NASA Technical Memorandum 110314

Robot Position Sensor Fault Tolerance

Hal A. Aldridge
Langley Research Center, Hampton, Virginia

January 1997

National Aeronautics and
Space Administration
Langley Research Center
Hampton, Virginia 23681-0001

Table of Contents

List of Figures

List of Tables

1. Introduction and Literature Search

1.1 Justification for fault tolerance in robotics

Robots are finding new applications in critical space, nuclear, and underwater tasks. These tasks are being driven by the need for maintenance and construction operations to be performed in areas where it is hazardous for humans to work. While current advanced laboratory manipulators are capable of successfully completing many of the tasks, it is difficult to convince mission planners to rely on robot completion of critical tasks. This lack of confidence necessitates expensive preparations for possibly dangerous human intervention.

One reason for this lack of confidence is the fault tolerance of most existing laboratory robot designs. Classically designed serial manipulators contain many single point failures. Failures can lead to the inability of the robot to complete the assigned tasks or, if not handled properly, robot runaway. In the robot engineer's efforts to demonstrate the ability of the robot to complete a candidate task, the inclusion of fault tolerance in the robot design or control system is normally an afterthought. The lack of fault tolerance in the resulting robot system can lead to spectacular failures further reinforcing the user's decision to make robotics an expensive convenience, not a necessity.

The ability to implement advanced machine intelligence which can guarantee high level task completion is beyond the scope of current technology. However, the ability to monitor robot systems to check for mechanical or electrical failures is possible. If properly designed, robot systems can detect and recover from many failure modes. The key is that the robot mechanisms must be properly designed. A fault tolerant control system added at the last minute can do little make a robot fault tolerant without proper sensor information and robot mechanical ability.

Robot manipulator designs can incorporate redundant sensors and actuators and can be kinematically designed to continue operation after joint failure. Most of these techniques require new robot designs or major modifications to existing designs. In practice, it is sometimes advantageous to use an existing, proven robot design. A new, fault tolerant design can be failure prone due to minor faults that have been fixed in existing designs. Existing designs may already be operational or in late planing stages. The cost of significant changes in these designs to incorporate fault tolerance using existing techniques can be prohibitive.

One failure mode that has received little attention is joint position sensor failure. While joint position sensors are arguably the most important type of sensor in robot designs, their failure has been dismissed as easily compensated for by the inclusion of backup position sensors. While including backup position sensors is effective, it can force major changes to optimized robot joint designs. Unfortunately, this chapter will show that other existing techniques for joint position determination, while adequate for instantaneous failure detection and identification, are not practical for extended use in unstructured environments. The trivial solution to this problem to lock the joint with the failed sensor as the control system would do with a failed actuator. Since the existing robot is not kinematically optimized for fault tolerance needlessly locking a joint can be disastrous to the robot's ability to complete a task. As a result, position feedback adequate for continued operation of the failed joint is even more essential in existing robot designs than in fault tolerant designs.

The goal of this dissertation is to propose a new method for joint position determination and develop a control system capable of satisfactory operation during position sensor failure. Several key problems must be solved by the proposed system:

1. The sensors required by the proposed system must currently be incorporated in advanced robot designs or be proven sensors which can be incorporated into robot designs without major changes to the robot system
2. The proposed system must be suitable for extended use in unstructured environments
3. The control system must be stable without position sensor feedback and track desired trajectories adequately with the feedback from the backup sensors
4. The incorporation of fault tolerance should not significantly degrade performance during normal operation
5. The control system must be implementable on reasonable computer hardware

An important consideration in developing techniques for sensor based control is the performance of the control algorithms on a real system with real sensors. Unexpected sensor effects in real systems can invalidate control laws that worked well in simulation. All new algorithms developed in this dissertation will be experimentally tested on an advanced robot system of the type that would be considered for fault tolerant modifications.

Sections 2 through 5 of this chapter will detail the state of the art in fault tolerant and robust design techniques. Special emphasis will be given to the applications of joint torque sensors and Cartesian accelerometers in robotics. Section 6 will introduce the proposed position sensor fault tolerant control system utilizing virtual passive joint control for stability and two methods utilizing accelerometer feedback for joint position determination without integration.

1.2 General purpose fault tolerance

Fault tolerance is a necessary part of critical dynamic systems. The fault tolerance process can be divided into two major components. First, a fault must be detected and identified by the system. The system must then decide how to recover from the fault and implement the appropriate actions.

1.2.1 Failure detection and identification (FDI)

Once a fault occurs, it must be detected and isolated before it can be corrected. A good overview of FDI techniques can be found in a text by Patton et al. [1]. To determine whether a fault has occurred, the effects of the fault must be observable. For example, if a motor is supposed to turn at 1000rpm, a sensor in the system must be able to detect that the motor is turning at that speed. The simplest way to accomplish this task is to place a tachometer on the motor and directly monitor the motor's speed. As the number of system states that must be monitored increases, the number of sensors required for direct monitoring can become impractical or some states may not be directly observable. These problems lead to using analytical methods to observe the system states that must be monitored. These analytical methods incorporate a system model along with available system outputs to produce the appropriate quantities. Techniques include banks of Kalman filters and Leunberger observers [2], parity space approaches [3], and parameter estimation techniques [4]. These methods determine a residual, the difference between observed and desired system states. Unfortunately, due to errors in the system model used to determine the residual, a non-zero residual does not imply a system failure. Several authors have proposed techniques for robust residual generation [5,6,7,8]. Limiting the number of false alarms due to modeling errors is currently one of the main emphasis of FDI research.

