INERTIAL-SPACE DISTURBANCE REJECTION FOR ROBOTIC MANIPULATORS

NAGW-1333

by

Kevin Holt

Rensselaer Polytechnic Institute
Electrical, Computer, and Systems Engineering Department
Troy, New York 12180-3590

November 1992

CIRSSE REPORT #130

CONTENTS

LIST OF TABLES

LIST OF FIGURES

NOTATION

The following is a summary of some of the notation and conventions used throughout this report:

1. The coordinate frames of the robot are labeled 1 – 3 (platform) and 4 – 9 (PUMA). Frame 0 is the inertial frame, and frame E is the end-effector frame.

2. ${}^{k}p_{i,j} \in \Re^3$ is the vector describing the position of frame j with respect to frame i, expressed in the coordinates of frame k. Note that ${}^{k}p_{i,j} = -{}^{k}p_{j,i}$.

3. ${}^{i}_{j}R \in \Re^{3\times3}$ is the rotation matrix describing the orientation of frame j with respect to frame i.

4. ${}^{i}_{j}T \in \Re^{4\times4}$ is the homogeneous transformation describing the position and orientation of frame j with respect to frame i:

$$ {}^{i}_{j}T \triangleq \begin{bmatrix} {}^{i}_{j}R & p_{i,j} \\ 0 & 1 \end{bmatrix} $$

5. ${}^{k}\tilde{p}_{i,j} \in \Re^{3\times3}$ is the cross product matrix associated with the vector ${}^{k}p_{i,j}$, expressed in the coordinates of frame k:

$$ {}^{k}\tilde{p}_{i,j} \triangleq \begin{bmatrix} 0 & -{}^{k}p_{i,j}(z) & {}^{k}p_{i,j}(y) \\ {}^{k}p_{i,j}(z) & 0 & -{}^{k}p_{i,j}(x) \\ -{}^{k}p_{i,j}(y) & {}^{k}p_{i,j}(x) & 0 \end{bmatrix} $$

where ${}^{k}p_{i,j}(x)$, ${}^{k}p_{i,j}(y)$, and ${}^{k}p_{i,j}(z)$ are the components of ${}^{k}p_{i,j}$. By "cross product matrix", it is meant that, for all $w \in \Re^3$,

$$ {}^{k}\tilde{p}_{i,j}w = {}^{k}p_{i,j} \times w $$

Note that ${}^{l}_{k}R\,{}^{k}\tilde{p}_{i,j}\,{}^{k}_{l}R = {}^{l}\tilde{p}_{i,j}$.

6. ${}^k du_{i,j} \in \Re^6$ is the differential displacement of frame j with respect to frame i, expressed in the coordinates of frame k. The first three components of this vector are the differential translation and the last three are the differential rotation:

$$ {}^k du_{i,j} \triangleq \begin{bmatrix} {}^k dp_{i,j} \\ {}^k d\phi_{i,j} \end{bmatrix} $$

7. ${}^k J_{i,j} \in \Re^{6\times n}$ is the Jacobian relating differential joint displacements to the differential displacement of frame j with respect to frame i, expressed in the coordinates of frame k.

8. ${}^k \Phi_{j,l} \in \Re^{6\times 6}$ is the transformation that maps ${}^k J_{i,j}$ to ${}^k J_{i,l}$:

$$ {}^k \Phi_{j,l} \triangleq \begin{bmatrix} I & -{}^k \tilde{p}_{j,l} \\ 0 & I \end{bmatrix} $$

9. ${}^m_k \mathbf{R} \in \Re^{6\times 6}$ is the transformation that maps ${}^k J_{i,j}$ to ${}^m J_{i,j}$:

$$ {}^m_k \mathbf{R} \triangleq \begin{bmatrix} {}^m_k R & 0 \\ 0 & {}^m_k R \end{bmatrix} $$

10. Trigonometric functions may be abbreviated by their first letter; for example, $S_i = \sin(q_i)$ and $C_{ij} = \cos(q_i + q_j)$.

ACKNOWLEDGEMENT

The author wishes to acknowledge Dr. Alan A. Desrochers for his guidance and support during the preparation of this thesis. The author would also like to express his thanks to Dr. Steven H. Murphy and Dr. John T. Wen for their innumerable contributions and invaluable feedback, and to the students and staff at CIRSSE for their tireless efforts to maintain a quality research environment.

ABSTRACT

This report investigates the disturbance rejection control problem for a 6-DOF PUMA manipulator mounted on a 3-DOF platform. A control algorithm is designed to track the desired position and attitude of the end-effector in inertial space, subject to unknown disturbances in the platform axes. Conditions for the stability of the closed-loop system are derived. The performance of the controller is compared for step, sinusoidal, and random disturbances in the platform rotational axis and in the neighborhood of kinematic singularities.

CHAPTER 1
INTRODUCTION

1.1 Motivation

One of the main research objectives at the Center for Intelligent Robotic Systems for Space Exploration (CIRSSE) is to demonstrate the feasibility of using robotic manipulators for on-orbit tasks. In particular, robotic manipulators have been proposed as a means of reducing the amount of extra vehicular activity (EVA) time required for space station assembly and maintenance. The proposed scenario involves a robotic manipulator attached to some mobile platform, such as a spacecraft, satellite, or the space station itself.

Although certain on-orbit tasks will require only joint-space control, others will require motion with respect to an inertial or Local Vertical Local Horizontal (LVLH) reference frame [1]. In the latter case, disturbances in the platform position and attitude may prevent the manipulator from successfully completing the task. One possibility is to make course corrections using reaction wheels or jets; however, the disturbances may exceed the saturation limits of the reaction mechanism [2]. Additionally, this approach could lead to excessive attitude control fuel consumption, limiting the useful on-orbit life of the system [3]. This report explores a second possibility, namely, using the *manipulator* to compensate for platform disturbances.

1.2 Past Research

The problem of controlling a robotic manipulator on a mobile platform has received considerable attention in the past few years. Joshi and Desrochers [4, 5] designed a nonlinear feedback control law to carry out tasks (with respect to the robot base frame) in the presence of roll, pitch and yaw disturbances in the platform

axes. Dubowsky, Vance, and Torres [2] proposed a time-optimal planning algorithm for a robotic manipulator mounted on a spacecraft, subject to saturation limits in the attitude control reaction jets. Papadopoulos and Dubowsky [3] developed a general framework for analyzing the control of free-floating space manipulator systems. Most recently, Torres and Dubowsky [6] have presented a technique called the enhanced disturbance map to find manipulator trajectories that reduce the effect of disturbances in the spacecraft position and attitude.

One common assumption in the literature is that the disturbance signal is exactly known. If this is the case, then the end-effector location can be calculated without relying on direct end-point sensing. However, this assumption is invalid if there is a significant delay in the platform position and attitude measurements, or if the kinematics of the platform are not well known, or if the platform is a non-rigid structure (such as the proposed Space Station *Freedom* [7]). In the more likely case that only the *nominal* platform location and *upper bound* on the disturbance signal are known, direct end-point sensing is needed to measure the end-effector location.

1.3 Report Objective and Organization

The goal of this report is to investigate the problem of controlling a robotic manipulator in the presence of disturbances in the platform axes. Specifically, a controller is designed to track the desired position and attitude of the end-effector with respect to the inertial reference frame using end-point feedback. The platform operating point and the maximum deviation from the operating point are assumed to be known. The controller design, analysis, implementation, and performance are illustrated for a 6-DOF PUMA manipulator mounted on a 3-DOF platform.

The remainder of this report is organized as follows:

Chapter 2 defines a transformation from task space to joint space, called the approximate pseudoinverse Jacobian, which is both singularity-free and computationally efficient.

Chapter 3 examines the disturbance rejection control problem from a kinematic perspective and develops a control law for disturbance rejection based on the approximate pseudoinverse Jacobian.

Chapter 4 describes CIRSSE's robotic testbed and the software implementation of the controller on the testbed.

Chapter 5 presents several sets of experimental results. The performance of the controller is compared for various classes of disturbance signals and at the singularities of the Jacobian.

Chapter 6 summarizes this report and discusses some future directions for this area of research.

CHAPTER 2
THE APPROXIMATE PSEUDOINVERSE JACOBIAN

In the inertial-space control problem, the desired end-effector trajectory is specified in task coordinates (in this case, inertial coordinates), while the actual control takes place on the joint level. Hence, some mapping between task and joint space is required. For disturbance rejection, the transformation that maps the displacement of the end-effector to joint displacements, i.e., the inverse Jacobian, is of particular interest. However, this transformation is ill-defined for certain manipulator configurations. This chapter presents an alternative mapping, called the approximate pseudoinverse Jacobian, which is defined for all manipulator configurations.

There are five sections in this chapter. Section 2.1 discusses the forward and inverse Jacobians for the 6-DOF PUMA arm. Section 2.2 reviews the singular value decomposition and the pseudoinverse. Section 2.3 presents the definition and properties of the approximate pseudoinverse Jacobian. Section 2.4 compares the pseudoinverse and approximate pseudoinverse near singularities, as well as the cost of computing each solution. Finally, Section 2.5 summarizes the main results from this chapter.

2.1 Background

The Jacobian matrix (or *Jacobian*) is a mapping from joint space to task (Cartesian) space. It maps differential changes in joint position to differential changes in Cartesian position and orientation according to the following relationship:

$$du = J(q)dq \tag{2.1}$$

where $du \in \Re^6$ is the differential Cartesian displacement vector (linear and angular).

$q \in \Re^n$ is the vector of joint positions, $dq \in \Re^n$ is the vector of differential joint displacements, and $J \in \Re^{6\times n}$ is the Jacobian matrix. The Jacobian also relates joint velocities to Cartesian velocities.

The inverse mapping, when it exists, is given by

$$dq = J^{-1}(q)du \tag{2.2}$$

In order for J^{-1} to exist, J must be square (6×6) and full rank. The *singularities* of J are those points where the Jacobian loses rank, i.e., rank$(J) < 6$. The singularities of the PUMA Jacobian are discussed further in Section 2.1.4.

2.1.1 Coordinate Frames

The kinematic frames of the PUMA and platform are shown in Figure 2.1. The coordinate frame assignments follow the Modified Denavit-Hartenberg convention, in which coordinate frame i is attached to link i, with the origin on the axis of joint i [8]. The kinematic parameters of the PUMA and platform are listed in Table 2.1.

frame number, i	α_{i-1}	a_{i-1} (m)	d_i (m)	θ_i
1	$-90°$	0.32000	q_1	$0°$
2	$90°$	0.00000	0.54400	q_2
3	$-90°$	0.00000	0.00000	q_3
4	$90°$	0.00000	0.82800	q_4
5	$-90°$	0.00000	0.24300	q_5
6	$0°$	0.43182	-0.09391	q_6
7	$90°$	-0.02031	0.43300	q_7
8	$-90°$	0.00000	0.00000	q_8
9	$90°$	0.00000	0.00000	q_9

Table 2.1: Kinematic Parameters

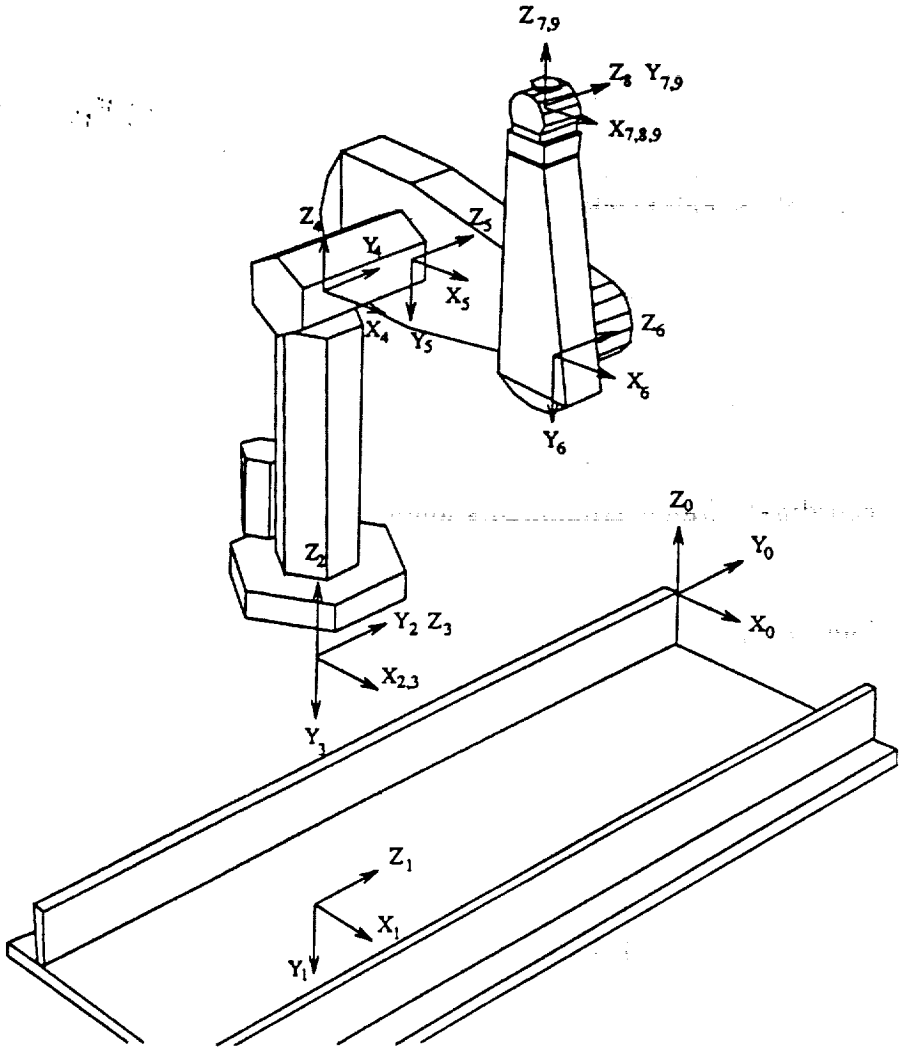

Figure 2.1: Coordinate Frame Assignments

2.1.2 Velocity and Coordinate Frame Transformations

The notation ${}^k du_{i,j}$ will be used to denote the differential displacement of frame j with respect to frame i, expressed in the coordinates of frame k. With this notation, Equation (2.1) is written as

$$ {}^k du_{i,j} = {}^k J_{i,j} dq \tag{2.3} $$

Frames i and j will be referred to as the *reference frame* and the *velocity frame*, respectively. Frame k will be referred to as the *coordinate frame*.

The velocity frame of the Jacobian can be changed through the transformation

$$ \begin{aligned} {}^k J_{i,l} &= \begin{bmatrix} I & -{}^k\tilde{p}_{j,l} \\ 0 & I \end{bmatrix} {}^k J_{i,j} \\ &\triangleq {}^k\Phi_{j,l}\, {}^k J_{i,j} \end{aligned} \tag{2.4} $$

where ${}^k\tilde{p}_{j,l}$ is the cross product matrix associated with the position vector ${}^k p_{j,l}$ (cf. the Notation at the beginning of this report). This transformation is useful for finding the differential displacement of the end-effector, ${}^k du_{i,E}$, given the position vector ${}^k p_{9,E}$ (obtained from the tool transform) and the Jacobian ${}^k J_{i,9}$.

The coordinate frame of the Jacobian can be changed via the transformation

$$ \begin{aligned} {}^m J_{i,j} &= \begin{bmatrix} {}^m_k R & 0 \\ 0 & {}^m_k R \end{bmatrix} {}^k J_{i,j} \\ &\triangleq {}^m_k \mathbf{R}\, {}^k J_{i,j} \end{aligned} \tag{2.5} $$

where ${}^m_k R$ is the rotation matrix describing the orientation of frame k with respect to frame m. Combining (2.4) and (2.5) results in

$$ {}^m J_{i,l} = {}^m_k \mathbf{R}\, {}^k\Phi_{j,l}\, {}^k J_{i,j} \tag{2.6} $$

2.1.3 Forward and Inverse PUMA Jacobians

Finding the Jacobian and its inverse expressed in any arbitrary coordinate frame can be computationally expensive. However, it is possible to take advantage of coordinate frame transformations to find the simplest Jacobian matrix [9]. For the PUMA arm, the Jacobian matrix is simplest when expressed in frame 6 [10]:

$$ {}^6J_{3,9} = \begin{bmatrix} -(d_5+d_6)C_{56} & d_7+a_5S_6 & d_7 & 0 & 0 & 0 \\ (d_5+d_6)S_{56} & a_6+a_5C_6 & a_6 & 0 & 0 & 0 \\ a_5C_5+a_6C_{56}+d_7S_{56} & 0 & 0 & 0 & 0 & 0 \\ -S_{56} & 0 & 0 & 0 & -S_7 & C_7S_8 \\ -C_{56} & 0 & 0 & -1 & 0 & -C_8 \\ 0 & 1 & 1 & 0 & C_7 & S_7S_8 \end{bmatrix} \tag{2.7} $$

Note also that this Jacobian matrix is in lower block triangular form. This is due to the geometry of spherical wrist arms; i.e., the fact that the origins of the last three frames coincide. The following compact notation will be used to denote the matrix ${}^6J_{3,9}$:

$$ {}^6J_{3,9} = \begin{bmatrix} B & 0 \\ D & E \end{bmatrix} \tag{2.8} $$

where B, D, and E are 3×3 submatrices of the Jacobian.

The inverse Jacobian, when it exists, can also be written in block matrix form (see Kailath [11], p. 656):

$$ {}^6J_{3.9}^{-1} = \begin{bmatrix} B & 0 \\ D & E \end{bmatrix}^{-1} = \begin{bmatrix} B^{-1} & 0 \\ -E^{-1}DB^{-1} & E^{-1} \end{bmatrix} \tag{2.9} $$

This expression is only defined when J is full rank, or equivalently, when B and E are full rank. The singularities of J are those points where either $\text{rank}(B) < 3$ or $\text{rank}(E) < 3$.

2.1.4 Singularities of the PUMA Jacobian

The singularities of the PUMA Jacobian can be found by solving for the roots of the determinant of J:

$$\begin{aligned} \det(J) &= \det(B)\det(E) \\ &= a_5(a_6S_6 - d_7C_6)(a_5C_5 + a_6C_{56} + d_7S_{56})S_8 \\ &= 0 \end{aligned} \tag{2.10}$$

When the first factor in (2.10) vanishes, the PUMA is at the *Arm Fully Stretched singularity.* Setting this factor to zero and solving for q_6 yields

$$q_6 = \tan^{-1}\left(\frac{d_7}{a_6}\right) + n\pi, \quad n = 0, \pm 1, \pm 2, \ldots \tag{2.11}$$

The Arm Fully Stretched configuration is classified as a *workspace boundary singularity* [8]. This singularity occurs whenever the arm switches between the `flex` and the `noflex` configurations (see [12] for the definitions of the PUMA poses). At the Arm Fully Stretched singularity, the end-effector cannot instantaneously move in certain linear directions; for example, any differential translation dp which exceeds the workspace boundary is physically unachievable.

The second factor in (2.10) corresponds to the *Hand Over Head singularity.* Setting this factor to zero and solving for q_5 yields

$$q_5 = \tan^{-1}\left(\frac{a_5 + a_6C_6 + d_7S_6}{a_6S_6 - d_7C_6}\right) + n\pi, \quad n = 0, \pm 1, \pm 2, \ldots \tag{2.12}$$

The Hand Over Head configuration is a classified as a *workspace interior singularity* [8], and corresponds to changing between the `right` and `left` configurations [12]. As in the Arm Fully Stretched singularity, certain instantaneous linear directions cannot be achieved at the Hand Over Head configuration. For example, if

$q_2 = q_3 = q_4 = 0$ and Equation (2.12) is satisfied, then instantaneous motion in the inertial Y direction is impossible.

The third factor in (2.10) corresponds to the *Wrist singularity.* Setting this factor to zero and solving for q_8 yields

$$q_8 = n\pi, \quad n = 0, \pm 1, \pm 2, \ldots \tag{2.13}$$

The Wrist singularity also occurs in the workspace interior, when the arm switches between the `flip` and `noflip` configurations [12]. At the Wrist singularity, certain instantaneous angular directions cannot be achieved; for example, if the arm is in the "home" position (as in Figure 2.1), the end-effector cannot instantaneously rotate about the inertial X axis.

2.2 Pseudoinverse Jacobian

2.2.1 Motivation

The usual method of dealing with singularities of the Jacobian is to avoid them. For example, Baillieul, Hollerbach, and Brockett [13] proposed using kinematic redundancy[1] to steer around workspace interior singularities. This approach is not applicable to the disturbance rejection problem, however, since a sufficiently large disturbance could force the manipulator into a singular configuration. There are practical problems with singularity avoidance as well. For instance, the manipulator must avoid not just singular *points*, but singular *regions*, since the norm of J^{-1} becomes very large in the neighborhood of a singularity. For disturbance rejection, then, it would be desirable to have a mapping from task space to joint space which is well-behaved near singularities. This section examines one such candidate, the *pseudoinverse Jacobian*, denoted by $J^{\dagger}$.

[1] A kinematically redundant manipulator has more degrees of freedom than required to reach every point in the workspace with arbitrary orientation; hence, $n > 6$.

In robotics literature, the pseudoinverse is often used in the context of path planning or control for kinematically redundant manipulators, to overcome the difficulty of J being a nonsquare matrix. Roboticists usually define $J^\dagger$ as

$$J^\dagger \triangleq \begin{cases} J^T(JJ^T)^{-1} & m \leq n \\ J^{-1} & m = n \\ (J^TJ)^{-1}J^T & m \geq n \end{cases} \tag{2.14}$$

Clearly, this method of computing $J^\dagger$ does not address the issue of singularities since it still relies on matrix inversion. A more general approach to computing the pseudoinverse, based on the *singular value decomposition*, is presented below.

2.2.2 The Singular Value Decomposition

The singular value decomposition (or "SVD") is the unique factorization of any $m \times n$ matrix J into the product of two orthonormal matrices and a matrix whose off-diagonal elements are zero and whose diagonal elements are the *singular values* of J. This factorization is expressed below:

$$J = U\Sigma V^T \tag{2.15}$$

where U is an $m \times m$ orthonormal matrix, V is an $n \times n$ orthonormal matrix, and Σ is the $m \times n$ matrix of singular values. For notational purposes, it will be assumed that $m \leq n$, although all of the results are still valid for $m > n$.

Since U and V are nonsingular, J and Σ have the same rank. Thus, if $\text{rank}(J) = r$, the first r singular values of J will be nonzero and the last $m - r$ equal to zero. Furthermore, it can be shown that the singular values are the nonnegative square roots of the eigenvalues of JJ^T [14]. Let the singular values be ordered as $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_m$. Then, Σ is written as

$$\Sigma = \begin{bmatrix} \sigma_1 & & & & & \\ & \ddots & & & & \\ & & \sigma_r & & & \\ & & & 0 & & \\ & & & & \ddots & \\ & & & & & 0 \end{bmatrix} = \begin{bmatrix} \Sigma_r & 0 \\ 0 & 0 \end{bmatrix} \tag{2.16}$$

The singular values can be used to measure how close J is to being singular. One such measure, the *condition number*, is defined below:

$$\text{cond}(J) \triangleq \frac{\sigma_1}{\sigma_m} \tag{2.17}$$

A matrix is nearly singular or *ill-conditioned* when the condition number is very large (or infinite). Another measure of singularity, commonly used by roboticists, is the *measure of manipulability* [15]:

$$\begin{aligned} \text{MOM}(J) &\triangleq \sqrt{\det(JJ^T)} \\ &= \sqrt{\det(\Sigma\Sigma^T)} \\ &= \sigma_1\sigma_2\cdots\sigma_m \end{aligned}$$

The measure of manipulability behaves like the inverse of the condition number, in that $\text{MOM}(J) \to 0$ as $\text{cond}(J) \to \infty$. If J is square, it is easy to see that

$$\text{MOM}(J) = \det(J) \tag{2.18}$$

2.2.3 Definition of the Pseudoinverse

Consider for the moment the task of inverting Σ when $m = n$. If Σ is full rank, then all of the singular values will be nonzero, and the inverse is simply

$$\Sigma^{-1} = \text{diag}(\frac{1}{\sigma_1}, \frac{1}{\sigma_2}, \ldots, \frac{1}{\sigma_m}) \tag{2.19}$$

In the event that a singular value σ_i is zero, Σ^{-1} does not exist. The pseudoinverse is defined by replacing these $1/\sigma_i$'s with zero:

$$\Sigma^{\dagger} \triangleq \text{diag}(\frac{1}{\sigma_1}, \frac{1}{\sigma_2}, \ldots, \frac{1}{\sigma_r}, 0, \ldots, 0) \tag{2.20}$$

By this definition, a singular value must be exactly zero for Σ to be singular. However, Σ will be numerically ill-conditioned when one or more of the σ_i's are very small. In practice, it is useful to define a *singular value threshold*, σ_{min}, below which any singular value is considered to be zero.

For the general case when Σ is not necessarily square, the pseudoinverse is defined as

$$\Sigma^{\dagger} \triangleq \begin{bmatrix} \frac{1}{\sigma_1} & & & & & \\ & \ddots & & & & \\ & & \frac{1}{\sigma_r} & & & \\ & & & 0 & & \\ & & & & \ddots & \\ & & & & & 0 \end{bmatrix} = \begin{bmatrix} \Sigma_r^{-1} & 0 \\ 0 & 0 \end{bmatrix} \tag{2.21}$$

The concept of the pseudoinverse can easily be extended to arbitrary matrices. Recall that the singular value decomposition factors J into the product $U\Sigma V^T$. Since the matrices U and V are orthonormal, $U^{-1} = U^T$ and $V^{-1} = V^T$. Thus, the pseudoinverse of J is

$$J^{\dagger} \triangleq V\Sigma^{\dagger}U^T \tag{2.22}$$

2.2.4 The Moore-Penrose Conditions

The pseudoinverse can also be defined by four algebraic properties, known as the Moore-Penrose conditions:

$$JJ^{\dagger}J = J \tag{2.23}$$

$$J^{\dagger}JJ^{\dagger} = J^{\dagger} \tag{2.24}$$

$$(JJ^{\dagger})^{T} = JJ^{\dagger} \tag{2.25}$$

$$(J^{\dagger}J)^{T} = J^{\dagger}J \tag{2.26}$$

The first condition (Equation (2.23)) is also the definition of a *generalized inverse.* That is, any matrix J^{-} which satisfies the property $JJ^{-}J = J$ is a generalized inverse of J. Similarly, (2.24) is the definition of a *reflexive generalized inverse* [15]. It is straightforward to verify that $J^{\dagger}$ is the unique matrix that satisfies all four conditions [14].

2.2.5 Properties of the Pseudoinverse

Several important properties of the pseudoinverse are listed below.

1. If J is square, then $J^{\dagger} = J^{-1}$ when J is nonsingular.

2. If J is singular and $du \in \mathcal{R}(J)$, then there are an infinite number of vectors dq that satisfy Equation (2.1). The pseudoinverse selects the least-squares solution; that is, $dq = J^{\dagger}du$ is the solution with the smallest 2-norm.

3. If J is singular and $du \notin \mathcal{R}(J)$, then there are no vectors dq that satisfy (2.1). The pseudoinverse constructs a "solution" vector that minimizes the norm of the residual; that is, $dq = J^{\dagger}du$ minimizes $\|Jdq - du\|_2$.

There are many other interesting properties of the pseudoinverse and the singular value decomposition that are not directly related to this discussion. The reader is referred to [14] or [16] for additional information.

2.3 Approximate Pseudoinverse Jacobian

2.3.1 Motivation

The pseudoinverse has one serious drawback, which is the high cost of computing the singular value decomposition. The SVD algorithm uses a series of Householder transformations to reduce the input matrix to diagonal form [17]. Since this is an $\mathcal{O}(N^3)$ operation, finding the SVD for the 6×6 Jacobian matrix can be too costly to implement in real-time (see Table 2.2 at the end of this chapter).

This motivated the search for yet another alternative to the inverse Jacobian, with the additional constraint that the number of computations be kept to a minimum. The alternative presented in this section is called the *approximate pseudoinverse Jacobian*, and is denoted by $J^{\ddagger}$.

2.3.2 Definition of the Approximate Pseudoinverse

The basic idea behind the approximate pseudoinverse is to use the partitioned form of J (cf. Equation (2.8)) and perform the SVD on the submatrices B and E. This reduces the number of computations by a factor of four, since two 3×3 singular value decompositions is an $\mathcal{O}(2(N/2)^3)$ operation.

The definition of the approximate pseudoinverse Jacobian is

$$J^{\ddagger} \triangleq \begin{bmatrix} B^{\dagger} & 0 \\ -E^{\dagger} D B^{\dagger} & E^{\dagger} \end{bmatrix} \tag{2.27}$$

where B, D, and E are defined as in (2.8). It should be noted that if J had a block-diagonal instead of a block-triangular structure (i.e., if the linear and angular

subspaces of $\Re^6$ were completely decoupled) then the approximate pseudoinverse would be identical to the pseudoinverse.

2.3.3 Properties of the Approximate Pseudoinverse

Several properties of the approximate pseudoinverse are stated below.

