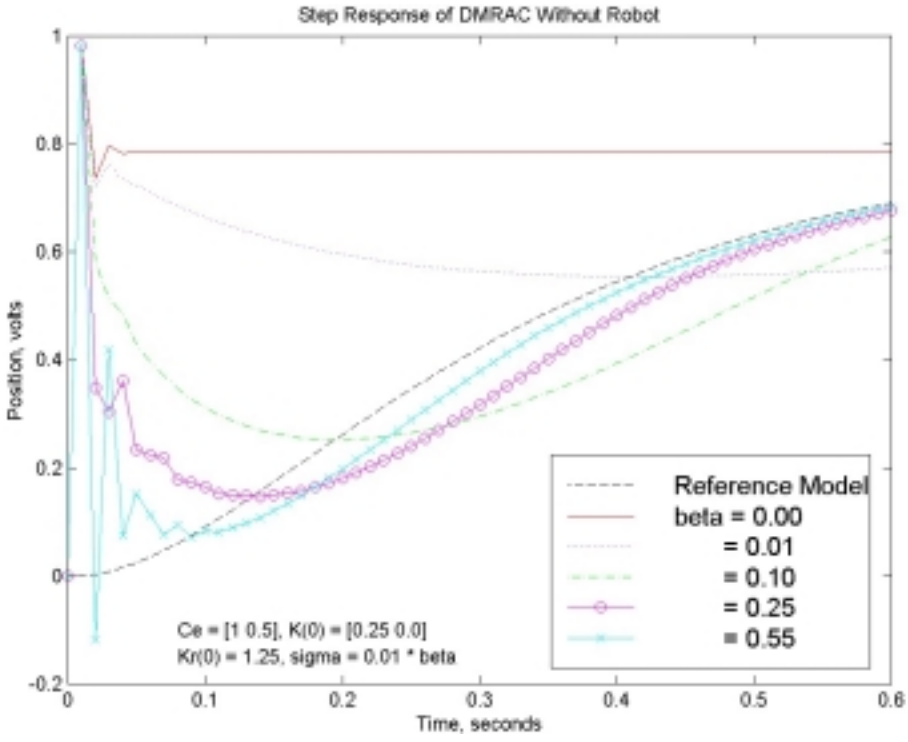

Figure 5.13: Decentralized Model Reference Adaptive Controller Tests Without Robot, Varying β: Position

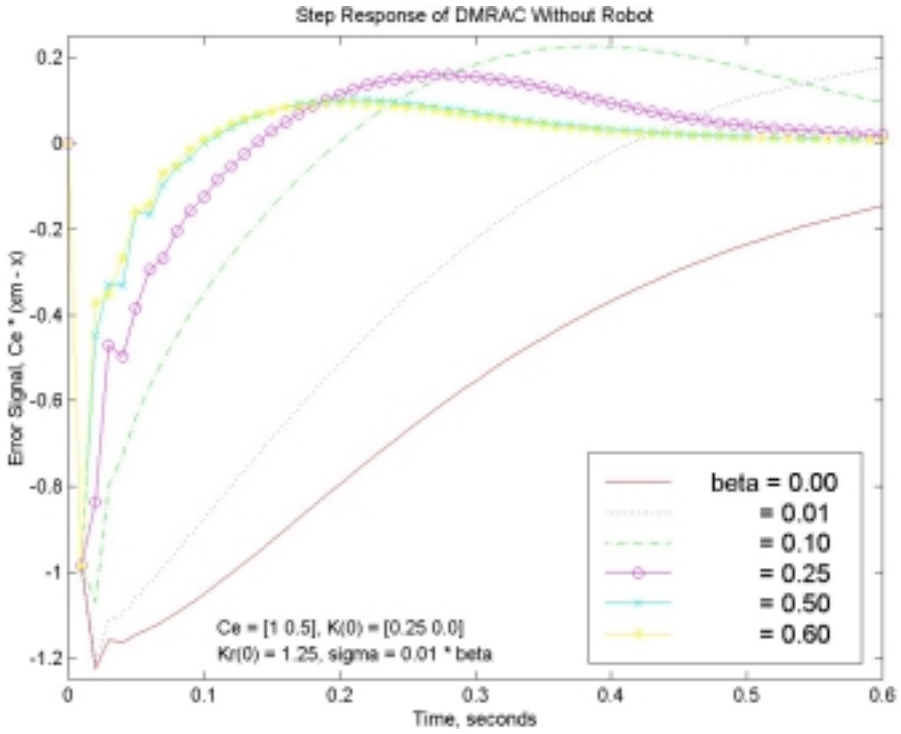

Figure 5.14: Decentralized Model Reference Adaptive Controller Tests Without Robot, Varying β: Error Signal

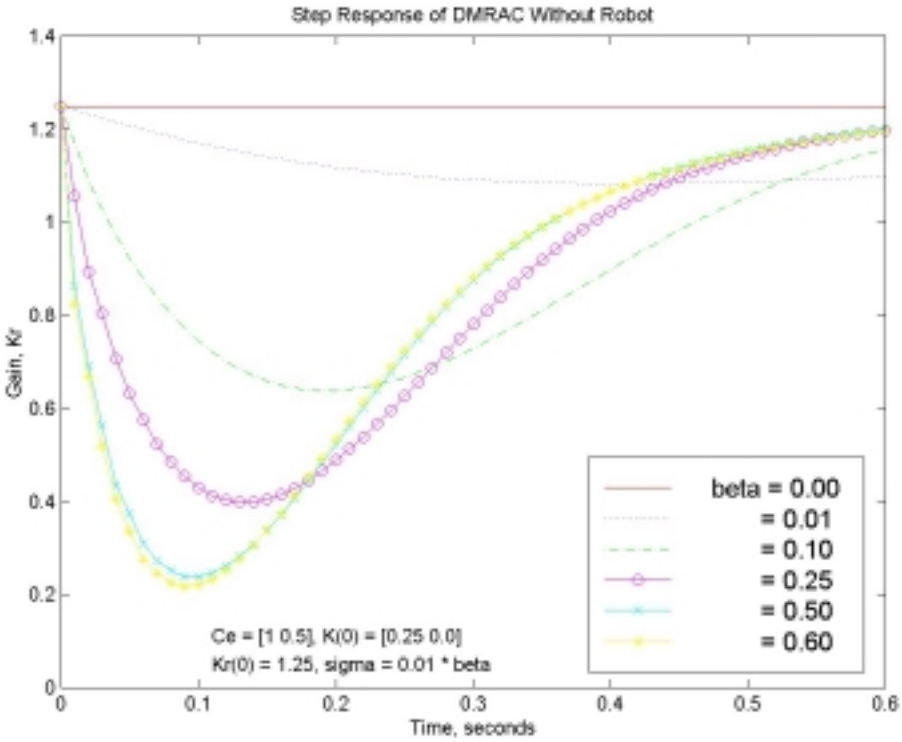

Figure 5.15: Decentralized Model Reference Adaptive Controller Tests Without Robot, Varying β: $\mathbf{K}_r$

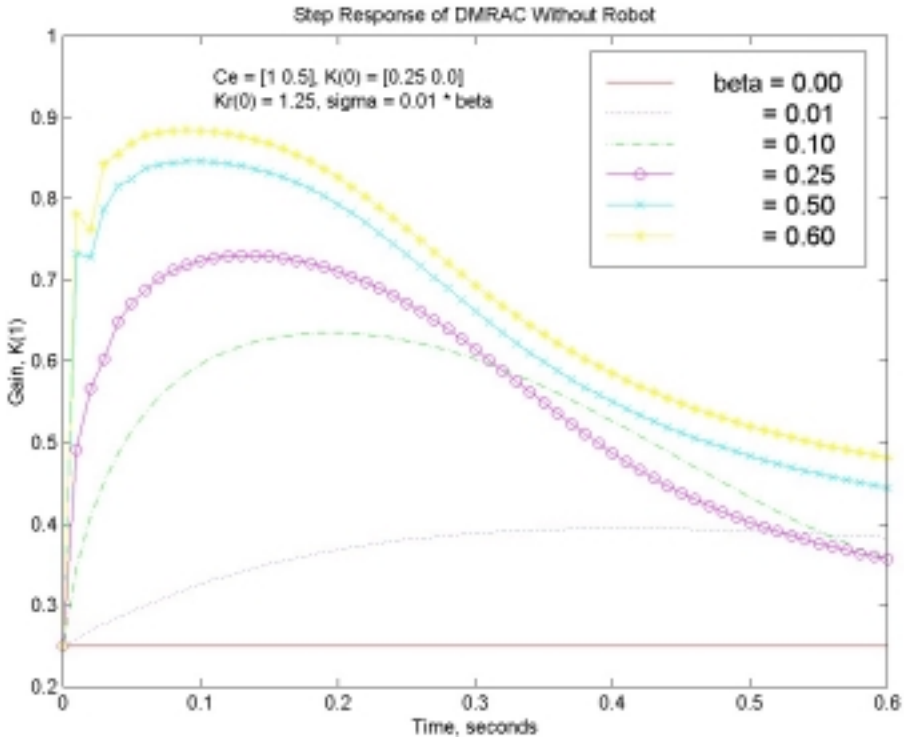

Figure 5.16: Decentralized Model Reference Adaptive Controller Tests Without Robot, Varying β: $\mathbf{K}_1$

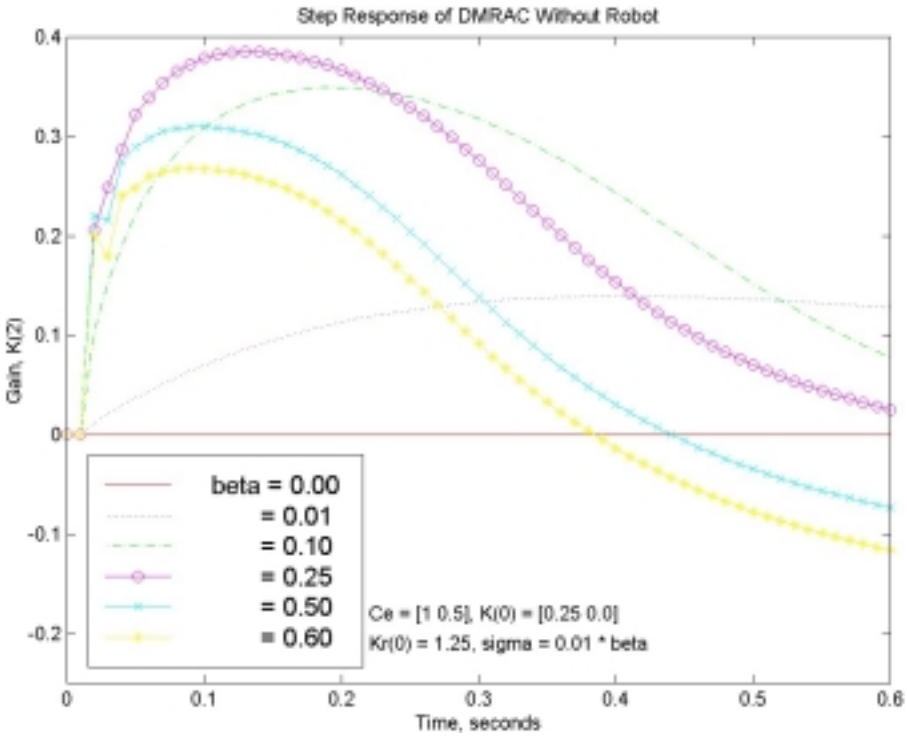

Figure 5.17: Decentralized Model Reference Adaptive Controller Tests Without Robot, Varying β: $\mathbf{K}_2$

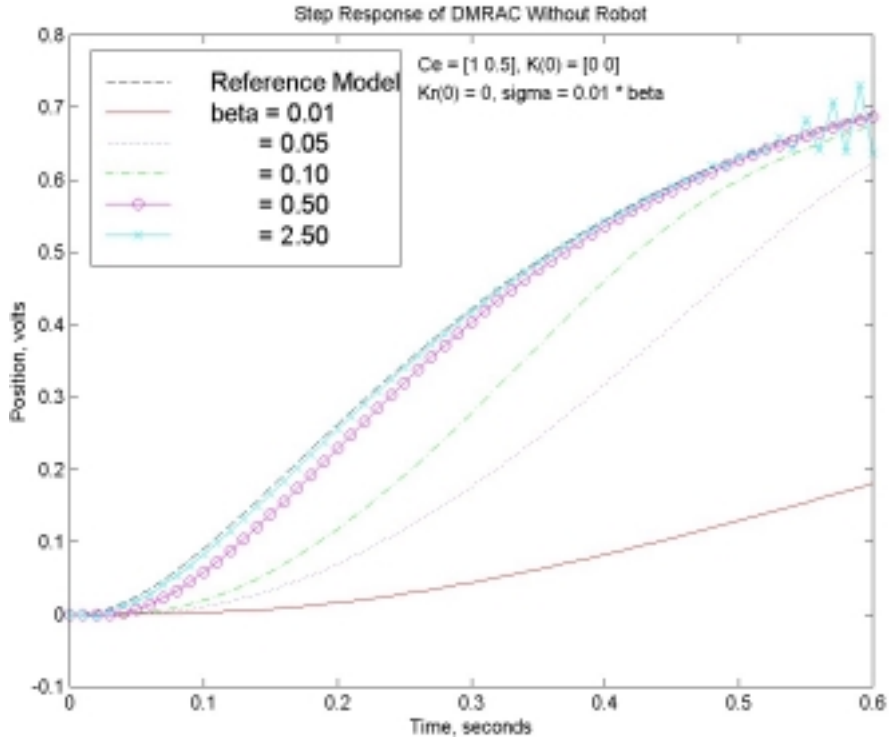

Figure 5.18: Decentralized Model Reference Adaptive Controller Tests Without Robot, Varying β: Position

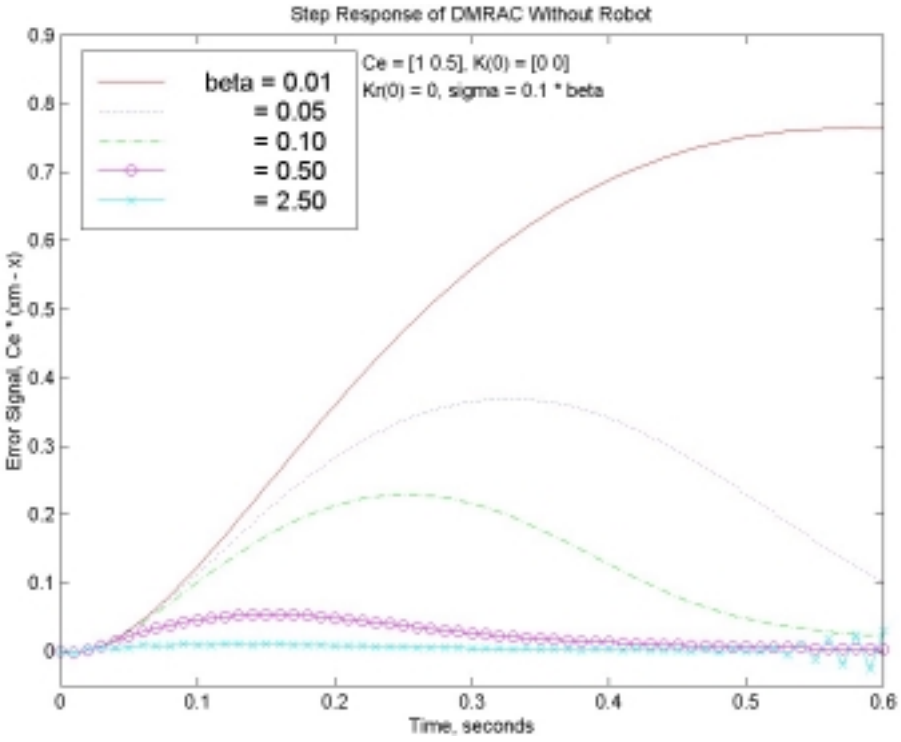

Figure 5.19: Decentralized Model Reference Adaptive Controller Tests Without Robot, Varying β: Error Signal