1.2.2 Fault recovery

Once a fault has been identified, the operator or control system must determine how to handle the fault. The main choices are to halt the appropriate parts of the system or to use other means to bypass the fault and continue operation. To be able to continue operation, the system must have some redundant characteristics. Direct redundancy, such as backup amplifiers, sensors, or motors, can be built into the system. When a failure occurs, the system is switched to the backup device. Direct redundancy, while very effective, can be expensive and difficult to implement because each critical system must have a backup system capable of replicating its outputs. A more complex method of failure recovery involves exploiting the analytical redundancy in a system. An example of an application utilizing analytical redundancy in a multi-engine aircraft would be using differential engine thrust to control airplane yaw instead of a failed rudder. The use of analytic redundancy requires that the control system be reconfigured to utilize the working components to perform the task of the failed components. Work on reconfigurable control systems can be found in [9,10,11,12,13].

1.3 Robot fault tolerance

Existing general purpose FDI and fault recovery techniques can be applied to robot systems. Due to the highly nonlinear nature of robot systems, some general purpose methods become cumbersome quickly. The following sections detail specific applications of fault tolerance in robot systems.

1.3.1 Robot joint failure detection and identification

Direct measurement of joint position and velocity is built into most robot systems. As a result, the control system can monitor the joint actuators to determine if they are performing as commanded. A problem occurs in this direct measurement when the system is moving. Due to errors in the control model, the robot may not track the desired trajectory exactly. The FDI system must be able to determine if this error is due to a failure or a control error. Choosing constant, large acceptable error bounds to reduce false alarms during movement will reduce the effectiveness of the FDI system. Horak [14] proposed a method to determine allowable tracking errors in general dynamic systems due to bounded modeling error. The complex nature of robot systems makes this method computationally intensive. Exploiting the structure of the robot control equations, Visinsky et al. [15] developed an efficient method to determine acceptable dynamic tracking errors due to modeling errors.

Parameter identification methods can be used in robot fault detection. An example of this application can be found in Feryermuth [16]. The author uses on-line parameter identification techniques to compare actual parameters such as link inertia, link mass, and drive friction with previously computed quantities. The author then relates changes in these parameters over time to faults in the robot systems such as drive train distortion, motor heating, overloads (dynamic and static), and brake defects.

In another paper by Visinsky et al. [17], the authors introduce some limited analytical sensor redundancy. To monitor motor and sensor faults, the authors compare sensed position and sensed velocity, sensed position and computed acceleration, sensed velocity and computed acceleration, and sensed velocity and computed jerk. By determining which of these quantities are outside acceptable error bounds, this technique will allow for the isolation of the failed sensor or actuator.

1.3.2 Robot joint failure recovery

When a robot joint failure occurs, the control system has three options. If the failure is not recoverable, i.e. is the robot does not have the direct or analytical redundancy to continue to operate, the control system must halt the task. If direct redundancy exists, the control system can switch to the backup system and continue the task. If analytical redundancy exists, the control system can be reconfigured to use other joint actuators or sensors to complete the task.

Development work has been done on joint designs that incorporate direct actuator redundancy [18]. In these designs, the innovative work is in reducing the weight and size of the redundant actuator package while maintaining performance. If a robot design does not incorporate directly redundant actuators, another option is presented by Bergerman and Xu [19]. Assuming the failed actuator is not frozen and a joint brake is available, the authors present a variable structure based control system to treat the failed joint as a passive actuator and control its movement by dynamic effects.

If the joint is locked, the kinematic design of the robot will determine if it can still complete a task given a set of locked joints. Several papers present an analysis of crippled robot motion and discuss techniques to design redundant and non-redundant robot kinematics to allow for specific task completion [20,21,22,23]. Paredis et al. [21] described techniques for fault tolerant robot design given a set of trajectory points a robot must be able to reach. Paredis and Khosla [24] presented global trajectory planning algorithms to avoid unfavorable joint robot positions for fault tolerant manipulators before and after failure occurs. When the decision is made to lock a joint, the control system must be reconfigured. This reconfiguration will change the manipulator parameters resulting in a control discontinuity that could effect the current end-effector trajectory. Two papers by Ting et al. [25,26] detail methods to reduce this reconfiguration error. Since the other joints in the system must produce the torque required to complete the task in this failure recovery mode, it is possible that some joints could be called upon to produce more torque than they are capable. Ting et al. [27] presents a method to properly redistribute the torque in kinematically redundant manipulators during failure.

Another approach to joint failure recovery is to make the robot modular. This modularity will allow joints to be added and removed as necessary to bypass joint failures. The design of the Reconfigurable Modular Manipulator System is discussed in [28] and [29]. In Kelmar and Khosla [30], the authors present a method to automatically determine the forward and inverse kinematics of the reconfigured robot. The new control system for the reconfigured manipulator can be implemented quickly and efficiently using the Chimera real time operating system [31] in conjunction with the Onika graphical programming environment [32].

The robot control system must have a high level control strategy to implement a fault detection and isolation process and, when the fault is found, a fault recovery procedure. Visinsky et al [17,33] propose a layered approach to the high level control. On the highest layer, the authors propose an expert system based supervisory system to reconfigure the system given inputs from the fault detection layer and the robot servo layer.