1. $J^\ddagger = J^{-1}$ when J is nonsingular.

2. $J^\ddagger$ does *not* satisfy the Moore-Penrose conditions when J is singular.

3. Properties (2) - (3) of Section 2.2.5 can be extended to the approximate pseudoinverse by partitioning $\Re^6$ into the linear and angular subspaces. Let $dp, d\phi \in \Re^3$ be the linear and angular components of du, respectively, and let $dq_1, dq_2 \in \Re^3$ be the components of dq. Then, the approximate pseudoinverse solution is

$$\begin{bmatrix} dq_1 \\ dq_2 \end{bmatrix} = \begin{bmatrix} B^\dagger & 0 \\ -E^\dagger D B^\dagger & E^\dagger \end{bmatrix} \begin{bmatrix} dp \\ d\phi \end{bmatrix} \tag{2.28}$$

If J is singular, the approximate pseudoinverse finds the minimum norm solution *as if dp and $d\phi$ were decoupled*; that is, $dq = J^\ddagger du$ minimizes $\|B dq_1 - dp\|_2$ and $\|E dq_2 - d\phi\|_2$.

2.4 Comparison

2.4.1 Behavior Near Singularities

Figure 2.2 compares the 2-norm, or the maximum singular value, of $J^\dagger$ (solid curve), $J^\ddagger$ (dashed curve), and J^{-1} (dotted curve) in the vicinity of the Hand Over Head singularity. Figures 2.3 and 2.4 show the behavior near the Arm Fully Stretched and Wrist singularities, respectively.

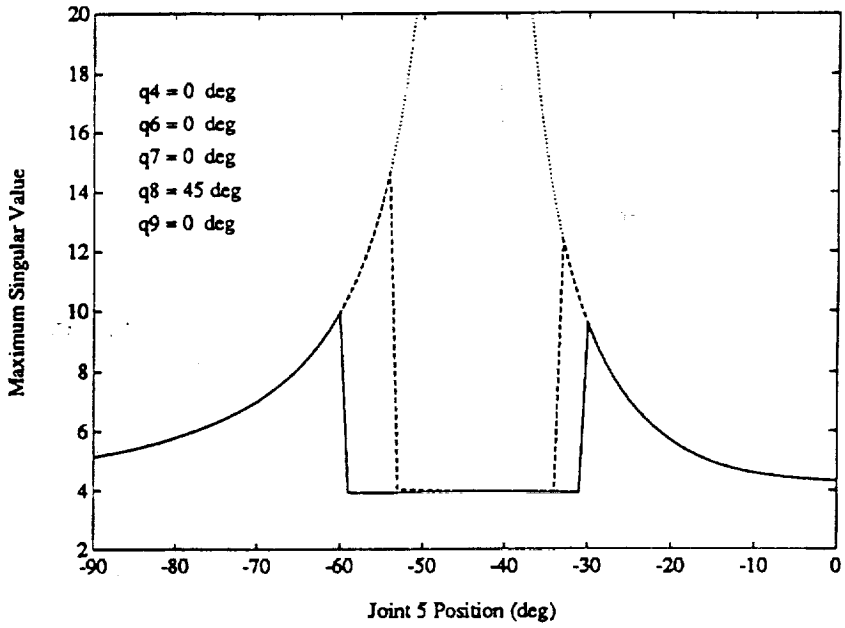

Figure 2.2: 2-Norms of $J^{\dagger}$ (solid curve), $J^{\ddagger}$ (dashed curve), and J^{-1} (dotted curve) Near Hand Over Head Singularity

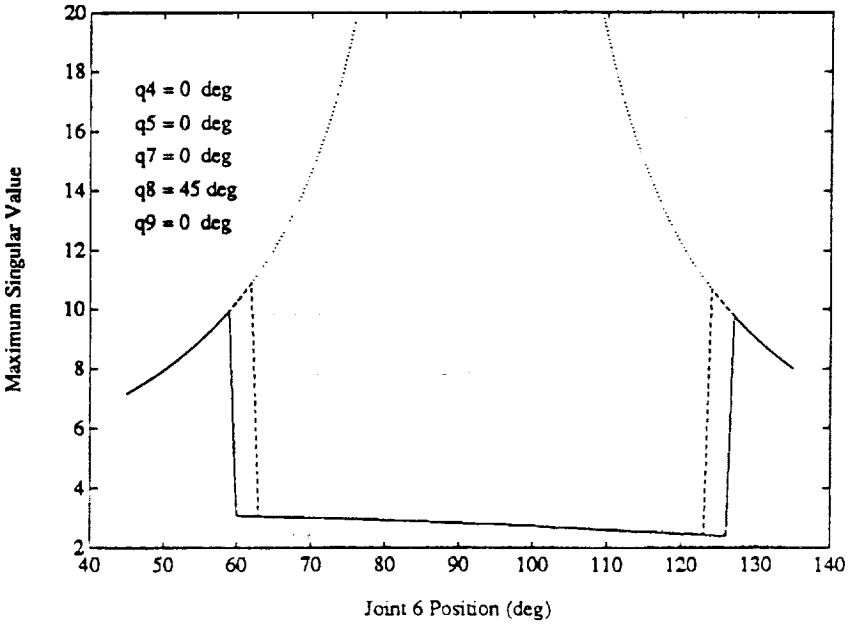

Figure 2.3: 2-Norms of $J^{\dagger}$ (solid curve), $J^{\ddagger}$ (dashed curve), and J^{-1} (dotted curve) Near Arm Fully Stretched Singularity

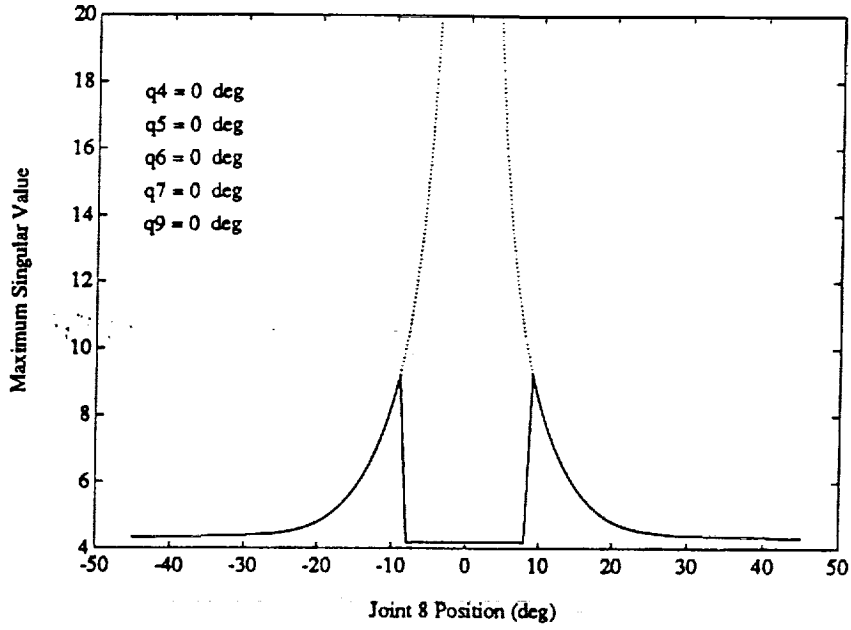

Figure 2.4: 2-Norms of $J^\dagger$ (solid curve), $J^\ddagger$ (dashed curve), and J^{-1} (dotted curve) Near Wrist Singularity

The discontinuities in $\left\|J^\dagger\right\|_2$ and $\left\|J^\ddagger\right\|_2$ occur when the smallest nonzero singular value, σ_r, falls below the threshold value, σ_{min}. This threshold is an important parameter; setting σ_{min} to a relatively small value will shrink the width of the "well" about the singular point, thus extending the range over which $J^\dagger = J^{-1}$ and $J^\ddagger = J^{-1}$. The side-effect is that the norm will be very large and highly discontinuous near the singularity. By the same token, setting σ_{min} to a relatively large value will reduce the discontinuity in the norm by increasing the width of the singular region. A threshold value of $\sigma_{min} = 0.1$ was used to generate Figures 2.2 - 2.4.

2.4.2 Bound on Approximation Error

The pseudoinverse and approximate pseudoinverse Jacobians are identical only when J is nonsingular. In order to characterize the difference in behavior at a singularity, some measure of the *approximation error* is needed.

Recall from Section 2.2.4 that a generalized inverse J^- of a matrix J is defined by the property

$$JJ^-J = J$$

Since the pseudoinverse satisfies this property, a reasonable way to measure the approximation error is to see "how close" $J^\dagger$ is to being a true generalized inverse using the following norm:

$$\left\| JJ^\dagger J - J \right\|_2$$

An upper bound on the approximation error will now be derived using this norm. Consider the matrix

$$\begin{aligned} JJ^\dagger J &= \begin{bmatrix} BB^\dagger B & 0 \\ DB^\dagger B + EE^\dagger D(I - B^\dagger B) & EE^\dagger E \end{bmatrix} \\ &= \begin{bmatrix} BB^\dagger B & 0 \\ D - (I - EE^\dagger)D(I - B^\dagger B) & EE^\dagger E \end{bmatrix} \end{aligned} \tag{2.29}$$

Subtracting J yields

$$\begin{aligned} JJ^\dagger J - J &= \begin{bmatrix} BB^\dagger B - B & 0 \\ -(I - EE^\dagger)D(I - B^\dagger B) & EE^\dagger E - E \end{bmatrix} \\ &= -\begin{bmatrix} B(I - B^\dagger B) & 0 \\ (I - EE^\dagger)D(I - B^\dagger B) & (I - EE^\dagger)E \end{bmatrix} \\ &= -\begin{bmatrix} I & 0 \\ 0 & I - EE^\dagger \end{bmatrix} \begin{bmatrix} B & 0 \\ D & E \end{bmatrix} \begin{bmatrix} I - B^\dagger B & 0 \\ 0 & I \end{bmatrix} \end{aligned} \tag{2.30}$$

When both B and E are singular, the approximation error is bounded as follows:

$$\begin{aligned} \left\| JJ^{\ddagger}J - J \right\|_2 &= \left\| \begin{bmatrix} I & 0 \\ 0 & I - EE^{\dagger} \end{bmatrix} \begin{bmatrix} B & 0 \\ D & E \end{bmatrix} \begin{bmatrix} I - B^{\dagger}B & 0 \\ 0 & I \end{bmatrix} \right\|_2 \\ &\leq \|J\|_2 \end{aligned} \tag{2.31}$$

If B is nonsingular, a less conservative upper bound can be found:

$$\begin{aligned} \left\| JJ^{\ddagger}J - J \right\|_2 &= \left\| \begin{bmatrix} I & 0 \\ 0 & I - EE^{\dagger} \end{bmatrix} \begin{bmatrix} 0 & 0 \\ 0 & E \end{bmatrix} \right\|_2 \\ &\leq \|E\|_2 \end{aligned} \tag{2.32}$$

Likewise, when E is nonsingular the upper bound reduces to

$$\begin{aligned} \left\| JJ^{\ddagger}J - J \right\|_2 &= \left\| \begin{bmatrix} B & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} I - B^{\dagger}B & 0 \\ 0 & I \end{bmatrix} \right\|_2 \\ &\leq \|B\|_2 \end{aligned} \tag{2.33}$$

Finally, if both B and E are nonsingular, the approximate pseudoinverse is identical to the pseudoinverse:

$$\left\| JJ^{\ddagger}J - J \right\|_2 = 0 \tag{2.34}$$

2.4.3 Computation Time

Table 2.2 compares the computation times of the the inverse, pseudoinverse, and approximate pseudoinverse Jacobians for each coordinate frame. As predicted, the approximate pseudoinverse is about four times faster to compute than the pseudoinverse. Note that the computation times are largest for frame 0, since the solution is first computed in frame 6 and then transformed into the desired frame k using ${}^{6}_{k}\mathrm{R}$.

Coordinate Frame k	Computation Time		
	${}^{k}J_{3,E}^{-1}$	${}^{k}J_{3,E}^{\dagger}$	${}^{k}J_{3,E}^{\ddagger}$
0	1.31 ms	25.31 ms	6.38 ms
1	1.31 ms	25.31 ms	6.38 ms
2	1.19 ms	25.31 ms	6.25 ms
3	1.09 ms	24.98 ms	6.11 ms
4	0.97 ms	24.65 ms	5.98 ms
5	0.97 ms	24.65 ms	5.98 ms
6	0.88 ms	24.98 ms	5.98 ms
7	0.82 ms	24.98 ms	5.84 ms
8	0.81 ms	24.98 ms	5.85 ms
9	0.81 ms	24.65 ms	5.85 ms
E	0.95 ms	25.31 ms	6.11 ms

Table 2.2: Computation Times for ${}^{k}J_{3,E}^{-1}$, ${}^{k}J_{3,E}^{\dagger}$, and ${}^{k}J_{3,E}^{\ddagger}$

Hence, transforming the solution into frame 0 requires the most computationally expensive rotation matrix.

The inverse, pseudoinverse, and approximate pseudoinverse Jacobian solutions were implemented in the C programming language using the GNU[2] gcc Version 2.2.2 compiler. The data in Table 2.2 was collected by timing the software on a Motorola MVME 147SA-2 Single Board Computer.

2.5 Summary

A nonsingular mapping from task space to joint space, the approximate pseudoinverse Jacobian, was defined in this chapter. The approximate pseudoinverse was compared to the inverse and pseudoinverse in terms of the computational cost and the behavior of the norm near kinematic singularities. From this comparison, it can be concluded that the approximate pseudoinverse is the clear choice for real-time control.

[2]Copyright (C) 1989, 1991 Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA.

CHAPTER 3
A KINEMATIC CONTROL LAW FOR DISTURBANCE REJECTION

This chapter focuses on the development and analysis of a control law for disturbance rejection based on the approximate pseudoinverse Jacobian. The organization of this chapter is as follows. Section 3.1 gives an overview of the inertial-space disturbance rejection control problem. Section 3.2 proposes a kinematic control law and develops an expression for the closed-loop system. Section 3.3 derives an upper bound on the control gain for closed-loop stability. Section 3.4 discusses several controller design and implementation issues, and Section 3.5 summarizes this chapter.

3.1 Overview

3.1.1 Kinematic vs. Dynamic Control

Any inertial-space controller must take into account both the kinematics and the dynamics of the manipulator. The design approach followed in this report is to partition the control into two separate loops: a kinematic loop, which outputs position setpoints for each joint based on the inertial-space error, and a dynamic loop, which outputs torques for each motor based on the joint-space error.

There are several advantages to decoupling the control in this manner. First, it allows the control designer to build and tune each loop independently. The dynamic loop, for example, can be tuned by looking only at the joint-space errors, and the kinematic loop can be tuned by assuming that the joint-level control is perfect. A second advantage is that the two controllers can run in parallel and at different sampling rates, provided that the position setpoints are buffered. For example, the dynamic loop could be implemented in hardware at a faster sampling rate than the

kinematic loop. Finally, a number of dynamic control laws, such as PID, PD-plus-gravity, computed-torque, and sliding mode control, have already been developed for robot manipulators [18]. The remainder of this chapter will concentrate on the kinematic control loop, with the assumption that a dynamic controller is already available.

3.1.2 Problem Formulation

The control problem that will be addressed in this chapter can be briefly stated as follows. Consider a 6-DOF PUMA manipulator mounted on a 3-DOF platform. The goal is to maintain the desired position and attitude of the end-effector with respect to the inertial reference frame (frame 0), subject to arbitrary disturbances in the platform axes. The following information is assumed to be available:

1. $\theta \in \Re^6$, the PUMA joint positions
2. $\eta_o \in \Re^3$, the nominal platform joint positions
3. $\bar{\delta} \in \Re^3$, the maximum deviations from the nominal platform joint positions
4. ${}^0u_{0,\varepsilon} \in \Re^6$, the inertial end-effector location

Two factors contribute to the motion of the end-effector: the differential displacement of the PUMA joints, which can be measured, and the differential displacement of the platform joints, which is unknown. Let δ denote the disturbance signal and let dv be the component of the end-effector motion caused by the differential displacement of the platform joints. Then, the differential end-effector displacement can be written as

$$\begin{aligned} {}^0du_{0,\varepsilon} &= {}^0J_{3,\varepsilon}(\eta_o + \delta, \theta)d\theta + dv \\ &= {}^0_3\mathbf{R}(\eta_o + \delta)\,{}^3J_{3,\varepsilon}(\theta)d\theta + dv \end{aligned} \tag{3.1}$$

Note that coordinate frame transformations have been applied to isolate the dependence of the PUMA Jacobian on the platform joint positions.

3.2 Discrete-Time System Analysis

3.2.1 Discrete-Time Approximation

A discrete-time model of the system will now be derived by approximating the differential quantities in (3.1) with displacements. The underlying assumption here is that the sampling period, ΔT, is sufficiently small (i.e., the sampling rate is much higher than the bandwidth of the system).

Define Δu_k as $\Delta u_k \triangleq u_k - u_{k-1}$, where the subscript k denotes the kth sample step. In the limit as ΔT goes to zero, the displacement Δu_k equals the differential du:

$$\lim_{\Delta T \to 0} \Delta u_k = du \tag{3.2}$$

Similarly, $\Delta\theta_k \to d\theta$ and $\Delta v_k \to dv$ as $\Delta T \to 0$. Therefore, the discrete-time approximation is

$$\begin{aligned} du &\approx \Delta u_k = u_k - u_{k-1} \\ d\theta &\approx \Delta\theta_k = \theta_k - \theta_{k-1} \\ dv &\approx \Delta v_k = v_k - v_{k-1} \end{aligned} \tag{3.3}$$

and the discrete version of (3.1) is

$${}^0u_k - {}^0u_{k-1} = {}^0_3\mathrm{R}(\eta_o + \delta_k)\,{}^3J_{3,E}(\theta_k)\Delta\theta_k + \Delta v_k \tag{3.4}$$

where the subscripts denoting the reference and velocity frames of du have been dropped to avoid confusion with the time index.

3.2.2 Proposed Control Law

Let ${}^0u_d$ be the desired position and orientation of the end-effector along some specified trajectory. The control objective is to drive the end-effector to this position and orientation:

$$ {}^0u_k \rightarrow {}^0u_d \quad \text{as} \quad k \rightarrow \infty \tag{3.5} $$

Ideally, the control objective could be achieved in minimum time by computing the PUMA joint displacements $\Delta\theta_d$ needed to cancel out the inertial-space error. However, exact cancellation would require complete knowledge of the disturbance signal. The next best solution then is to compute a $\Delta\theta_d$ which *approximately* cancels out the inertial-space error. With this goal in mind, the proposed control law is as follows:

$$ \Delta\theta_d = {}^3J^{\ddagger}_{3,E}(\theta_k)\,{}^3_0\mathbf{R}(\eta_o)K_c({}^0u_d - {}^0u_k) \tag{3.6} $$

where $K_c \in \Re^{6\times 6}$ is a matrix of control gains. Note that K_c can be used to weight certain components of the inertial-space error less than others; for example, setting the first column of K_c to zero would eliminate any control in the inertial X direction. Note also that this control law is essentially an inertial-space "spring", whose "stiffness" is determined by K_c. (Damping is assumed to be provided by the dynamic controller). Equation (3.6) will be referred to as the $J^{\ddagger}$ *control law* in the sequel.

3.2.3 Closed-Loop System

A simple expression for the closed-loop system can be derived by assuming that there is a one period delay in the control actuation:

$$ \Delta\theta_{k+1} = \Delta\theta_d \tag{3.7} $$

Equation (3.7) basically means that the joint-level servo control is assumed to be "perfect"; i.e., the arm achieves the desired setpoint θ_d within one sample step of the $J^\ddagger$ controller. Substituting (3.6) and (3.7) into (3.4) results in

$$ {}^0u_k - {}^0u_{k-1} = {}^0_3\mathbf{R}(\eta_o + \delta_k)\,{}^3J_{3,E}(\theta_k)\,{}^3J^\ddagger_{3,E}(\theta_{k-1})\,{}^3_0\mathbf{R}(\eta_o)K_c({}^0u_d - {}^0u_{k-1}) + \Delta v_k \quad (3.8) $$

In order to simplify this expression, define the quantity

$$ M_{k,k-1} \triangleq {}^0_3\mathbf{R}(\eta_o + \delta_k)\,{}^3J_{3,E}(\theta_k)\,{}^3J^\ddagger_{3,E}(\theta_{k-1})\,{}^3_0\mathbf{R}(\eta_o)K_c \quad (3.9) $$

Rewriting (3.8) in terms of $M_{k,k-1}$, it is easy to see that the closed-loop system is linear with time-varying coefficients:

$$ {}^0u_k = (I - M_{k,k-1})\,{}^0u_{k-1} + M_{k,k-1}\,{}^0u_d + \Delta v_k \quad (3.10) $$

A block diagram of the closed-loop system is shown in Figure 3.1.

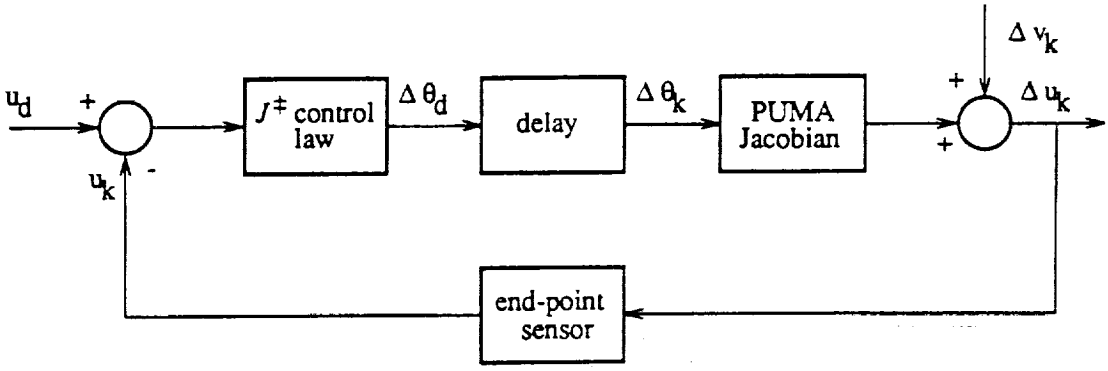

Figure 3.1: Block Diagram of Closed-Loop System

3.3 Stability Analysis

3.3.1 Spectrum of Closed-Loop System

The stability of the closed-loop system can be completely characterized by the spectrum of $I - M_{k,k-1}$. The necessary and sufficient condition for stability is that, for all $k > 0$, the eigenvalues Λ_i of $I - M_{k,k-1}$ must lie in the unit circle in the Λ-plane:

$$|\Lambda(I - M_{k,k-1})| \leq 1 \quad \forall k > 0 \tag{3.11}$$

Equivalently, the eigenvalues λ_i of $M_{k,k-1}$ must lie in a circle of unit radius centered at the point (1.0,0.0) in the Λ-plane. This can be verified by defining $\lambda_i \triangleq 1 - \Lambda_i$ and substituting into the characteristic polynomial:

$$\begin{aligned} p(\Lambda_i) &= \det(\Lambda_i I - (I - M_{k,k-1})) \\ &= -\det((1 - \Lambda_i)I - M_{k,k-1}) \\ &= -\det(\lambda_i I - M_{k,k-1}) \end{aligned} \tag{3.12}$$

Hence, the λ_i's are the eigenvalues of $M_{k,k-1}$.

The stability condition will now be expressed in terms of the matrix $M_{k,k-1}$. Define α to be the maximum angle of rotation of the eigenvalues of $M_{k,k-1}$:

$$\alpha \triangleq \sup_{i,k} \arg(\lambda_i) \tag{3.13}$$

and let (x_0, y_0) be a point on the shifted unit circle in the Λ-plane such that $\arg(x_0 + jy_0) = \alpha$ (see Figure 3.2). If ρ is the distance from the origin to (x_0, y_0), then the stability criterion can be restated as follows:

$$\bar{\sigma}(M_{k,k-1}) \leq \rho \quad \forall k > 0 \tag{3.14}$$

where $\bar{\sigma}(M_{k,k-1})$ denotes the maximum singular value of $M_{k,k-1}$.

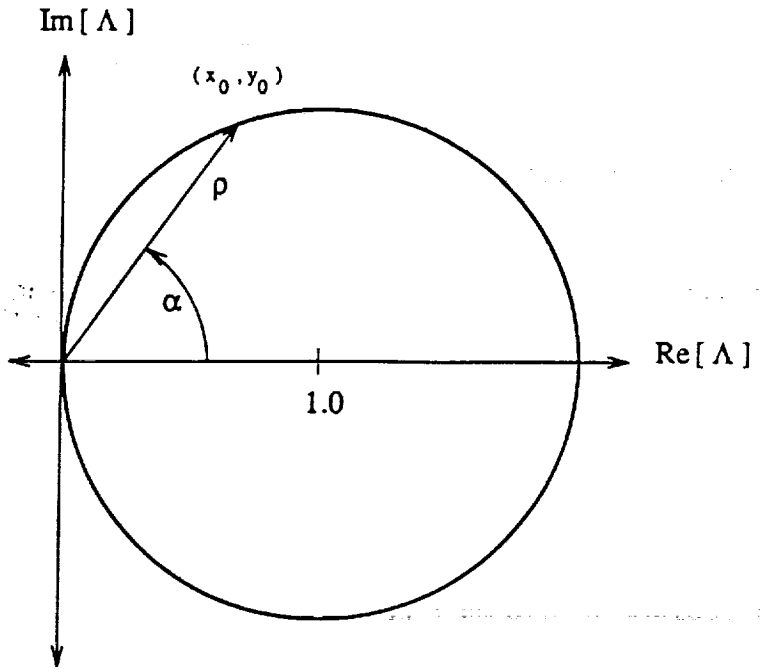

Figure 3.2: Region of Stability in the Λ-Plane

It is straightforward to find a relationship between α and ρ. The point (x_0, y_0) must simultaneously satisfy the following set of equations:

$$x_0^2 + y_0^2 = \rho^2 \tag{3.15}$$

$$(x_0 - 1)^2 + y_0^2 = 1 \tag{3.16}$$

Solving for x_0 and y_0 gives

$$(x_0, y_0) = (\frac{\rho^2}{2}, \frac{1}{2}\rho\sqrt{4 - \rho^2}) \tag{3.17}$$

Hence, α and ρ are related by

$$\alpha = \tan^{-1}(\frac{y_0}{x_0})$$

$$= \tan^{-1}\left(\frac{\sqrt{4-\rho^2}}{\rho}\right)$$
$$= \tan^{-1}\left(\sqrt{\frac{4}{\rho^2}-1}\right) \tag{3.18}$$

or, solving for ρ,

$$\rho = \frac{2}{\sqrt{\tan^2\alpha + 1}} \tag{3.19}$$

The condition for stability can therefore be written as

$$\bar{\sigma}(M_{k,k-1}) \leq \frac{2}{\sqrt{\tan^2\alpha + 1}} \quad \forall k > 0 \tag{3.20}$$

This equation will be used in Section 3.3.3 to find an upper bound on the control gain, K_c. As a first step toward deriving this bound, it is necessary to examine the spectrum of the matrix ${}^6J_{3,9}{}^6J_{3,9}^\ddagger$.

3.3.2 Spectrum of $JJ^\ddagger$

Using the compact notation for ${}^6J_{3,9}$, the matrix ${}^6J_{3,9}{}^6J_{3,9}^\ddagger$ can be written as

$${}^6J_{3,9}{}^6J_{3,9}^\ddagger = \begin{bmatrix} B & 0 \\ D & E \end{bmatrix}\begin{bmatrix} B^\dagger & 0 \\ -E^\dagger D B^\dagger & E^\dagger \end{bmatrix}$$
$$= \begin{bmatrix} BB^\dagger & 0 \\ (I-EE^\dagger)DB^\dagger & EE^\dagger \end{bmatrix} \tag{3.21}$$

Since this matrix is block triangular, the spectrum is simply the union of the eigenvalues of $BB^\dagger$ and $EE^\dagger$:

$$\lambda({}^6J_{3,9}{}^6J_{3,9}^\ddagger) = \lambda(BB^\dagger) \bigcup \lambda(EE^\dagger) \tag{3.22}$$

The following theorem completely specifies the spectrum of an arbitrary matrix times its pseudoinverse.