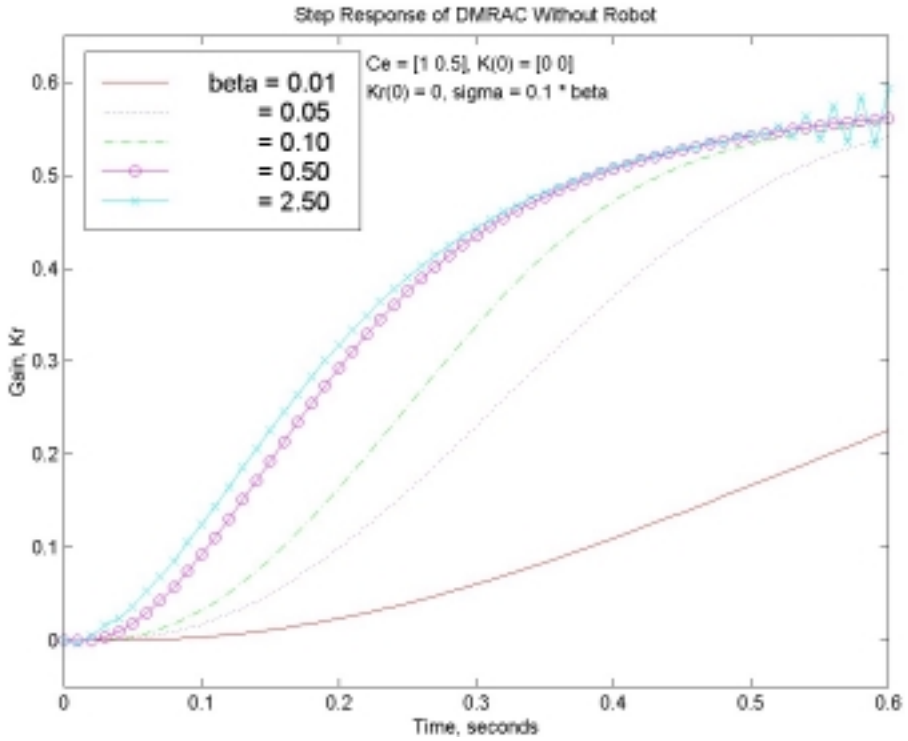

Figure 5.20: Decentralized Model Reference Adaptive Controller Tests Without Robot, Varying β: $\mathbf{K}_r$

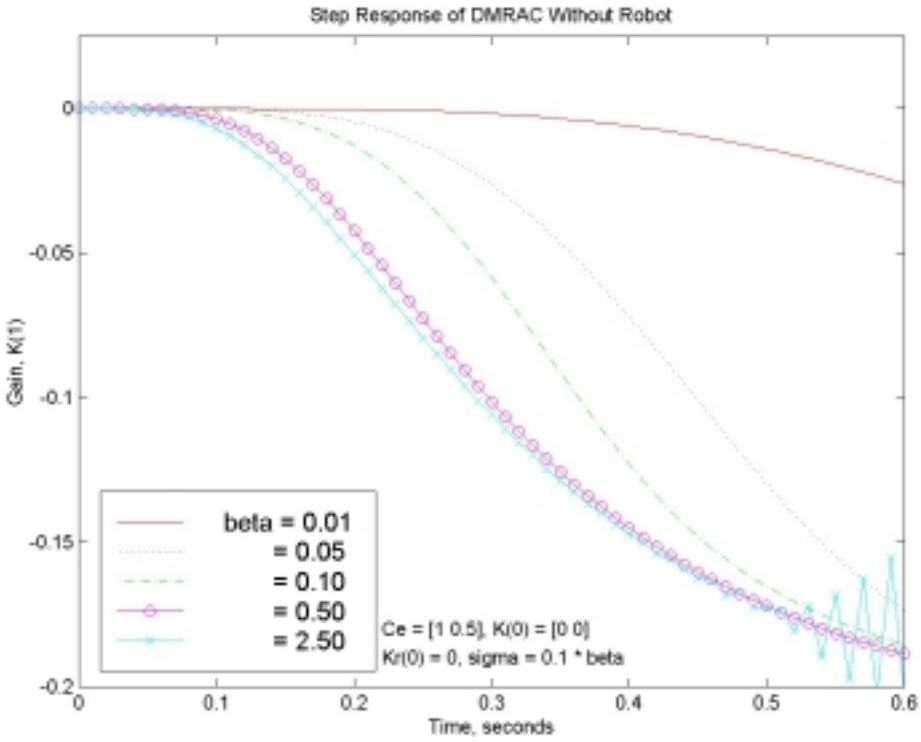

Figure 5.21: Decentralized Model Reference Adaptive Controller Tests Without Robot, Varying β: $\mathbf{K}_1$

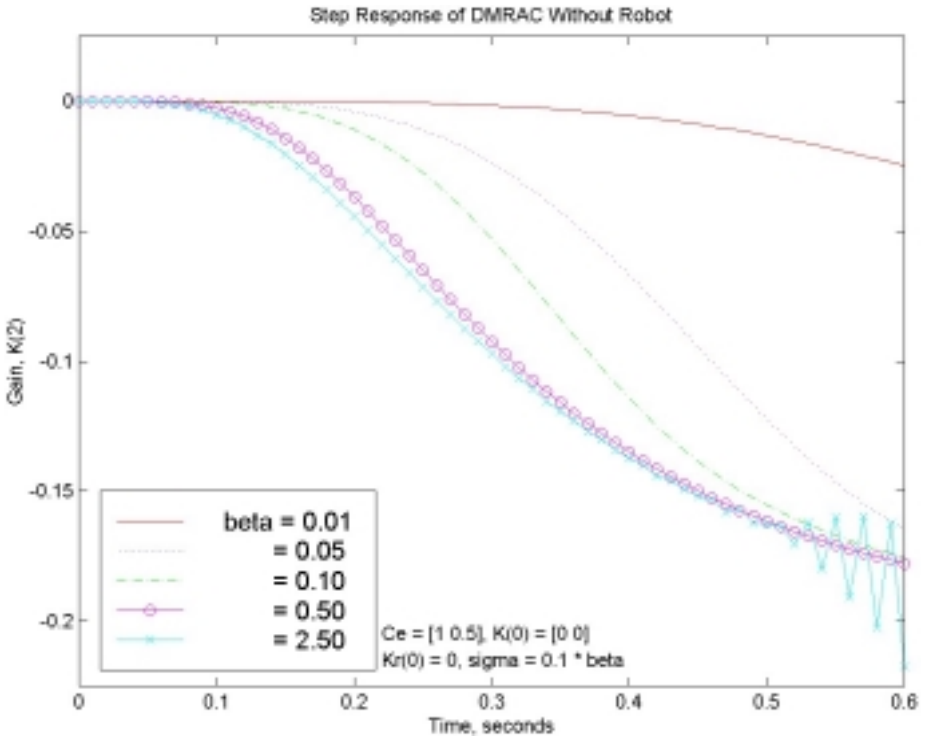

Figure 5.22: Decentralized Model Reference Adaptive Controller Tests Without Robot, Varying β: $\mathbf{K}_2$

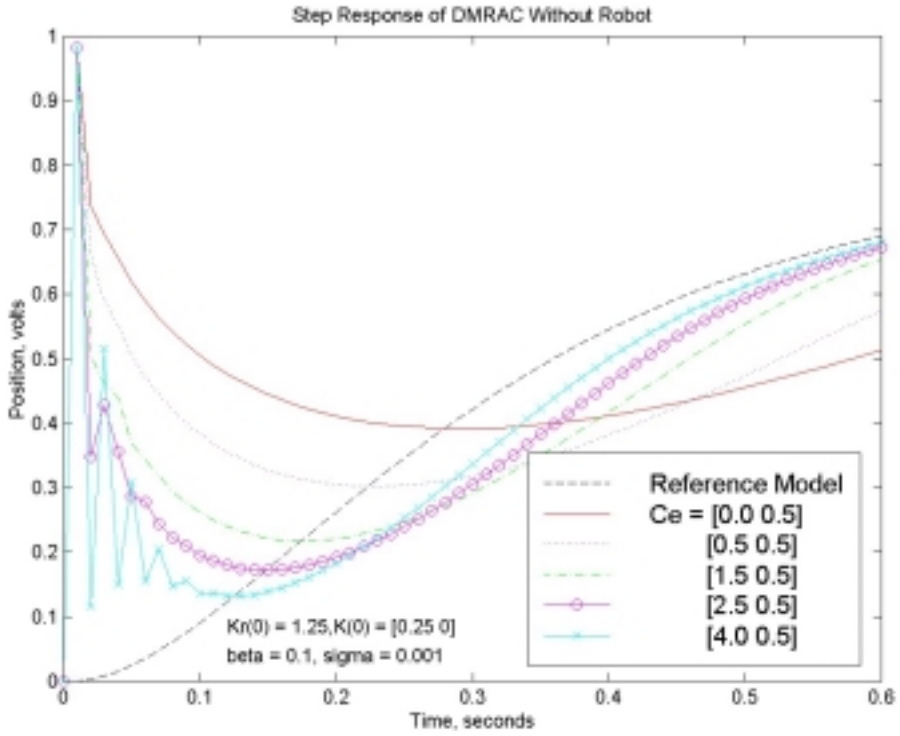

Figure 5.23: Decentralized Model Reference Adaptive Controller Tests Without Robot, Varying $\mathbf{C}_e$: Position

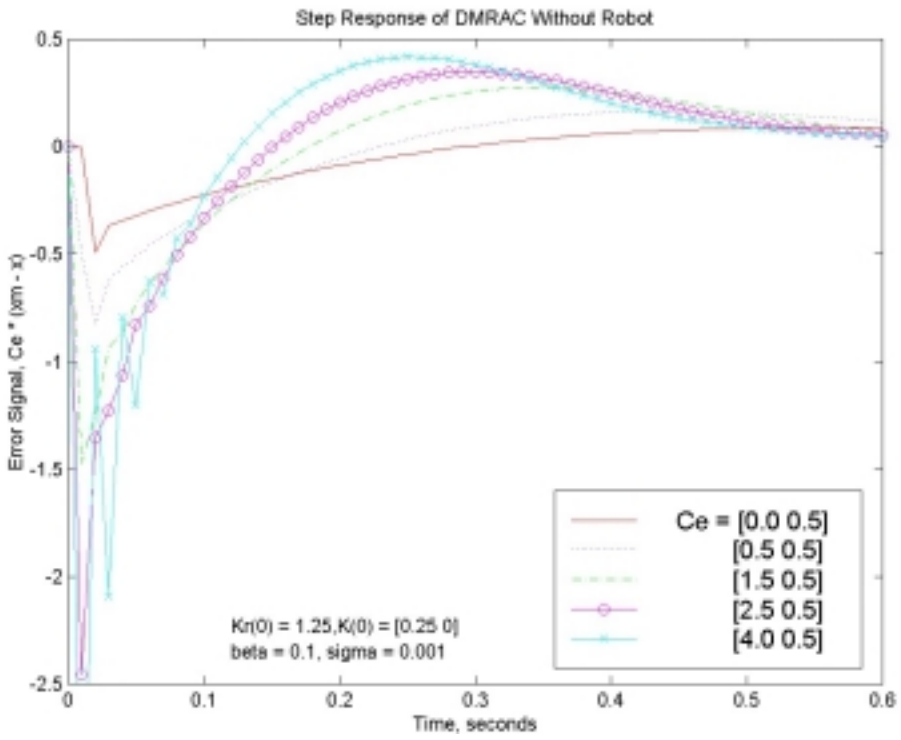

Figure 5.24: Decentralized Model Reference Adaptive Controller Tests Without Robot, Varying $\mathbf{C}_e$: Error Signal

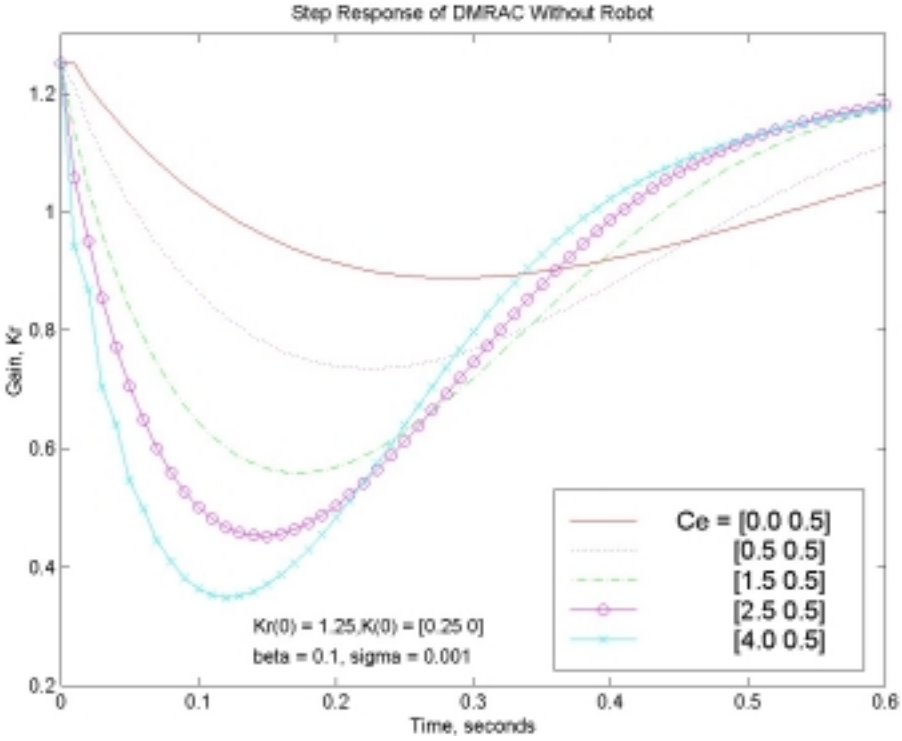

Figure 5.25: Decentralized Model Reference Adaptive Controller Tests Without Robot, Varying $\mathbf{C}_e$: $\mathbf{K}_r$

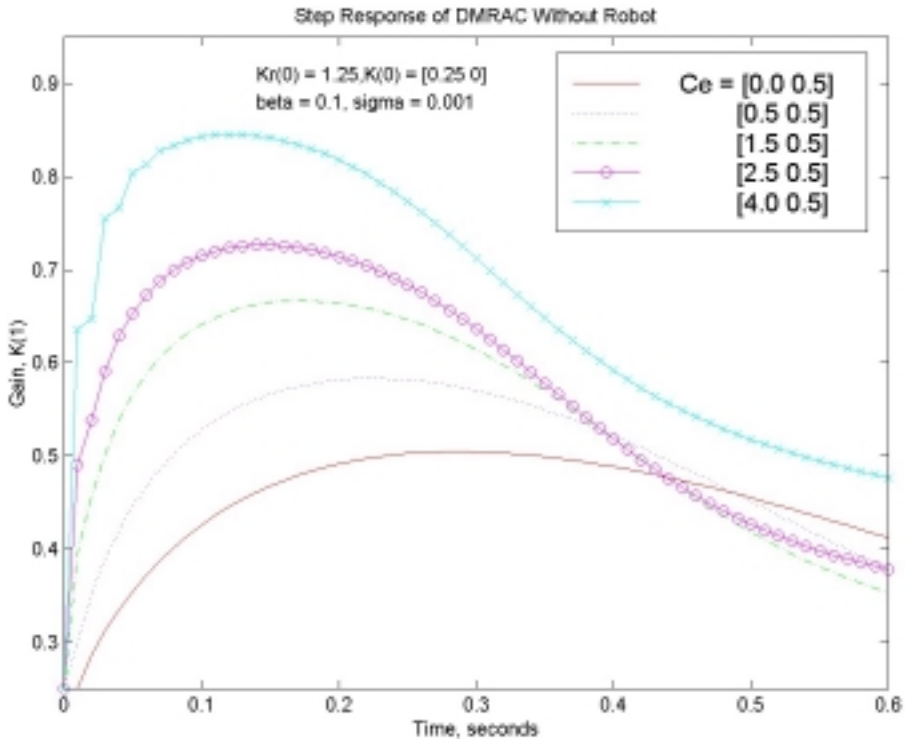

Figure 5.26: Decentralized Model Reference Adaptive Controller Tests Without Robot, Varying $\mathbf{C}_e$: $\mathbf{K}_1$