1.4 Robust Control

Robust joint control methods can be important to fault tolerant architectures. If a robot joint is locked during failure, the two links to which it is attached must be treated as one. Since the original links contained modeling errors in their dynamic parameters, the dynamic parameters of the new link can have an even greater modeling error. Several texts and survey papers have summaries of existing techniques [34,35,36]. The most popular robust control algorithms employ versions of the computed torque controllers, variable structure controllers, and adaptive controllers. Each controller type has its own advantages. Computed torque variations are the most popular and are very effective. However, large gains may be required to obtain robustness. Variable structure controllers, while in theory robust to parameter variations, require control switching that can excite high frequency modes and are not fully realizable in real systems. Adaptive controllers require some time to improve the initial parameters and, as a result, are best suited to repetitive operations.

1.4.1 Applications of torque sensors in robust control

Although joint torque sensors have become an integral part of advanced robot designs, their range of applications has been limited. Their primary uses are to verify joint torque commands and to implement a torque controller to allow for better torque command tracking. Some attempts have been made to use all joint torque sensors in the robot system along with the manipulator Jacobian to control end-effector forces. This force control method proves inferior to end-effector force sensor based methods due to modeling errors and the usually lower resolution of joint torque sensors.

An innovative application of joint torque sensors is in the area of robust control. Papers by Kosuge et al. [37] and Imura et al. [38] advocate using torque sensors to decouple the torque required to drive the joints from the torque required to move the links. In these implementations, joint motor parameters determine the portion of the torque command required to make the motor respond to a joint position trajectory while the torque sensor determines the resistance caused by the link to the joint motion. That sensed resistance is added to the torque required to drive the joint motor to form the complete torque command to the joint. Both methods only consider direct drive joints. Problems with the methods include the simplicity of the joint model used and joint torque sensor effects such as bias, hysteresis, and noise.

1.5 Position sensor fault tolerance

Most robot control techniques, with the exception of pure force or velocity based control, are dependent on accurate joint position sensing. As a result, a fault tolerant manipulator must have a FDI and failure recovery strategy for its position sensors. The most basic method of accomplishing the FDI task is to build directly redundant position sensors into the joint. Analytical redundancy between joint position and joint velocity sensors can also be exploited instantaneously as detailed in Visinsky et al. [17] for FDI. If the robot end-effector can be tracked, then a deviation from the trajectory calculated from the joint sensors and that determined from the end-effector tracking would imply a joint sensor failure. This method is problematic due to the difficulty of accurate end-effector tracking in unstructured environments, possible problems in identifying which joint sensor is in error (especially in redundant robots), and the slow update rate of some tracking methods.

After a joint position sensor failure is identified, a recovery process must be undertaken. If directly redundant position sensors are available, then control is switched to the redundant sensor and operation can continue. However, if only analytical redundancy exists, then the reconfiguration decision is more difficult. While joint velocity sensors can be used instantaneously to detect joint position sensor failure, integrating the sensed velocity over time will lead to an unacceptable position offset. The difficulty of end-effector tracking in unstructured environments and its slow update rate make end-effector tracking an unsuitable candidate to generate position feedback. Although reasonable methods exist to determine if joint position sensor failure has occurred using analytical redundancy, there does not appear to be a satisfactory method to continue operation after a joint position sensor failure without direct sensor redundancy. It would be advantageous to have a system internal to the robot utilizing analytical redundancy that provides joint position data at a high rate for FDI and failure recovery.

If a robot joint loses all position information, there is no other choice but to lock the joint in its current position, at least temporarily. At that point, hopefully some experiments can be conducted to verify the final joint position and some crippled operation can continue. The locking of the joint is usually accomplished by engaging the joint brake. Unfortunately, some robot designs do not incorporate joint brakes and joint brakes can fail. If the joint position information will be available from a recovery method that takes a few seconds to calibrate or is intermittent from a partially failed sensor, repeatedly locking and unlocking the joint brake can lead to unwanted vibrations and other wear on the joint drive. It would be advantageous to have a controller that could stabilize a robot joint without position information and would not unnecessarily shock the system or otherwise degrade performance when active. Such a controller could be activated whenever position information was suspect.

In the situation that a position sensor failure is detected but the failure is diagnosed as a position offset or the recovery method has an offset problem, if the offset is not exactly known control stability problems could occur. Most robust control designs are not designed to deal with position error. Due to the high gains that some controllers use to overcome parameter error, the controllers could be especially susceptible to position error. The current robust control methods that utilize joint torque sensors could be resilient to any errors caused by unknown link position due to their direct sensing of link static and dynamic effects instead of relying on position dependent model calculations.

1.5.1 Applications of accelerometers in position sensor fault tolerance

Accelerometers have found applications in such diverse fields as automotive safety [39,40] and control of space structures [41]. Accelerometers have also found applications in robotics. The main use of accelerometers in robotics has been in flexibility control [42,43,44]. Accelerometers have also been used in system identification [45], runaway detection [46,47], orientation measurement in cooperation with inclinometers [48], and sensing slip in grasping operations [49]. Of these applications, only the runaway detection and orientation measurement applications relate to fault tolerance. While the accelerometer based runway detection systems are efficient in detecting rapid joint runaway conditions, they have problems with slow drifts and provide no method to recover from the runaway condition. Orientation measurement relates to fault tolerance in that it could be used to compare the sensed orientation of robot links with the orientation calculated from position sensors for FDI. In the orientation measurement application, the accelerometers are utilized only for high frequency components of the orientation. An inclinometer provides steady state and low frequency information canceling drifts and offsets caused by the double integration of the accelerometer information. Problems with the double integration of accelerations to obtain positions due to various sensor effects has limited the utility of accelerometers in position sensor fault tolerance.