Theorem 3.1 *If $A \in \Re^{m\times n}$, with $m \leq n$, and* $\text{rank}(A) = r$, *then the spectrum of $AA^\dagger$ consists of $m - r$ eigenvalues at zero and r eigenvalues at one.*

Proof: Let $A = U\Sigma V^T$ be the singular value decomposition of A. Then,

$$\begin{aligned} AA^\dagger &= U\Sigma V^T V\Sigma^\dagger U^T \\ &= U\begin{bmatrix} \Sigma_r & 0 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} \Sigma_r^{-1} & 0 \\ 0 & 0 \end{bmatrix}U^T \\ &= U\begin{bmatrix} I_r & 0 \\ 0 & 0 \end{bmatrix}U^T \end{aligned} \tag{3.23}$$

Partition U into the column vectors $[u_1\ u_2\ \ldots\ u_n]$. Equation (3.23) can then be written as the sum of the outer products of the first r columns of U:

$$\begin{aligned} AA^\dagger &= u_1u_1^T + u_2u_2^T + \ldots + u_ru_r^T \\ &= \sum_{i=1}^{r} u_i\rangle\langle u_i \end{aligned} \tag{3.24}$$

The eigenvalues λ_i of $AA^\dagger$ are the solutions to

$$(AA^\dagger)\xi_i = \lambda_i\xi_i \tag{3.25}$$

It will now be shown that the eigenvectors ξ_i are the columns of U and the corresponding eigenvalues are

$$\lambda_i = \begin{cases} 1 & 1 \leq i \leq r \\ 0 & r+1 \leq i \leq m \end{cases} \tag{3.26}$$

First, consider $1 \leq i \leq r$. Since U is orthonormal, then for all $j, k \in \{1 \ldots r\}$,

$$u_j^T u_k = \begin{cases} 1 & j = k \\ 0 & j \neq k \end{cases} \tag{3.27}$$

Substituting $\xi_i = u_i$ in (3.25) and using Equations (3.24) and (3.27) results in

$$\begin{aligned}(AA^\dagger)u_i &= (u_1u_1^T + u_2u_2^T + \ldots + u_ru_r^T)u_i \\ &= u_1u_1^Tu_i + \ldots + u_iu_i^Tu_i + \ldots u_ru_r^Tu_i \\ &= u_i \end{aligned} \tag{3.28}$$

which implies that $\lambda_i = 1$. Now consider $r+1 \leq i \leq m$. Since

$$u_j^Tu_k = 0 \tag{3.29}$$

for all $j \in \{1 \ldots r\}$ and $k \in \{r+1 \ldots m\}$, then

$$\begin{aligned}(AA^\dagger)u_i &= (u_1u_1^T + u_2u_2^T + \ldots + u_ru_r^T)u_i \\ &= 0 \end{aligned} \tag{3.30}$$

which implies that $\lambda_i = 0$. ■

Returning to the original problem, suppose that $\text{rank}(B) = r$ and $\text{rank}(E) = s$. By Theorem 3.1, the complete spectrum of ${}^6J_{3,9}{}^6J_{3,9}^\dagger$ is

$$\lambda({}^6J_{3,9}{}^6J_{3,9}^\dagger) = \{\underbrace{1, \ldots, 1}_{r+s}, 0, \ldots, 0\} \tag{3.31}$$

3.3.3 Bound on Control Gain

One final condition is needed to find an upper bound on K_c. Observe that, since J is a continuous operator,

$$\|J(\theta_k) - J(\theta_{k-1})\| \to 0 \quad \text{as} \quad \|\Delta\theta_k\| \to 0 \tag{3.32}$$

In other words, for $\Delta\theta_k$ sufficiently small, J is approximately constant or *slowly time-varying*. Thus, for sufficiently small joint displacements, $M_{k,k-1} \rightarrow M_k$, where M_k is defined as

$$M_k \triangleq {}^0_3\mathbf{R}(\eta_o + \delta_k)\,{}^3J_{3,\mathcal{E}}(\theta_k)\,{}^3J^{\dagger}_{3,\mathcal{E}}(\theta_k)\,{}^3_0\mathbf{R}(\eta_o)K_c \tag{3.33}$$

The results from Sections 3.3.1 and 3.3.2 can now be used to find a condition on K_c for stability. Applying velocity and coordinate transformations to (3.33),

$$M_k = {}^0_6\mathbf{R}(\eta_o + \delta_k, \theta_k)\,{}^6\Phi_{9,\mathcal{E}}(\theta_k)\,{}^6J_{3,9}(\theta_k)\,{}^6J^{\dagger}_{3,9}(\theta_k)\,{}^6\Phi^{-1}_{9,\mathcal{E}}(\theta_k)\,{}^6_0\mathbf{R}(\eta_o, \theta_k)K_c \tag{3.34}$$

Since velocity and coordinate frame transformations are orthogonal,

$$\begin{aligned} \bar{\sigma}(M_k) &\leq \bar{\sigma}({}^6J_{3,9}(\theta_k)\,{}^6J^{\dagger}_{3,9}(\theta_k))\bar{\sigma}(K_c) \\ &\leq \bar{\sigma}(K_c) \end{aligned} \tag{3.35}$$

Hence, a sufficient condition for stability is

$$\bar{\sigma}(K_c) \leq \frac{2}{\sqrt{\tan^2\alpha + 1}} \tag{3.36}$$

3.4 Controller Design

3.4.1 Attitude Error

An important design consideration is the method used to calculate the attitude error. So far, it has been assumed that the position and orientation of the end-effector are represented by the vectors ${}^0p_k$ and ${}^0o_k$, and the inertial-space error is computed as

$${}^0u_d - {}^0u_k = \begin{bmatrix} {}^0p_d - {}^0p_k \\ {}^0\varphi_d - {}^0o_k \end{bmatrix} \tag{3.37}$$

If the orientation is represented by the rotation matrix ${}^0_E R$, however, then the components of ${}^0\phi_k$ must be extracted from ${}^0_E R$ before Equation (3.37) can be applied. Unfortunately, this approach runs into singularity problems at certain orientations. A more stable method is to use the *attitude error matrix*, defined as

$$\Delta R \triangleq {}^0_E R_d \, {}^0_E R_k^T \tag{3.38}$$

where ${}^0_E R_d$ and ${}^0_E R_k$ are the desired and actual rotation matrices. In the limit as the rotations about the inertial X, Y, and Z axes approach zero, it can be shown [19] that

$$\begin{aligned} \Delta R &\rightarrow dR \\ &\triangleq \begin{bmatrix} 1 & -d\phi_z & d\phi_y \\ d\phi_z & 1 & -d\phi_x \\ -d\phi_y & d\phi_x & 1 \end{bmatrix} \end{aligned} \tag{3.39}$$

The components $d\phi_x$, $d\phi_y$, and $d\phi_z$ represent the differential rotations about the inertial X, Y, and Z axes. Thus, for small (i.e., less than 180°) rotations about X, Y, and Z, the angular part of the inertial-space error can be formed by taking the (3,2), (1,3), and (2,1) components of ΔR:

$${}^0\phi_d - {}^0\phi_k \approx \begin{bmatrix} \Delta R(3,2) & \Delta R(1,3) & \Delta R(2,1) \end{bmatrix}^T \tag{3.40}$$

3.4.2 Design Parameters

The $J^\ddagger$ controller has two design parameters: the control gain, K_c, and the minimum singular value, σ_{min}. Some guidelines for selecting these parameters are discussed below.

The selection of the control gain is greatly simplified by restricting K_c to be a scalar times the identity matrix:

$$K_c = k_c I, \quad 0 \leq k_c \leq 2 \tag{3.41}$$

The parameter k_c controls the spectral radius of M_k. For example, if $k_c = 0.5$, then the eigenvalues of M_k will lie on a circle of radius 0.5 in the Λ-plane (or at zero, if J is singular). The region of stability can then be found by applying Equation (3.18). It is easy to verify that for $k_c = 0$, the system can tolerate up to 90° rotation in the eigenvalues of M_k (i.e., $\alpha = 90°$), and for $k_c = 2$, the system is marginally stable (i.e., $\alpha = 0°$). Thus, the choice of the control gain is a trade-off between performance (large k_c) and robustness (large α).

It is straightforward to choose a stable k_c if $\bar{\delta}$ is known *a priori*. (Recall that $\bar{\delta}$ is the vector of maximum deviations in the platform joint positions.) Let $\bar{\lambda}$ denote the spectrum of the matrix ${}^0_3\mathbf{R}(\eta_o + \bar{\delta}){}^3_0\mathbf{R}(\eta_o)$. By invoking the slowly time-varying condition, α can be approximated as follows:

$$\alpha \approx \sup_i \arg(\bar{\lambda}) \tag{3.42}$$

and k_c is calculated as

$$k_c = \frac{2}{\sqrt{\tan^2 \alpha + 1}} \tag{3.43}$$

The selection of σ_{min} is essentially a trade-off between tracking accuracy and the norm of the control signal. Recall from Section 2.4.1 that increasing σ_{min} increases the width of the singular region and consequently reduces the norm of $J^\ddagger$ at the boundary of the singular region. In terms of disturbance rejection, increasing σ_{min} causes the control in the direction of the singularity to shut off earlier, resulting in a larger tracking error. The advantage to increasing σ_{min} is that the norm of $\Delta\theta_d$ will be smaller (and less discontinuous) at the boundary of the singular region. Therefore, the selection of σ_{min} should be based on the desired upper bound on the

norm of $\Delta\theta_d$, which in turn is dictated by the saturation limits of the joint-level controller.

3.5 Summary

The design and analysis of a kinematic control law for inertial-space disturbance rejection was described in this chapter. A discrete-time model of the closed-loop system was derived, and a sufficient condition for closed-loop stability was found. The selection of the controller design parameters and the computation of the attitude error were also discussed.

CHAPTER 4
IMPLEMENTATION ON A ROBOTIC TESTBED

This chapter gives an overview of CIRSSE's robotic testbed and some of the software used in the implementation of the $J^{\ddagger}$ controller on the testbed. Section 4.1 describes the platform carts and the PUMA arms. Section 4.2 details the hardware-level interface and real-time operating systems. Sections 4.3 and 4.4 discuss the software used to control the robots, and Section 4.5 summarizes this chapter.

4.1 Robot Hardware

4.1.1 Platform Carts

The platform system, custom built by K.N. Aronson, Inc. of Arcade, NY, consists of two 3-DOF carts on a 12 ft linear track. The platform joints are labeled 1 – 3 for the left cart and 10 – 12 for the right cart. Joint 1 provides translational motion for the left cart along the track, while joints 2 and 3 provide tilt and pan, respectively. A diagram of the platform system is shown in Figure 4.1.

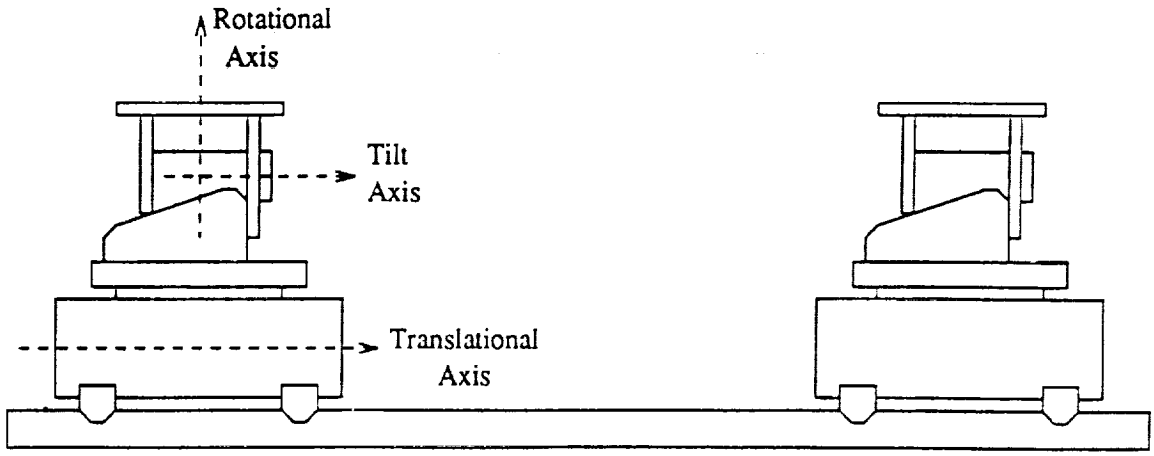

Figure 4.1: 3-DOF Platform Carts

4.1.2 PUMA Arms

Mounted on the platform system is a pair of 6-DOF PUMA arms, built by Unimation, Inc. of Danbury, CT. The joints of the left arm (Unimation model 560) are labeled 4 – 9 and the joints of the right arm (Unimation model 600) are labeled 13 – 18. The left PUMA and platform cart are shown in Figure 4.2.

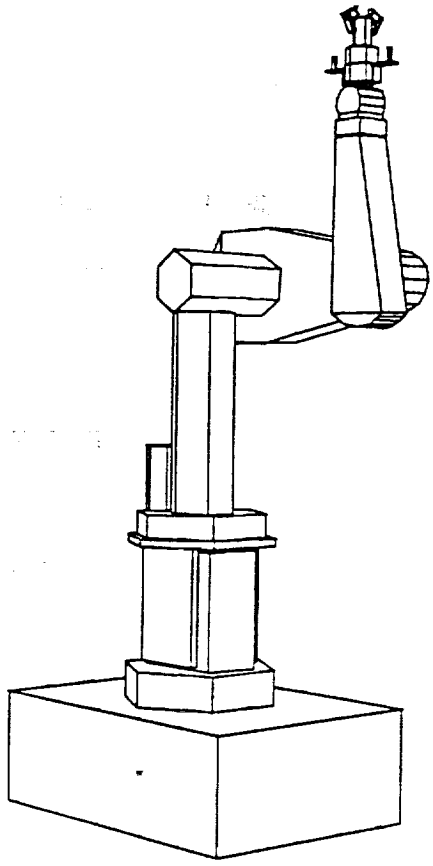

Figure 4.2: Left PUMA and Platform Cart

Each PUMA arm is equipped with a force/torque sensor and a pneumatic gripper. Additionally, two cameras are mounted on a bracket located at the flange of the left PUMA. The physical dimensions of the force/torque sensor, camera mount, and gripper are taken into account by the tool transform, ${}^{9}_{E}T$, which specifies the position and orientation of the end-effector frame. For the left robot, the origin

of frame E is located between the jaws of the gripper, 23.9 cm from the origin of frame 9 [20].

4.2 Computer Control System

4.2.1 Hardware Interface

The platform and PUMA robots are controlled from a VME chassis which contains a number of hardware components distributed across the bus. The bulk of the real-time computation takes place on three Motorola MVME 147SA-2 and two Motorola MVME 135 Single Board Computers (SBCs), labeled CPU 0 - CPU 5. A Motorola MVME 224-1 Shared Memory board provides a common address space for the CPUs.

The platform encoders are accessed via three Whedco VME-3570-1 Dual Channel Encoder Interface boards. A Datel DVME-628V D/A board is used to drive the platform servo motors. Digital lines, such as platform power, limit switches, and emergency stop switches are interfaced through a VME Microsystems VMIVME-2532A High-Voltage Digital I/O board.

The encoder, torque, and power signals for the PUMA robots are handled by two Unimation controller boxes outside of the VME chassis. They are connected to the VME chassis by two VMEbus to Q-Bus adapters.

The five SBCs are installed on an Ethernet backplane, which allows communication between the VME chassis, a separate Datacube VME chassis (for computer vision), and the Sun workstations on the CIRSSE network.

4.2.2 Operating Systems

Each CPU runs under VxWorks[1], a UNIX-compatible real-time operating system. Among other things, the VxWorks kernel supports priority based scheduling,

[1]Wind River Systems, Alameda, CA.

intertask communication, synchronization, interrupt handling, and memory management.

However, VxWorks does *not* provide a mechanism for tasks on separate CPUs to communicate with each other. In order to facilitate interprocessor communication, the CIRSSE Testbed Operating System (CTOS) was developed [21]. CTOS enables tasks to communicate asynchronously via message passing.

In addition to interprocessor communication, CTOS also supports interchassis communication. For example, CTOS allows a task on CPU 5 to send and receive messages from a task on a Sun workstation (running under UNIX). This communication bridges the gap between synchronous (real-time) and asynchronous (non-real-time) tasks.

4.3 Motion Control System

The Motion Control System (MCS) is a collection of real-time software components that provides joint-level servo control, force/torque-based control, setpoint interpolation, and trajectory generation. The portions of the MCS relevant to this report, as well as the software implementation of the $J^{\ddagger}$ control law, are discussed below.

The MCS is loaded onto CPUs 0 – 5 at boot-time and can easily be configured to meet the needs of a particular experiment. For this thesis, the $J^{\ddagger}$ controller was used in place of the standard MCS trajectory generator.

4.3.1 Channel I/O Drivers

The platform and PUMA channel drivers are responsible for handling the robot I/O, including: torque commands, power and brake commands, emergency stop and limit switch status, encoder positions, and encoder calibration. The channel drivers run at the servo rate, which is typically $1/0.0045\ s^{-1}$.

The torque and position information for each joint is mapped onto a unique *slot* in shared memory, and can be accessed using the library `chanLib`. This allows tasks on other CPUs (e.g., servo controllers) to exchange data with the channel drivers in a synchronous fashion. Asynchronous information, such as power and calibration commands, is sent via CTOS messages.

4.3.2 Inertial End-Point Sensor Driver

A separate driver was written for this thesis to measure the location of the end-effector in inertial space. In lieu of a direct end-point sensor, the forward kinematics are used to compute ${}^0_E T_k$, the homogeneous transform describing the current position and attitude of the end-effector with respect to frame 0. (Note that this software is a temporary substitute for direct end-point feedback; the forward kinematics can *not* be used in practice since the platform joint positions are needed to calculate ${}^0_E T_k$.) The end-effector transform is stored in a shared memory slot and can be accessed via the library `chanIESLib`. The inertial end-point sensor driver was implemented with a sampling rate of $1/0.0036\ s^{-1}$.

4.3.3 Joint-level Servo Controllers

The platform and PUMA controllers compute the torques required to servo each joint to the desired setpoint. Position and velocity setpoints are passed to the controllers via the library `interpLib`, which uses linear interpolation to smooth the desired trajectory. The controllers run at the servo rate, in lock step with the channel drivers.

The control algorithm for the PUMA is based on the well-known Proportional-plus-Integral-plus-Derivative (PID) control law [22, 23]. To reduce the coupling between the joints, the PID torques are multiplied by the diagonal terms of $M(\theta)$,

the mass matrix[2]. Gravity compensation was also added to further reduce the position error. Thus, the control law for the PUMA arm is

$$\Gamma = M(\theta_k)(K_P(\theta_d - \theta_k) + K_I \sum_{i=0}^{k}(\theta_d - \theta_k)\Delta T + K_D(\dot{\theta_d} - \dot{\theta_k})) + g(\theta_k) \qquad (4.1)$$

where

Γ	is the 6×1 vector of joint torques
$M(\theta_k)$	is the 6×6 mass matrix (diagonal terms only)
$\theta_d - \theta_k$	is the 6×1 vector of position errors
$\dot{\theta_d} - \dot{\theta_k}$	is the 6×1 vector of velocity errors
K_P	is a 6×6 diagonal matrix of proportional gains
K_I	is a 6×6 diagonal matrix of integral gains
K_D	is a 6×6 diagonal matrix of velocity gains
$g(\theta_k)$	is the 6×1 vector of gravity torques
ΔT	is the sampling period

In addition, a first order low-pass filter is used to attenuate the noise in the joint velocity estimates, $\dot{\theta_k}$. The control algorithm for the platform is identical to (4.1), with $M = I$ and $g = 0$.

4.3.4 $J^{\ddagger}$ Controller

The $J^{\ddagger}$ controller functions like a trajectory generator, in that it supplies position setpoints to the PUMA servo controller through the `interpLib` interface (velocity setpoints are set to zero). The position setpoints are calculated by adding the control vector $\Delta\theta_d$ to the the current joint positions θ_k, where $\Delta\theta_i$ is computed as in Equation (3.6). The library `jacLib`, in particular the function `jacPumaAprxPseudoInv()`, is used to find the approximate pseudoinverse solution. The sampling rate of this controller is $1/0.0081\ s^{-1}$.

[2] $M(\theta)$ is the matrix that multiplies $\ddot{\theta}$ in the Lagrange-Euler dynamics of the PUMA.

The inputs to the $J^\sharp$ controller are the desired and current end-effector transforms, ${}^0_E T_d$ and ${}^0_E T_k$, from which the inertial-space error is extracted (cf. Section 3.4.1). The current end-effector transform is read from shared memory using the inertial end-point sensor library described in Section 4.3.2. The desired transform is read from a file during controller initialization. Ideally, the desired transform would be specified on-line by a task-space trajectory generator; however, this functionality was not available at the time of this thesis.

4.3.5 State Manager

The MCS State Manager coordinates the bootstrapping phase of the Motion Control System by sending messages to the various components (channel drivers, controllers, etc.) at boot-time. The State Manager also implements a simple state machine for the testbed. The five states of the MCS are:

Cold - MCS initialization.

Reserve - Application reserves slots.

Active - Robot power on, brakes on.

Motion - Robot power on, brakes off.

Emergency Stop - Emergency stop button pressed. Robot power off.

4.4 Software Libraries

In addition to the Motion Control System, several libraries of routines were used for this thesis. These libraries are briefly described below.

4.4.1 Transform Library

The Transform Library, or `transLib`, is a collection of routines that operate on homogeneous transforms. In particular, the routines `transInvert()` and

`transMultiply()` are used by the $J^{\ddagger}$ controller to perform the transform inversion and multiplication required to compute ΔR, the attitude error matrix. A CIRSSE Technical Memorandum describing the Transform Library is forthcoming.

4.4.2 Kinematics Library

The Kinematics Library, or `kinLib`, includes functions to perform the forward and inverse kinematics. The routine `kinFwd()` is used by the inertial end-point sensor driver to compute the current end-effector transform ${}^{0}_{E}T_{k}$. The functional interface for the Kinematics Library is described in [12].

4.4.3 Jacobian Library

The Jacobian Library, or `jacLib`, contains routines for computing the solutions to the forward, forward transpose, inverse, pseudoinverse, and approximate pseudoinverse Jacobian equations. The implementation details as well as the functional interface for the Jacobian Library are explained in [24].

The approximate pseudoinverse solution uses an algorithm developed by Press, Flannery, et al. [17] to perform the singular value decomposition. This algorithm can be found in the library `slvLinEqn`.

4.5 Summary

The major hardware and software components of the CIRSSE testbed were described in this chapter. The real-time implementation of the joint-level PID controller and the $J^{\ddagger}$ kinematic controller were discussed, as well as some of the supporting software, such as the inertial end-point sensor library and Jacobian library.

CHAPTER 5
EXPERIMENTAL RESULTS

This chapter presents the results of four sets of experiments utilizing the testbed. The goal of the experiments was to demonstrate the performance of the $J^\ddagger$ control law under various operating conditions. The first three sets of experiments focused on the time response of the closed-loop system for the following classes of disturbances:

1. Step disturbances in the platform joints
2. Sinusoidal disturbances in the platform joints
3. Random disturbances in the platform joints

The majority of disturbances that are likely to be encountered by the robot can be decomposed into signals belonging to these three classes; for example, an impulsive disturbance can be approximated as a combination of positive and negative step disturbances. The fourth set of experiments was aimed at understanding the open-loop characteristics of the control law in the neighborhood of singularities.

This chapter is organized into five sections. Sections 5.1 – 5.3 discuss the performance of the $J^\ddagger$ control law for step, sinusoidal, and random disturbances in the platform rotation. (Results for the platform translational and tilt axes are qualitatively similar, and are excluded for the sake of brevity.) Section 5.4 examines the behavior of the $J^\ddagger$ control law near the singularities of the PUMA. Section 5.5 presents some conclusions based on the experimental results.

In this chapter, the term *orientation error* will refer to the equivalent angle of rotation ϕ_e of the attitude error matrix ΔR (cf. Equation (3.38)). The orientation error is found by computing the equivalent axis and angle representation of ΔR:

$$\Delta R = e^{\tilde{k}\phi_e}, \quad 0 \leq \phi_e \leq 180^\circ \tag{5.1}$$

where $\tilde{k} \in \Re^3$ is the normalized axis of rotation and ϕ_e is the scalar representing the equivalent angle of rotation. An algorithm for extracting $\tilde{k}$ and ϕ_e from an attitude matrix is given in [19].

5.1 Step Disturbances in Platform Rotation

This section analyzes the time response of the closed-loop system for 10°, 20°, and 30° step disturbances in the platform rotation. For each case, the control gain K_c was set to identity.

5.1.1 10° Step Disturbance

Figure 5.1 shows the inertial-space errors errors when a 10° step disturbance is applied to the platform rotational joint. The linear (X, Y, and Z) components of the error are shown in the upper plot and the orientation error in the lower plot. The components of $\Delta\theta_d$, the control vector, are plotted in Figure 5.2.

	Maximum Overshoot	4% Settling Time
X	$1.527 \times 10^{+0}$ cm	1.54 s
Y	$3.825 \times 10^{+0}$ cm	0.84 s
Z	6.366×10^{-1} cm	1.70 s
ϕ_e	$3.503 \times 10^{+0}$ deg	1.00 s

Table 5.1: Maximum Overshoot and 4% Settling Time for 10° Step Disturbance in Platform Rotation

Table 5.1 lists the maximum overshoot and 4% settling time for the X, Y, Z, and orientation errors. The 4% settling time refers to the time required for the error to enter and remain within $\pm\epsilon$ of zero, where ϵ is 4% of the peak absolute error.

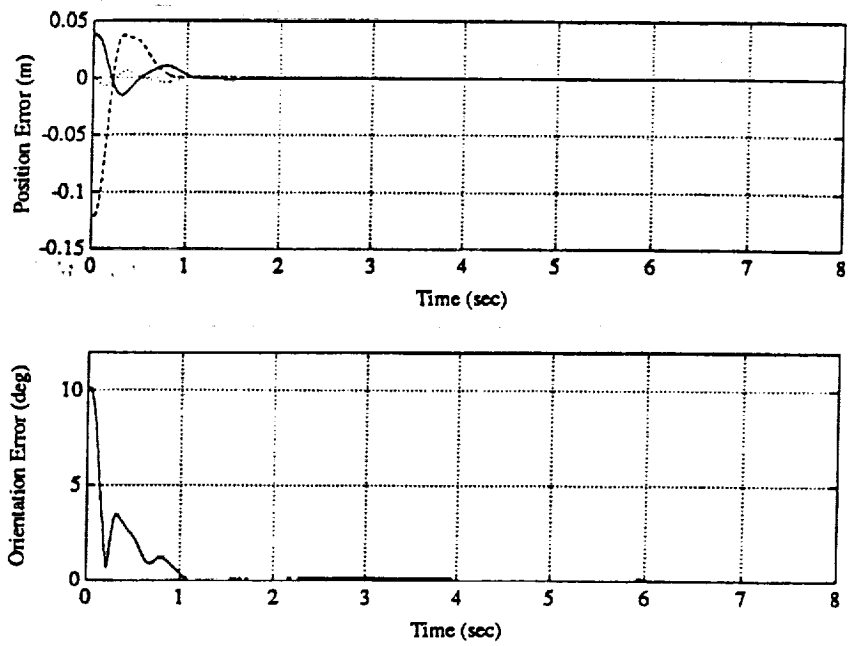

Figure 5.1: Position Error (X – solid curve; Y – dashed curve; Z – dotted curve) and Orientation Error for 10° Step Disturbance in Platform Rotation

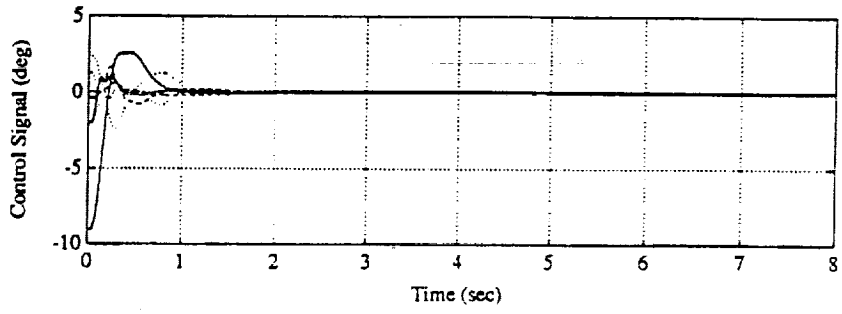

Figure 5.2: Control Signals ($\Delta\theta_d(1)$, $\Delta\theta_d(4)$ – solid curves; $\Delta\theta_d(2)$, $\Delta\theta_d(5)$ – dashed curves; $\Delta\theta_d(3)$, $\Delta\theta_d(6)$ – dotted curves) for 10° Step Disturbance in Platform Rotation

5.1.2 20° Step Disturbance

Figures 5.3 and 5.4 show the inertial-space errors and control signals for a 20° step disturbance. The settling time and overshoot for X, Y, Z, and ϕ_e are listed in Table 5.2.