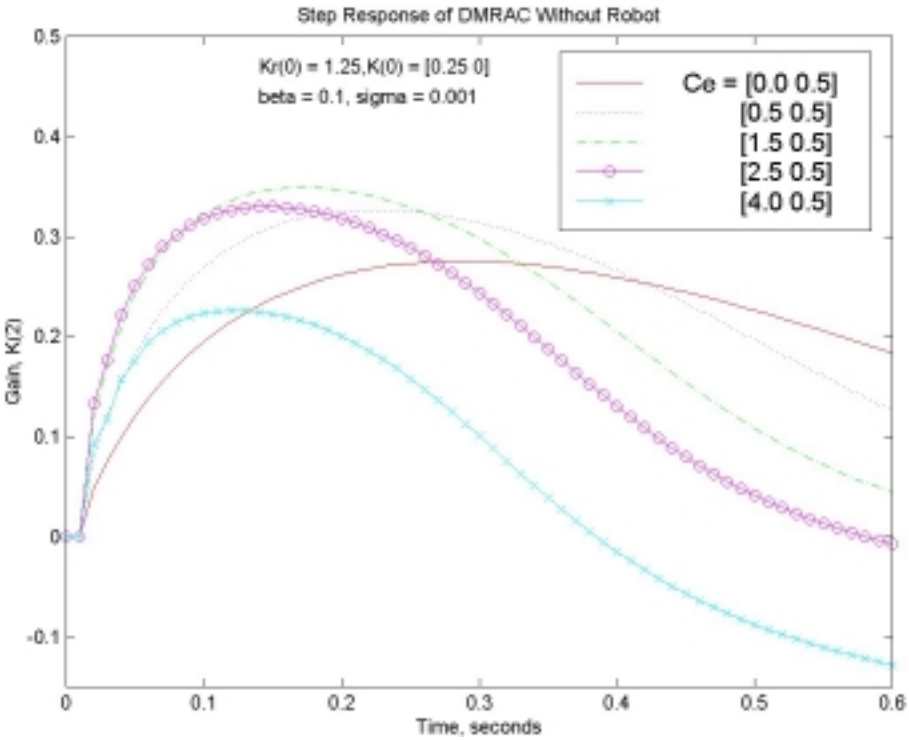

Figure 5.27: Decentralized Model Reference Adaptive Controller Tests Without Robot, Varying $\mathbf{C}_e$: $\mathbf{K}_2$

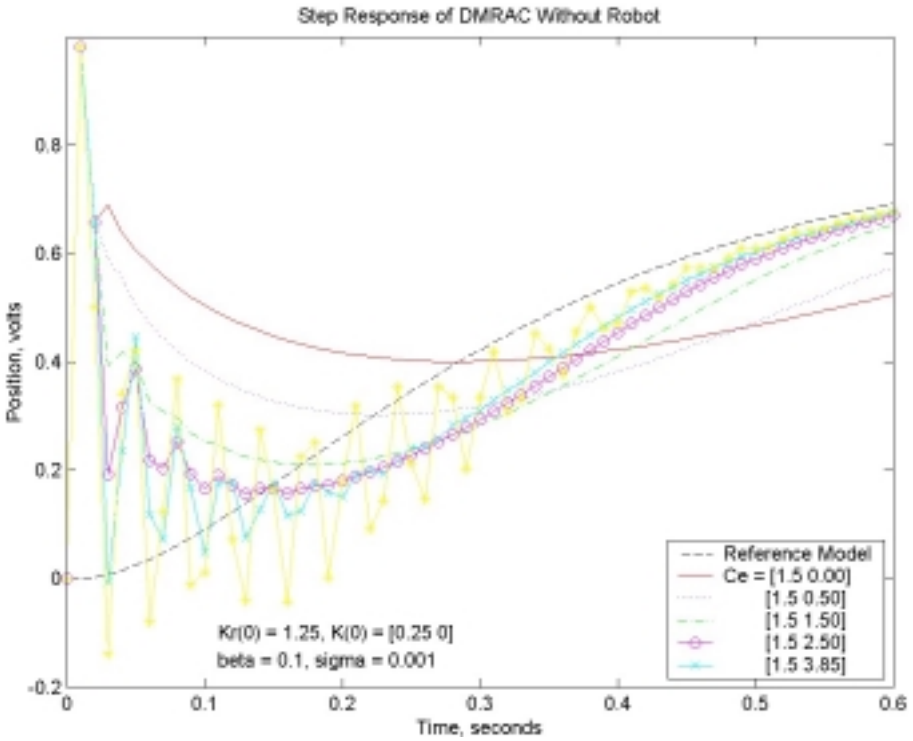

Figure 5.28: Decentralized Model Reference Adaptive Controller Test Without Robot, Varying $\mathbf{C}_{e2}$: Position

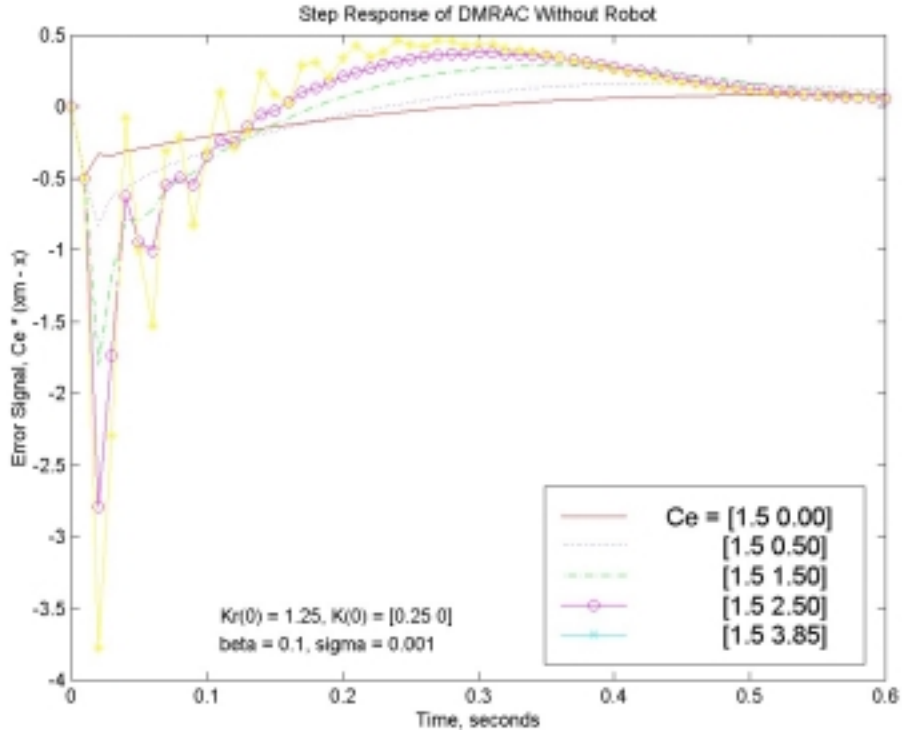

Figure 5.29: Decentralized Model Reference Adaptive Controller Test Without Robot, Varying $\mathbf{C}_{e2}$: Error Signal

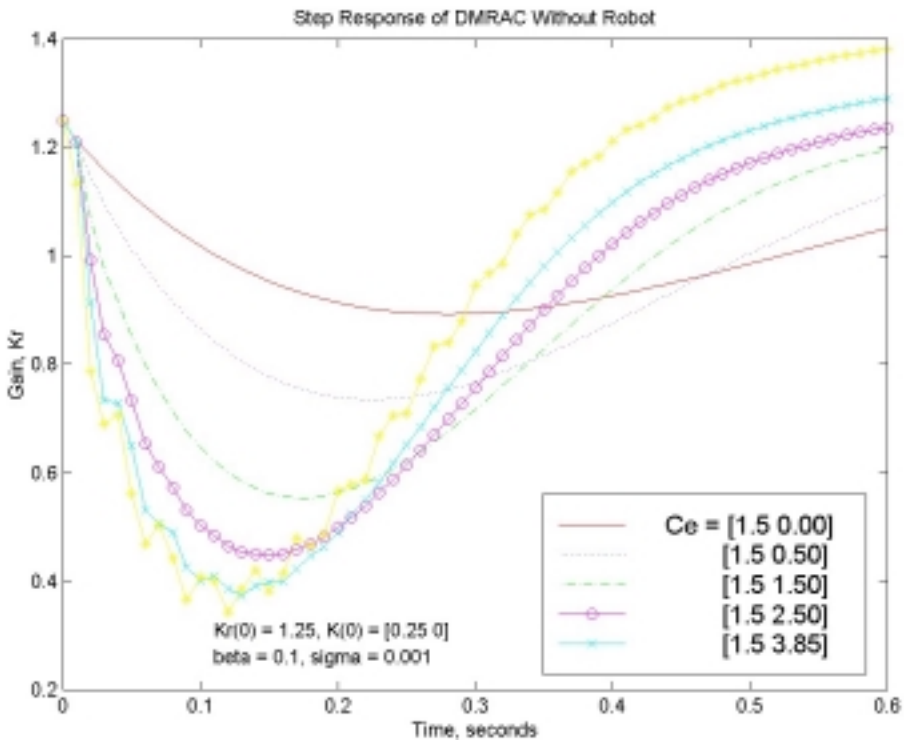

Figure 5.30: Decentralized Model Reference Adaptive Controller Test Without Robot, Varying $\mathbf{C}_{e2}$: $\mathbf{K}_r$

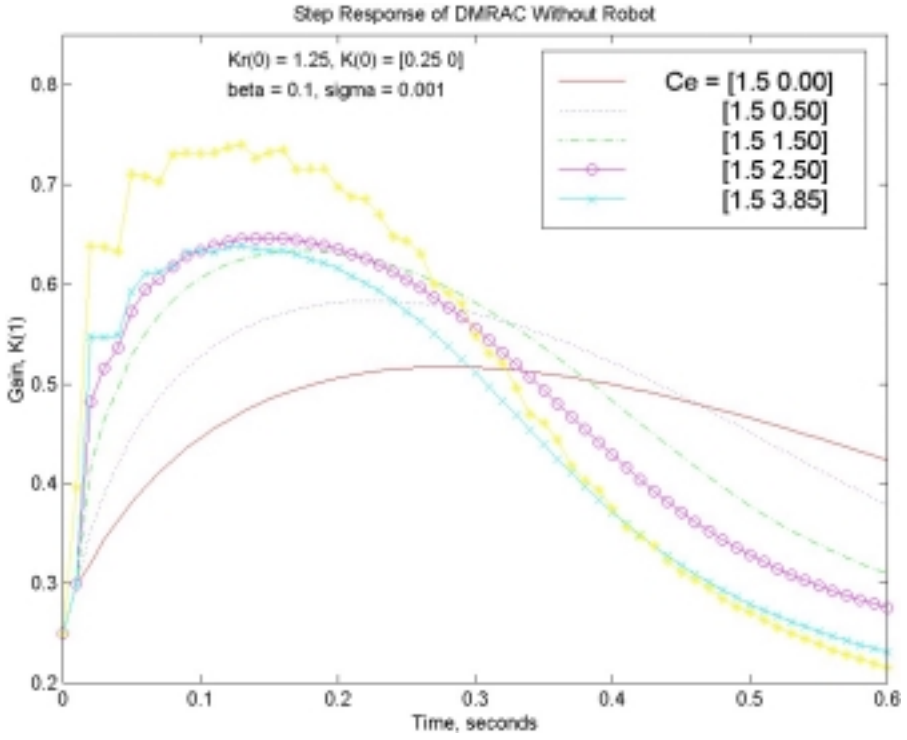

Figure 5.31: Decentralized Model Reference Adaptive Controller Test Without Robot, Varying $\mathbf{C}_{e2}$: $\mathbf{K}_1$

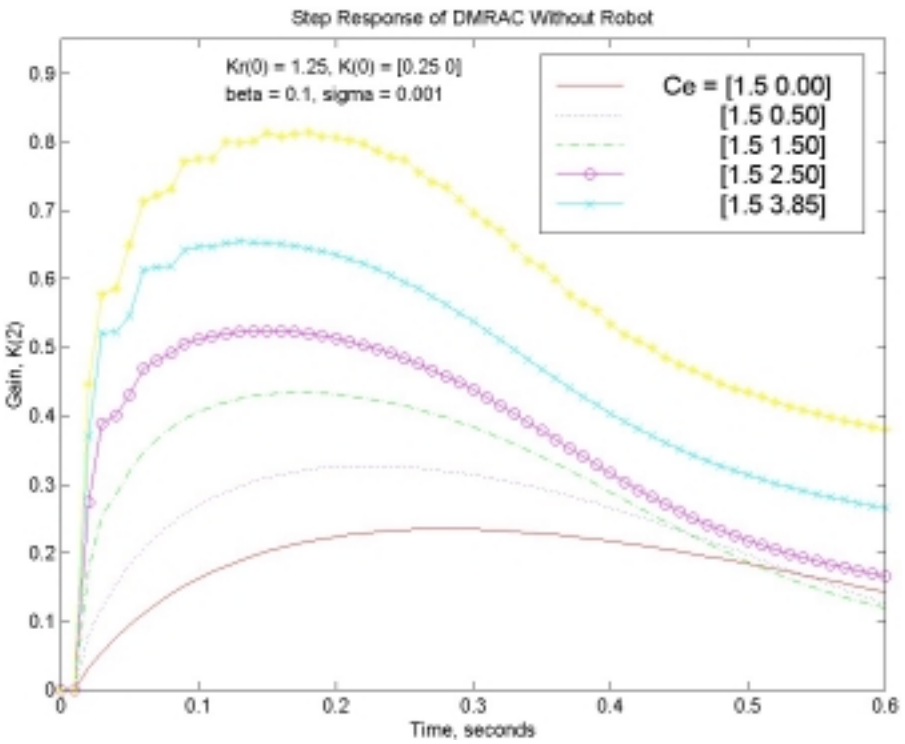

Figure 5.32: Decentralized Model Reference Adaptive Controller Test Without Robot, Varying $\mathbf{C}_{e2}$: $\mathbf{K}_2$

5.4 Decentralized MRAC with Robot in the Loop

For the final set of tests, the decentralized model reference adaptive controller was attached to one link of the robot, and its performance was examined. The controller using the $\mathbf{K}$ and $\mathbf{K}_r$ found in Section 5.2 and the $\mathbf{C}_e$ calculated in Section 4.3 was unstable for almost all non-zero βs. Setting the initial gains to zero did not improve the performance. The values of β that did provide a stable response yielded a response almost identical to that of the constant-gain controller.

At this point, the experiments performed to determine the motor constant, described in Section 4.2.2, were reexamined, but no error was found. Since $\mathbf{C}_e$ could not be found using the method described in Section 3.2, experiments were run to try to find it empirically. This required varying $\mathbf{C}_{e1}$, $\mathbf{C}_{e2}$, and β independently. A samples of the tests run can be seen in Figures 5.33-5.54. After many attempts to find a stable combination of β and $\mathbf{C}_e$, a stable response was not found for any non-zero value of β.

The method of determining the velocity as the derivative of the position measurement induced a significant amount of noise. Because of the robot's support structure limitations, it was not possible to add a velocity measurement sensor to the robot. Therefore, attempts were made to reduce velocity measurement noise by filtering, as shown in Figures 5.55-5.60. A seven-pole Butterworth filter was added to the block diagram shown in Figure 4.12. The cut-off frequency was varied to see if the performance could be improved, but filtering the velocity signal only induced lag in the controller, which led to oscillations and instability.