1.6 Research problems to be solved in dissertation and contributions

This literature search has identified that joint position sensor fault tolerance is an essential characteristic for robust robot controller design. While currently available methods can be used to detect and identify joint position sensor failure, without direct sensor redundancy there is no satisfactory method to recover from the failure. It would be advantageous to have system internal to the robot utilizing analytical redundancy that provides high rate joint position data for FDI and failure recovery. The goal of this dissertation is to provide a practical method for continued operation during joint position sensor failure. An effort was made to verify all theory in this dissertation on a real system. To that end, Chapters 2 through 7 of this dissertation contain three related pairs chapters, one with theory and simulation and the second with experimental results.

The predominate joint position sensors used in robotics currently, optical encoders and inductive resolvers, provide the most noise free, accurate, directly measured position information currently available. Any analytical method developed will be nosier, less accurate, and more susceptible to bias error. As a result, the joint controller will need to be robust. Analytical methods could require some startup time after a failure is detected. As a result, the controller must be able to tolerate the sudden, temporary loss of position data without instability. If the loss of position data continues and if the joint is not equipped with a joint brake, the controller should attempt to slow the joint to a stop and attempt to resist further movement to allow for continued operation of other joints. Ideally, this controller would have other beneficial qualities that would enhance operation of a joint with an operational joint position sensor. Chapter 2 of this dissertation will present a controller with all the desired properties based on joint torque sensor feedback, a linear joint model, and virtual passive control. Chapter 3 will explore the characteristics of this controller on three joints of a laboratory manipulator. In chapters 5 and 7, up to six joints of the manipulator will be controlled by this new method with position feedback of varying quality.

Although the controller developed can stabilize a joint without position feedback, it cannot servo to a position without position feedback. A method to provide joint position during sensor failure is still required. Utilizing double integration, perfect accelerometers would be ideal for this purpose. Current accelerometer designs are small, require little power, and are easily incorporated into existing robot designs. Unfortunately, sensor effects such as noise and bias make the double integration based technique impractical. This dissertation will develop two techniques for using accelerometers to determine joint position without integration. The first technique is a joint specific method presented in chapter 4. Experimental results obtained by instrumenting three joints of a laboratory robot are presented in chapter 5. Usage of this method in both FDI and failure recovery is discussed. The other position determination method is a system wide method discussed in chapter 6. This method reduces the amount of accelerometers required for fault tolerance over the joint specific method but has increased computational complexity and reduced convergence properties. Chapter 7 details experiments conducted using the system wide method to provide fault tolerance to different sets of up to three simultaneously failed position sensors in both FDI and fault recovery. These techniques utilize the robot kinematic equations and a known acceleration field. Computational methods for real-time implementation of these techniques are also presented.

Existing fault tolerant control systems can incorporate the new controller and position determination methods. The position determined by the joint specific or system wide methods can be used by existing FDI methods to determine if a joint position sensor has failed. At failure, the fault tolerant control system decides if recovery is possible given the characteristics of the system and the task. If recovery is possible, the position determined by the either accelerometer based algorithm can be used along with the virtual passive based torque controller to continue operation as shown in this dissertation. The scope of this dissertation does not include the advancement of FDI or reconfiguration techniques. The goal is to produce a sensor/controller combination with the desired characteristics to provide for continued operation without joint position feedback utilizing existing FDI and reconfiguration techniques.

The research contributions presented in this dissertation will be:

1. The design, development, and experimental testing of a virtual passive joint torque controller which improves position tracking performance during normal joint operation without a complex dynamic model and can stabilize a robot joint without position feedback.
2. The design, development, and experimental testing of a joint specific method of determining joint position using Cartesian accelerometers without integration for FDI and the integration of the joint specific position determination method and the virtual passive torque control method for failure recovery.
3. The design, development, and experimental testing of a system wide method of determining joint position using Cartesian accelerometers without integration for FDI and the integration of the system wide position determination method and the virtual passive torque control method for failure recovery.

2. Dissipative Controller Design for Robot Systems Using Joint Torque Sensors and a Joint Model

2.1 Introduction

Control of robotic systems is a difficult problem due to the nonlinearity of the complex system equations. Several techniques to control these nonlinear system equations have been studied. Some methods, such as Jacobian linearization[50] and pseudolinearization[51], have attempted to linearize the system and apply classical linear system control techniques to the problem. These methods are only valid close to a linearization point or trajectory. The feedback linearization[52] technique attempts to linearize the equations over a large workspace and has been popular in the robotic control literature for some time. Feedback linearization requires good knowledge of the system parameters and states, else some of the nonlinearity will not be canceled out. Variable structure control is a purely nonlinear control method which has been a popular research topic[53]. Although variable structure control is robust, the fast switching required by the controller to maintain this robustness is difficult to achieve without chatter. These are only a few of the many control methods that have been applied to robot systems.

The main reason for the nonlinearity in the equations is the need to calculate the dynamic effects on the structure. Controlling the nonlinear plant based on the full dynamic equations is not the only method for controlling a multi-body dynamic system such as a robot manipulator. If the joints have a torque sensor along the drive axis, the problem can be reduced to controlling the individual joint dynamics. Work in this area has been published recently[37,38]. These methods use known, mostly linear, electric motor driven joint models to control joint acceleration and velocity. These methods still require measurement of joint position and velocity to compute the control inputs.