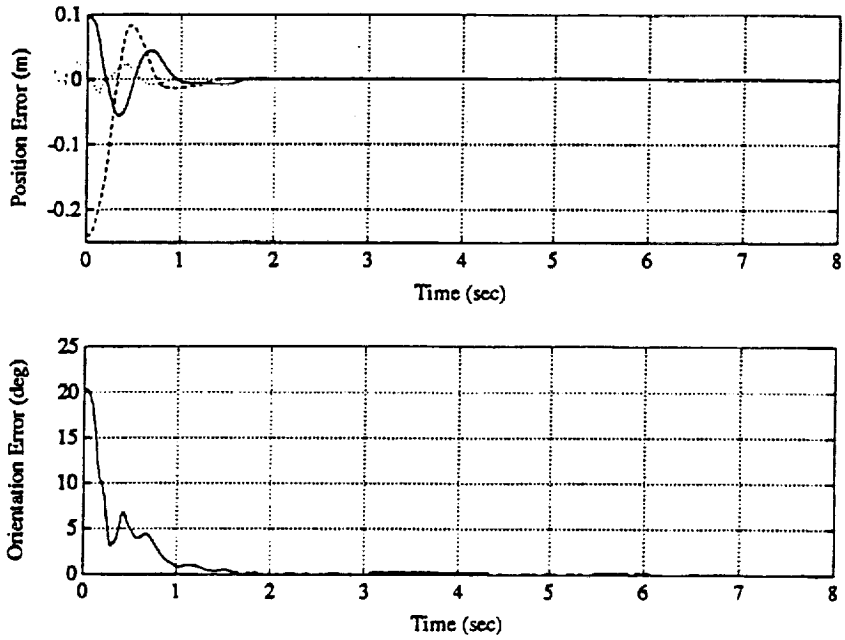

Figure 5.3: Position Error (X – solid curve; Y – dashed curve; Z – dotted curve) and Orientation Error for 20° Step Disturbance in Platform Rotation

	Maximum Overshoot	4% Settling Time
X	$5.526 \times 10^{+0}$ cm	1.60 s
Y	$8.283 \times 10^{+0}$ cm	1.33 s
Z	$2.435 \times 10^{+0}$ cm	2.06 s
ϕ_e	$6.806 \times 10^{+0}$ deg	1.27 s

Table 5.2: Maximum Overshoot and 4% Settling Time for 20° Step Disturbance in Platform Rotation

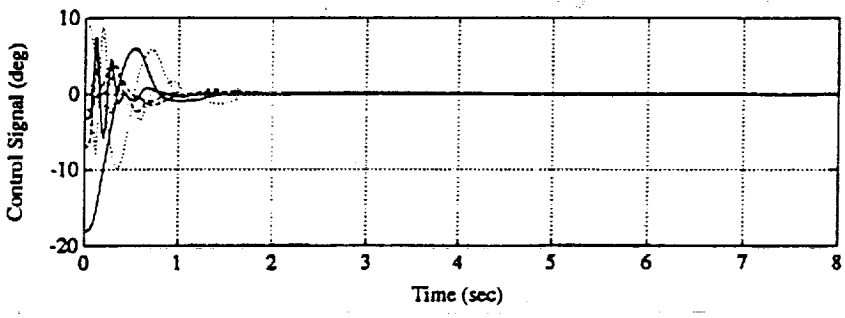

Figure 5.4: Control Signals ($\Delta\theta_d(1)$, $\Delta\theta_d(4)$ – solid curves; $\Delta\theta_d(2)$, $\Delta\theta_d(5)$ – dashed curves; $\Delta\theta_d(3)$, $\Delta\theta_d(6)$ – dotted curves) for 20° Step Disturbance in Platform Rotation

5.1.3 30° Step Disturbance

The inertial-space errors and control signals for the 30° case are shown in Figures 5.5 and 5.6. The maximum overshoot and settling time for each coordinate are displayed in Table 5.3.

	Maximum Overshoot	4% Settling Time
X	$1.737\times10^{+1}$ cm	1.97 s
Y	$1.706\times10^{+1}$ cm	2.43 s
Z	$1.253\times10^{+1}$ cm	1.66 s
ϕ_e	$2.055\times10^{+1}$ deg	2.08 s

Table 5.3: Maximum Overshoot and 4% Settling Time for 30° Step Disturbance in Platform Rotation

5.1.4 Comparison

The maximum overshoot and settling time provide a measure of the relative degree of stability of the closed-loop system. For example, the maximum overshoot in the X direction is about 1.5 cm for the 10° case, 5.5 cm for the 20° case, and 17.4 cm for the 30° case, indicating that the degree of stability decreases as the magnitude of the disturbance increases. Likewise, the settling times are generally

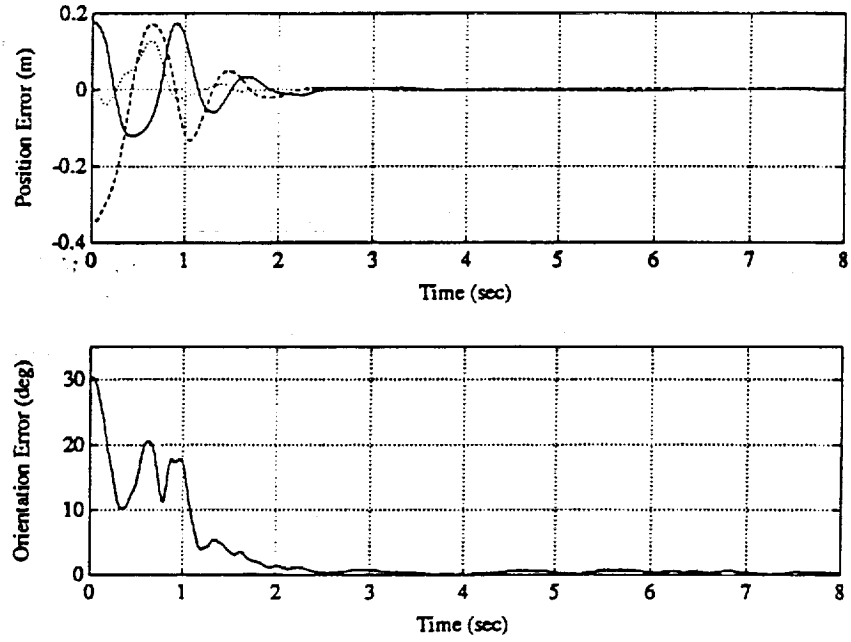

Figure 5.5: Position Error (X – solid curve; Y – dashed curve; Z – dotted curve) and Orientation Error for 30° Step Disturbance in Platform Rotation

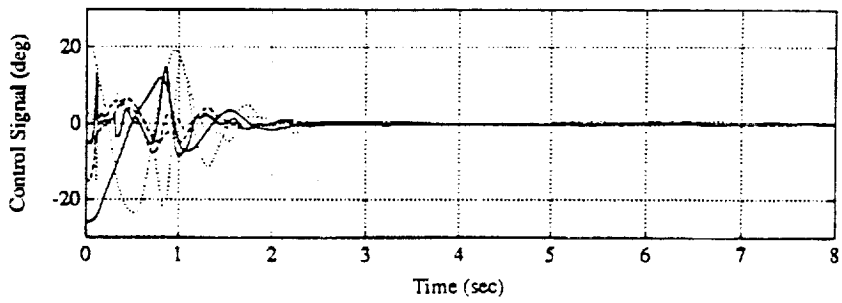

Figure 5.6: Control Signals ($\Delta\theta_z(1)$, $\Delta\theta_z(4)$ – solid curves; $\Delta\theta_z(2)$, $\Delta\theta_z(5)$ – dashed curves; $\Delta\theta_z(3)$, $\Delta\theta_z(6)$ – dotted curves) for 30° Step Disturbance in Platform Rotation

longer for larger magnitude disturbances.

Figure 5.7 shows a plot of the position error in the $X - Y$ plane for 10°, 20°, and 30° step disturbances. This view corresponds to looking in the negative Z direction, or "down", from directly above the robot (see Figure 2.1). It is interesting to note that for the 10° case, the end-effector converges to the desired position along a roughly straight-line path, while for the 30° case, the path resembles a spiral. This spiraling is caused by the error in the approximate pseudoinverse Jacobian matrix, which is computed using the nominal platform position instead of the true platform position.

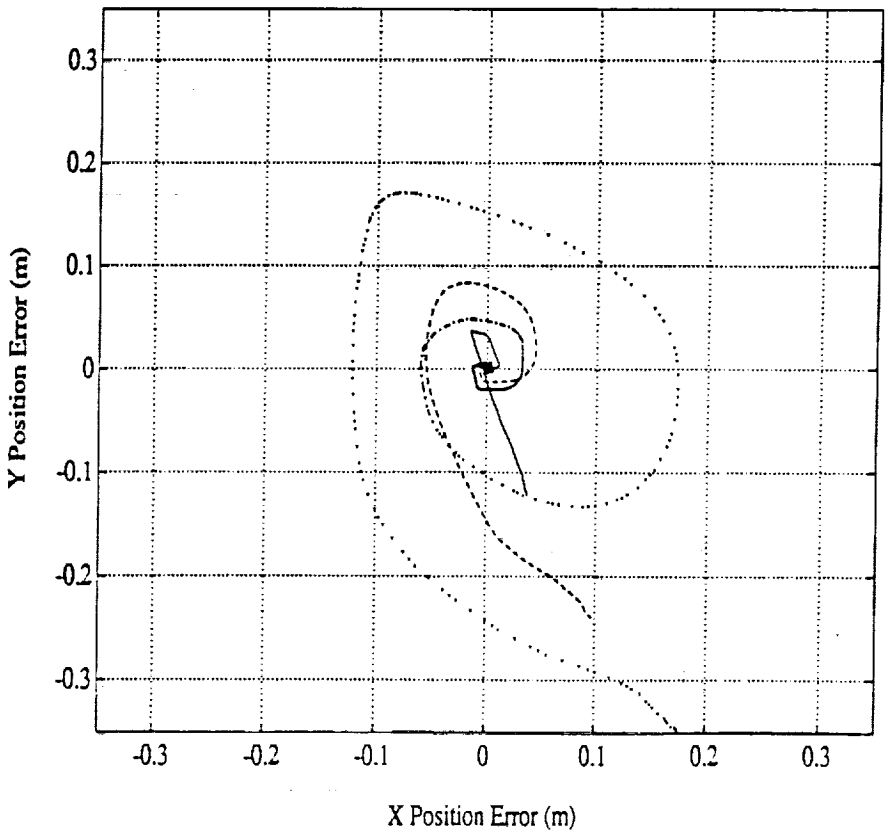

Figure 5.7: Position Errors for 10° (solid curve), 20° (dashed curve), and 30° (dotted curve) Step Disturbances in Platform Rotation

5.2 Sinusoidal Disturbances in Platform Rotation

This section compares the time response of the system (with $K_c = I$) for 16 second, 8 second, and 4 second period sinusoidal disturbances in the platform rotation.

5.2.1 16 Second Period Sinusoidal Disturbance

Figure 5.8 shows the X, Y, and Z position errors, with and without disturbance rejection, for a 10° amplitude, 16 second period sinusoidal disturbance in the platform rotation. The orientation error, with and without disturbance rejection, is shown in Figure 5.9. Table 5.4 displays the largest absolute error and mean-square error for each coordinate.

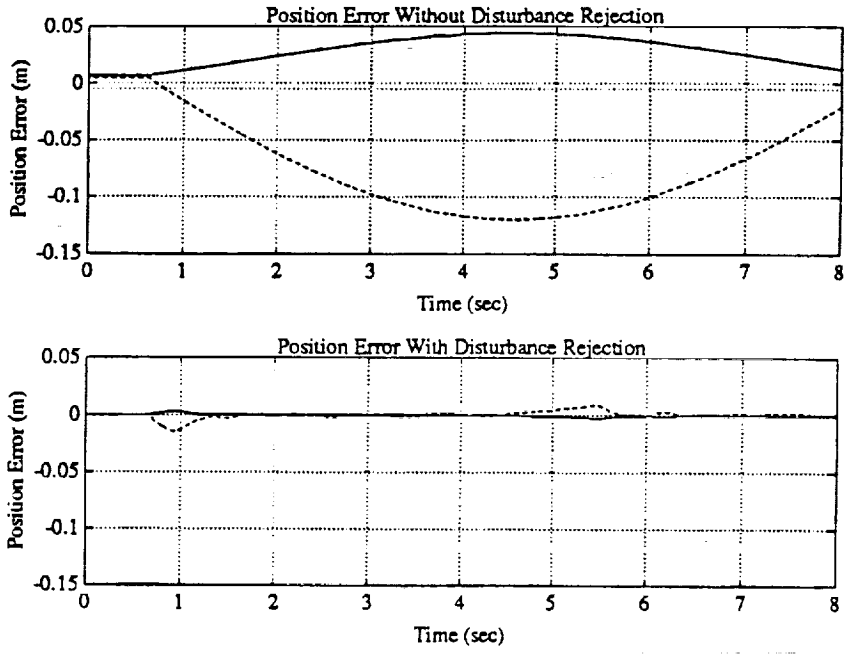

Figure 5.8: Position Errors (X – solid curves; Y – dashed curves; Z – dotted curves) for 10° Amplitude, 16 Second Period Sinusoidal Disturbance in Platform Rotation

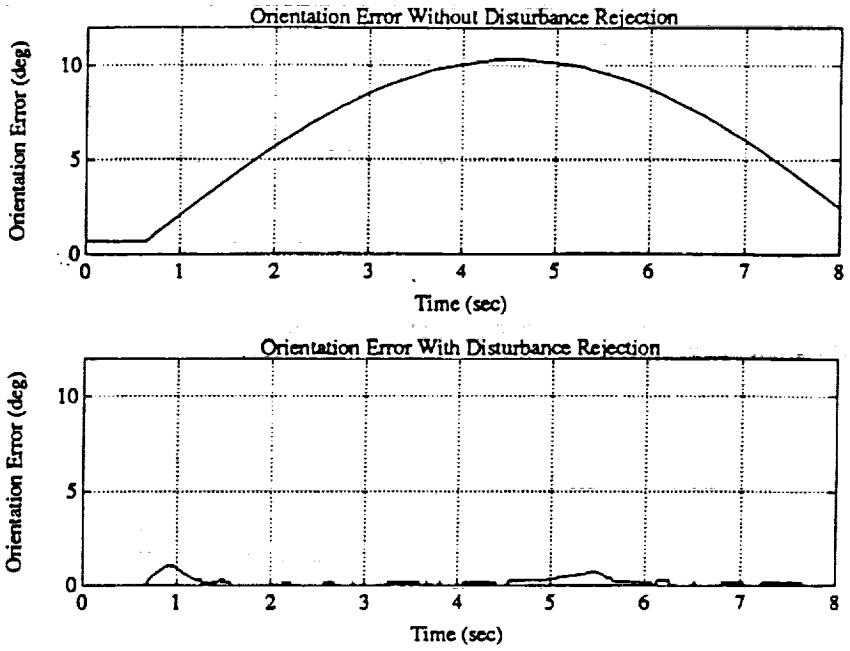

Figure 5.9: Orientation Errors for 10° Amplitude, 16 Second Period Sinusoidal Disturbance in Platform Rotation

	Without Disturbance Rejection		With Disturbance Rejection	
	Max Error	MSE	Max Error	MSE
X	$4.414 \times 10^{+0}$ cm	$9.641 \times 10^{+0}$ cm^2	3.240×10^{-1} cm	7.220×10^{-3} cm^2
Y	$1.200 \times 10^{+1}$ cm	$6.915 \times 10^{+1}$ cm^2	$1.451 \times 10^{+0}$ cm	1.275×10^{-1} cm^2
Z	5.110×10^{-1} cm	2.511×10^{-1} cm^2	6.900×10^{-2} cm	4.364×10^{-4} cm^2
ϕ_e	$1.034 \times 10^{+1}$ deg	$5.332 \times 10^{+1}$ deg^2	$1.057 \times 10^{+0}$ deg	7.178×10^{-2} deg^2

Table 5.4: Maximum and Mean-Square Errors for 10° Amplitude, 16 Second Period Sinusoidal Disturbance in Platform Rotation

5.2.2 8 Second Period Sinusoidal Disturbance

Figures 5.10 and 5.11 show the position and orientation errors for a 10° amplitude, 8 second period sinusoidal disturbance in the platform rotation. Table 5.5 shows the maximum and mean-square position and orientation errors.

	Without Disturbance Rejection		With Disturbance Rejection	
	Max Error	MSE	Max Error	MSE
X	$3.924\times10^{+0}$ cm	$4.230\times10^{+0}$ cm^2	4.940×10^{-1} cm	3.251×10^{-2} cm^2
Y	$1.298\times10^{+1}$ cm	$8.166\times10^{+1}$ cm^2	$2.171\times10^{+0}$ cm	3.336×10^{-1} cm^2
Z	1.900×10^{-2} cm	3.459×10^{-4} cm^2	1.310×10^{-1} cm	2.509×10^{-3} cm^2
ϕ_e	$1.014\times10^{+1}$ deg	$4.865\times10^{+1}$ deg^2	$1.665\times10^{+0}$ deg	2.172×10^{-1} deg^2

Table 5.5: Maximum and Mean-Square Errors for 10° Amplitude, 8 Second Period Sinusoidal Disturbance in Platform Rotation

5.2.3 4 Second Period Sinusoidal Disturbance

Figures 5.12 – 5.13 show the position and orientation errors for a 10° amplitude, 4 second period sinusoidal disturbance in the platform rotational joint. Table 5.6 shows the maximum and mean-square errors for each coordinate.

	Without Disturbance Rejection		With Disturbance Rejection	
	Max Error	MSE	Max Error	MSE
X	$4.364\times10^{+0}$ cm	$4.135\times10^{+0}$ cm^2	6.840×10^{-1} cm	1.136×10^{-1} cm^2
Y	$1.434\times10^{+1}$ cm	$7.602\times10^{+1}$ cm^2	$2.989\times10^{+0}$ cm	$1.731\times10^{+0}$ cm^2
Z	4.610×10^{-1} cm	6.697×10^{-2} cm^2	2.690×10^{-1} cm	1.274×10^{-2} cm^2
ϕ_e	$1.028\times10^{+1}$ deg	$4.512\times10^{+1}$ deg^2	$2.486\times10^{+0}$ deg	$1.174\times10^{+0}$ deg^2

Table 5.6: Maximum and Mean-Square Errors for 10° Amplitude, 4 Second Period Sinusoidal Disturbance in Platform Rotation

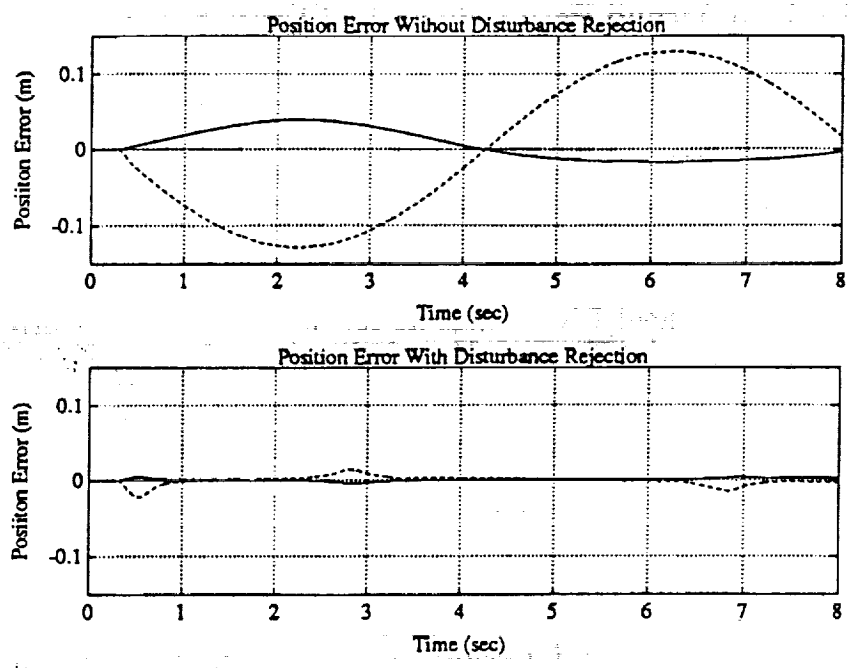

Figure 5.10: Position Errors (X – solid curves; Y – dashed curves; Z – dotted curves) for 10° Amplitude, 8 Second Period Sinusoidal Disturbance in Platform Rotation

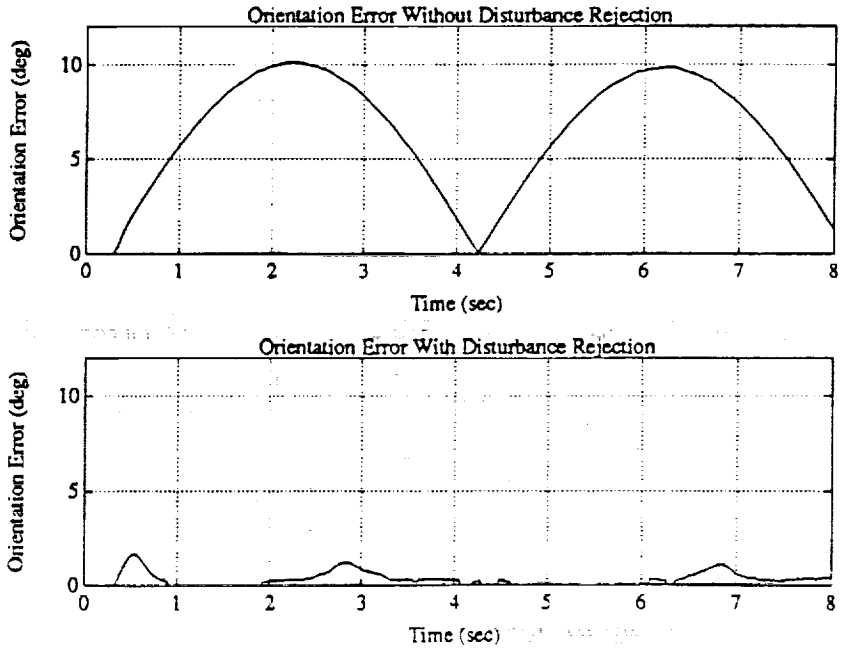

Figure 5.11: Orientation Errors for 10° Amplitude, 8 Second Period Sinusoidal Disturbance in Platform Rotation

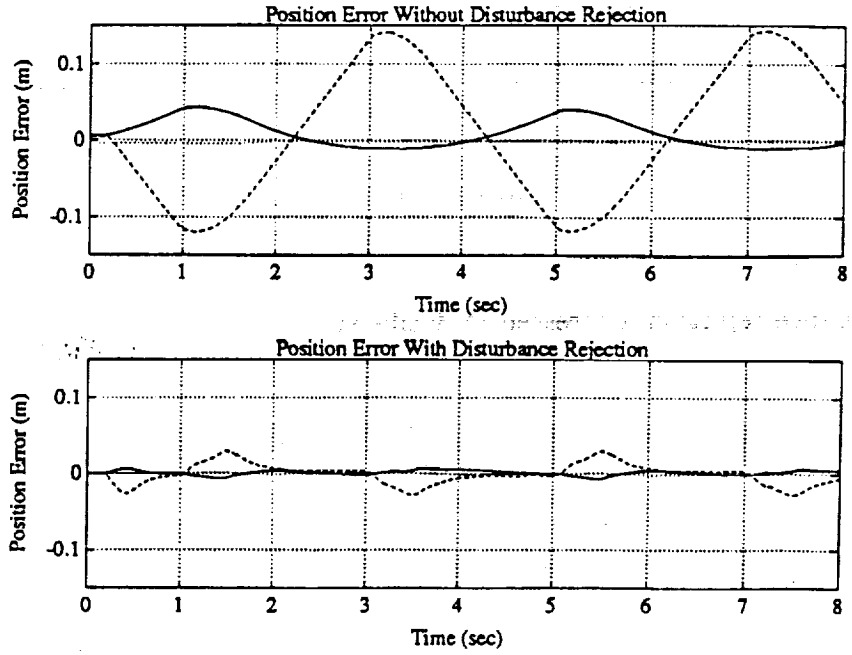

Figure 5.12: Position Errors (X – solid curves; Y – dashed curves; Z – dotted curves) for 10° Amplitude, 4 Second Period Sinusoidal Disturbance in Platform Rotation

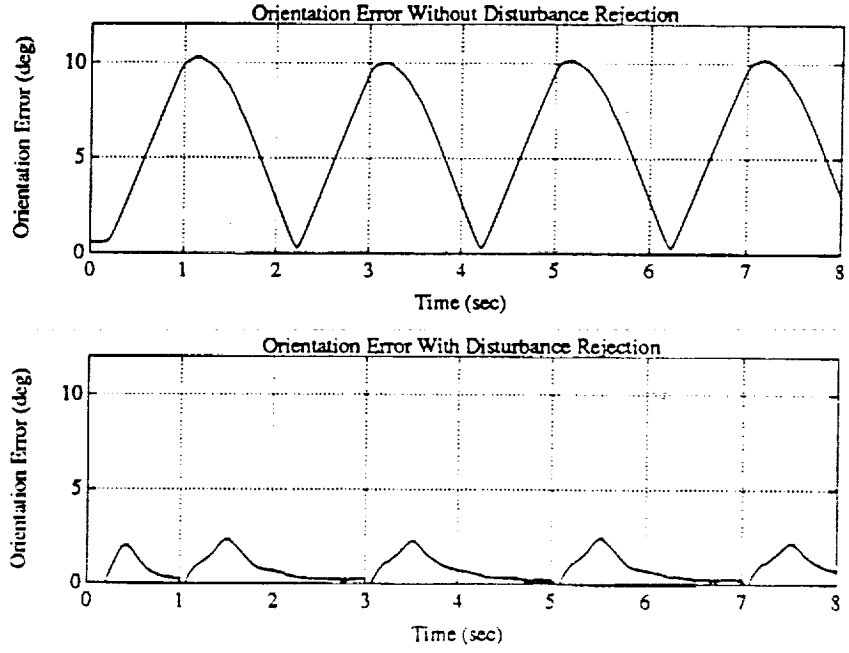

Figure 5.13: Orientation Errors for 10° Amplitude, 4 Second Period Sinusoidal Disturbance in Platform Rotation

5.2.4 Comparison

Comparing Figures 5.8 – 5.13, it can be concluded that the quality of the disturbance rejection diminishes as the frequency of the disturbance signal increases. One measure of the quality of the disturbance rejection is the *mean-square error attenuation*, defined as the ratio of the mean-square errors with and without disturbance rejection, expressed in decibels:

$$A_{\text{MSE}} \triangleq 20 \log_{10}(\varepsilon'_{\text{MS}}/\varepsilon_{\text{MS}}) \quad \text{dB} \tag{5.2}$$

where ε'_{MS} and ε_{MS} are the mean-square errors with and without disturbance rejection, respectively. Table 5.7 lists the A_{MSE} values for the 16, 8, and 4 second period sinusoidal disturbances. Note that for the 8 second case, the A_{MSE} value in the Z direction is positive, indicating that the error was amplified instead of attenuated. However, the actual mean-square error in this case is only 2.5×10^{-3} cm^2 (see Table 5.5).

	$T = 16$ s	$T = 8$ s	$T = 4$ s
X	$-6.251\times10^{+1}$ dB	$-4.229\times10^{+1}$ dB	$-3.122\times10^{+1}$ dB
Y	$-5.469\times10^{+1}$ dB	$-4.778\times10^{+1}$ dB	$-3.285\times10^{+1}$ dB
Z	$-5.520\times10^{+1}$ dB	$+1.721\times10^{+1}$ dB	$-1.441\times10^{+1}$ dB
ϕ_e	$-5.742\times10^{+1}$ dB	$-4.700\times10^{+1}$ dB	$-3.169\times10^{+1}$ dB

Table 5.7: Attenuation of Mean-Square Errors for 16, 8, and 4 Second Period, 10° Amplitude Sinusoidal Disturbances in Platform Rotation

5.3 Random Disturbances in Platform Rotation

Two types of stochastic disturbance signals are considered in this section: random noise with a uniform distribution, and random noise with a normal distribution. The control gain was set to identity, as in the previous sections.

5.3.1 Random Disturbance With Uniform Distribution

Figures 5.14 and 5.15 show the position and orientation errors for a random noise disturbance in the platform rotation, uniformly distributed in the interval $(-0.5°, +0.5°)$. The notation Unif$(-0.5°, +0.5°)$ will be used to represent this disturbance signal. Table 5.8 lists the maximum and mean-square for each coordinate.

	Without Disturbance Rejection		With Disturbance Rejection	
	Max Error	MSE	Max Error	MSE
X	$1.184 \times 10^{+0}$ cm	2.384×10^{-1} cm^2	8.440×10^{-1} cm	1.062×10^{-1} cm^2
Y	$5.939 \times 10^{+0}$ cm	$6.511 \times 10^{+0}$ cm^2	$3.799 \times 10^{+0}$ cm	$2.362 \times 10^{+0}$ cm^2
Z	3.010×10^{-1} cm	1.856×10^{-2} cm^2	4.900×10^{-2} cm	4.681×10^{-4} cm^2
ϕ_e	$4.037 \times 10^{+0}$ deg	$2.966 \times 10^{+0}$ deg^2	$2.911 \times 10^{+0}$ deg	$1.397 \times 10^{+0}$ deg^2

Table 5.8: Maximum and Mean-Square Errors for Unif$(-0.5°, +0.5°)$ Random Disturbance in Platform Rotation

5.3.2 Random Disturbance With Normal Distribution

Figures 5.16 – 5.17 display the position and orientation errors for a zero mean, 0.25° standard deviation Gaussian noise disturbance in the platform rotation (denoted by $\mathcal{N}(0, 0.25°)$). The maximum and mean-square errors are given in Table 5.9.