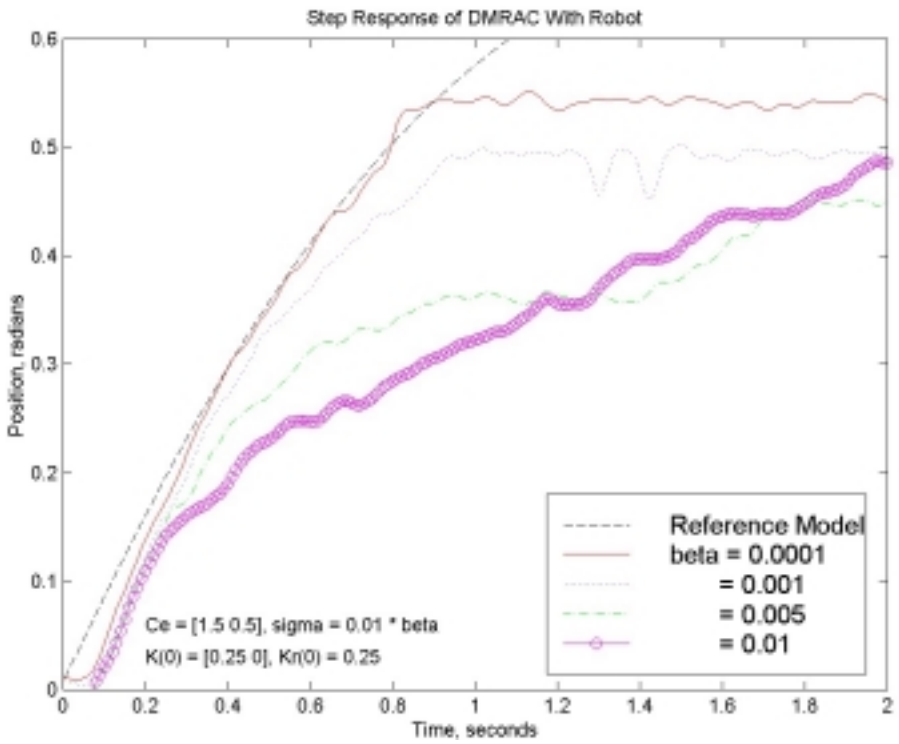

Figure 5.33: Decentralized Model Reference Adaptive Controller Test With Robot, Varying β: Position

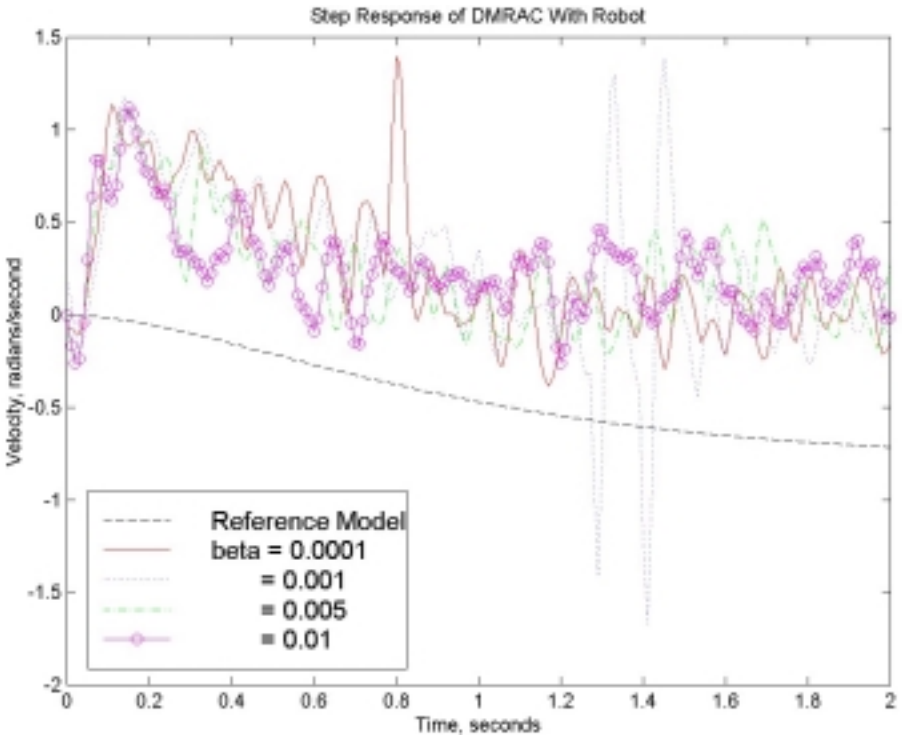

Figure 5.34: Decentralized Model Reference Adaptive Controller Test With Robot, Varying β: Velocity

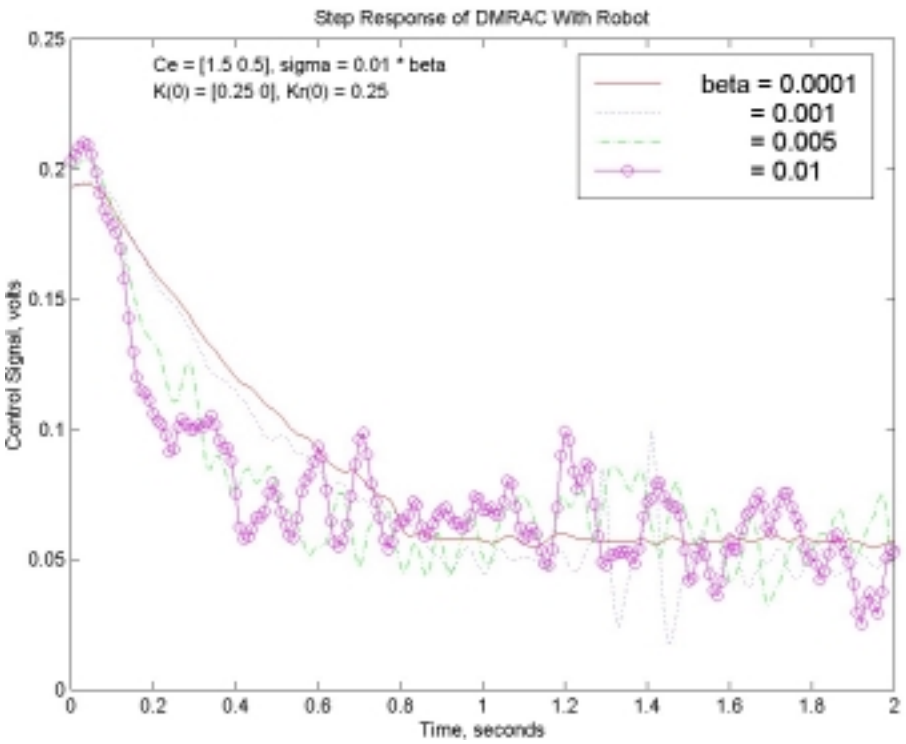

Figure 5.35: Decentralized Model Reference Adaptive Controller Test With Robot, Varying β: Control Signal

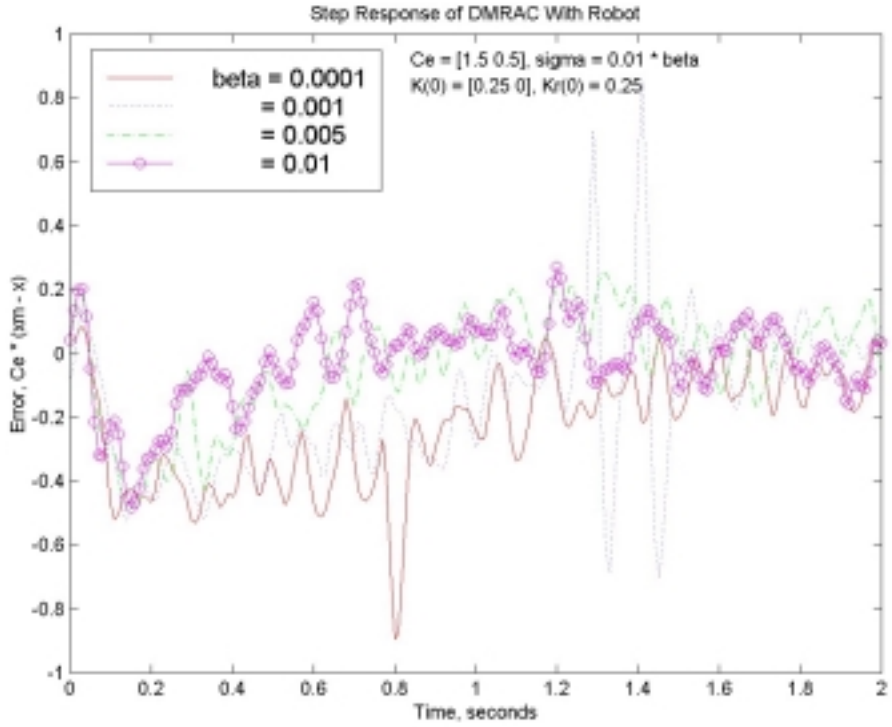

Figure 5.36: Decentralized Model Reference Adaptive Controller Test With Robot, Varying β: Error Signal

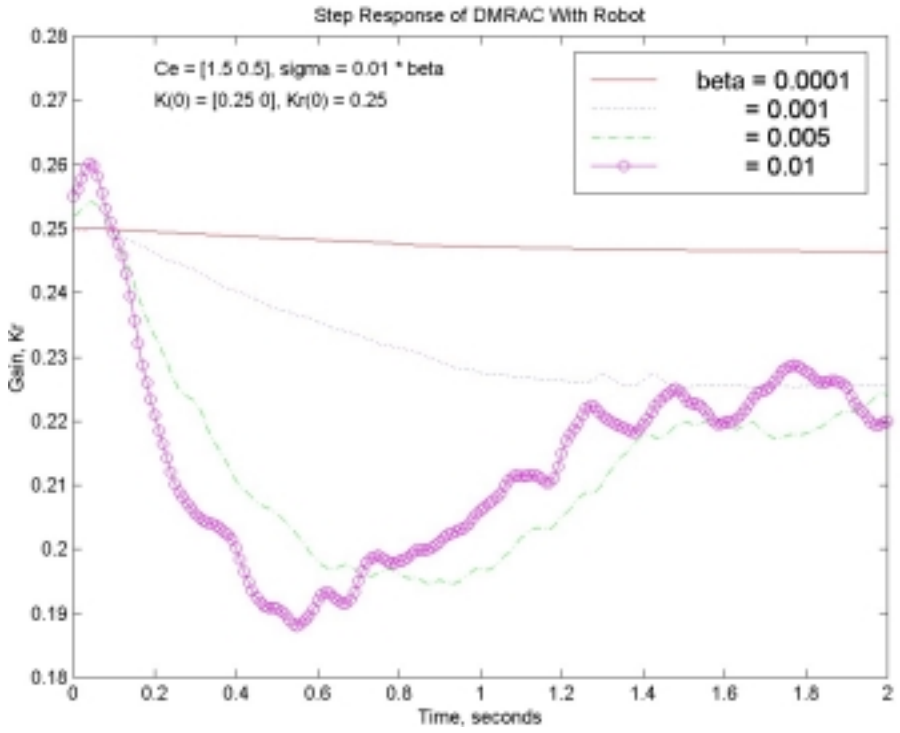

Figure 5.37: Decentralized Model Reference Adaptive Controller Test With Robot, Varying β: $\mathbf{K}_r$

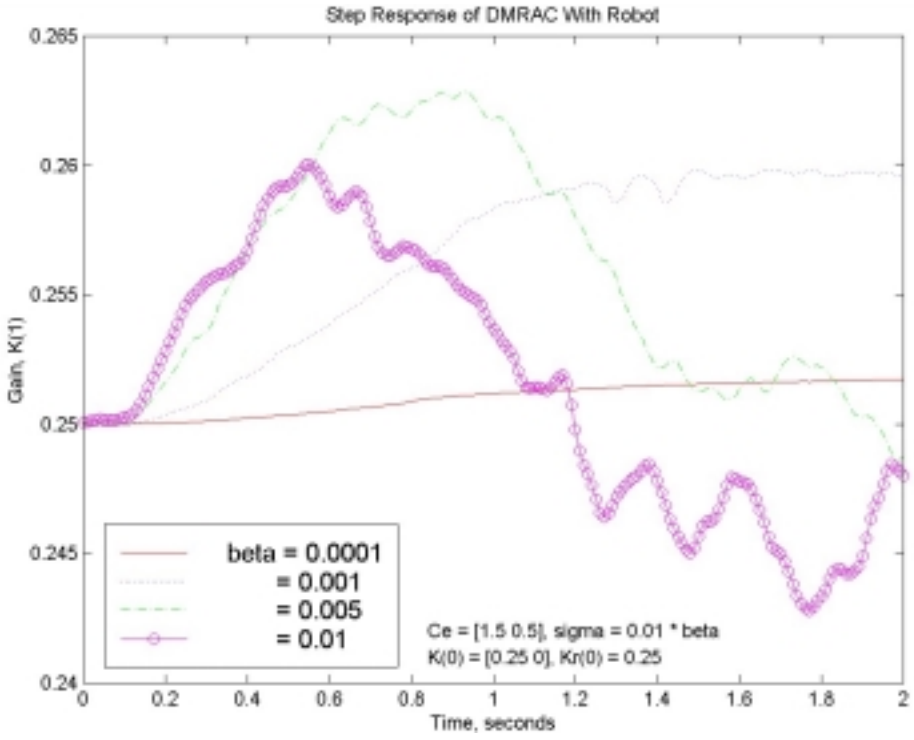

Figure 5.38: Decentralized Model Reference Adaptive Controller Test With Robot, Varying I: $\mathbf{K}_1$

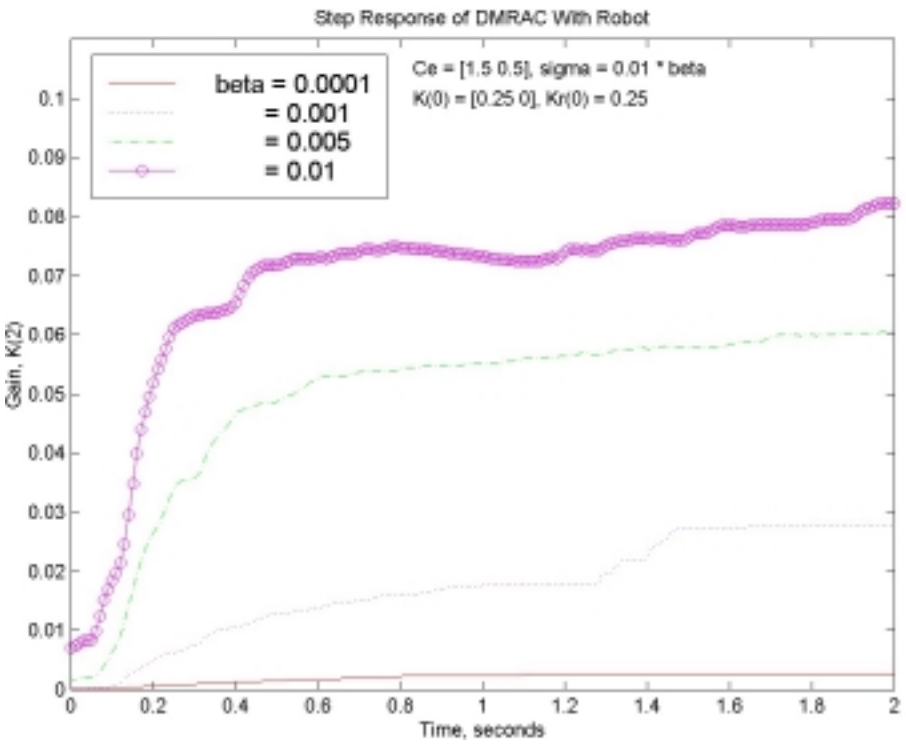

Figure 5.39: Decentralized Model Reference Adaptive Controller Test With Robot, Varying β: $\mathbf{K}_2$

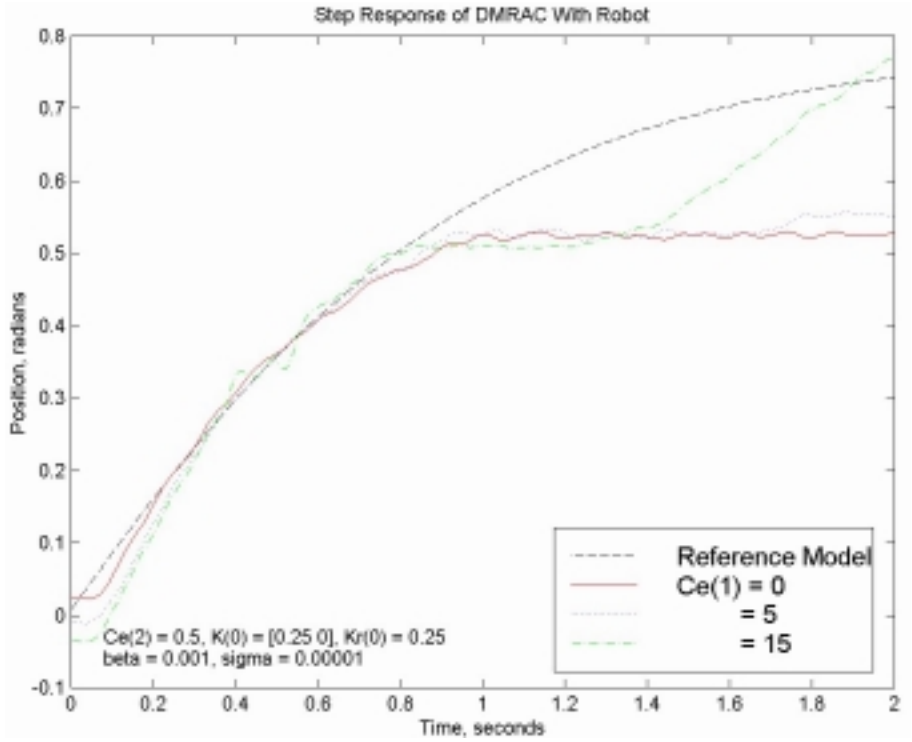

Figure 5.40: Decentralized Model Reference Adaptive Controller Test With Robot, Varying $\mathbf{C}_{e1}$: Position