Passivity based control methods have been applied to multi-body dynamic control in robotics [54,55] and vibration control of space structures[56]. The problem with the basic passivity control approach is that it requires velocity feedback. The virtual passive dynamic control approach [57] has been successful in stabilizing systems with displacement, velocity, or a combination of acceleration with displacement and velocity feedback.

This chapter develops a method to stabilize a robot manipulator with joint torque sensors without directly measuring joint position, velocity, or acceleration. The torque sensor output will be used in conjunction with a simple joint model and the virtual passive dynamic-based control technique to quickly dissipate the kinetic energy in the robot system. A Lyaponov stability proof will be presented for the control technique.

The robustness of the system will be discussed and simulations of the control system will be shown. Chapter 3 will discuss experimental results.

2.2 Dynamic Model

The following derivation is based on a model of a direct drive, electric motor driven, revolute joint with an output torque sensor presented in Kosuge[37]. This joint is assumed to be part of a serial linkage consisting of other revolute joints and rigid links. The structure of the proposed joint is shown in Fig. 2.1. Each joint is assumed to have two parameters, rotor inertia and viscous friction.

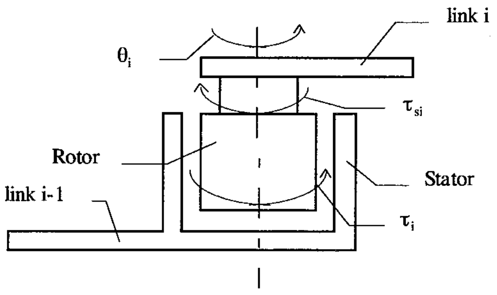


Figure 2.1: Diagram of proposed direct drive joint

Paraphrasing the development in Kosuge[37], the model of the joint is determined by applying a Newton-Euler iterative dynamics[58] approach to a rigid, serially linked structure with revolute joints. This derivation included the motor inertial and viscous friction terms. The full equation for the joint torque was computed. The terms required to compute the torque were divided into three groups: terms which depend only on the i^{th} joint, terms depending on other joints which effect the i^{th} joint, and terms that involve link inertias, masses, and lengths. All equations in this chapter are given in the robot base coordinate system for consistency with Kosuge. The joint torque model resulting from this derivation is found in Eq. (2.1).

$$\tau_i = m_i\ddot{\theta}_i + \tau_{si} + v_i\dot{\theta}_i + f_i \tag{2.1}$$

$$M_{ri} \equiv A_{i-1}^{0} M_i A_0^{i-1} \tag{2.2}$$

$$m_i \equiv z_0^T M_{ri} z_0 \tag{2.3}$$

$$\omega_i = \omega_{i-1} + z_{i-1}\dot{\theta}_i \tag{2.4}$$

$$\dot{\omega}_i = \dot{\omega}_{i-1} + z_{i-1}\ddot{\theta}_i + \omega_{i-1} \times (z_{i-1}\dot{\theta}_i) \tag{2.5}$$

$$
\begin{aligned}
f_i \equiv z_0^T [M_{ri}(A_{i-1}^0 \dot{\omega}_{i-1}) + A_{i-1}^i (A_i^0 \omega_i) \\
\times (A_i^{i-1} M_{ri} A_{i-1}^i A_i^0 \omega_i) + M_{ri}(A_{i-1}^0 \omega_{i-1} \times z_0 \dot{\theta}_i)]
\end{aligned}
\tag{2.6}
$$

Where,

M_i	Inertia matrix of the rotor of the i^{th} joint
v_i	Coefficient of the viscous friction torque acting on the rotor around the joint axis z_{i-1}
A_j^i	Rotation matrix
M_{ri}	Rotor inertia matrix in reference frame
τ_i	Torque at joint input
τ_{si}	The sensed torque along the z axis at the joint output
θ_i	Joint position
ω_i	Angular velocity
z_0	Unit z vector $[0\ 0\ 1]^T$

Joint specific terms are evident in the model while link specific terms are contained in the torque measurement along the axis of rotation. With the exception of the nonlinear term f_i, the elements of Eq. (2.1) depend only on values of the i^{th} joint.

In later sections, the goal will be to develop a controller that will produce u_i, the control input, that will stabilize a joint described by Eq. (2.1). If $\tau_i = u_i$, the control input, then Eq. (2.1) can be solved in terms of the sensed torque.

$$
\tau_{si} = -m_i \ddot{\theta}_i - v_i \dot{\theta}_i - f_i + u_i \tag{2.7}
$$

where

$$
\tau_{si} = \tau_{xi} + f_i \tag{2.8}
$$

and

$$
\tau_{xi} = -m_i \ddot{\theta}_i - v_i \dot{\theta}_i + u_i \tag{2.9}
$$

In section 2.4, exact knowledge of the term τ_{xi} is assumed to derive a stabilizing control input for the system. In section 2.5, the robustness of the system to inexact knowledge of τ_{xi} is discussed.

2.3 Virtual passive dynamic control

The controller design technique used in section 2.4 is similar to the passive dynamic control technique presented in Juang[51]. This technique is based on the concept that a mechanical system can be represented by a second order system with inertial, damping and stiffness related parameters. An active feedback controller can be designed with its dynamics equivalent to a mechanical system. The resulting controller is

$$H_M \ddot{x}_c + H_D \dot{x}_c + H_K x_c + g(y_s) = u \tag{2.10}$$

where y_s is the measured system output, g is a user defined function, x_c is the controller state vector of dimension n_c, and H_M, H_D, and H_K are the controller mass, damping, and stiffness matrices respectively. These matrices are design parameters and can be chosen to meet performance and stability requirements. The function g is an arbitrary function of the measured system output, y_s. These outputs can be system states or combinations of system states.