	Without Disturbance Rejection		With Disturbance Rejection	
	Max Error	MSE	Max Error	MSE
X	$1.664 \times 10^{+0}$ cm	6.275×10^{-1} cm^2	8.760×10^{-1} cm	1.373×10^{-1} cm^2
Y	$5.501 \times 10^{+0}$ cm	$6.625 \times 10^{+0}$ cm^2	$3.789 \times 10^{+0}$ cm	$2.534 \times 10^{+0}$ cm^2
Z	4.900×10^{-2} cm	1.304×10^{-3} cm^2	1.590×10^{-1} cm	5.640×10^{-3} cm^2
ϕ_e	$4.789 \times 10^{+0}$ deg	$5.332 \times 10^{+0}$ deg^2	$2.899 \times 10^{+0}$ deg	$1.515 \times 10^{+0}$ deg^2

Table 5.9: Maximum and Mean-Square Errors for $\mathcal{N}(0, 0.25°)$ Random Disturbance in Platform Rotation

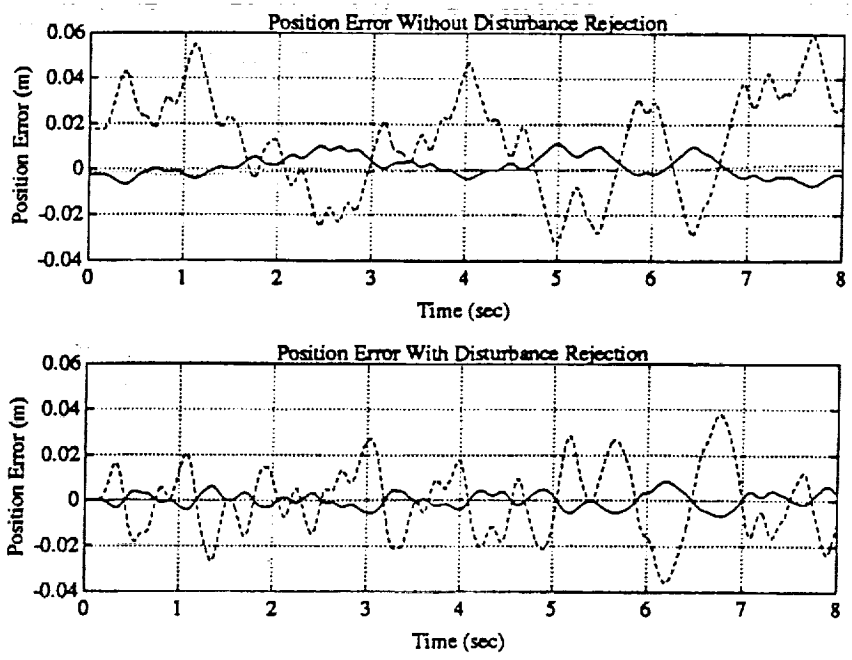

Figure 5.14: Position Errors (X – solid curves; Y – dashed curves; Z – dotted curves) for Unif$(-0.5°, +0.5°)$ Random Disturbance in Platform Rotation

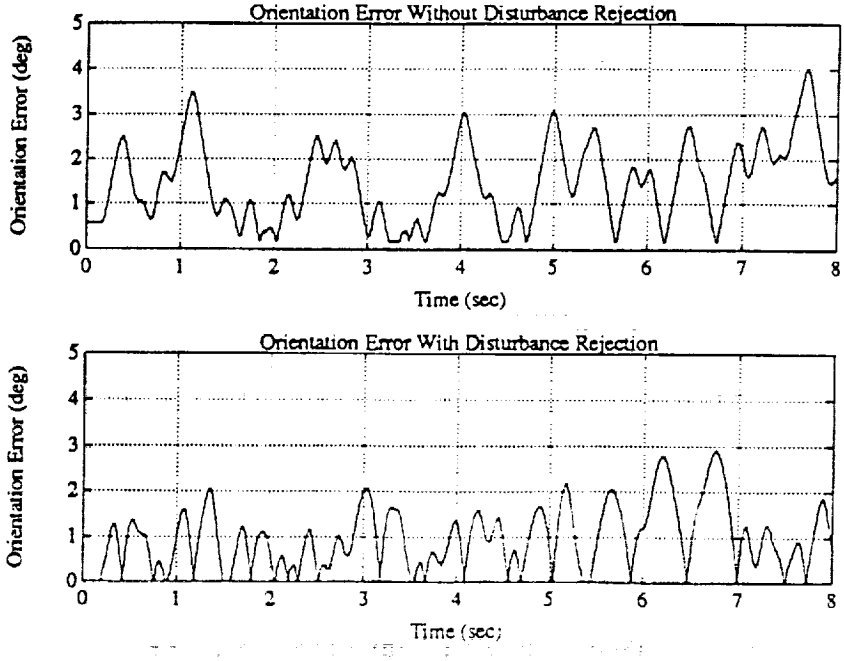

Figure 5.15: Orientation Errors for Unif$(-0.5°, +0.5°)$ Random Disturbance in Platform Rotation

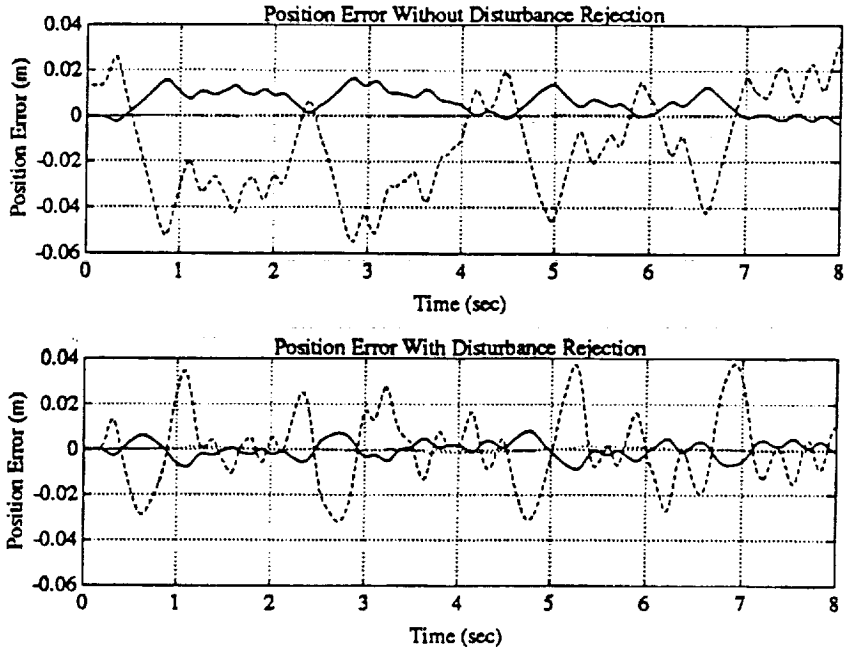

Figure 5.16: Position Errors (X – solid curves; Y – dashed curves; Z – dotted curves) for $\mathcal{N}(0, 0.25°)$ Random Disturbance in Platform Rotation

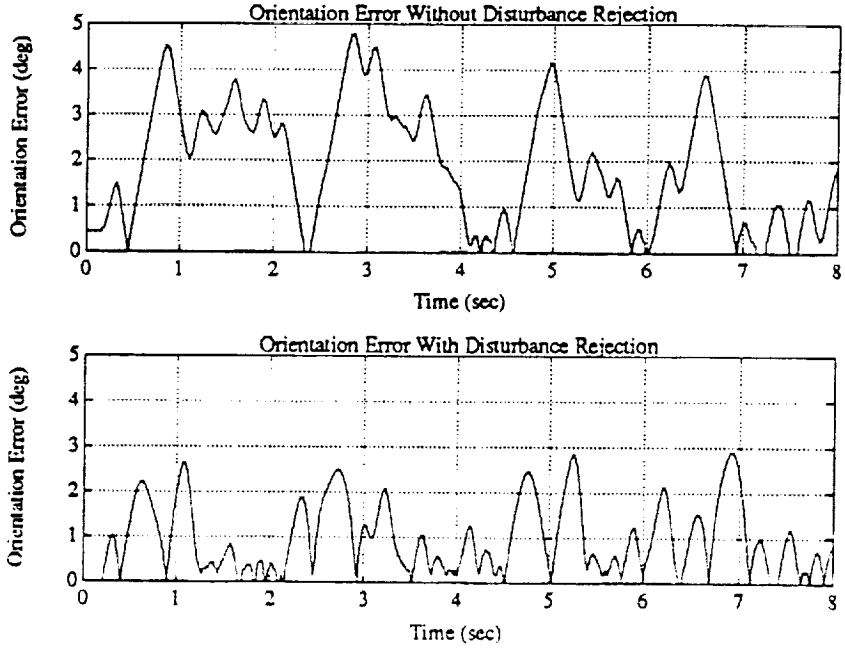

Figure 5.17: Orientation Errors for $\mathcal{N}(0, 0.25°)$ Random Disturbance in Platform Rotation

5.3.3 Comparison

Table 5.10 displays the mean-square error attenuations for uniform and Gaussian random noise in the platform rotation. The A_{MSE} values indicate that the performance is similar for both cases. In comparison with the results for sinusoidal disturbances, however, the quality of disturbance rejection is significantly less, since the random disturbance signals are of much higher bandwidth.

	$\text{Unif}(-0.5°, +0.5°)$	$\mathcal{N}(0, 0.25°)$
X	$-7.022\times10^{+0}$ dB	$-1.320\times10^{+1}$ dB
Y	$-8.807\times10^{+0}$ dB	$-8.347\times10^{+0}$ dB
Z	$-3.196\times10^{+1}$ dB	$+1.272\times10^{+1}$ dB
ϕ_e	$-6.542\times10^{+0}$ dB	$-1.093\times10^{+1}$ dB

Table 5.10: Attenuation of Mean-Square Errors for $\text{Unif}(-0.5°, +0.5°)$ and $\mathcal{N}(0, 0.25°)$ Random Disturbances in Platform Rotation

5.4 Behavior Near Singularities

In the experiments discussed so far, the manipulator was able to maintain the desired end-effector position and orientation without being forced into a singular configuration. This section examines the behavior of the $J^{\ddagger}$ controller when the arm is at or near each of the three PUMA singularities.

5.4.1 Arm Fully Stretched Singularity

Figure 5.18 shows the vector of open-loop control signals near the Arm Fully Stretched singularity. The minimum singular value parameter, σ_{min}, was set to 0.1. At this value of σ_{min}, the control in the direction of the workspace boundary becomes very weak approximately 30° from the singular point. This prevents the end-effector from getting too close to the workspace boundary. Consequently, the

manipulator will not switch between the `flex` and `noflex` configurations while the $J^{\ddagger}$ controller is running.

If the parameter σ_{min} is sufficiently small, however, the width of the singular region will be reduced to the point where the control signal for joint 6 ($\Delta\theta_d(3)$) could drive the arm through the singularity. This may lead to an undesirable "chattering" behavior, in which the arm rapidly oscillates between the `flex` and `noflex` configurations.

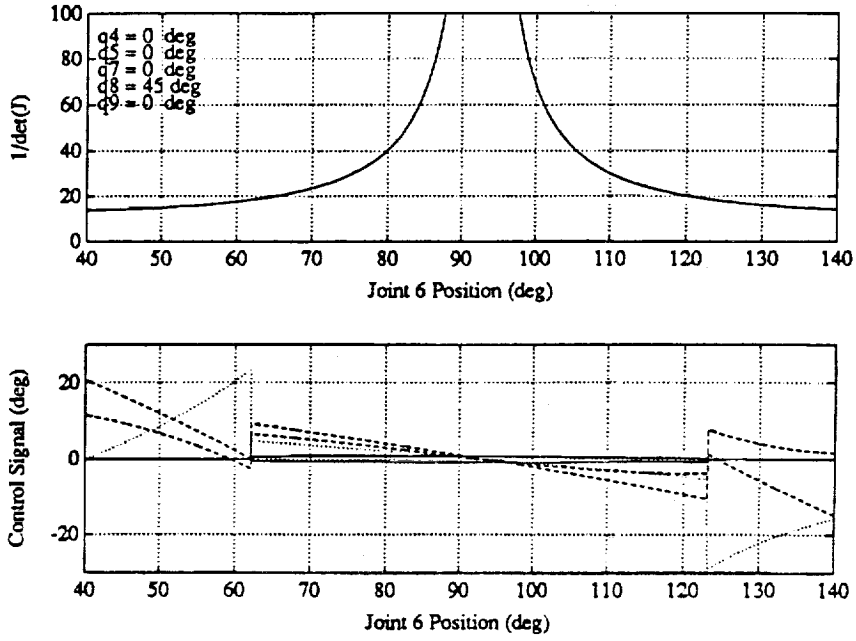

Figure 5.18: Behavior of $1/\det(J)$ and Open-Loop Control Signals ($\Delta\theta_d(1)$, $\Delta\theta_d(4)$ – solid curves; $\Delta\theta_d(2)$, $\Delta\theta_d(5)$ – dashed curves; $\Delta\theta_d(3)$, $\Delta\theta_d(6)$ – dotted curves) Near Arm Fully Stretched Singularity

5.4.2 Hand Over Head Singularity

Figure 5.19 shows the open-loop control signals in the vicinity of the Hand Over Head singularity, with $\sigma_{min} = 0.1$. At about 10° from the singular point, the control in the "forbidden" directions (c.f. Section 2.1.4) becomes very weak. Unlike

the Arm Fully Stretched singularity, this does not prevent the manipulator from changing configurations; however, it does mean that the end-effector will be unable to track certain linear components of the desired trajectory while the arm is in the singular region.

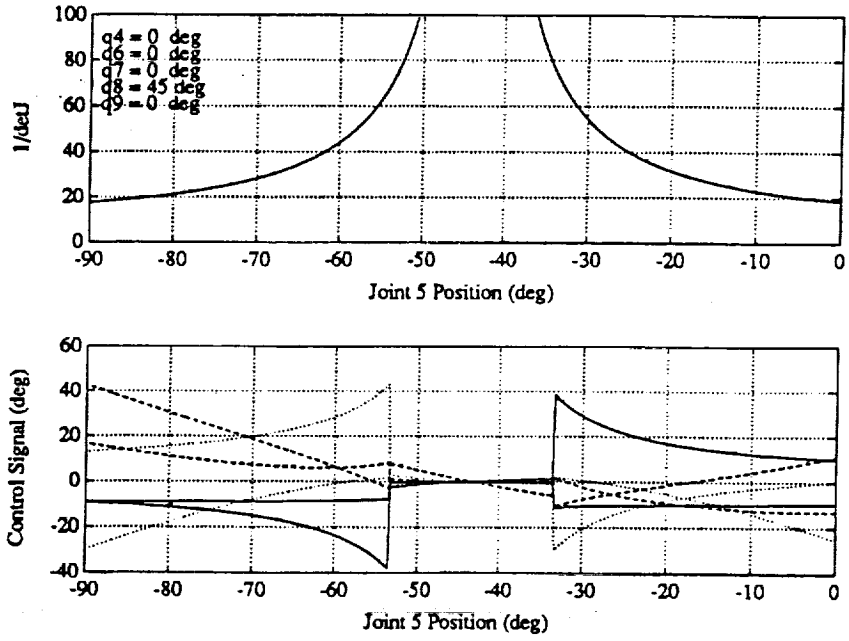

Figure 5.19: Behavior of $1/\det(J)$ and Open-Loop Control Signals ($\Delta\theta_d(1)$, $\Delta\theta_d(4)$ – solid curves; $\Delta\theta_d(2)$, $\Delta\theta_d(5)$ – dashed curves; $\Delta\theta_d(3)$, $\Delta\theta_d(6)$ – dotted curves) Near Hand Over Head Singularity

5.4.3 Wrist Singularity

Figure 5.20 shows the open-loop control signals near the Wrist singularity, with $\sigma_{min} = 0.1$. The control signals for joints 7 and 9 ($\Delta\theta_d(4)$ and $\Delta\theta_d(6)$) go to zero about 8° from the singular point. As in the Hand Over Head singularity, this does affect the ability to change configurations. However, the end-effector will be unable to track certain angular components of the desired trajectory when the position of joint 8 is within 8° of zero.

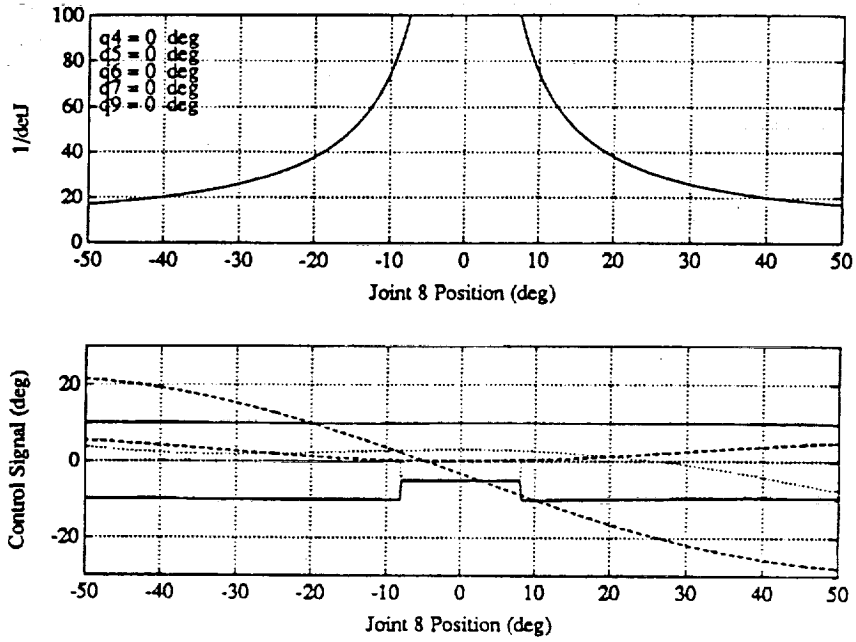

Figure 5.20: Behavior of $1/\det(J)$ and Open-Loop Control Signals ($\Delta\theta_d(1)$, $\Delta\theta_d(4)$ – solid curves; $\Delta\theta_d(2)$, $\Delta\theta_d(5)$ – dashed curves; $\Delta\theta_d(3)$, $\Delta\theta_d(6)$ – dotted curves) Near Wrist Singularity

5.5 Summary

Several important conclusions can be drawn from the experimental results presented in this chapter. These conclusions are summarized below:

1. The relative stability of the closed-loop system is a function of the amplitude of the disturbance signal.

By comparing the maximum overshoot and settling time, it was argued that the system was less stable for the 30° step input than for the 10° step input. This observation agrees with the stability analysis presented in Section 3.3, since α is directly related to the maximum disturbance amplitude (cf. Equation 3.42). With $K_c = I$, the system is stable for platform rotational disturbances less than 60°.

2. The relative performance of the controller is a function of the frequency of the disturbance signal.

For the 16 second sinusoidal disturbance, the mean-square error attenuation was very good (about -55 dB), but for the 4 second sinusoidal disturbance, the attenuation was markedly less (about -30 dB). In other words, the $J^\ddagger$ controller is like a high-pass filter; the lowest frequency components of the disturbance signal are attenuated the most.

3. The control in certain directions becomes very weak near singularities.

This implies that there may be an unavoidable tracking error in the "forbidden" directions when the arm is at or near a singularity. This also prevents the arm from switching between the `flex` and `noflex` configurations near the workspace boundary.

CHAPTER 6
CONCLUSION

6.1 Report Summary and Conclusions

This report described the design, analysis, implementation, and performance of a kinematic controller for inertial-space disturbance rejection. First, the problem of mapping end-effector displacements to joint displacements was considered. The approximate pseudoinverse Jacobian was presented as a computationally efficient and well-defined solution to this problem. Next, a kinematic control algorithm, the $J^{\ddagger}$ control law, was proposed. A discrete-time model of the closed-loop system was derived, and the stability of the system was shown to be related to the upper bound on the disturbance and the selection of the control gain. The real-time implementation of the controller on CIRSSE's robotic testbed was then discussed, as well as the hardware and software components of the testbed used in this thesis. Finally, some experimental results were presented, comparing the performance of the controller for step, sinusoidal, and random disturbances in the platform rotational axis, and at the singularities of the PUMA.

In conclusion, the $J^{\ddagger}$ controller has been demonstrated to be very effective for rejecting the low-frequency components of an arbitrary disturbance signal. The controller was shown to be robust with respect to relatively large magnitude disturbances and in the neighborhood of kinematic singularities. The modest computational requirements of the algorithm, coupled with the fact that precise knowledge of the disturbance signal is not required, suggest that this controller is a practical solution to the inertial-space disturbance rejection control problem.

6.2 Future Research

Several recommendations for future directions in this area of research are discussed below.

1. Implement direct inertial end-point sensing.

Recall from Section 4.3.2 that the inertial end-point sensor driver calculates the inertial end-effector position and attitude using the forward kinematics. In practice, however, additional sensors are needed to measure the end-effector location since the platform joints are not exactly known. This driver should be replaced when direct end-point feedback is available.

2. Incorporate joint limit constraints into the kinematic control algorithm.

Although the joint limits of the manipulator are usually taken into account by the path planner, a large enough magnitude disturbance could force one or more joints to its limit. It would be desirable to avoid this situation by augmenting the kinematic control law with joint limit constraints.

3. Design a better dynamic control algorithm.

The performance of the system could be improved by using a better joint-level controller. The limiting factor in the PID control algorithm used for this research appears to be velocity noise, which arises from backward differencing the joint positions. This noise could be reduced by Kalman filtering or by directly measuring the joint velocities with tachometers.

4. Investigate alternative kinematic control algorithms.

Several alternatives to the approximate pseudoinverse Jacobian exist for transforming between joint and task space. For instance, the inverse kinematics could be used

to map the inertial-space position and attitude to joint positions, or the transpose of the Jacobian could be used to map a "force-like" error (based on the inertial-space error) to joint torques.

5. Extend the results to free-floating space manipulator systems.

There are three issues which arise when dealing with free-floating systems that were not specifically addressed in this report. First, the dynamics of free-floating systems are more complicated than those of terrestrial systems; for example, there may be significant dynamic coupling between the manipulator and spacecraft, causing the spacecraft to react to manipulator motions. Second, the Jacobian of a free-floating system depends not only on the joint angles and kinematic parameters, but on the system mass and inertia properties [3]. Finally, a space manipulator may encounter *dynamic singularities*, depending on the history of the manipulator motion [3, 25]. The results presented in this report should be extended to encompass free-floating systems when these issues are better understood.

LITERATURE CITED

[1] M. R. Elgersma, G. Stein, M. R. Jackson, and J. Yeichner, "Robust Controllers for Space Station Momentum Management," *IEEE Control Systems Magazine*, vol. 12, pp. 14 – 22, October 1992.

[2] S. Dubowsky, E. E. Vance, and M. A. Torres, "The Control of Space Manipulators Subject to Spacecraft Attitude Control Saturation Limits," in *Proc. NASA Conf. Space Telerobotics*, (Pasadena, CA), January – February 1989.

[3] E. Papadopoulos and S. Dubowsky, "On the Nature of Control Algorithms for Space Manipulators," in *Proc. 1990 IEEE Robotics and Automation Conference*, (Cincinnati, OH), 1990.

[4] J. Joshi, "Modeling and Control of a Mobile Robot," Master's thesis, Rensselaer Polytechnic Institute, Troy, NY, August 1985.

[5] J. Joshi and A. A. Desrochers, "Modeling and Control of a Mobile Robot Subject to Disturbances," in *Proc. 1986 IEEE Robotics and Automation Conference*, (San Francisco, CA), 1986.

[6] M. A. Torres and S. Dubowsky, "Minimizing Spacecraft Attitude Disturbances in Space Manipulator Systems," *Journal of Guidance, Control, and Dynamics*, vol. 15, pp. 1010 – 1016, July – August 1992.

[7] J. Cheng, G. Ianculescu, C. S. Kenney, A. J. Laub, J. Ly, and P. M. Papadopoulos, "Control-Structure Interaction for Space Station Solar Dynamic Power Module," *IEEE Control Systems Magazine*, vol. 12, pp. 4 – 13, October 1992.

[8] J. J. Craig, *Introduction to Robotics, Mechanics & Control.* Reading, MA: Addison-Wesley, 1986.

[9] A. Fijany and A. Bejczy, "Efficient Jacobian Inversion for the Control of Simple Robot Manipulators," Engineering Memorandum, Jet Propulsion Laboratory, Pasadena, CA, April 1988.

[10] D. Sood, "Kinematics and Jacobian of the Platform and the Robot," Internal Communication, 1992.

[11] T. Kailath, *Linear Systems.* Englewood Cliffs, NJ: Prentice-Hall, 1980.

[12] J. F. Watson, III, "Testbed Kinematic Frames and Routines," CIRSSE Technical Memorandum 1 (v. 2), Rensselaer Polytechnic Institute, Troy, NY, August 1991.

[13] J. Baillieul, J. Hollerbach, and R. Brockett, "Programming and Control of Kinematically Redundant Manipulators," in *Proc. 24th IEEE Conf. Decision and Control*, (Las Vegas, NV), 1984.

[14] S. J. Leon, *Linear Algebra with Applications.* New York, NY: Macmillan, second ed., 1986.

[15] Y. Nakamura, *Advanced Robotics: Redundancy and Optimization.* New York, NY: Addison-Wesley, 1991.

[16] G. Strang, *Linear Algebra and its Applications.* San Diego, CA: Harcourt, Brace, Jovanovich, third ed., 1988.

[17] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes in C: The Art of Scientific Computing.* Cambridge, MA: Cambridge University Press, 1988.

[18] R. J. Schilling, *Fundamentals of Robotic Analysis and Control.* Englewood Cliffs, NJ: Prentice-Hall, 1990.

[19] R. P. Paul, *Robot Manipulators: Mathematics, Programming, and Control.* Cambridge, MA: The MIT Press, 1981.

[20] D. Swift, "Kinematic and Dynamic Parameters for the Testbed Grippers and Loads," CIRSSE Technical Memorandum 14 (v. 1), Rensselaer Polytechnic Institute, Troy, NY, January 1992.

[21] D. Lefebvre, "CTOS Tasks and Applications," CIRSSE Technical Memorandum 5 (v. 1), Rensselaer Polytechnic Institute, Troy, NY, August 1992.

[22] C. L. Phillips and H. T. Nagle, *Digital Control System Analysis and Design.* Englewood Cliffs, NJ: Prentice-Hall, 1984.

[23] J. R. Rowland, *Linear Control Systems.* New York, NY: John Wiley & Sons, 1986.

[24] K. Holt and L. S. Wilfinger, "Jacobian Library for the Testbed Arms," CIRSSE Technical Memorandum 17 (v. 1), Rensselaer Polytechnic Institute, Troy, NY, July 1992.

[25] E. Papadopoulos and S. Dubowsky, "On the Dynamic Singularities in the Control of Free-Floating Space Manipulators," in *Proc. of the ASME Winter Annual Meeting.* (San Fransisco), 1989.

[26] J. L. Cosentino, "Development of a Control System for a Pair of Robotic Platforms," Master's thesis, Rensselaer Polytechnic Institute, Troy, NY, August 1990.

[27] B. Armstrong, O. Khatib, and J. Burdick, "The Explicit Dynamic Model and Inertial Parameters of the PUMA 560 Arm," in *Proc. 1986 IEEE Robotics and Automation Conference*, (San Francisco, CA), pp. 510 - 518, March 1986.

[28] S. Murphy and D. Swift, "Dynamic Parameters and Inverse Dynamics for the PUMA 560," CIRSSE Technical Memorandum 13 (v. 1), Rensselaer Polytechnic Institute, Troy, NY, January 1992.

[29] D. P. Martin, J. Baillieul, and J. M. Hollerbach, "Resolution of Kinematic Redundancy Using Optimization Techniques," *IEEE Transaction on Automatic Control*, vol. 5, pp. 529 - 533, August 1989.

[30] L. M. Nugent, "Computational Reduction of PUMA 600 Jacobian Calculations, Inverse Jacobian Calculations, and the Lagrange-Euler Dynamics," Master's thesis, Rensselaer Polytechnic Institute, Troy, NY, February 1986.

[31] M. Leahy, L. Nugent, and G. Saridis, "Efficient PUMA Manipulator Jacobian Calculation and Inversion," *Journal on Robotic Systems*, vol. 4, no. 2, pp. 185 - 197, 1987.

[32] C.-J. Li, A. Henami, and T. S. Sankar, "An Efficient Computational Method of the Jacobian for Robot Manipulators," *Robotica*, vol. 9, pp. 231 - 234, 1991.