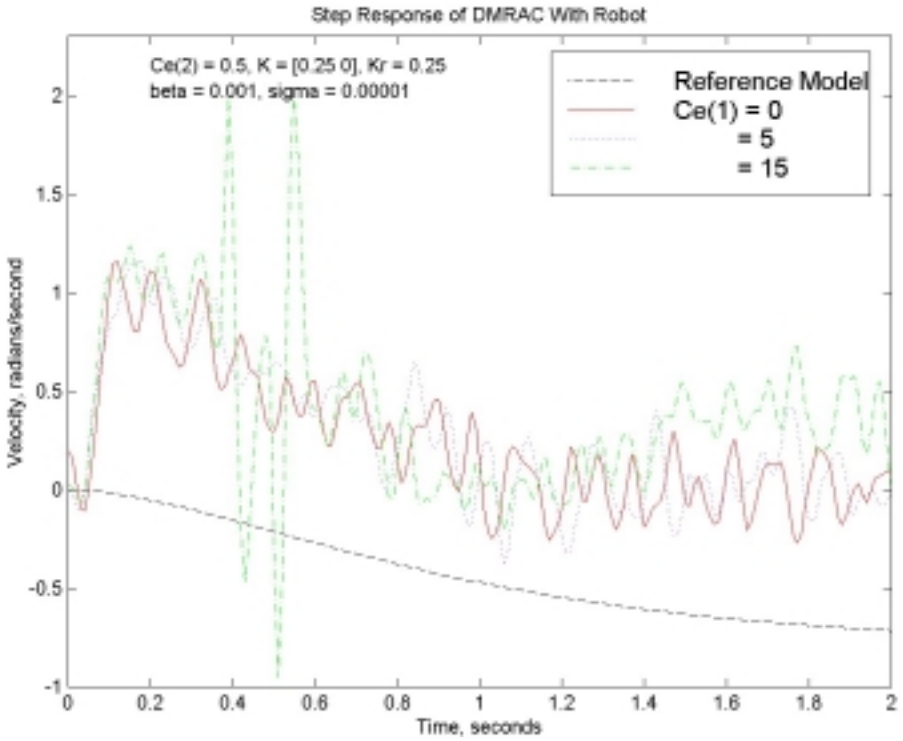

Figure 5.41: Decentralized Model Reference Adaptive Controller Test With Robot, Varying $\mathbf{C}_{e1}$: Velocity

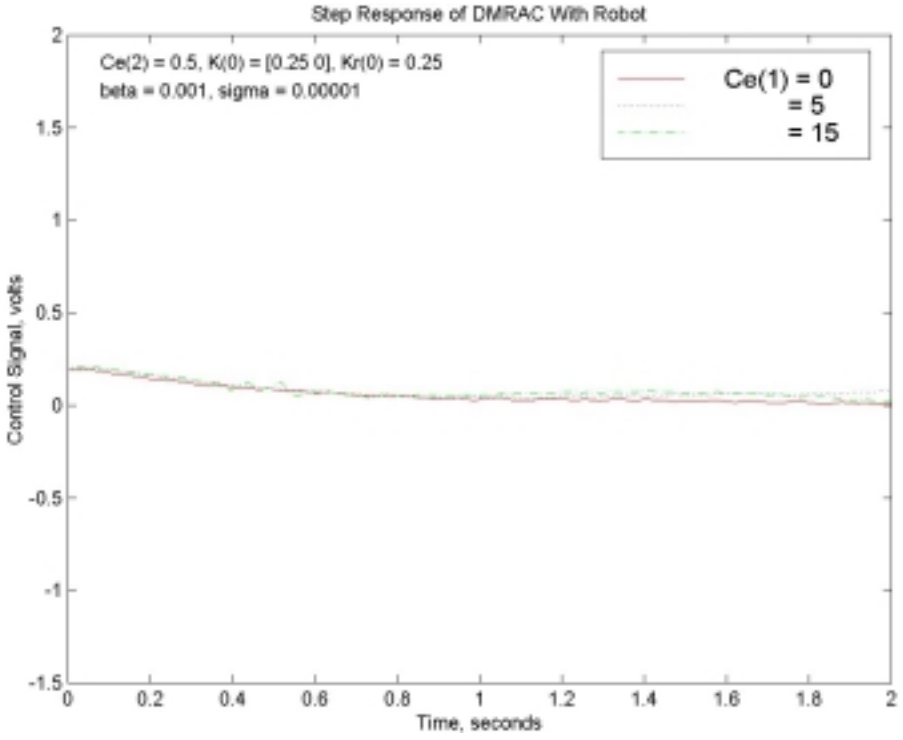

Figure 5.42: Decentralized Model Reference Adaptive Controller Test With Robot, Varying $\mathbf{C}_{e1}$: Control Signal

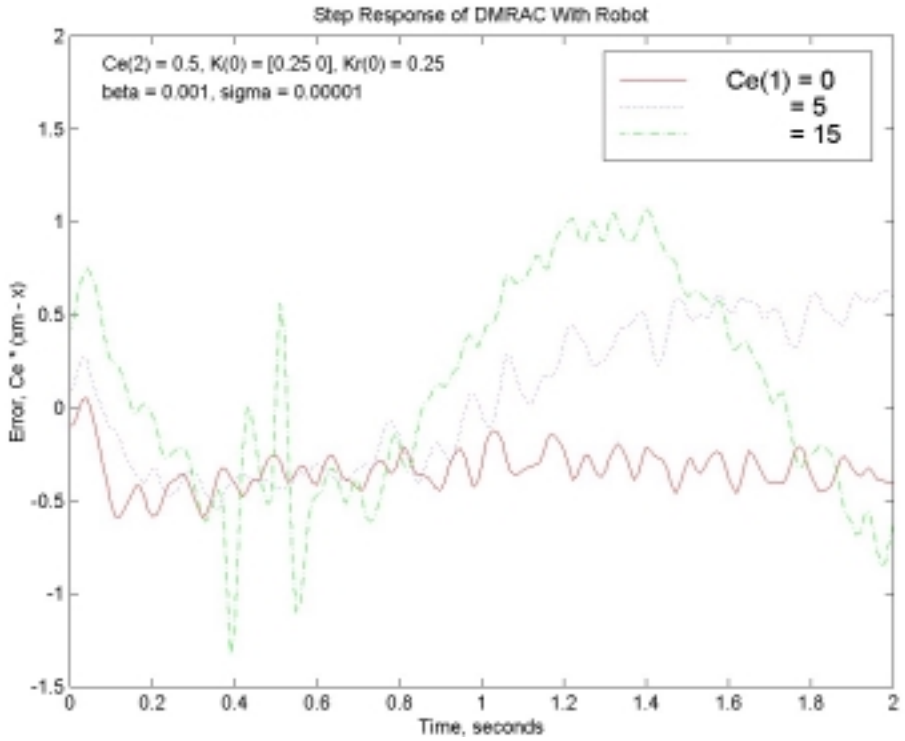

Figure 5.43: Decentralized Model Reference Adaptive Controller Test With Robot, Varying $\mathbf{C}_{e1}$: Signal

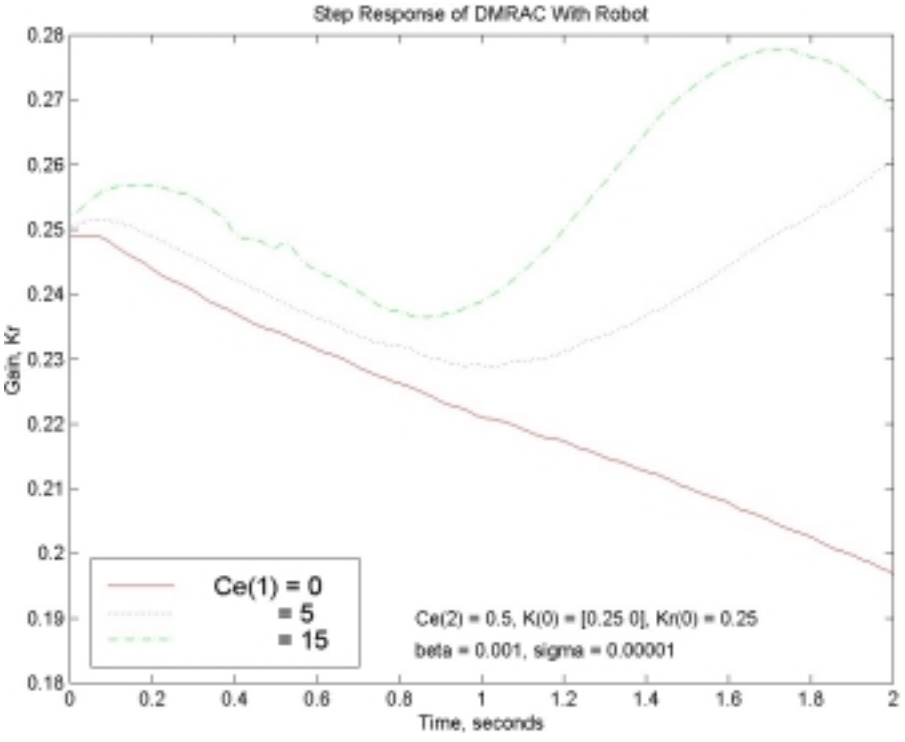

Figure 5.44: Decentralized Model Reference Adaptive Controller Test With Robot, Varying $\mathbf{C}_{e1}$: $\mathbf{K}_r$

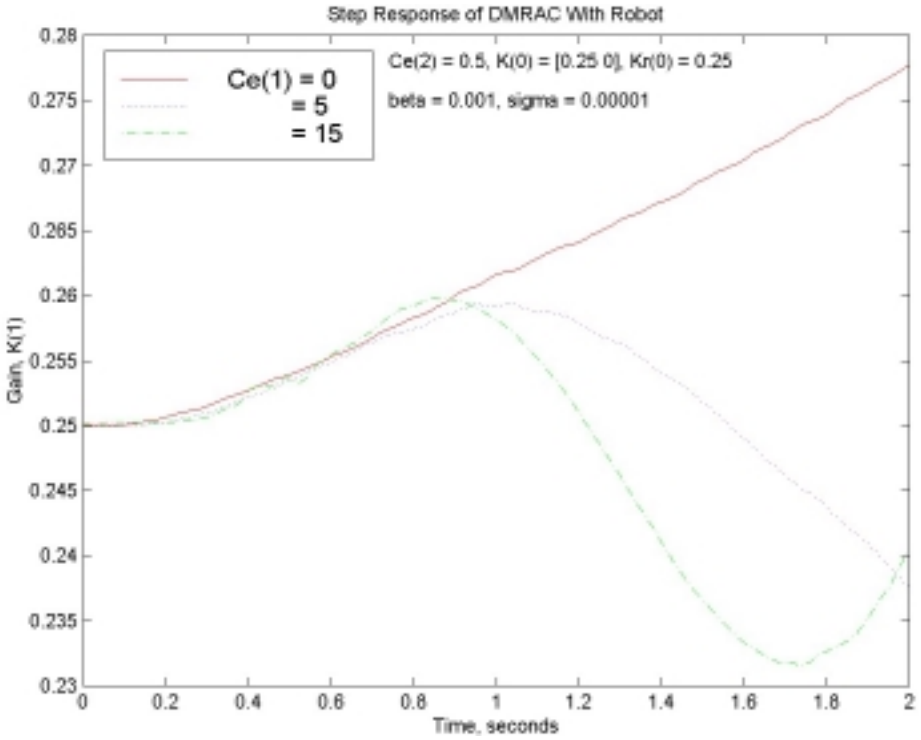

Figure 5.45: Decentralized Model Reference Adaptive Controller Test With Robot, Varying $\mathbf{C}_{e1}$: $\mathbf{K}_1$

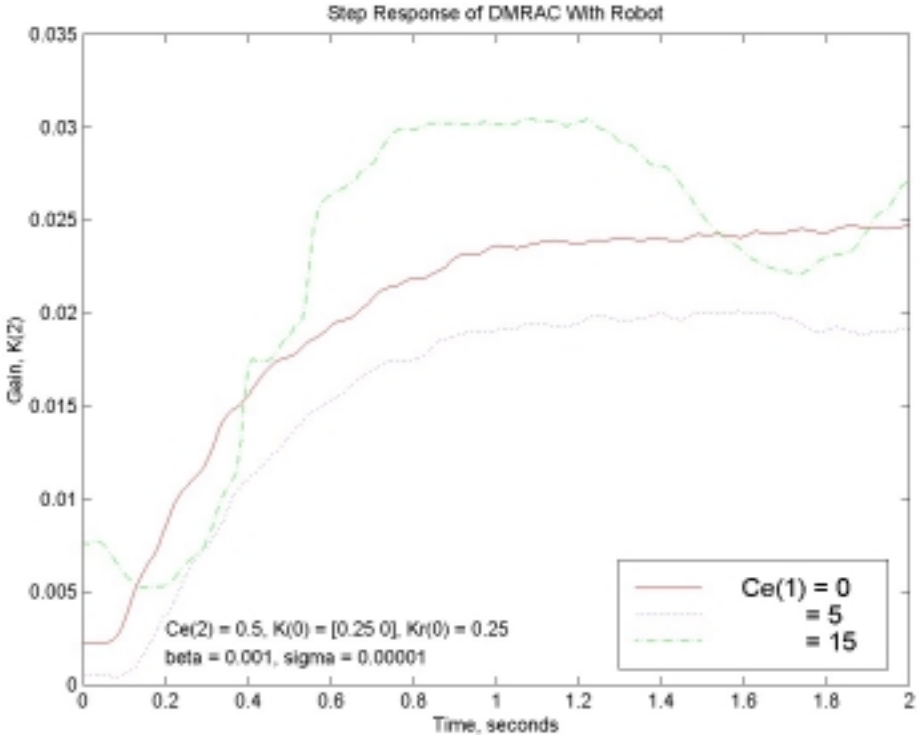

Figure 5.46: Decentralized Model Reference Adaptive Controller Test With Robot, Varying $\mathbf{C}_{e1}$: $\mathbf{K}_2$

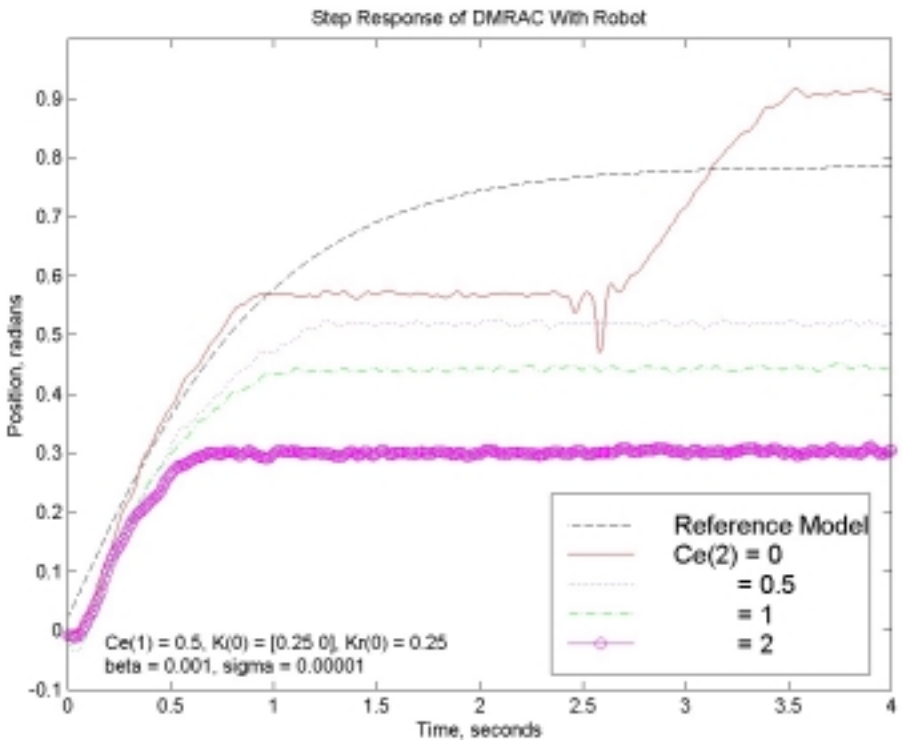

Figure 5.47: Decentralized Model Reference Adaptive Controller Test With Robot, Varying $\mathbf{C}_{e2}$: Position