The Lyaponov proof of the stability of the chosen control law will depend on the rate of change of the system kinetic energy. Let T be the total kinetic energy of a mechanical system (linear or nonlinear) with p control actuators at p physical locations described by p generalized coordinates x_{ai} and p control inputs u_i, $i=1,2,\ldots,p$. These generalized coordinates and their derivatives are physical quantities of the system. If a mechanical system is holonomic and scleronomic (no explicit time dependence), a basic result of analytical mechanics [53] that relates the time derivative of the total kinetic energy to the applied forces is:

$$\frac{dT}{dt} = u^T \dot{x}_a \tag{2.11}$$

where $x_a = (x_{a1}, x_{a2}, \ldots, x_{ap})^T$.

Choose the Lyaponov candidate function to be of the type:

$$L = T + q(x_a, \dot{x}_a, x_c, \dot{x}_c) \tag{2.12}$$

where q is an arbitrary function of the actuator and controller states and their rates. Taking the time derivative and substituting Eq. (2.11) yields,

$$\frac{dL}{dt} = u^T \dot{x}_a + \dot{q}(x_a, \dot{x}_a, \ddot{x}_a, x_c, \dot{x}_c, \ddot{x}_c) \tag{2.13}$$

If the control inputs, u_i, are designed to cause the rate of change of the Lyaponov function to be negative, the stability of the system is guaranteed by Lyaponov stability theory. This stability implies a constant decrease in the kinetic energy of the system.

Remembering that u is a function of system outputs, states, and controller states, it can be designed to cancel out and combine with terms in the derivative of the Lyaponov candidate function to result in an equation of the following form:

$$\frac{dL}{dt} = -\dot{x}_a^T D \dot{x}_a - \dot{x}_c^T R(x_a, \dot{x}_a, \ddot{x}_a, x_c, \dot{x}_c, \ddot{x}_c) \tag{2.14}$$

Where R is a linear function and D is a matrix involving the system damping. The desired function which implies a constantly decreasing Lyaponov function is,

$$\frac{dL}{dt} = -\dot{x}_a^T D \dot{x}_a - \dot{x}_c^T D_c \dot{x}_c \tag{2.15}$$

that can be obtained by making the equality,

$$R(x_a, \dot{x}_a, \ddot{x}_a, x_c, \dot{x}_c, \ddot{x}_c) = D_c \dot{x}_c \tag{2.16}$$

The controller state can be calculated using Eq. (2.16) and substituted into Eq. (2.10) to determine the u_i required to maintain the Lyaponov stability condition.

2.4 Torque Feedback

The goal of this section is to provide a Lyaponov stability proof for using virtual passive dynamic control with torque sensor feedback to stabilize a system. For simplicity of the proof, let the quantity τ_{xi} be known exactly. Inexact knowledge of τ_{xi} and its relationship to τ_{si} will be dealt with in section 2.5. Choose the Lyaponov function similar to Eq. (2.12) with $x_a = \theta$. A candidate Lyaponov function for the stability proof is:

$$L = T + \frac{1}{2}(\dot{x}_a + \dot{x}_c)^T K_\tau M_{rz} (\dot{x}_a + \dot{x}_c) + \frac{1}{2} x_c^T K_c x_c \tag{2.17}$$

$$M_{rz} = diag(m_1, \ldots, m_p) \tag{2.18}$$

Where K_τ and K_c are design matrices and M_{rz} represents the rotor inertia along the z axis of joints in the system. Differentiating (2.17),

$$\frac{dL}{dt} = u^T \dot{x}_a + (\ddot{x}_a + \ddot{x}_c)^T K_\tau M_{rz} (\dot{x}_a + \dot{x}_c) + \dot{x}_c^T K_c x_c \tag{2.19}$$

Now, select the control input u to be a function of τ_{xi}, the controller state vector, the rotor inertia, and the controller design matrix K_τ such that

$$u = K_\tau \tau_x - K_\tau M_{rz} \ddot{x}_c + K_\tau V \dot{x}_c \tag{2.20}$$

$$V = diag(v_1, \dots, v_p) \tag{2.21}$$

$$\tau_x = diag(\tau_{x1}, \dots, \tau_{xp}) \tag{2.22}$$

where τ_{xi} is defined in Eq. (2.9). Substituting the equation for τ_x into Eq. (2.20) yields

$$u = -K_\tau M_{rz}(\ddot{x}_a + \ddot{x}_c) - K_\tau V(\dot{x}_a - \dot{x}_c) + K_\tau u \tag{2.23}$$

Let,

$$R_\tau = (I - K_\tau)^{-1} K_\tau \tag{2.24}$$

Substituting (2.24) into (2.23),

$$u = -R_\tau M_{rz}(\ddot{x}_a + \ddot{x}_c) - R_\tau V(\dot{x}_a - \dot{x}_c) \tag{2.25}$$

If $R_\tau V$ and $R_\tau M_{rz}$ are chosen symmetric, replacing u in the time derivative of the Lyaponov function Eq. (2.19) produces:

$$\begin{aligned}\frac{dL}{dt} &= -(\ddot{x}_a + \ddot{x}_c)^T R_\tau M_{rz} \dot{x}_a - (\dot{x}_a - \dot{x}_c)^T R_\tau V \dot{x}_a \\ &\quad + (\ddot{x}_a + \ddot{x}_c)^T R_\tau M_{rz}(\dot{x}_a + \dot{x}_c) + \dot{x}_c^T K_c x_c\end{aligned} \tag{2.26}$$