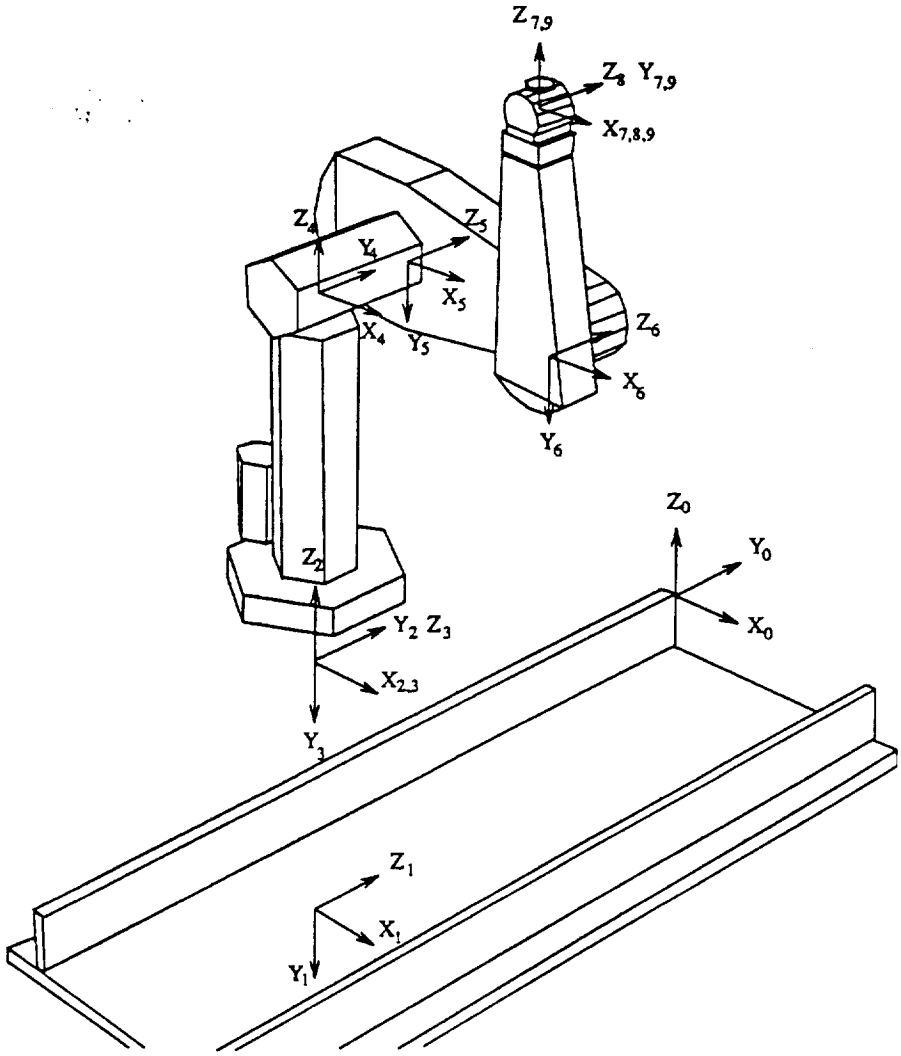

Figure 2.1: Coordinate Frame Assignments

1 & 2 blank

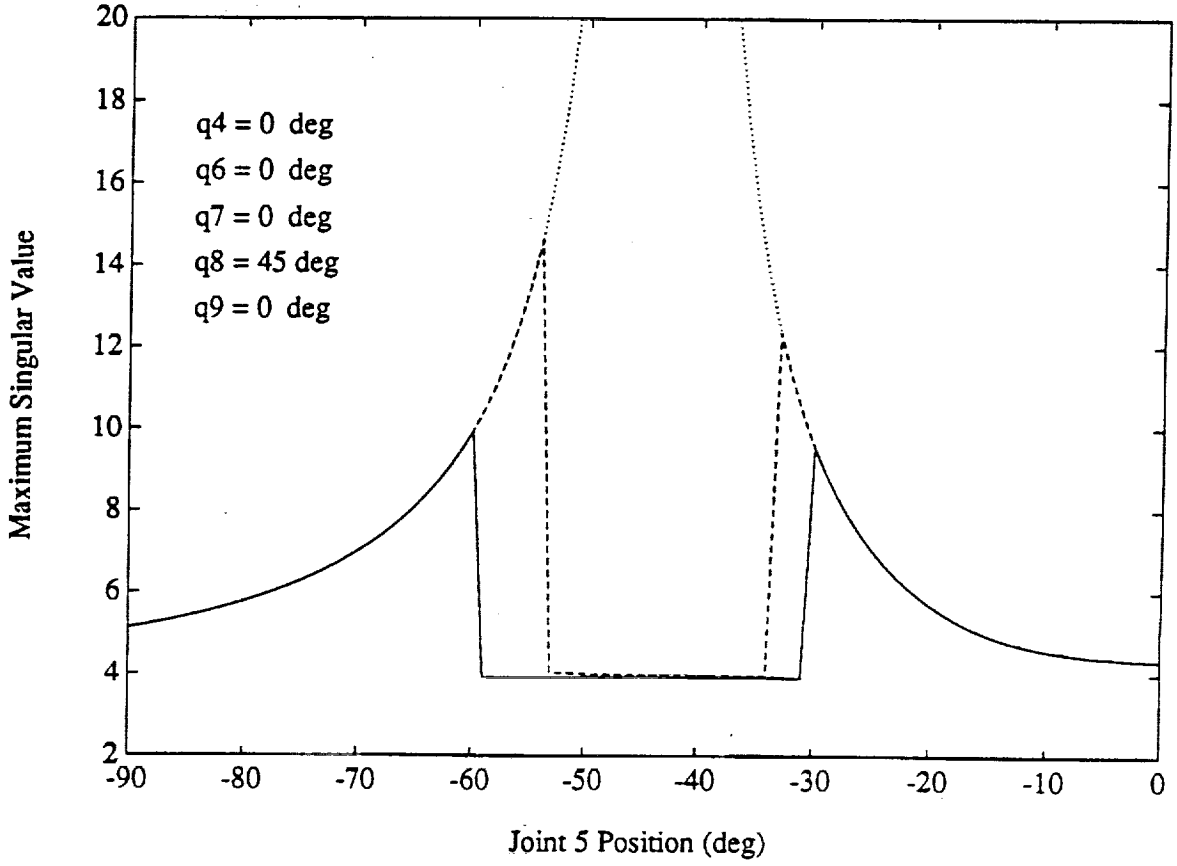

Figure 2.2: 2-Norms of $J^{\dagger}$ (solid curve), $J^{\ddagger}$ (dashed curve), and J^{-1} (dotted curve) Near Hand Over Head Singularity

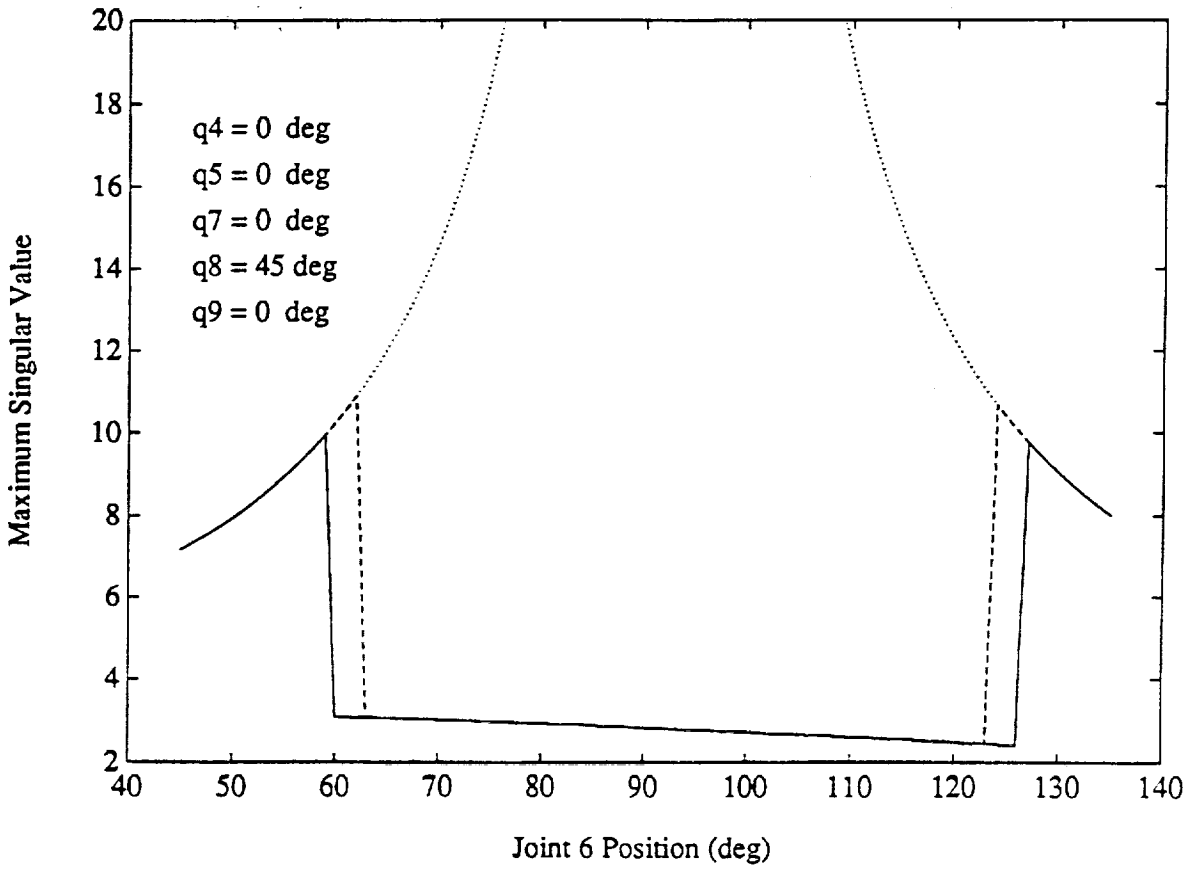

Figure 2.3: 2-Norms of $J^{\dagger}$ (solid curve), $J^{\ddagger}$ (dashed curve), and J^{-1} (dotted curve) Near Arm Fully Stretched Singularity

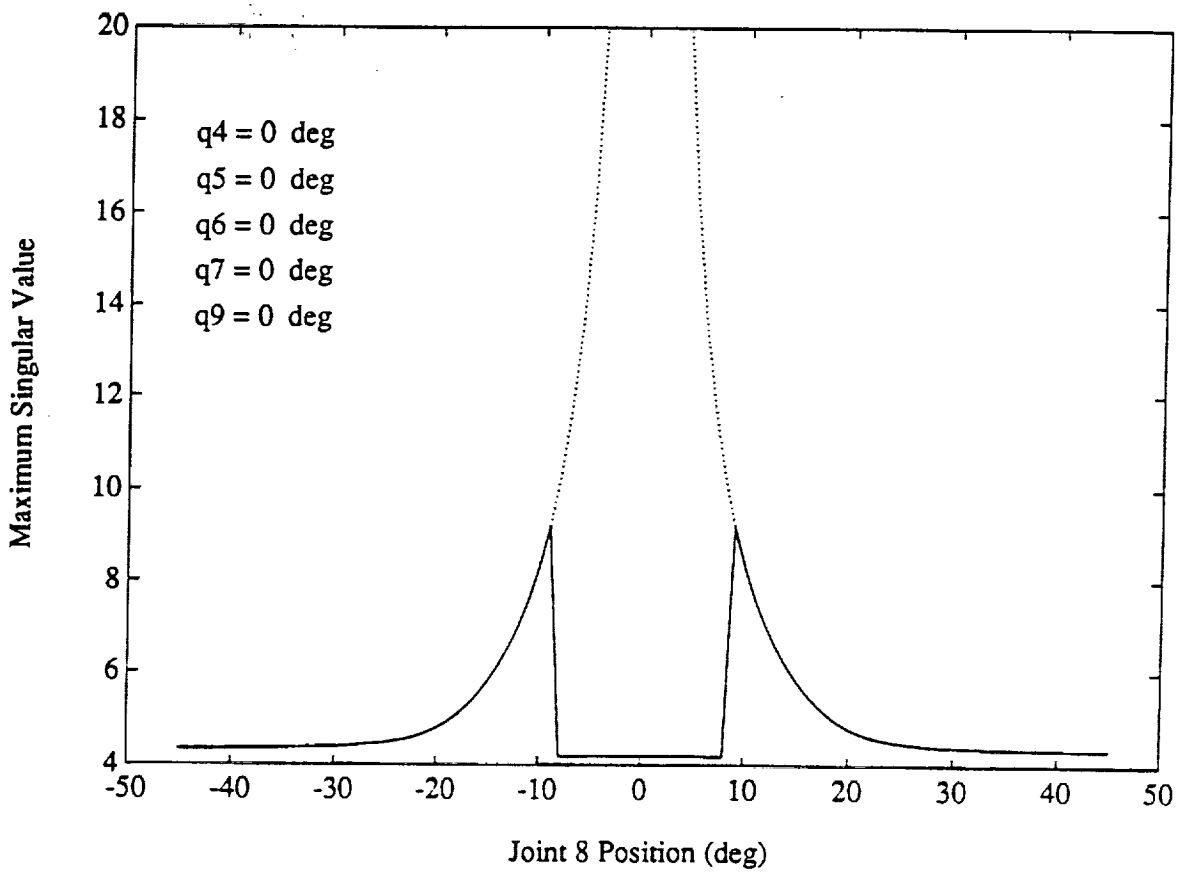

Figure 2.4: 2-Norms of $J^{\dagger}$ (solid curve), $J^{\ddagger}$ (dashed curve), and J^{-1} (dotted curve) Near Wrist Singularity

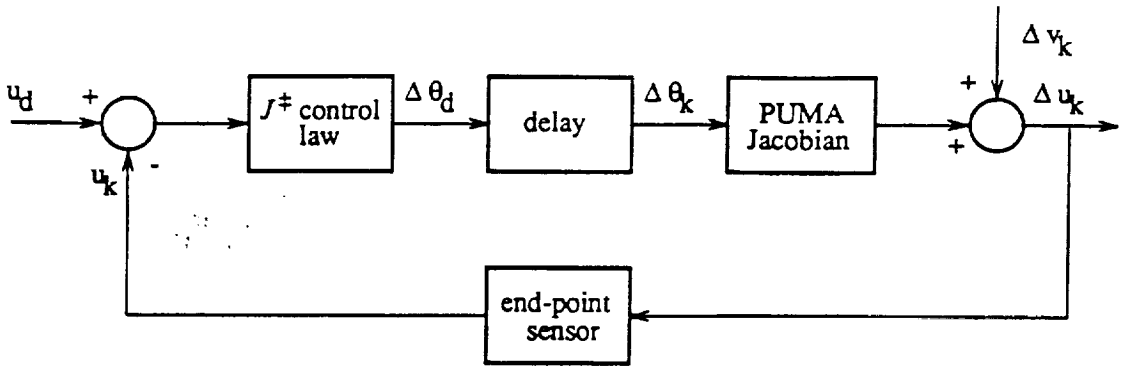

Figure 3.1: Block Diagram of Closed-Loop System

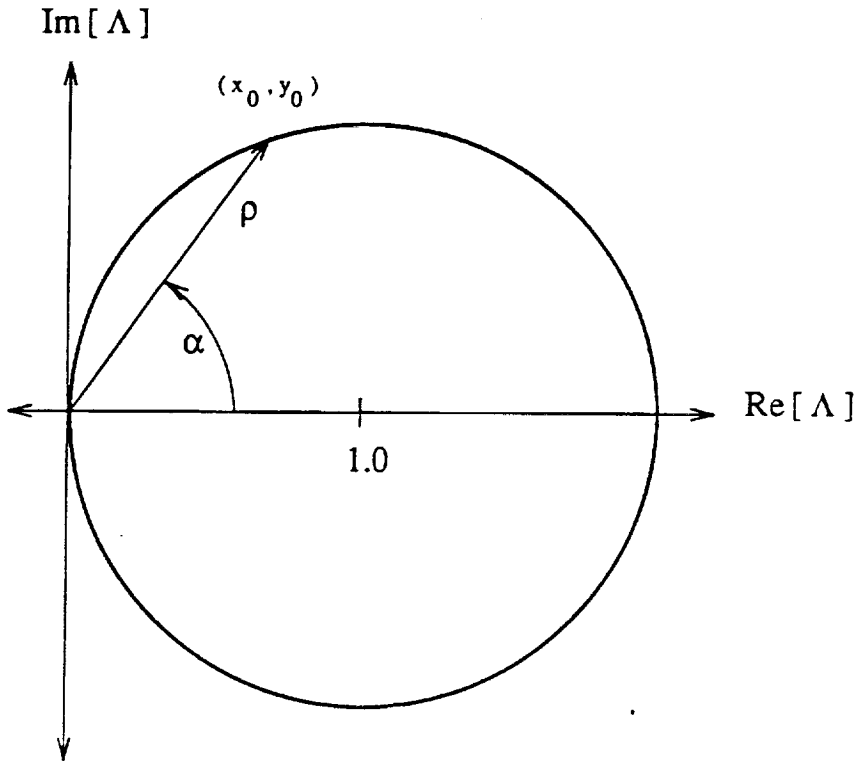

Figure 3.2: Region of Stability in the Λ-Plane

Page 7 blank

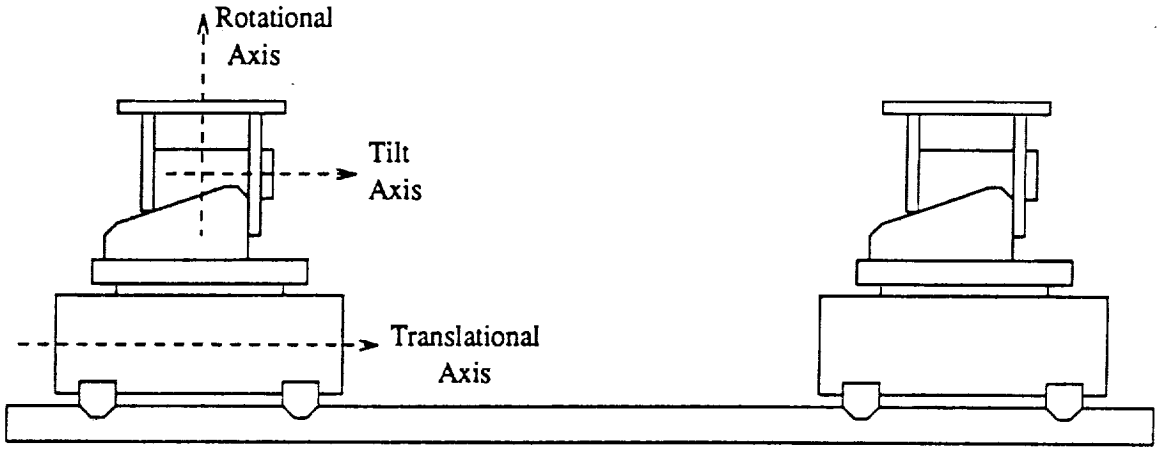

Figure 4.1: 3-DOF Platform Carts

Page 9 blank.

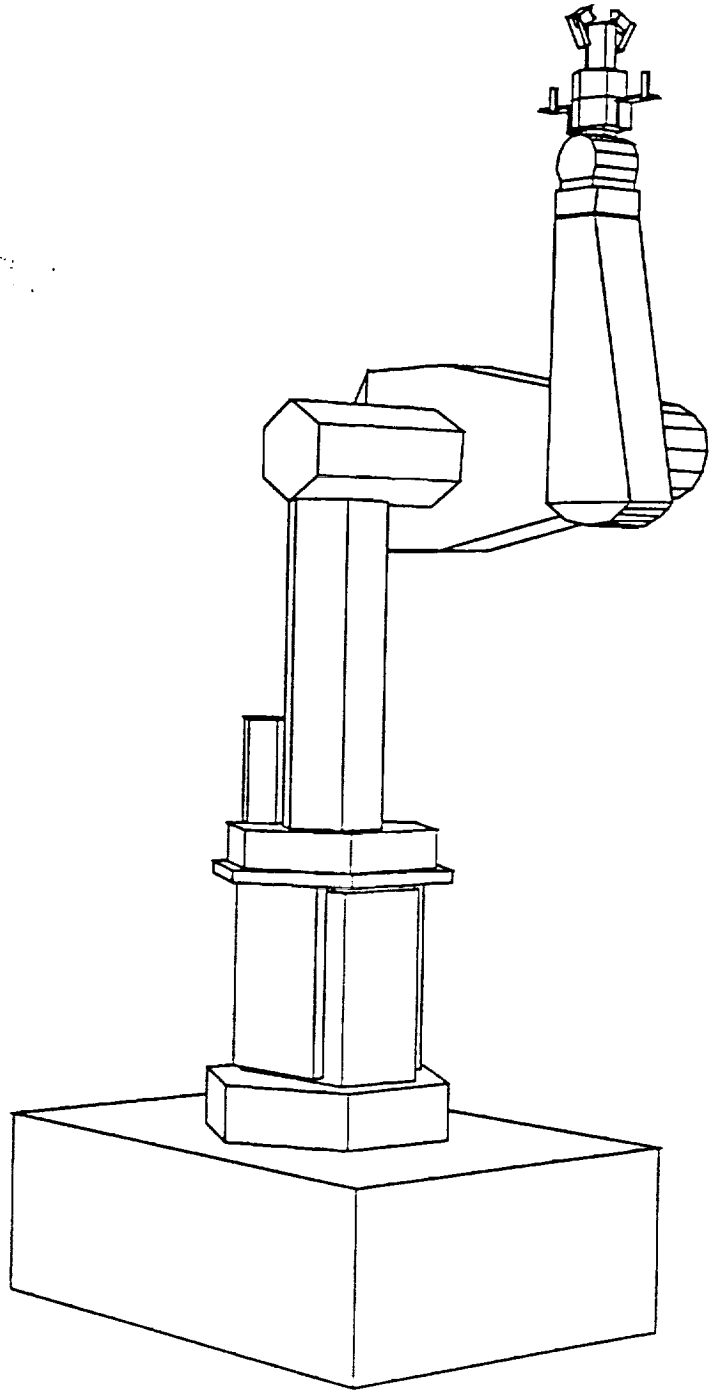

Figure 4.2: Left PUMA and Platform Cart

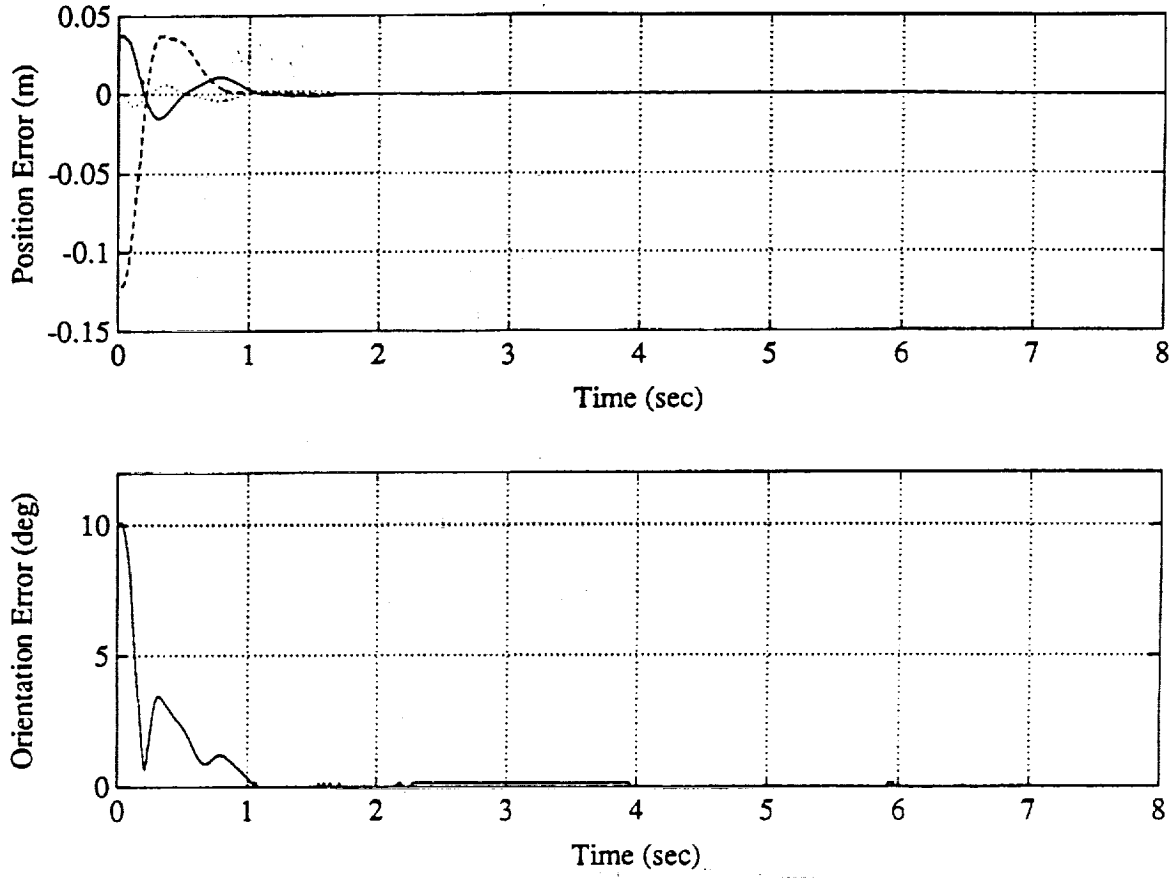

Figure 5.1: Position Error (X – solid curve; Y – dashed curve; Z – dotted curve) and Orientation Error for 10° Step Disturbance in Platform Rotation

Page 12 blank

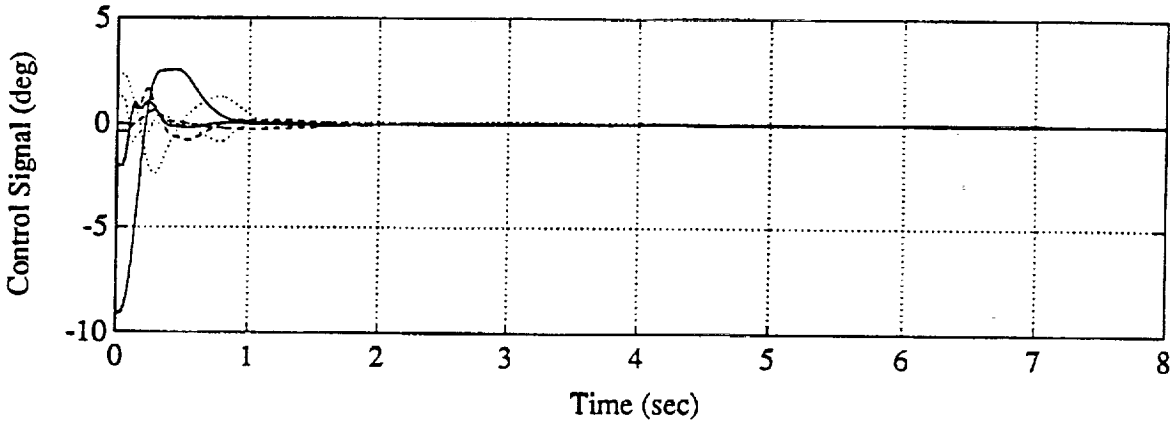

Figure 5.2: Control Signals ($\Delta\theta_d(1)$, $\Delta\theta_d(4)$ – solid curves; $\Delta\theta_d(2)$, $\Delta\theta_d(5)$ – dashed curves; $\Delta\theta_d(3)$, $\Delta\theta_d(6)$ – dotted curves) for 10° Step Disturbance in Platform Rotation