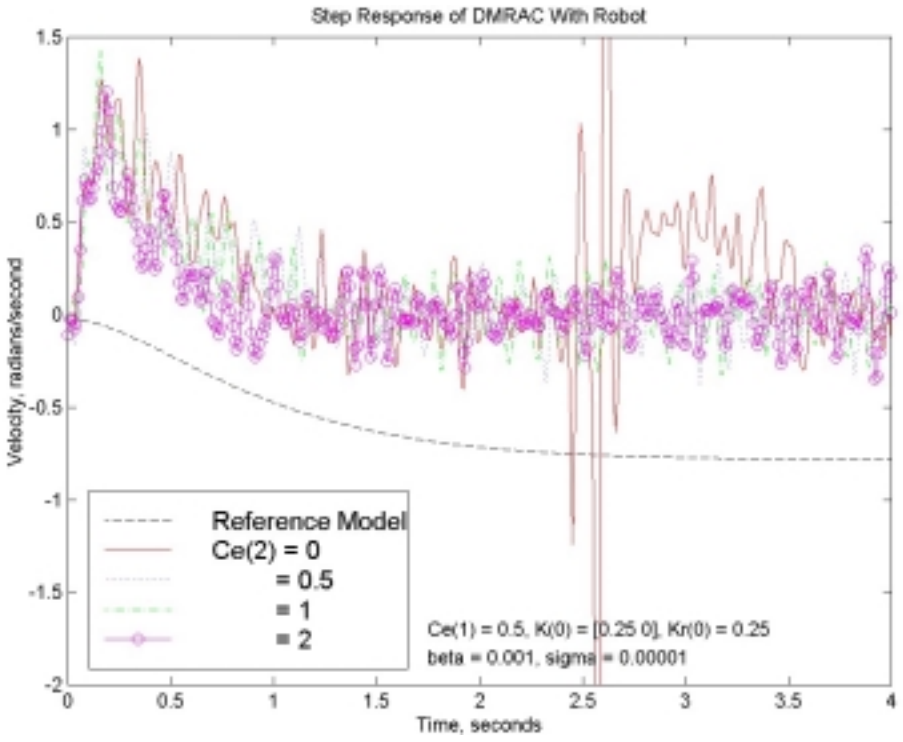

Figure 5.48: Decentralized Model Reference Adaptive Controller Test With Robot, Varying $\mathbf{C}_{e2}$: Velocity

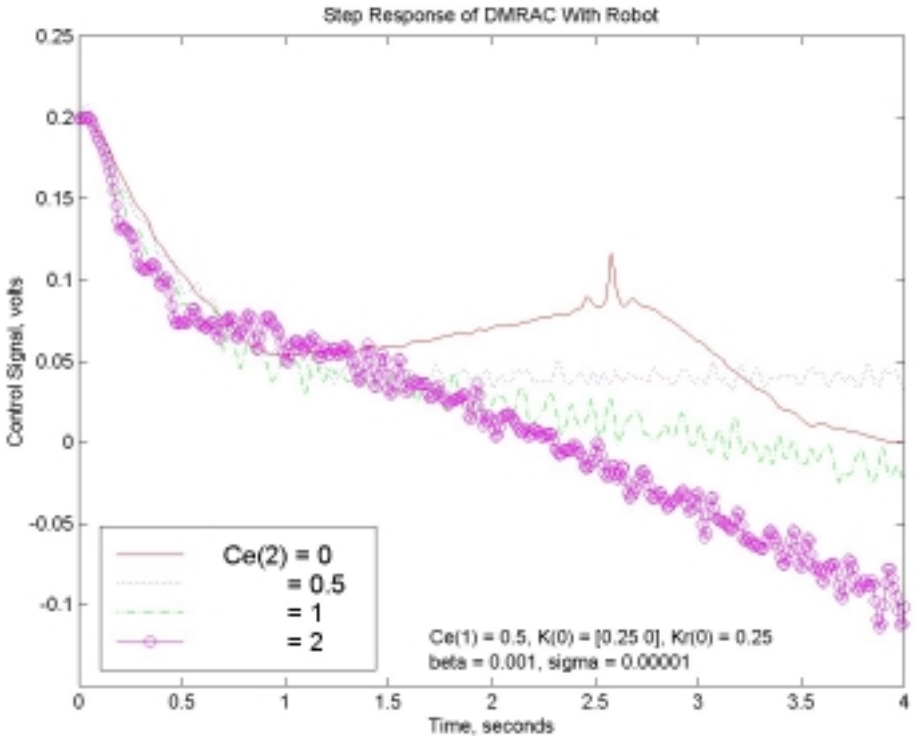

Figure 5.49: Decentralized Model Reference Adaptive Controller Test With Robot, Varying $\mathbf{C}_{e2}$: Control Signal

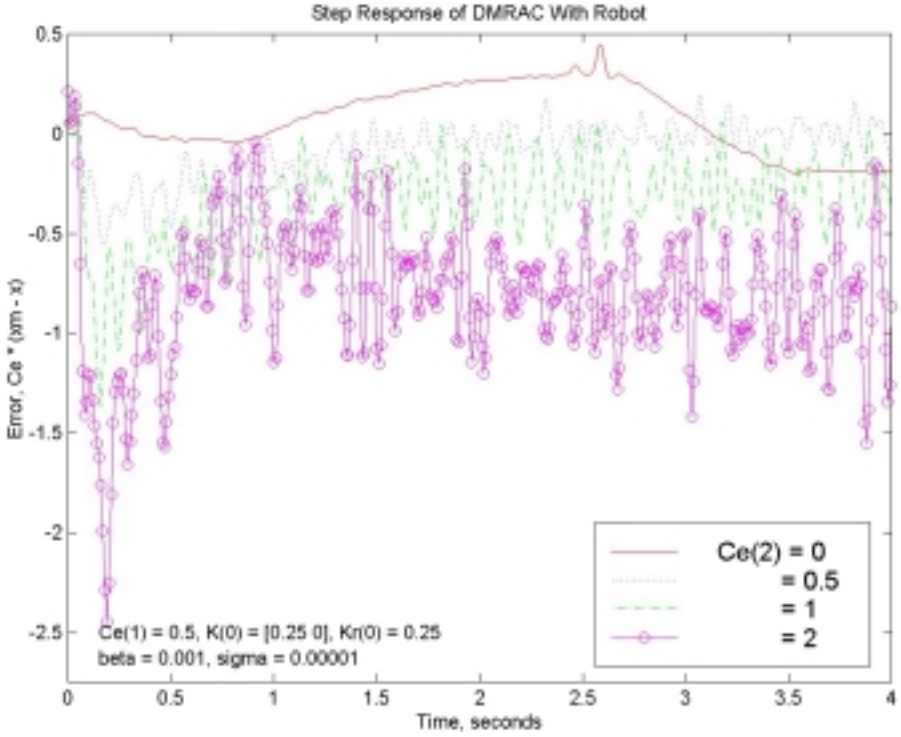

Figure 5.50: Decentralized Model Reference Adaptive Controller Test With Robot, Varying $\mathbf{C}_{e2}$: Error Signal

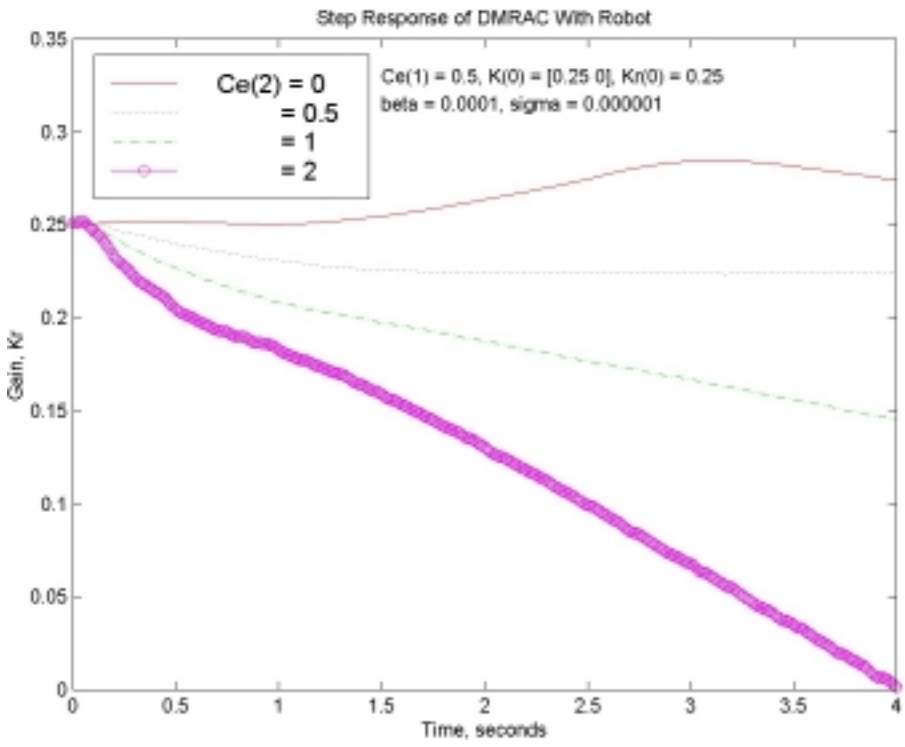

Figure 5.51: Decentralized Model Reference Adaptive Controller Test With Robot, Varying $\mathbf{C}_{e2}$: $\mathbf{K}_r$

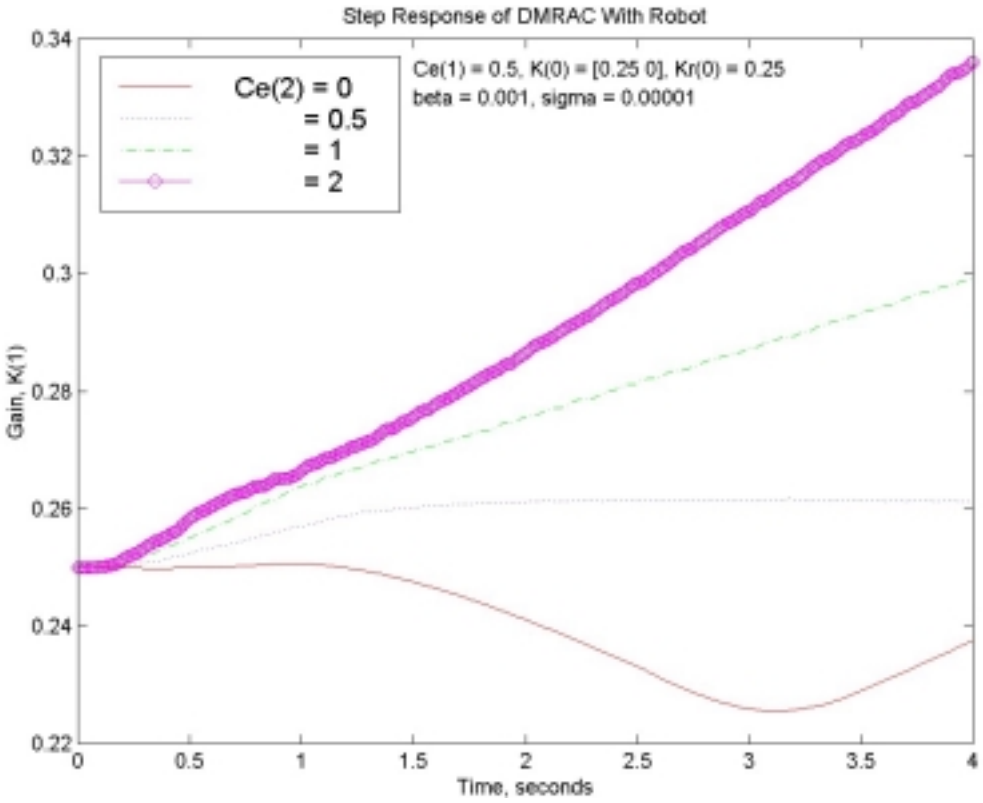

Figure 5.52: Decentralized Model Reference Adaptive Controller Test With Robot, Varying $\mathbf{C}_{e2}$: $\mathbf{K}_1$

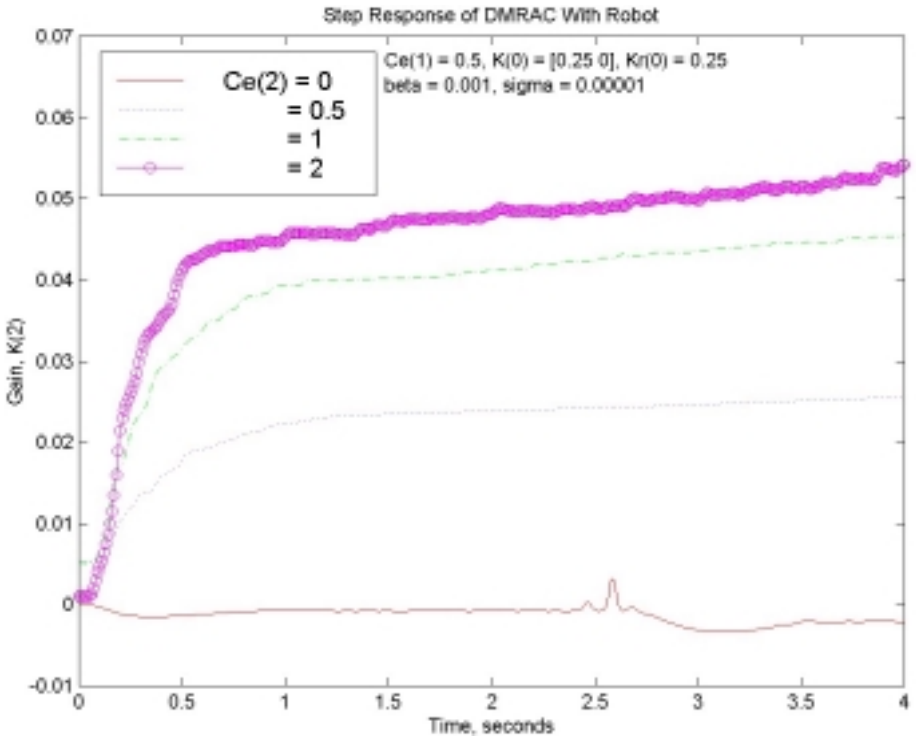

Figure 5.53: Decentralized Model Reference Adaptive Controller Test With Robot, Varying $\mathbf{C}_{e2}$: $\mathbf{K}_2$

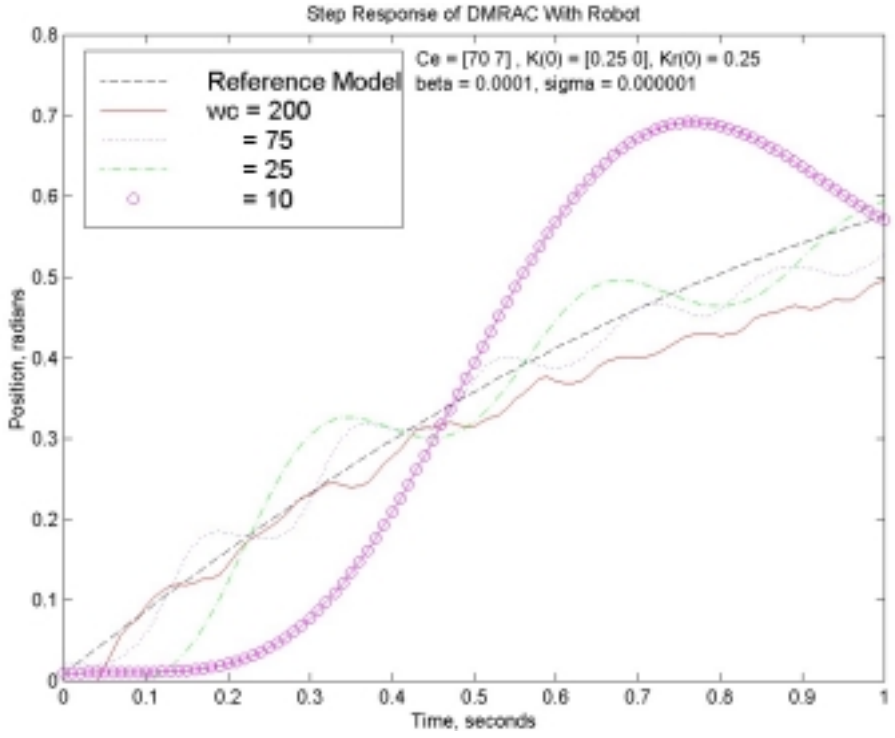

Figure 5.54: Decentralized Model Reference Adaptive Controller Test With Robot, Varying Filter Cut-off Frequency: Position