Canceling terms and reformatting,

$$\begin{aligned}\frac{dL}{dt} &= -\dot{x}_a^T R_\tau V \dot{x}_a + \dot{x}_c^T R_\tau V \dot{x}_a \\ &\quad + \dot{x}_c^T R_\tau M_{rz}(\ddot{x}_a + \ddot{x}_c) + \dot{x}_c^T K_c x_c\end{aligned} \tag{2.27}$$

or

$$\begin{aligned}\frac{dL}{dt} &= -\dot{x}_a^T R_\tau V \dot{x}_a + \dot{x}_c^T [R_\tau V \dot{x}_a \\ &\quad + R_\tau M_{rz}(\ddot{x}_a + \ddot{x}_c) + K_c x_c]\end{aligned} \tag{2.28}$$

Now, define the following equality

$$R_\tau V\dot{x}_a + R_\tau M_{rz}(\ddot{x}_a + \ddot{x}_c) + K_c x_c = -D_c \dot{x}_c \tag{2.29}$$

where D_c is a symmetric and positive-definite matrix. This equality produces a set of equations that calculate the controller state vector as a function of the sensed torque. Substituting Eq. (2.29) into Eq. (2.28), the time derivative of the Lyaponov function becomes:

$$\frac{dL}{dt} = -\dot{x}_a^T R_\tau V\dot{x}_a - \dot{x}_c^T D_c \dot{x}_c \tag{2.30}$$

This function results in a constantly decaying Lyaponov function if $R_\tau V$ and D_c are chosen to be positive definite and x_c is calculated using a reformatted version of Eq. (2.29),

$$R_\tau M_{rz}\ddot{x}_c + D_c\dot{x}_c + K_c x_c = -R_\tau(M_{rz}\ddot{x}_a + V\dot{x}_a) \tag{2.31}$$

Since only τ_x is known, substitute Eq. (2.9) and Eq. (2.22) into Eq. (2.31) and add $R_\tau u$ to both sides to replace dependence on measured joint acceleration and velocity with τ_x,

$$R_\tau M_{rz}\ddot{x}_c + D_c\dot{x}_c + K_c x_c + R_\tau u = R_\tau \tau_x \tag{2.32}$$

Using Eq. (2.32) and Eq. (2.20), the control input required to stabilize the system can be calculated as:

$$\begin{aligned} u = R_\tau \tau_x - R_\tau M_{rz}[(R_\tau M_{rz})^{-1}(-D_c\dot{x}_c - K_c x_c) \\ + M_{rz}^{-1}\tau_x)] + R_\tau V\dot{x}_c \end{aligned} \tag{2.33}$$

If R_τ and M_{rz} are invertible, as they should be since R_τ can be chosen positive definite and M_{rz} is the full rank diagonal joint rotor inertia matrix, Eq. (2.33) reduces to:

$$u = (R_\tau V + D_c)\dot{x}_c + K_c x_c \tag{2.34}$$

Although the torque sensor related term cancels out of u, the control input is not independent of τ_x because τ_x is used to form x_c and its derivative. With τ_x as the input, u' as the current torque command, and u as the output, the control signal required to stabilize the system can be calculated by the following set of linear equations.

$$\begin{bmatrix} \dot{x}_c \\ \ddot{x}_c \end{bmatrix} = \begin{bmatrix} 0 & I \\ -(R_\tau M_{rz})^{-1} K_c & -(R_\tau M_{rz})^{-1} D_c \end{bmatrix} \begin{bmatrix} x_c \\ \dot{x}_c \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ M_{rz}^{-1} & -M_{rz}^{-1} \end{bmatrix} \begin{bmatrix} \tau_x \\ u' \end{bmatrix} \tag{2.35}$$

$$u = \begin{bmatrix} K_c & R_\tau V + D_c \end{bmatrix} \begin{bmatrix} x_c \\ \dot{x}_c \end{bmatrix} \tag{2.36}$$

2.5 Stability robustness

The previous Lyaponov proof concentrated on a non-directly measured value, τ_{xi}, instead of the directly measured term, τ_{si}. It was also assumed that no modeling errors were present and that the nonlinear term, f_i, was calculable. If the joint in question does not have a position sensor and/or the controller did not have information from other joints, the nonlinear term is not calculable. If additive modeling errors are also present, the joint model becomes:

$$u_i = (m_i + \overline{m}_i)\ddot{\theta}_i + (v_i + \overline{v}_i)\dot{\theta}_i + \tau_{si} + f_i \tag{2.37}$$

Reformatting and solving for τ_{si} and treating f_i as an error,

$$\tau_{si} = \tau_{xi} - e_i \tag{2.38}$$

$$e_i = \overline{m}_i\ddot{\theta}_i + \overline{v}_i\dot{\theta}_i + f_i \tag{2.39}$$

From Eq. (2.39), it can be seen that if the kinetic energy in the joint declines, then the effects of the additive error terms are reduced. The magnitude of these errors should never be very high because the m_i term, which represents the rotor inertia along the z axis, should be known precisely from the motor manufacturer and v_i, the viscous friction term, while not easily modeled precisely can be closely approximated by a simple linear model. Since higher frictional forces enhance the dissipation of energy, stability will not be effected if the modeled friction is less than the actual friction. Due to the low relative magnitude of these errors, they can be considered disturbances and do not effect the overall stability of the system.