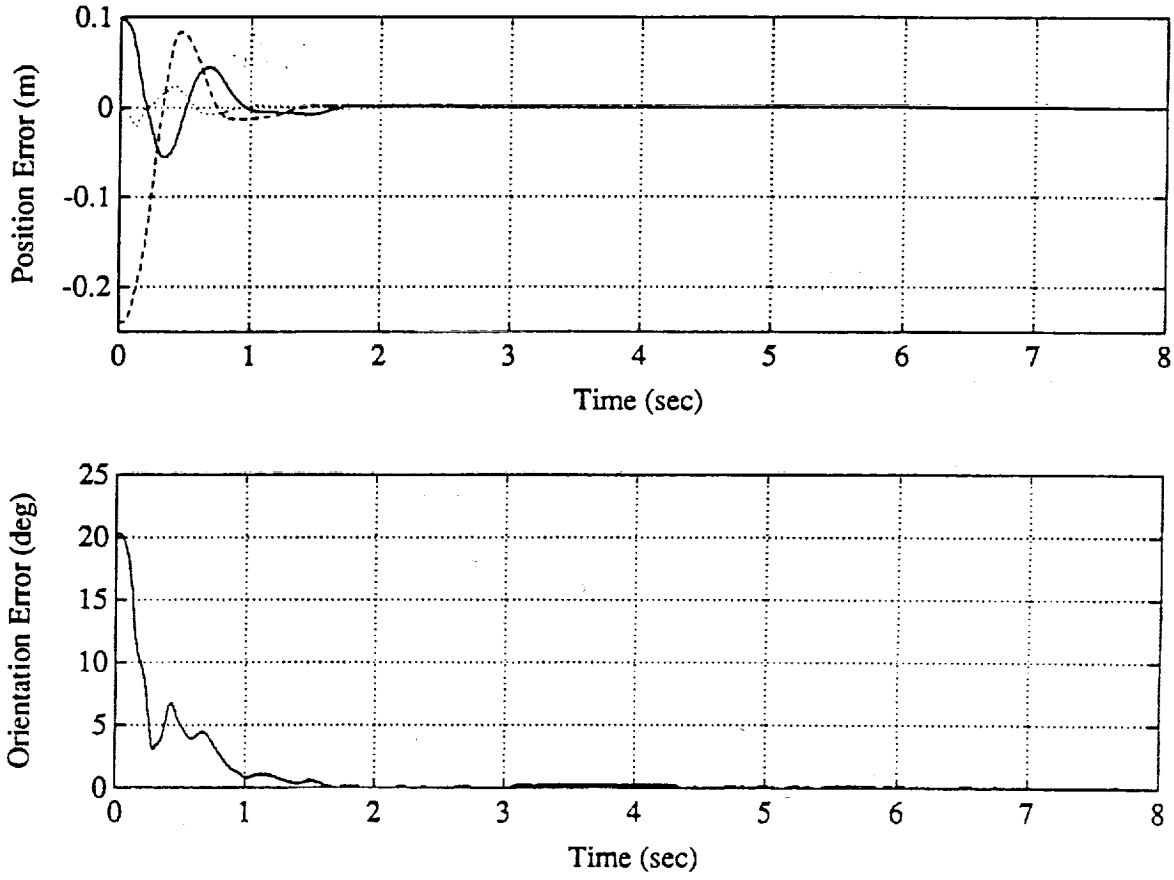

Figure 5.3: Position Error (X – solid curve; Y – dashed curve; Z – dotted curve) and Orientation Error for 20° Step Disturbance in Platform Rotation

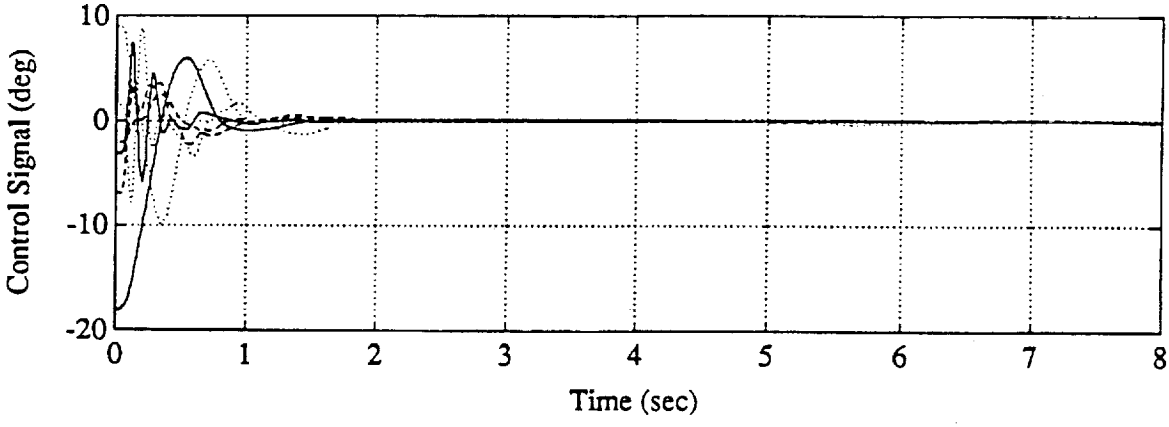

Figure 5.4: Control Signals ($\Delta\theta_d(1)$, $\Delta\theta_d(4)$ – solid curves; $\Delta\theta_d(2)$, $\Delta\theta_d(5)$ – dashed curves; $\Delta\theta_d(3)$, $\Delta\theta_d(6)$ – dotted curves) for 20° Step Disturbance in Platform Rotation

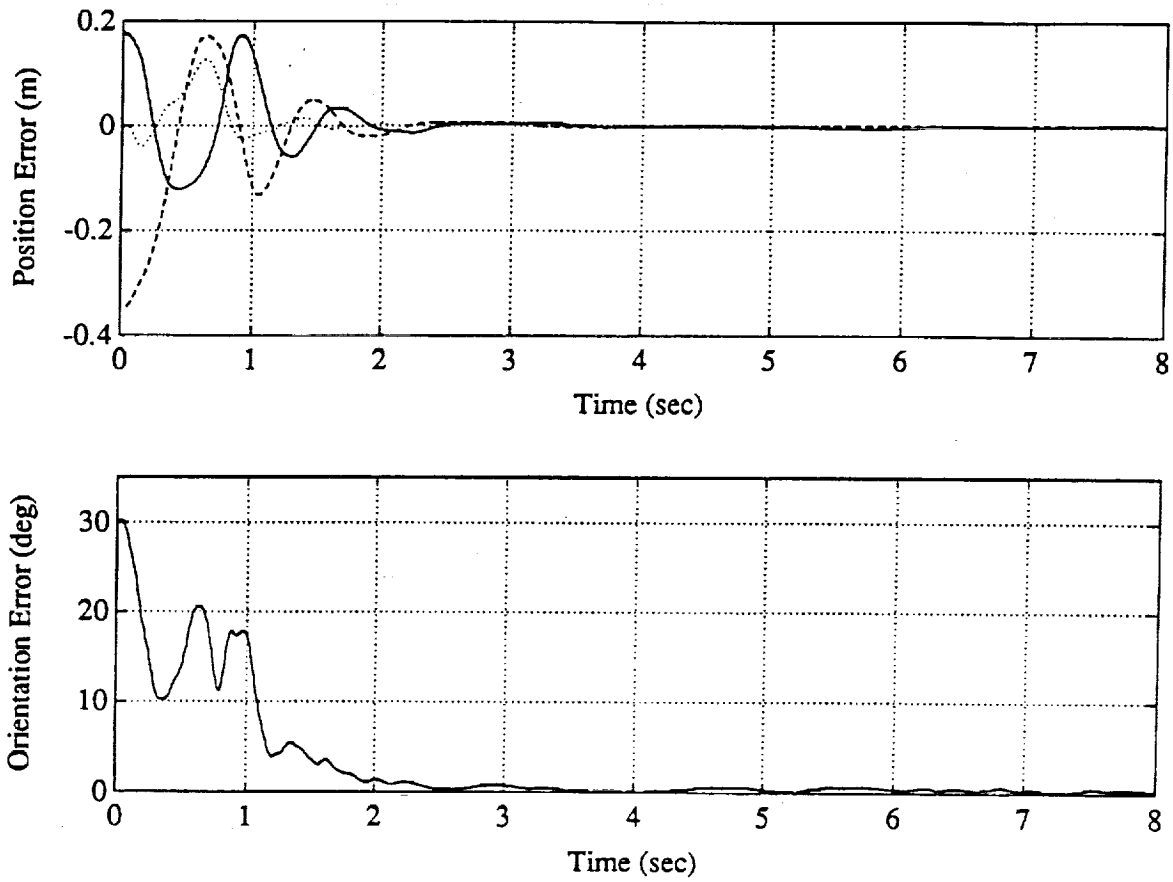

Figure 5.5: Position Error (X – solid curve; Y – dashed curve; Z – dotted curve) and Orientation Error for 30° Step Disturbance in Platform Rotation

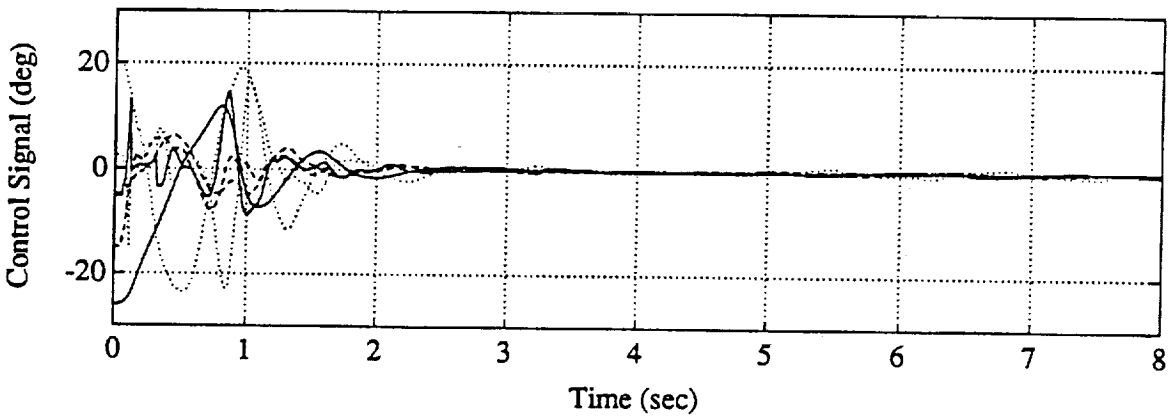

Figure 5.6: Control Signals ($\Delta\theta_d(1)$, $\Delta\theta_d(4)$ – solid curves; $\Delta\theta_d(2)$, $\Delta\theta_d(5)$ – dashed curves; $\Delta\theta_d(3)$, $\Delta\theta_d(6)$ – dotted curves) for 30° Step Disturbance in Platform Rotation

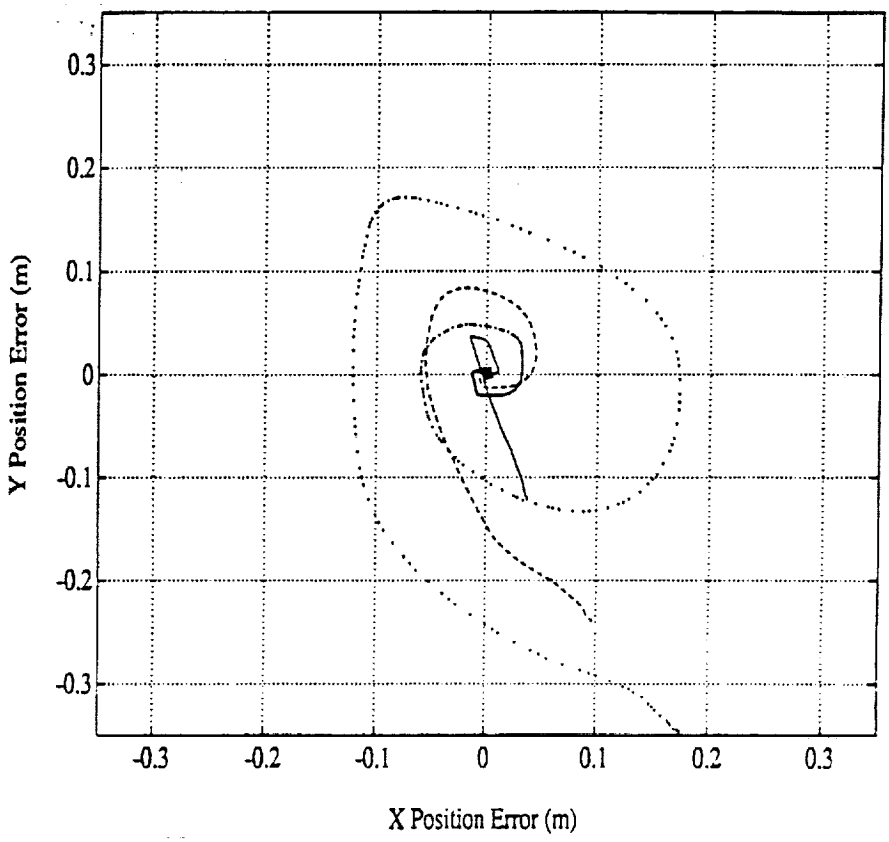

Figure 5.7: Position Errors for 10° (solid curve), 20° (dashed curve), and 30° (dotted curve) Step Disturbances in Platform Rotation

C-2

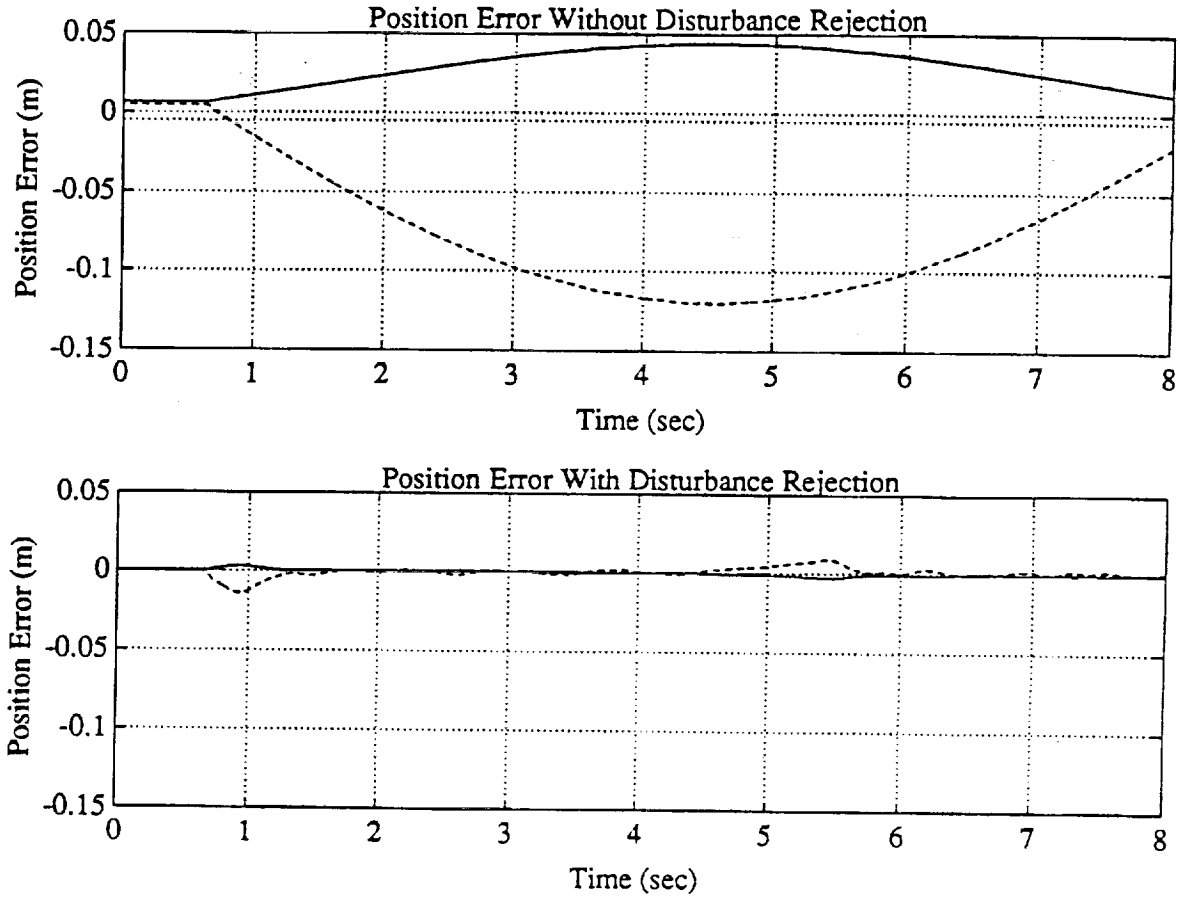

Figure 5.8: Position Errors (X – solid curves; Y – dashed curves; Z – dotted curves) for 10° Amplitude, 16 Second Period Sinusoidal Disturbance in Platform Rotation

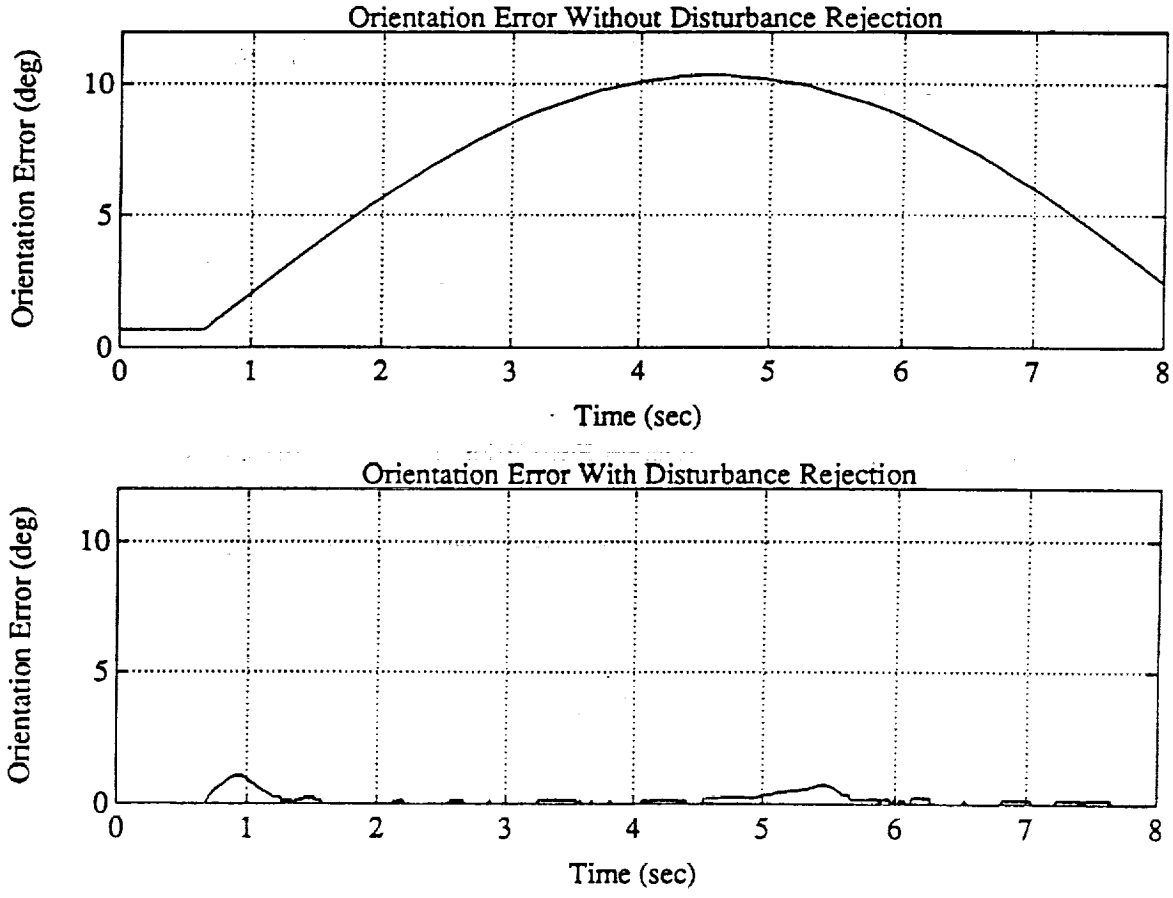

Figure 5.9: Orientation Errors for 10° Amplitude, 16 Second Period Sinusoidal Disturbance in Platform Rotation

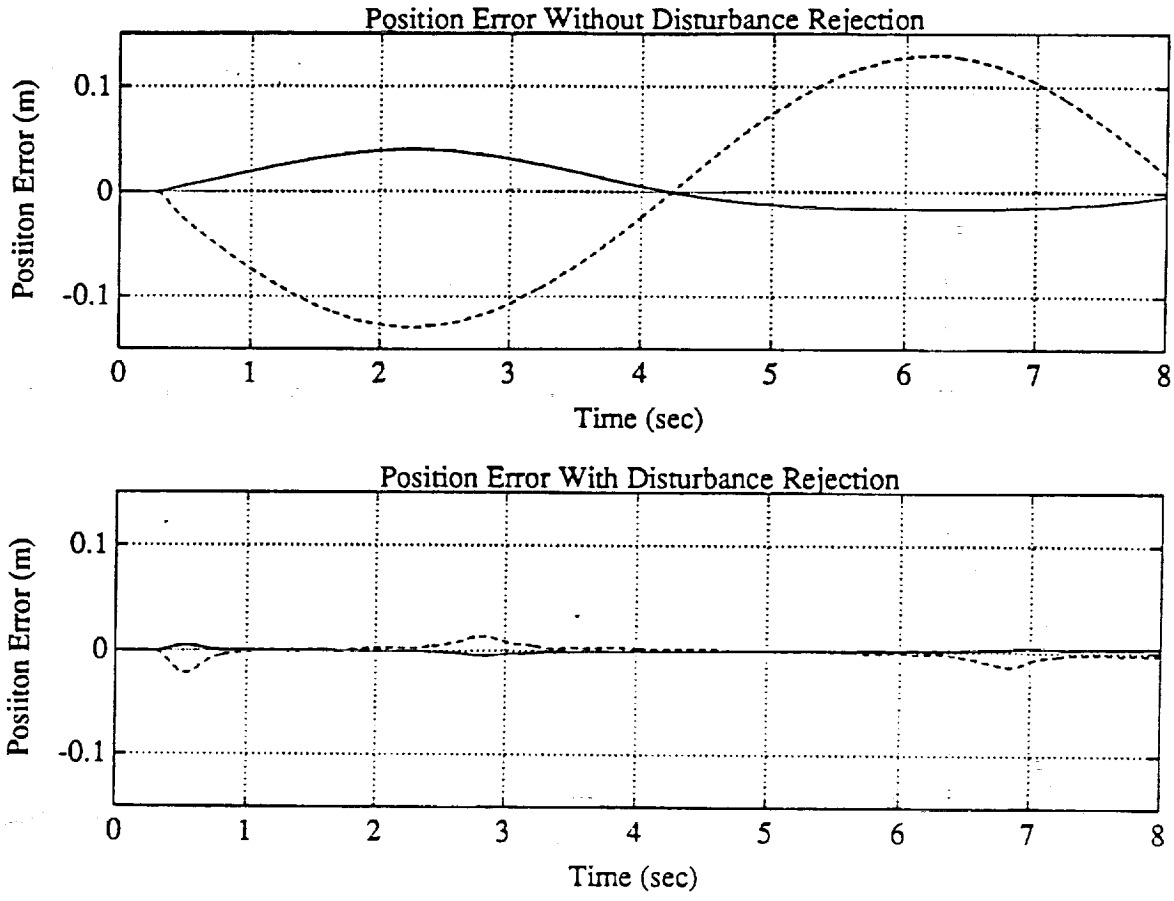

Figure 5.10: Position Errors (X – solid curves; Y – dashed curves; Z – dotted curves) for 10° Amplitude, 8 Second Period Sinusoidal Disturbance in Platform Rotation

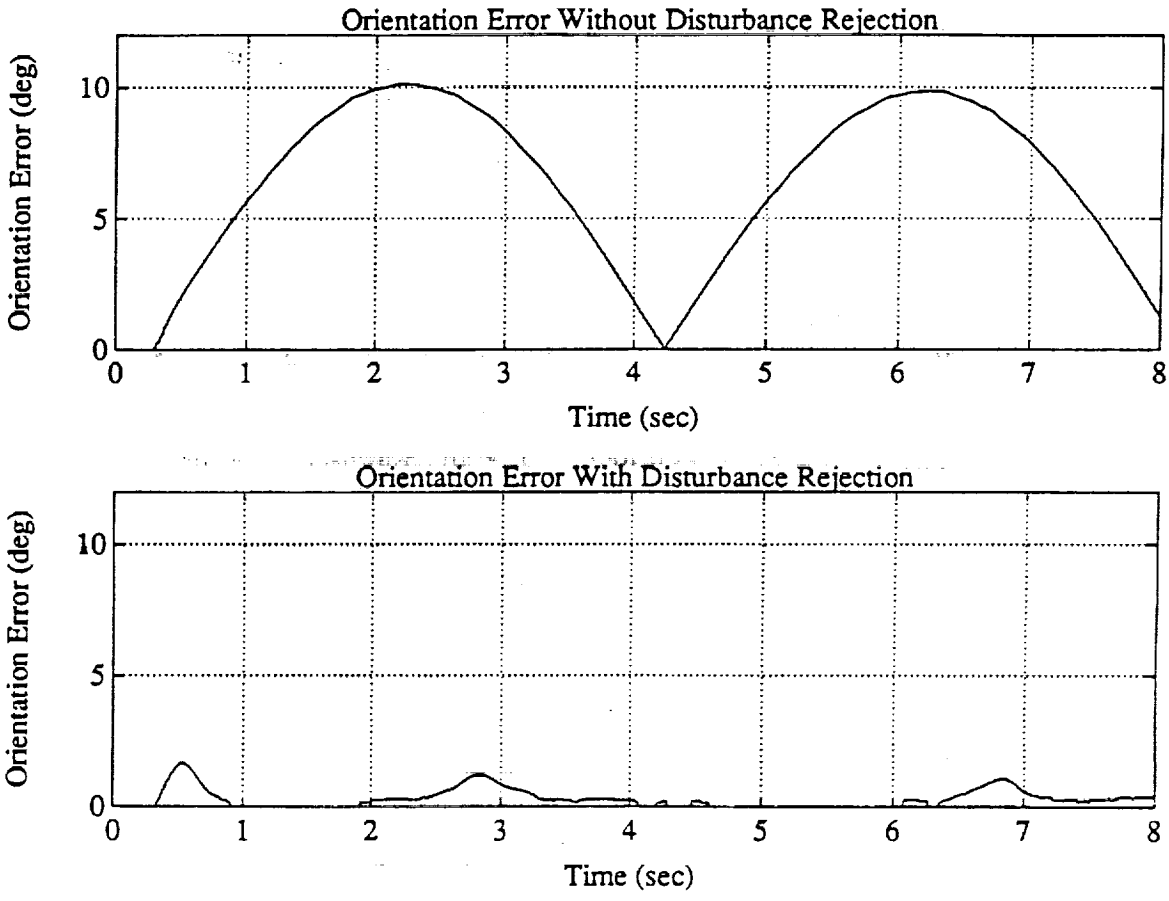

Figure 5.11: Orientation Errors for 10° Amplitude, 8 Second Period Sinusoidal Disturbance in Platform Rotation

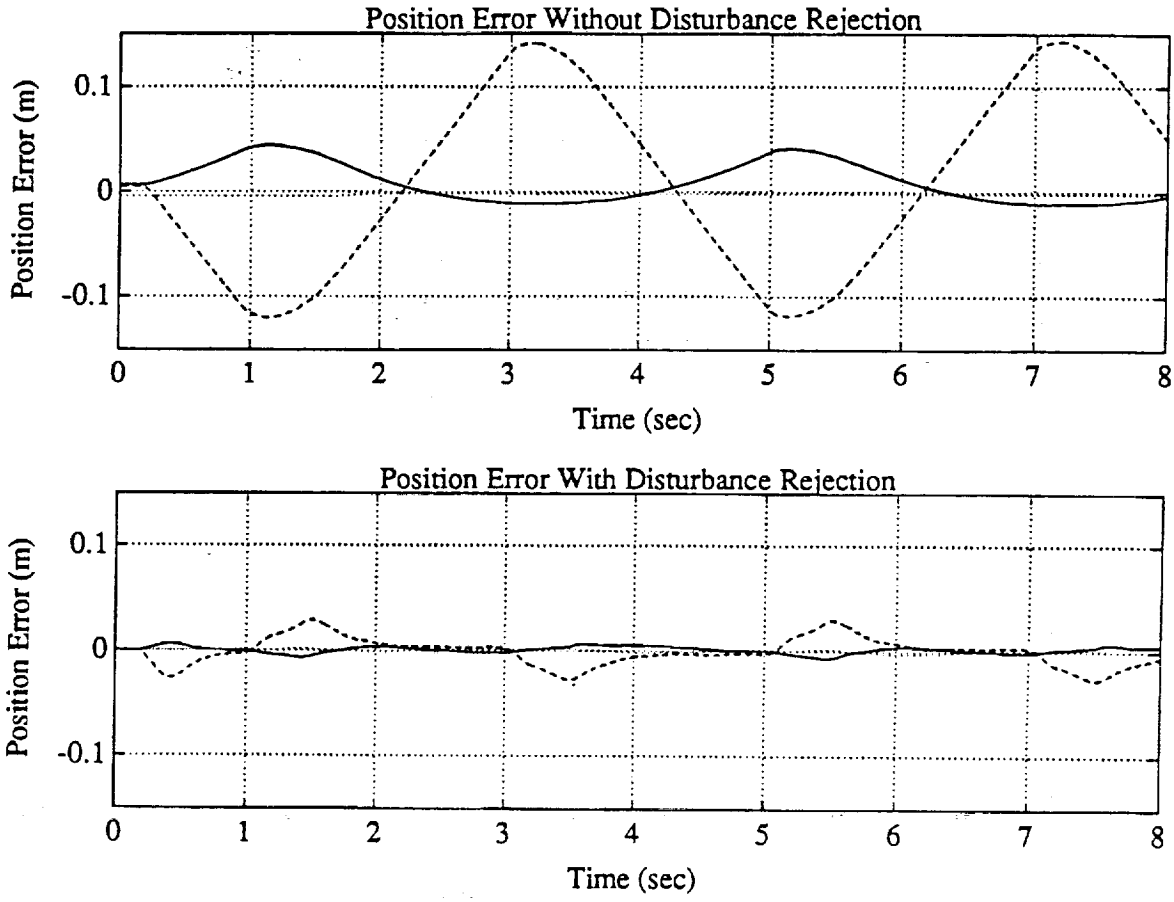

Figure 5.12: Position Errors (X – solid curves; Y – dashed curves; Z – dotted curves) for 10° Amplitude, 4 Second Period Sinusoidal Disturbance in Platform Rotation