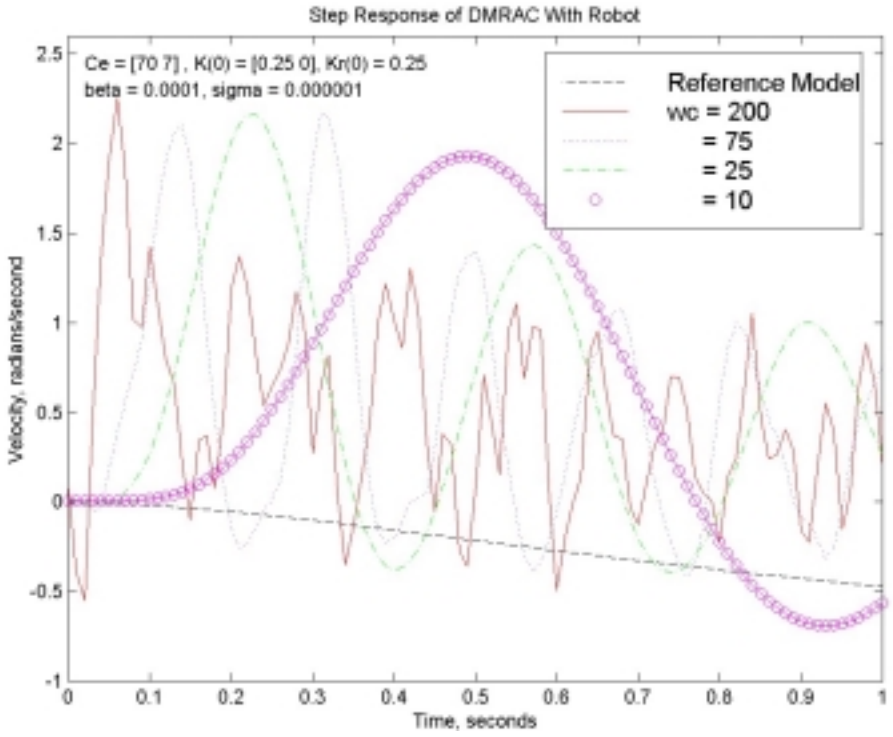

Figure 5.55: Decentralized Model Reference Adaptive Controller Test With Robot, Varying Filter Cut-off Frequency: Velocity

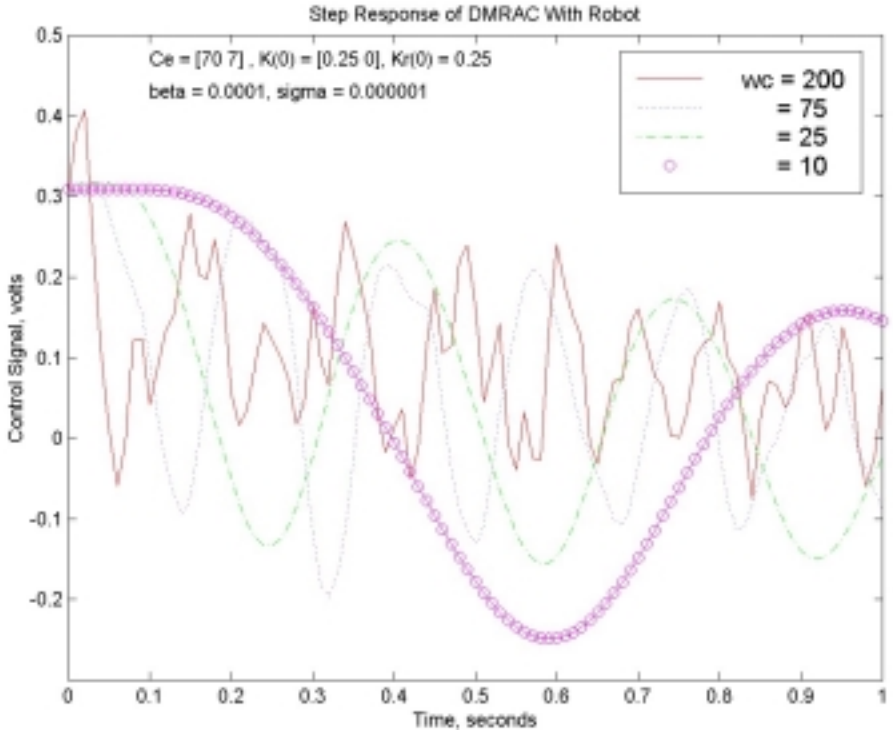

Figure 5.56: Decentralized Model Reference Adaptive Controller Test With Robot, Varying Filter Cut-off Frequency: Control Signal

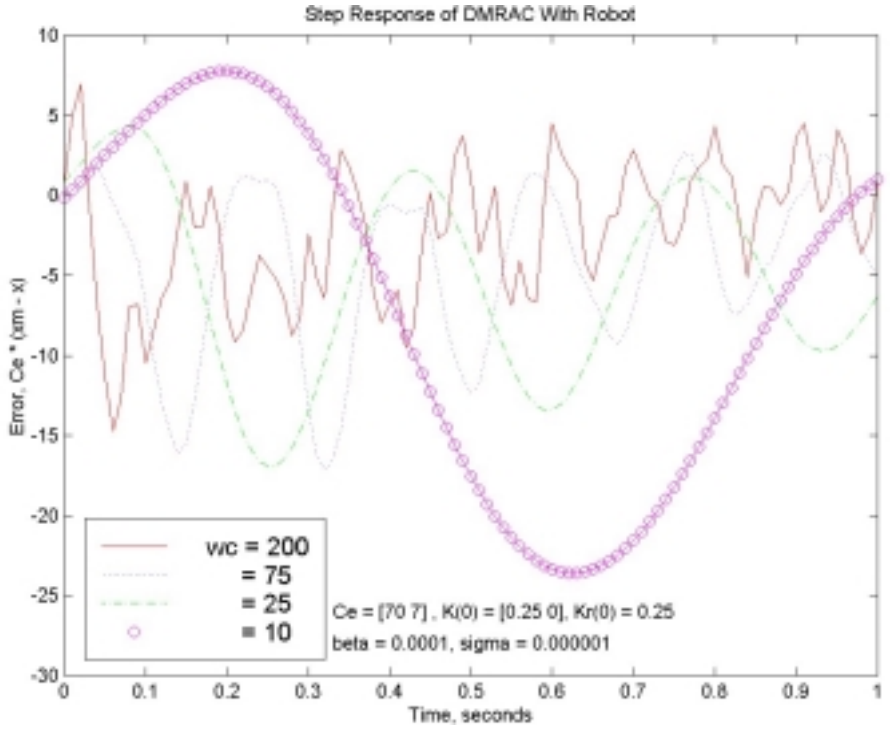

Figure 5.57: Decentralized Model Reference Adaptive Controller Test With Robot, Varying Filter Cut-off Frequency: Error Signal

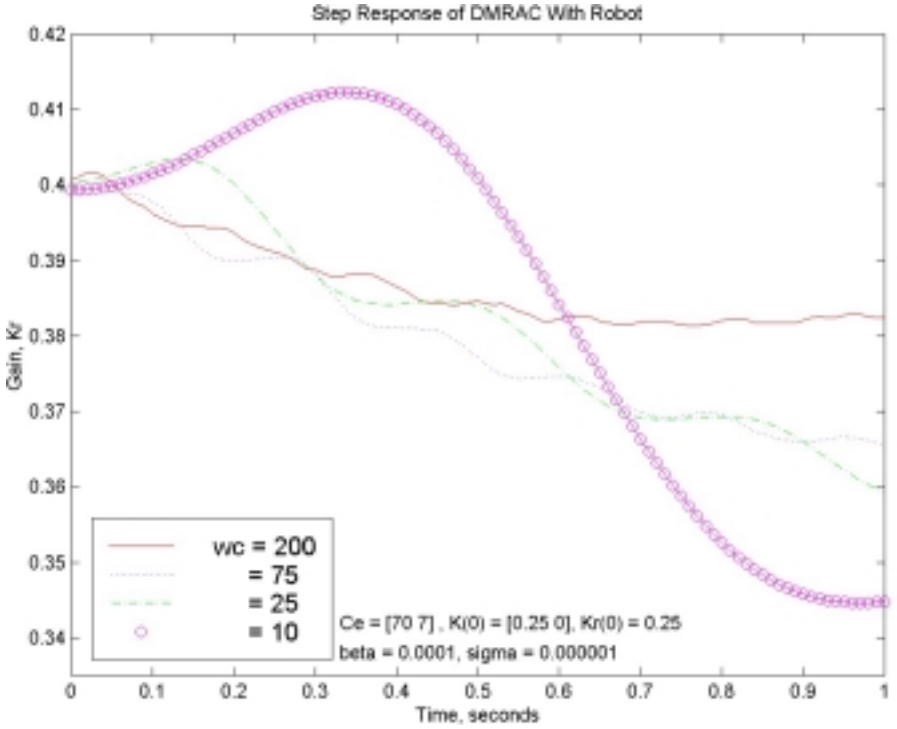

Figure 5.58: Decentralized Model Reference Adaptive Controller Test With Robot, Varying Filter Cut-off Frequency: $\mathbf{K}_r$

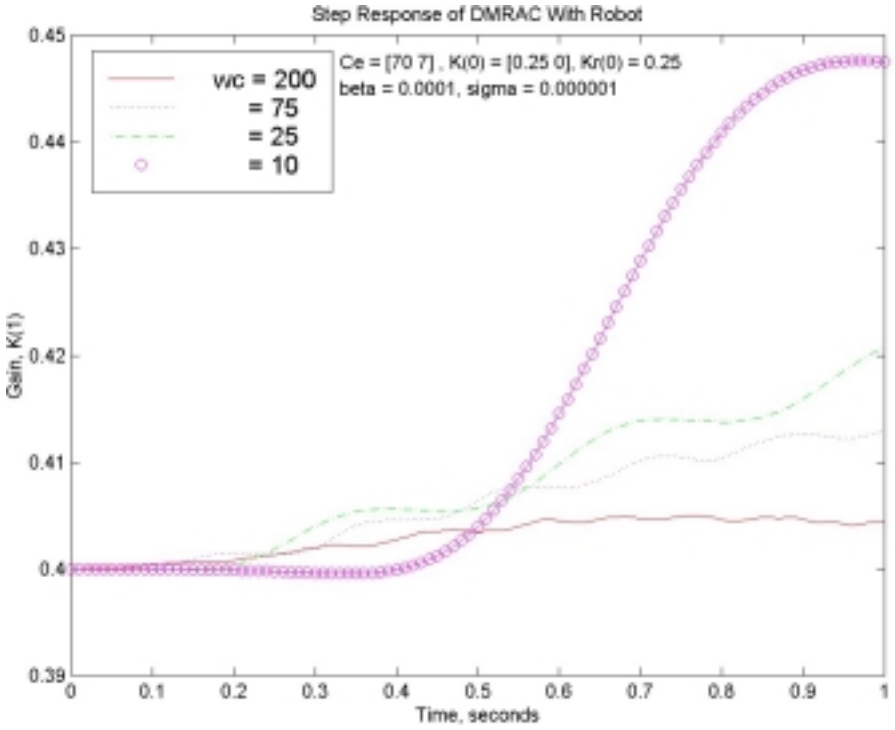

Figure 5.59: Decentralized Model Reference Adaptive Controller Test With Robot, Varying Filter Cut-off Frequency: $\mathbf{K}_1$

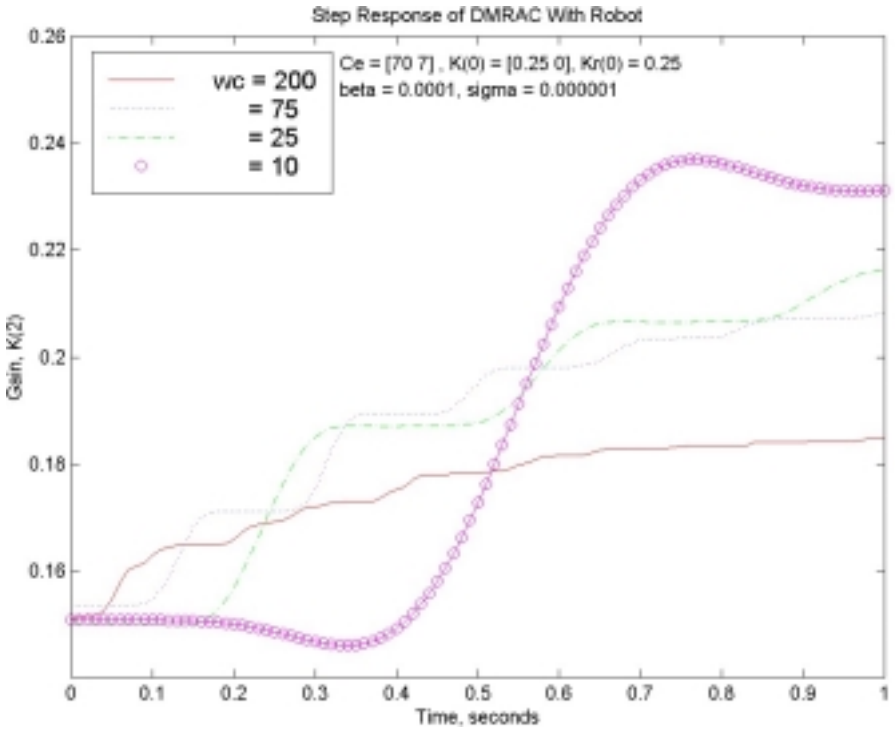

Figure 5.60: Decentralized Model Reference Adaptive Controller Test With Robot, Varying Filter Cut-off Frequency: $\mathbf{K}_2$

5.5 Concluding Remarks

A stable response from the mobile robot with the adaptive controller was not obtained using the controller parameters calculated in Chapter 4. The parameters used to calculate the initial gains and $\mathbf{C}_e$ were re-examined, but no error was found. Assuming that the instability was due to $\mathbf{C}_e$ having been calculated incorrectly, tests were performed to see if a value for $\mathbf{C}_e$ could be found that matched the model. These tests did not yield a stable response for non-zero values of β.

Velocity measurement noise was considered the next likely cause of instability, so adding joint velocity sensors to the robot was investigated. However, this was not possible because of the limited support-structure on the robot. Low-pass filtering was added to the joint velocity measurement in an attempt to improve the controller performance. Filter cut-off frequencies that were sufficiently low to remove the noise from the velocity measurement induced lag that caused the system to be unstable.

Chapter 6

Conclusions

6.1 Summary

The model reference adaptive controller's performance was not fully evaluated because a stable response was not obtained. With a simple state space controller, the robot could be made stable; however, the adaptive controller was unstable for all learning constants, β and σ, that changed the gains more than a few percent. Two possible reasons for the adaptive controller instability were theorized. The first involves the determination of the error scaling constant, C_e, and the second involves the estimation of the joint velocities using the derivative block in Simulink.

The first possible reason for the adaptive controller instability was the inability to obtain correct system identification for the robot. Incorrect system identification would result in the incorrect calculation of the error-scaling vector for each joint. Hyperstability is one of the model reference adaptive controller's requirements for stability, and if the error scaling is not appropriate for the system, the adaptive controller is not hyperstable. This is a possible explanation for the controller instability since consistent values for the motor constants, K_m and c could not be obtained. The three techniques used to determine the motor constants resulted in widely varying values, some differing by several orders of magnitude. Within each test method, the motor constants changed with test parameters, such as added inertia and input voltage.

Another possible cause for the adaptive controller's instability was calculating joint velocity using the derivative block in Simulink, rather than measuring it. The method of calculating velocity, as the change in position divided by the change in time, is very susceptible to noise. In this case, the noise was so large that the velocity measurement appeared random. In an attempt to get a useable velocity measurement, the signal was filtered. This reduced the noise, but also introduced a phase shift that may have contributed significantly to the system instability.

6.2 Future Studies

The analytical and numerical results modeled by Horner for a decentralized model reference adaptive controller on a mobile robot could not be verified due to hardware limitations. Future research should seek to resolve the issues encountered in this research.

The first suggestion for future research would be to repeat this research with a more suitable robot. A more suitable robot would have actuators whose torque and damping constants could be more accurately measured, or could be provided by the manufacturer, which would eliminate the difficulties encountered in this research in determining the motor constants. The robot should also provide a joint velocity measurement, which would eliminate the noise induced by calculating the velocity from the position measurement. Purchasing a robot that met these two requirements would substantially increase the cost of the project, but would further this research.