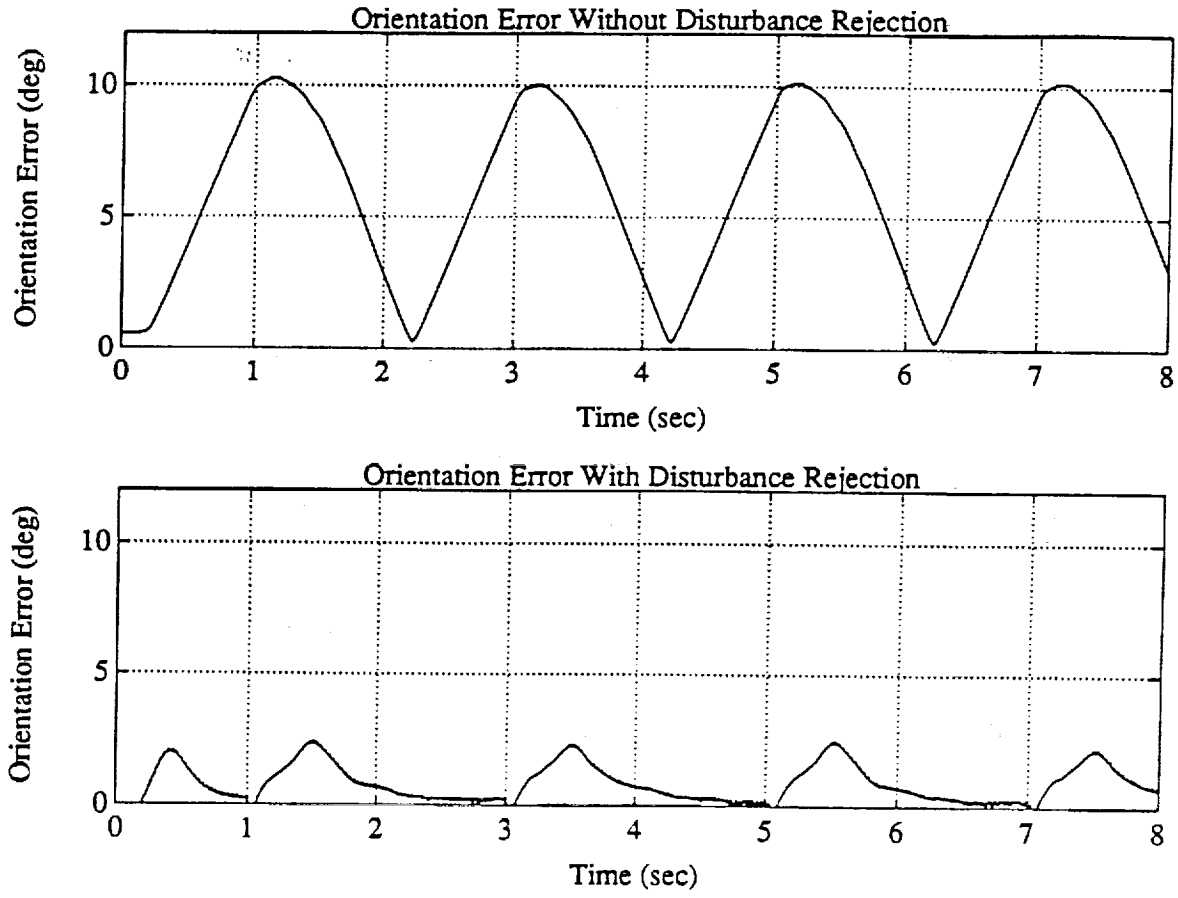

Figure 5.13: Orientation Errors for 10° Amplitude, 4 Second Period Sinusoidal Disturbance in Platform Rotation

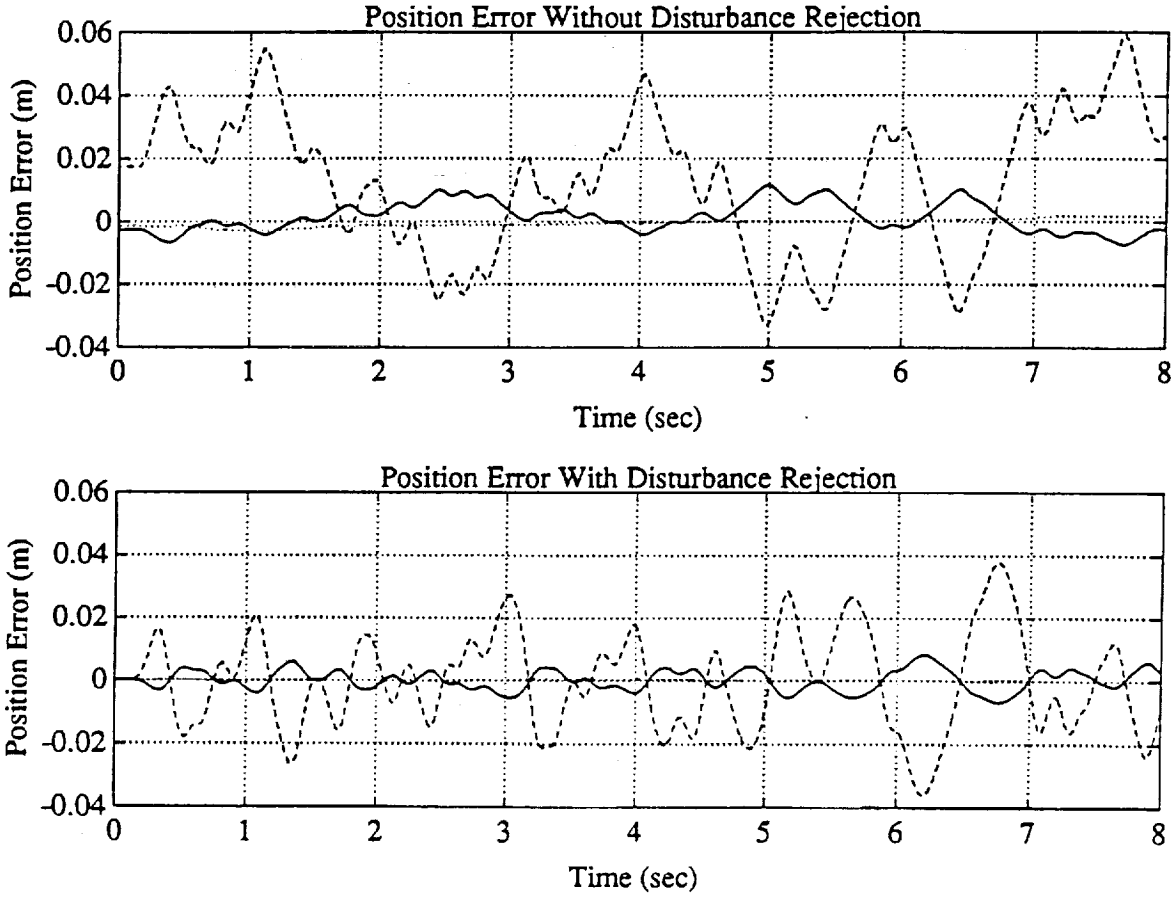

Figure 5.14: Position Errors (X – solid curves; Y – dashed curves; Z – dotted curves) for $\mathrm{Unif}(-0.5°, +0.5°)$ Random Disturbance in Platform Rotation

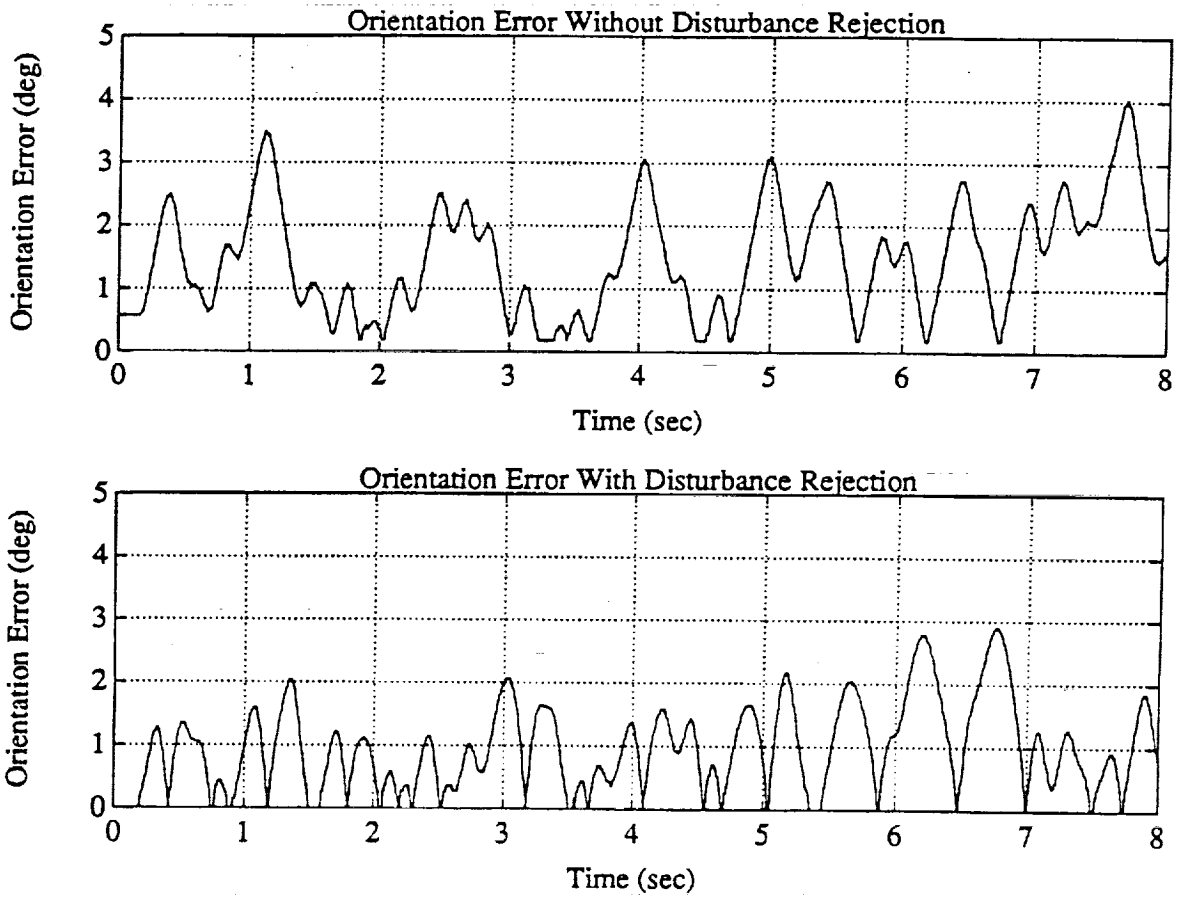

Figure 5.15: Orientation Errors for Unif$(-0.5°, +0.5°)$ Random Disturbance in Platform Rotation

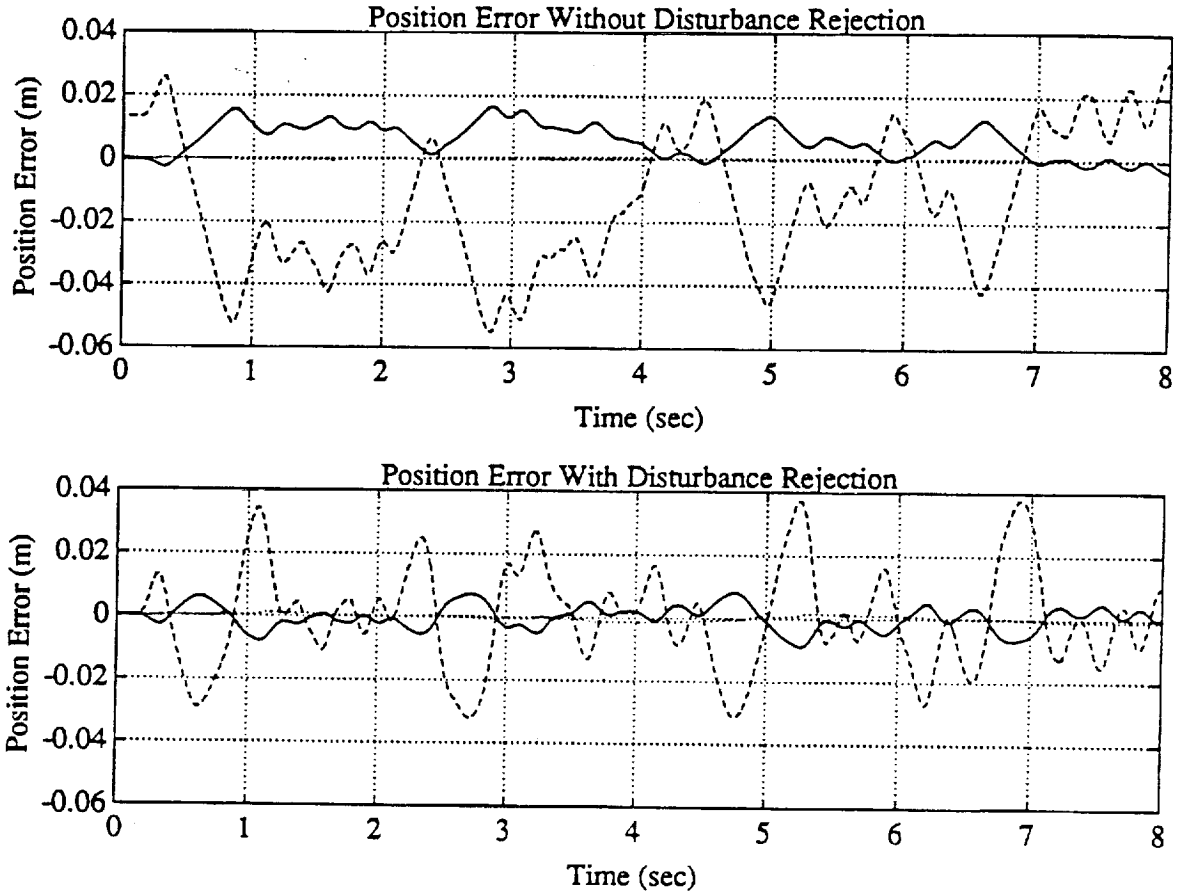

Figure 5.16: Position Errors (X – solid curves; Y – dashed curves; Z – dotted curves) for $\mathcal{N}(0, 0.25°)$ Random Disturbance in Platform Rotation

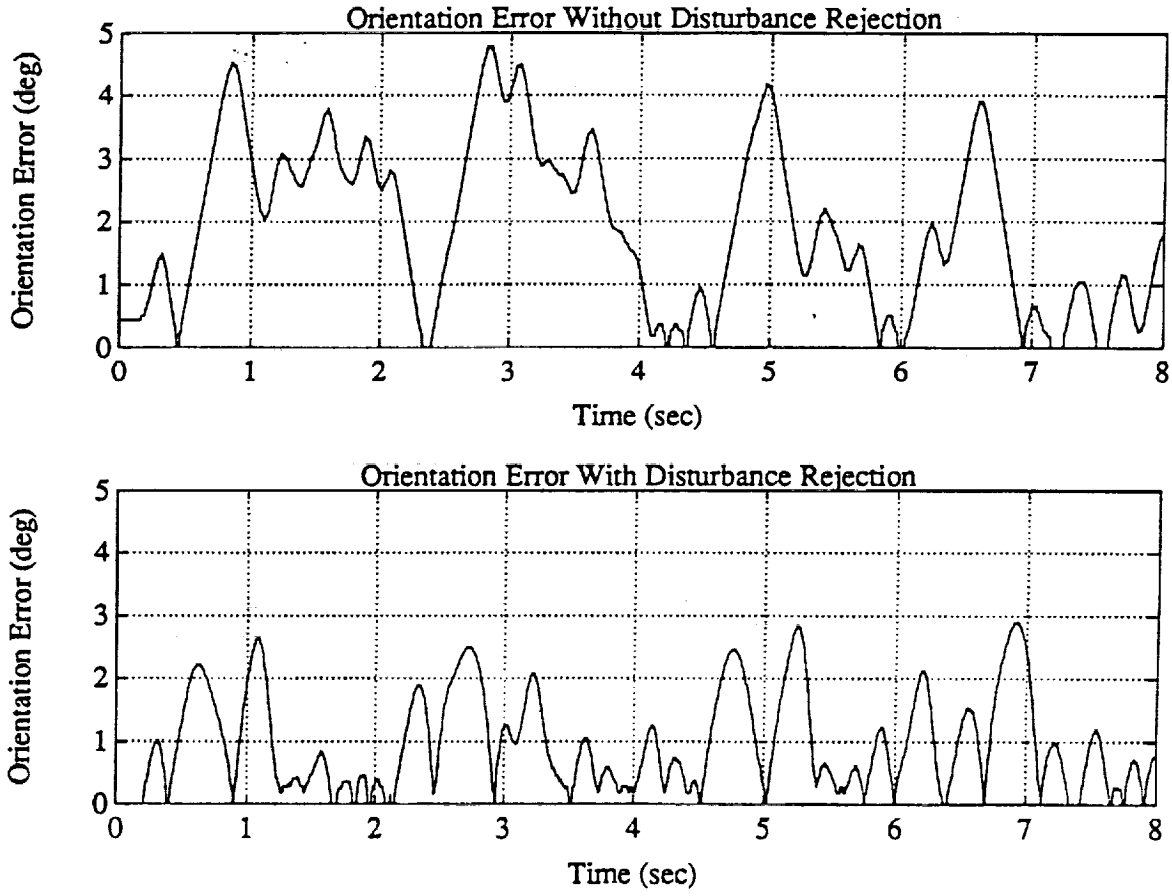

Figure 5.17: Orientation Errors for $\mathcal{N}(0, 0.25°)$ Random Disturbance in Platform Rotation

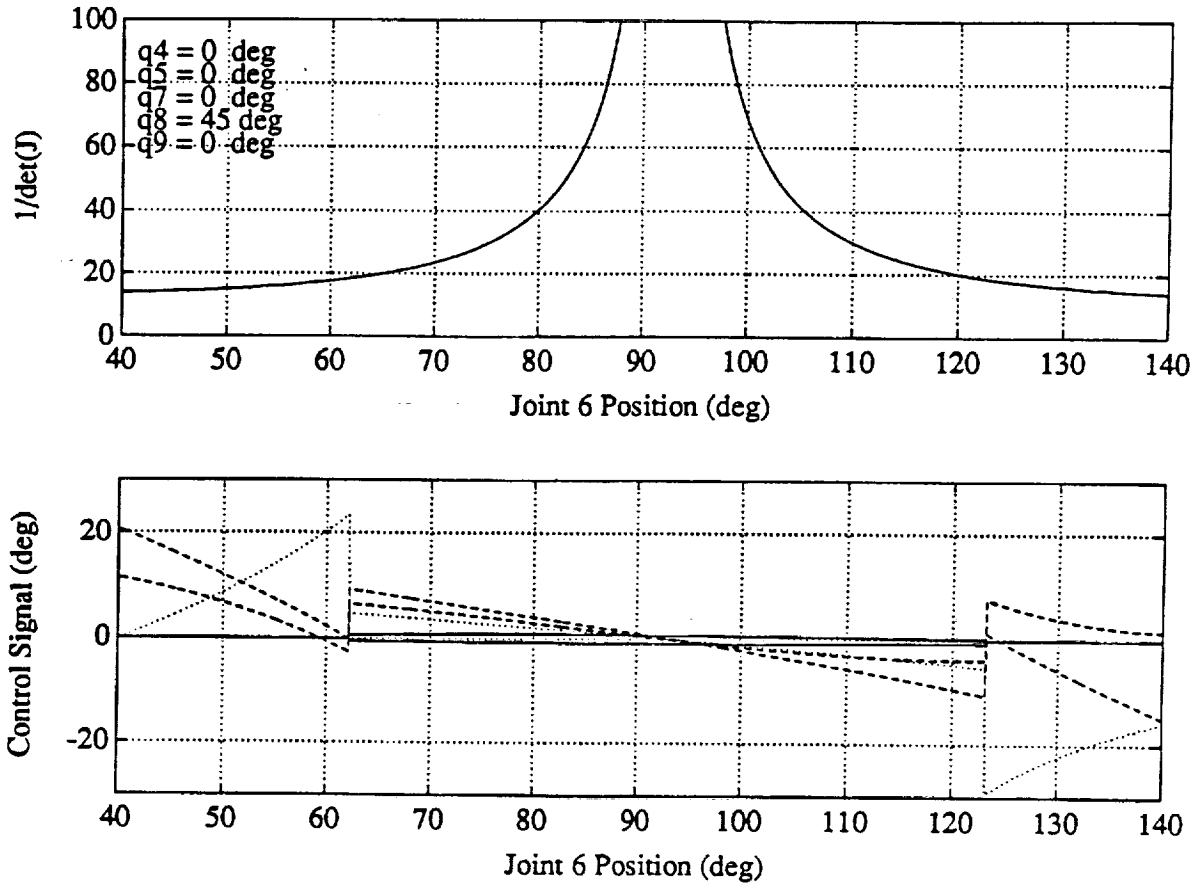

Figure 5.18: Behavior of $1/\det(J)$ and Open-Loop Control Signals ($\Delta\theta_d(1)$, $\Delta\theta_d(4)$ – solid curves; $\Delta\theta_d(2)$, $\Delta\theta_d(5)$ – dashed curves; $\Delta\theta_d(3)$, $\Delta\theta_d(6)$ – dotted curves) Near Arm Fully Stretched Singularity

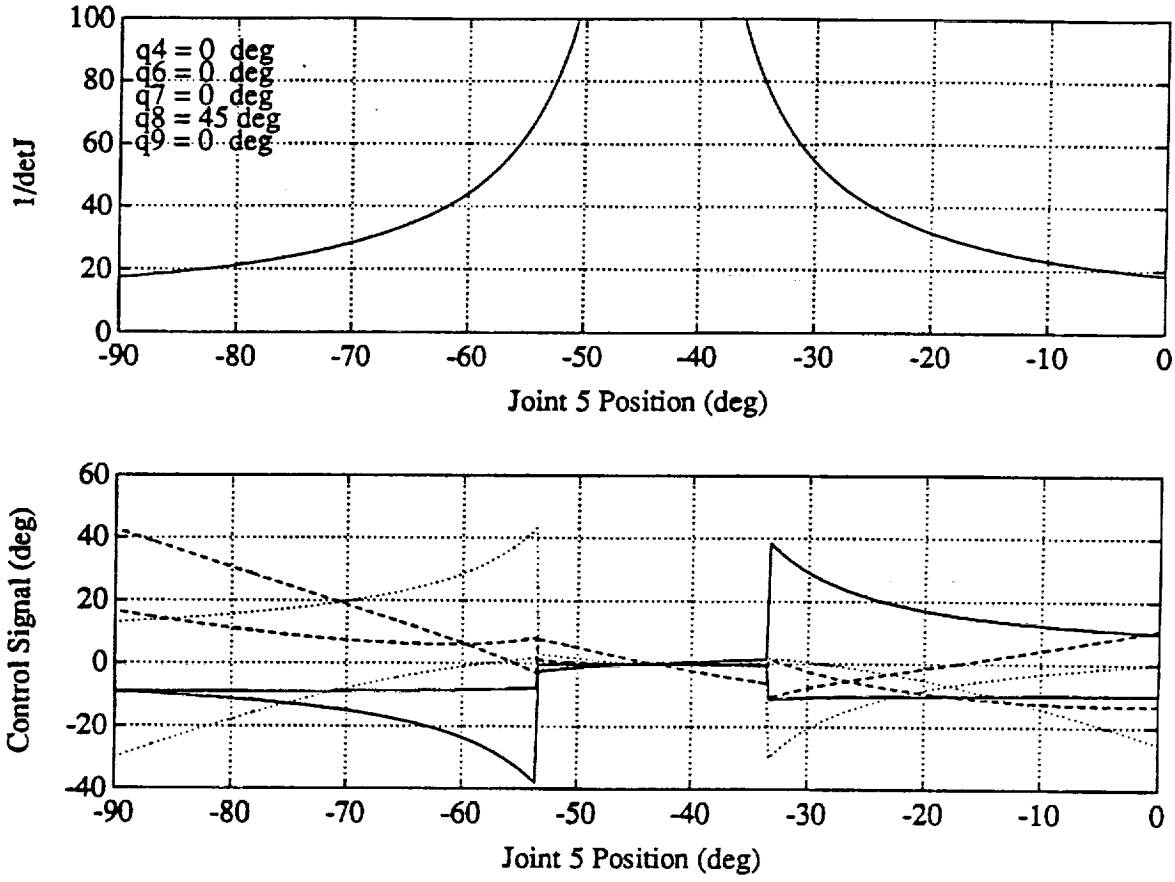

Figure 5.19: Behavior of $1/\det(J)$ and Open-Loop Control Signals ($\Delta\theta_d(1)$, $\Delta\theta_d(4)$ – solid curves; $\Delta\theta_d(2)$, $\Delta\theta_d(5)$ – dashed curves; $\Delta\theta_d(3)$, $\Delta\theta_d(6)$ – dotted curves) Near Hand Over Head Singularity

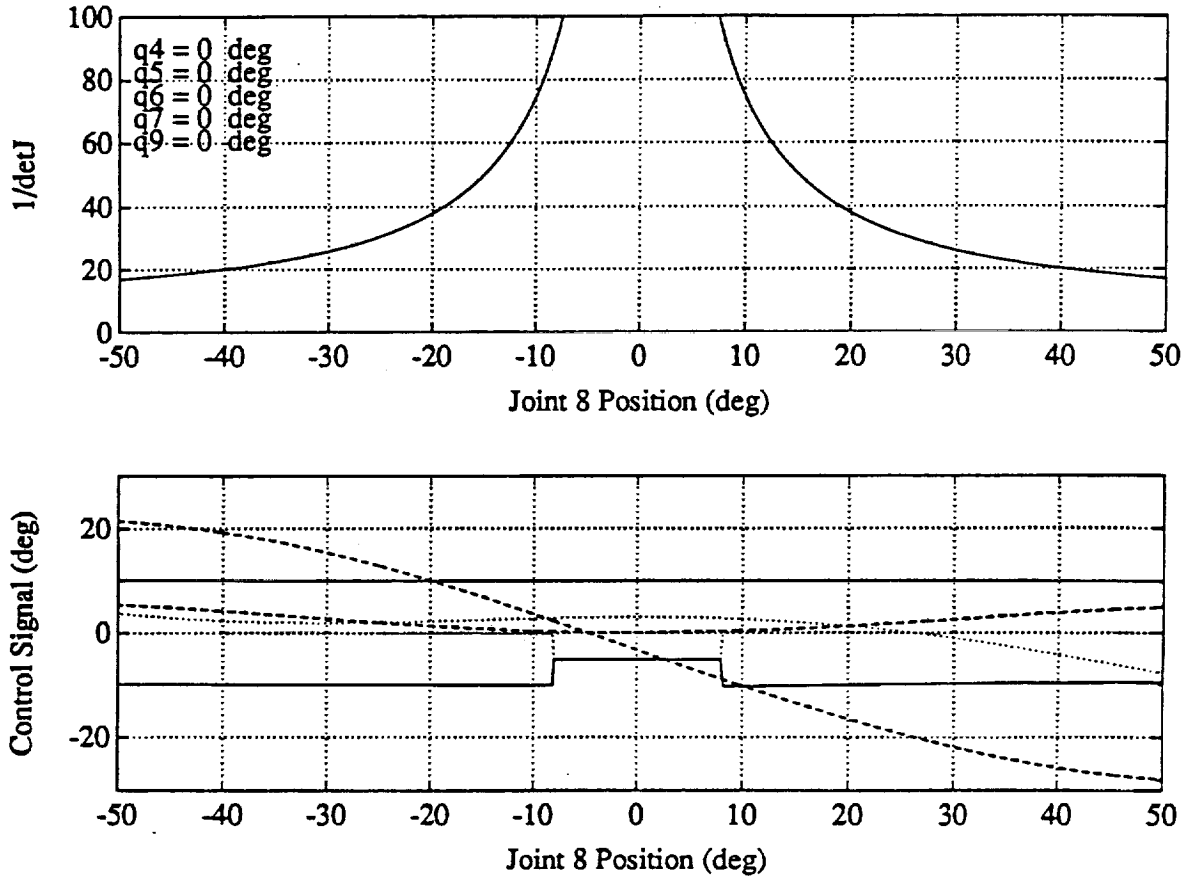

Figure 5.20: Behavior of $1/\det(J)$ and Open-Loop Control Signals ($\Delta\theta_d(1)$, $\Delta\theta_d(4)$ – solid curves; $\Delta\theta_d(2)$, $\Delta\theta_d(5)$ – dashed curves; $\Delta\theta_d(3)$, $\Delta\theta_d(6)$ – dotted curves) Near Wrist Singularity