Another possibility for future research would be to develop a state observer for estimating velocity. This was also a topic of future research suggested Horner's thesis, the base for this mobile robot research. Horner suggested using observers to reduce the number of required sensors and to reduce the effect of sensor failure. This particular line of research should be simulated before being testing experimentally.

Other possibilities for future research would be to simulate and then test experimentally other adaptive control methods such as Least Mean Squares (LMS), self-tuning regulators, fuzzy logic controllers, and neural network controllers.

References

1. Horner, A., *Design of a Model Reference Adaptive Control for a Mobile Robot*, Virginia Tech, Blacksburg, VA, 1996.

2. Craig, J. J., "Adaptive Control of Mechanical Manipulators," *Proceedings 1986 IEEE International Conference on Robotics and Automation*, pp. 190-195.

3. Dubowsky, S., and DesForges, D. T., "The Application of Model Referenced Adaptive Control to Robotic Manipulators," *Journal of Dynamic Systems, Measurements, and Control*, Volume 101, pp. 193-200, September 1979.

4. Donalson, D. D., and Leondes, C. T., "A Model Referenced Parameter Tracking Technique for Adaptive Control Systems," *Transactions of the IEEE on Applications and Industry*, Volume 82, Issue 68, pp. 241-262, September, 1963.

5. Sardar, H. M., and Ahmadian, M., "A Modified Model Referenced Adaptive Control Technique for Nonlinear Dynamics Systems," *Journal of Vibration and Acoustics*, Volume 114, pp. 154-160, April 1992.

6. Slotine, J. E., and Li, W., "On the Adaptive Control of Robot Manipulators," *The International Journal of Robotics Research*, Volume 6, Number 3, Fall 1997.

7. Mulders, P. C., Voorkamp, R. J., Jansen, J., and van der Wolf, A. C. H., "Adaptive Control of a Modular Robot System," *CIRP ANN.*, Volume 41, Number 1, pp. 403-406, 1992.

8. Pourbograt, F., "Adaptive Learning Control for Robots," *Proceedings of the 1988 IEEE International Conference on Robotics and Automation*, pp. 862-872.

9. Seraji, H., "Adaptive Independent Joint Control of Manipulators: Theory and Experiment," *Proceedings of the 1988 IEEE International Conference on Robotics and Automation*, pp. 854-861.

10. Fu, L., "Robust Adaptive Decentralized Control of Robot Manipulators," *IEEE Transactions on Automatic Control*, Volume 27, Number 1, pp. 106-110, January 1992.

11. Colbaugh, R. Seraji, H., and Glass, K., "Decentralized Adaptive Control of Manipulators," *Journal of Robotic Systems*, Volume 11, Number 5, pp 425-440, July 1994.

12. Oh, B. J., Jamshidi, M., and Seraji, H., "Decentralized Adaptive Control," *Proceedings of the 1988 IEEE International Conference on Robotics and Automation*, pp 1016-1021.

13. Yuan, B. and Book, W. J., "Decentralized Adaptive Control of Robot Manipulators with Robust Stabilization Design," *Proceedings of the American Control Conference*, pp 102-107, 1988.

14. Landau, Y. D., *Adaptive Control – The Model Reference Approach*, Marcel Dekker, Inc., New York, 1979.

15. Tarokh, M., "Decentralized Digital Adaptive Control of Robot Motion," *Proceedings of the 1990 IEEE International Conference on Robotics and Automation*, pp. 1410-1415.

16. Stoten, D. P., *Model Reference Adaptive Control of Manipulators*, John Wiley & Sons Inc., New York, 1990.

17. Gavel, D. T., and Hsia, T. C., "Decentralized Adaptive Control Experiments with PUMA Robot Arm," *Proceedings of the 1988 IEEE International Conference on Robotics and Automation*, pp. 1022-1027.

18. Tumeh, Z. S., "Decentralized Discrete Model Referenced Adaptive Manipulator Control," *Proceedings of the 1990 IEEE International Conference on Robotics and Automation*, pp. 1410-1415.

19. Franklin, G. F., Powell, J. D., and Workman, M. L., *Digital Control of Dynamic Systems*, Addison-Welsey, New York, 1994.

20. *The Student Edition of MATLAB: Version 4 User's Guide*, Mathworks, Inc., 1995.

21. *dSpace User's Guide*, dSpace.

22. Spasov, P., *Microcontroller Technology: the 68HC11*, Prentice Hall, Inc., Columbus, Ohio, 1996.

23. HC11: M68HC11 Reference Manual, Motorola, 1991.

24. Martin, F. G., "The Miniboard Technical Reference", http://wex.www.media.mit.edu/people/fredm/papers/mb/, MIT, October 1997.

25. Glass, S. W., Ranson, C. C., Reinholtz, C. F., and Calkins, J. M., "Modular robotic applications in nuclear power plant maintenance," *Proceedings of the American Power Conference*, Volume 1, pp 421-426, Illinois Institute of Technology, 1996

26. Greiner, H., Shectman, A., Won, C. Elsley, R, and Reith, P., "Autonomous legged underwater vehicles for near land warfare," *The 1996 IEEE Symposium on Autonomous Underwater Vehicle Technology*, pp, 41-48, 1996.

27. Siebert, M. W., and Keith, T. G., "NASA Mars Pathfinder mission," *Lubrication Engineering*, Volume 54, Number 12, pp 14-19, December, 1998.

Appendix A: Derivations

This section contains the several derivations that are not necessary to the understanding of the adaptive controller, but do assist in understanding how the controller was developed. Appendix A.1 is the derivation of the constant-gain error equation. Appendix A.2 is the derivation of $\mathbf{K}_r$. Appendix A.3 is the Derivation of $\mathbf{K}$. Appendix A.4 is the derivation of Erzburger's Condition for perfect model following [16]. Appendix A.5 is the derivation of the adaptive controller's error equation. Appendix A.6 is the derivation of $\mathbf{B}_m$ to provide zero steady state error.

A.1 Constant-Gain Controller Error Equation

Start with the definition of error Equation (3.5),

$$\mathbf{e}(t) = \mathbf{x}_m(t) - \mathbf{x}(t),$$

and take its derivative,

$$\dot{\mathbf{e}}(t) = \dot{\mathbf{x}}_m(t) - \dot{\mathbf{x}}(t).$$

Then substitute this equation for $\mathbf{x}(t)$ Equation (3.1) and $\mathbf{x}_m(t)$ Equation (3.3) to get

$$\dot{\mathbf{e}}(t) = \mathbf{A}_m\mathbf{x}_m(t) + \mathbf{B}_m r(t) - \left(\mathbf{A}\mathbf{x}(t) + \mathbf{B}u(t)\right) - f_{NL}(\mathbf{x},\mathbf{t}).$$

Then substitute the constant-gain control law Equation (3.6) for $u(t)$,

$$\dot{\mathbf{e}}(t) = \mathbf{A}_m\mathbf{x}_m(t) + \mathbf{B}_m r(t) - \left(\mathbf{A}\mathbf{x}(t) + \mathbf{B}\left(\mathbf{K}_r r(t) - \mathbf{K}\mathbf{x}(t)\right)\right) - f_{NL}(\mathbf{x},\mathbf{t}),$$

and simplify to get:

$$\dot{\mathbf{e}} = \mathbf{A}_m\mathbf{x}_m + \mathbf{B}_m r - \left(\left(\mathbf{A} - \mathbf{B}\mathbf{K}\right)\mathbf{x} + \mathbf{B}\mathbf{K}_r r\right) - f_{NL}(\mathbf{x},\mathbf{t}).$$

A.2: Derivation of the Controller Gain $\mathbf{K}_r$

Begin with the condition Equation (3.9b)

$$\mathbf{B}_m - \mathbf{B}\mathbf{K}_r = 0,$$

and solve for $\mathbf{K}_r$:

$$\mathbf{B}\mathbf{K}_r = \mathbf{B}_m$$

$$\left(\mathbf{B}^T\mathbf{B}\right)^{-1}\mathbf{B}^T\mathbf{B}\mathbf{K}_r = \left(\mathbf{B}^T\mathbf{B}\right)^{-1}\mathbf{B}^T\mathbf{B}_m$$

$$\mathbf{K}_r = \left(\mathbf{B}^T\mathbf{B}\right)^{-1}\mathbf{B}^T\mathbf{B}_m .$$

A.3: Derivation of the Matrix Gain K

Begin with the condition Equation (3.9c)

$$\left(\mathbf{A}_m - \mathbf{A} + \mathbf{BK}\right) = 0 ,$$

and solve for **K**:

$$\mathbf{BK} = \mathbf{A} - \mathbf{A}_m$$

$$\left(\mathbf{B}^T\mathbf{B}\right)^{-1}\mathbf{B}^T\mathbf{K} = \left(\mathbf{B}^T\mathbf{B}\right)^{-1}\mathbf{B}^T\left(\mathbf{A} - \mathbf{A}_m\right)$$

$$\mathbf{K} = \left(\mathbf{B}^T\mathbf{B}\right)^{-1}\mathbf{B}^T\left(\mathbf{A} - \mathbf{A}_m\right).$$

A.4: Erzburger's Conditions for Perfect Model Following

For the derivation of the first condition, begin with Equation (3.9b)

$$\mathbf{A}_m - \mathbf{A} - \mathbf{BK} = 0 ,$$

and solve for **K:**

$$\mathbf{BK} = \mathbf{A}_m - \mathbf{A}$$

$$\mathbf{B}^+\mathbf{BK} = \mathbf{B}^+\left(\mathbf{A}_m - \mathbf{A}\right)$$

$$\mathbf{K} = \mathbf{B}^+\left(\mathbf{A}_m - \mathbf{A}\right).$$

Then substitute **K** back into Equation (3.9b),

$$\left(\mathbf{A}_m - \mathbf{A}\right) - \mathbf{BB}^+\left(\mathbf{A}_m - \mathbf{A}\right) = 0 ,$$

and simplify,

$$\left(\mathbf{I}_n - \mathbf{BB}^+\right)\left(\mathbf{A} - \mathbf{A}_m\right) = 0 .$$

For the derivation of the second condition begin with Equation (3.9c),

$$\mathbf{B}_m - \mathbf{BK}_r = 0 ,$$

and solve for $\mathbf{K}_r$:

$$\mathbf{BK}_r = \mathbf{B}_m$$

$$\mathbf{B}^+\mathbf{BK}_r = \mathbf{B}^+\mathbf{B}_m$$

$$\mathbf{K}_r = \mathbf{B}^+\mathbf{B}_m .$$

Then substitute $\mathbf{K}_r$ back into Equation (3.9c),

$$\mathbf{B}_m - \mathbf{B}\mathbf{B}^+\mathbf{B}_m = 0,$$

and simplify:

$$\left(\mathbf{I}_n - \mathbf{B}\mathbf{B}^+\right)\mathbf{B}_m = 0.$$

A.5: Adaptive Controller Error Equation

Start with the definition of error Equation (3.5),

$$\mathbf{e}(t) = \mathbf{x}_m(t) - \mathbf{x}(t),$$

and take its derivative to get:

$$\dot{\mathbf{e}}(t) = \dot{\mathbf{x}}_m(t) - \dot{\mathbf{x}}(t).$$

Then substitute equation for $\mathbf{x}(t)$ Equation (3.1) and $\mathbf{x}_m(t)$ Equation (3.3) to get

$$\dot{\mathbf{e}}(t) = \mathbf{A}_m\mathbf{x}_m(t) + \mathbf{B}_m r(t) - \left(\mathbf{A}\mathbf{x}(t) + \mathbf{B}u(t) + f_{NL}(\mathbf{x},t)\right).$$

Next substitute the adaptive control law Equation (3.16) for $u(t)$ to get:

$$\dot{\mathbf{e}}(t) = \mathbf{A}_m\mathbf{x}_m(t) + \mathbf{B}_m r(t) - \mathbf{A}\mathbf{x}(t) - \mathbf{B}\left(\left(\mathbf{K}_r + \delta\mathbf{K}_r\right)r(t) - \left(\mathbf{K} - \delta\mathbf{K}\right)\mathbf{x}(t)\right) - f_{NL}(\mathbf{x},t).$$

Then simplify:

$$\dot{\mathbf{e}}(t) = \mathbf{A}_m\mathbf{x}_m(t) + \mathbf{B}_m r(t) - \mathbf{A}\mathbf{x}(t) - \mathbf{B}\mathbf{K}_r r(t) - \mathbf{B}\delta\mathbf{K}_r r(t) + \mathbf{B}\mathbf{K}\mathbf{x}(t) - \mathbf{B}\delta\mathbf{K}\mathbf{x}(t) - f_{NL}(\mathbf{x},t)$$

$$\dot{\mathbf{e}}(t) = \mathbf{A}_m\mathbf{x}_m(t) + \left(\mathbf{B}_m - \mathbf{B}\mathbf{K}_r\right)r(t) + \left(-\mathbf{A} + \mathbf{B}\mathbf{K}\right)\mathbf{x}(t) - \left(\mathbf{B}\delta\mathbf{K}_r r(t) + \mathbf{B}\delta\mathbf{K}\mathbf{x}(t) + f_{NL}(\mathbf{x},t)\right)$$

$$\dot{\mathbf{e}}(t) = \mathbf{A}_m\left(\mathbf{x}_m(t) - \mathbf{x}(t)\right) + \left(\mathbf{B}_m - \mathbf{B}\mathbf{K}_r\right)r(t) + \left(\mathbf{A}_m - \mathbf{A} + \mathbf{B}\mathbf{K}\right)\mathbf{x}(t) - \left(\mathbf{B}\delta\mathbf{K}_r r(t) + \mathbf{B}\delta\mathbf{K}\mathbf{x}(t) - f_{NL}(\mathbf{x},t)\right)$$

$$\dot{\mathbf{e}}(t) = \mathbf{A}_m\mathbf{e}(t) + \left(\mathbf{B}_m - \mathbf{B}\mathbf{K}_r\right)r(t) + \left(\mathbf{A}_m - \mathbf{A} + \mathbf{B}\mathbf{K}\right)\mathbf{x}(t) - \mathbf{B}\left(\delta\mathbf{K}_r r(t) + \delta\mathbf{K}\mathbf{x}(t) - f_{NL}(\mathbf{x},t)\right).$$

A.6: Derivation of Model Input Scaling Matrix $\mathbf{B_m}$

$\mathbf{B}_m$ is selected so that at steady state the system out put is equal to the model output, which is equal to the reference input.

At steady state, the state derivatives are zero, as indicated by Equation (3.31), i.e.,

$$\dot{\mathbf{x}}_{m,ss} = 0.$$

Combining the above, along with the model in Equation (3.3) yields:

$$\mathbf{A}_m\mathbf{x}_{m,ss} + \mathbf{B}_m\mathbf{r} = 0,$$

which can be simplified as:

$$\mathbf{x}_{m,ss} = -\mathbf{A}_m^{-1}\mathbf{B}_m r$$

$$\mathbf{y}_{m,ss} = \mathbf{C}\mathbf{x}_{m,ss} = -\mathbf{C}\mathbf{A}_m^{-1}\mathbf{B}_m r$$

Then, substituting the desired output from Equation (3.30), i.e.,

$$\mathbf{y}_{ss} = \mathbf{y}_{m,ss} = \mathbf{r},$$

results in:

$$1 = -\mathbf{C}\mathbf{A}_m^{-1}\mathbf{B}_m .$$

The above can be solved for $\mathbf{B}_m$ as follows:

$$\left(\mathbf{C}^T\mathbf{C}\right)^{-1}\mathbf{C}^T = -\left(\mathbf{C}^T\mathbf{C}\right)^{-1}\mathbf{C}^T\mathbf{C}\mathbf{A}_m^{-1}\mathbf{B}_m$$

$$\mathbf{C}^+ = -\mathbf{A}_m^{-1}\mathbf{B}_m$$

$$\mathbf{B}_m = -\mathbf{A}_m\mathbf{C}^+ .$$

Vita

Donald A Gardner was born on December 28, 1973 in Arlington, Virginia where he later attended Yorktown High School. He earned a Bachelor of Science Degree in Mechanical Engineering from Virginia Polytechnic Institute and State University in 1997. Upon graduation, Mr. Gardner spent a summer in Europe as an exchange student at the Fachhochschule Bochum. He worked with a German thesis student to develop a surfboard-like input device for a virtual reality program. Mr. Gardner then returned to Virginia Polytechnic Institute and State University to pursue a Master of Science Degree in Mechanical Engineering, concentrating his studies in control theory. Upon receiving his Master of Science Degree, Mr. Gardner joined Millennium Engineering and Integration in Arlington, Virginia where he performs design analysis for theatre ballistic missile defense.