For the i^{th} joint, it is evident that the additive errors will decline with the kinetic energy of the system, but it is not evident that the f_i term shares this property. The f_i term represents the dynamic effects of moving the joint through space. Each term of f_i is multiplied explicitly by the angular acceleration or the angular velocity of the i^{th} or $(i-1)^{th}$ link. If the base frame's angular acceleration and velocity are zero, Eq. (2.4) and Eq. (2.5) can be rewritten as:

$$\omega_i = \sum_{n=1}^{i} z_{n-1}\dot{\theta}_n \tag{2.40}$$

$$\dot{\omega}_i = \sum_{n=1}^{i} z_{n-1}\ddot{\theta}_n + \left(\sum_{n=1}^{i} z_{n-1}\dot{\theta}_n\right) \times (z_{i-1}\dot{\theta}_i) \tag{2.41}$$

The z axes in Eq. (2.40) and Eq. (2.41) are in base frame coordinates. The angular acceleration and velocity will decline as joint kinetic energy is dissipated because all terms in angular acceleration and velocity equations are multiplied by joint acceleration or velocity. Since all terms of f_i are multiplied by the declining angular terms, f_i will also decline with kinetic energy. An argument must still be made that the controller will remain passive during the disturbance caused by the nonlinear dynamic effect.

With the non-moving base frame assumption, the first joint has no nonlinear term because the first joint is not moving through space. Utilizing the previous discussion on the robustness to parameter error, the first joint will remain dissipative. As the first joint dissipates its kinetic energy, the components of the nonlinear terms in later joints in the kinematic chain are lowered. The nonlinear term of the second joint in the chain only depends on the motion of the first and second joints. If the motion of the first joint tends towards zero, only an increase in the velocity or acceleration of the second joint can cause the nonlinear term to increase in magnitude. In view of Eq. (2.6), the equation for f_i, the terms involving the velocity and acceleration of the second joint are in squared relationships and are multiplied by the full rotor inertia term, M_{ri}. Due to the size of practical rotor inertias and link velocities, these numbers can be considered to be of the same order of magnitude as the parameter errors and treated as a disturbance. Therefore, the second joint will remain dissipative. As the kinetic energy of the first and second joints decreases, the nonlinear effect on the third joint decreases. This argument can be continued for n joints. Note that the f_i term does not necessarily need to be treated as an unknown disturbance if the necessary sensing required to calculate it is available.

The previous arguments contend that the e_i term is small for a real system. Since e_i is based on physical parameters, it is bounded. The control system parameters can be tuned to deal with this bounded disturbance and maintain system stability. In addition, if stability is maintained, due to the dissipative nature of the controller, the e_i term will be reduced as the kinetic energy dissipates. With this argument, the e_i term can be ignored in the τ_{si} model. With these changes, τ_{si} can be modified to be approximately τ_{xi} and used to calculate the stabilizing control input.

The stability of the system is not only effected by disturbances that decay with kinetic energy. Sensor noise and unknown torque bias between the torque input and torque sensor output can be problematic. These errors will cause the control system to believe that the motor is rotating at a different velocity and acceleration. System noise will cause movement about the zero kinetic energy point as the signal to noise ratio decreases. Sensor bias will lead to the inability of the system to reach the zero kinetic energy point. In both cases, assuming reasonable error levels, as the system kinetic energy increases due to controller error, the signal due to the real motor parameters increases. The higher signal to error ratio then allows the controller to move back towards zero energy. Depending on the controller gains, the magnitude of the error, and the duration of the error, this reaction can lead to unstable behavior. Unfortunately, due to the low signal magnitude for real motors and the reduction of signal magnitude as energy decreases, the small, non-dissipating errors can adversely effect performance.

2.6 Extension to Geared Joints

The extension of the stability theory to a geared joint is reasonably straightforward. The motor model now considers a gear ratio, n_i, and a geartrain viscous friction term, v_i^G,

$$\tau_i = m_i\ddot{\theta}_{ig} + \frac{\tau_{si}}{n_i} + v_i\dot{\theta}_{ig} + \frac{f_i}{n_i} + \frac{v_i^G\dot{\theta}_{ig}}{n_i^2} \tag{2.42}$$

Note that nonlinear geartrain properties such as backlash and hysteresis will be discussed later in this section. The θ_{ig} terms in the geared joint model refer to the motor side of the joint, not the link side. Because the sensed torque is physically on the link side of the geartrain, its value are divided by the gear ratio. The nonlinear term is also divided by the gear ratio because its value is composed entirely on joint, not motor, output terms. Due to being on the reduced speed and torque side of the geartrain, the geartrain friction term is divided by the square of the gear ratio. The τ_{si} term can be recalculated for the geared joint as:

$$\tau_{si} = -n_i m_i\ddot{\theta}_{ig} - (n_i v_i + \frac{v_i^G}{n_i})\dot{\theta}_{ig} - f_i + n_i u_i \tag{2.43}$$

Changing to link coordinates, $\theta_{ig}=n_i\theta_i$.

$$\tau_{si} = -n_i^2 m_i\ddot{\theta}_i - (n_i^2 v_i + v_i^G)\dot{\theta}_i - f_i + n_i u_i \tag{2.44}$$

As before, a τ_{xi} term can be introduced in the τ_{si} equation.

$$\tau_{si} = n_i^2 \tau_{xi} - f_i \tag{2.45}$$

$$\tau_{xi} = -m_i \ddot{\theta}_i - v_i^{eq} \dot{\theta}_i + \frac{u_i}{n_i} \tag{2.46}$$

$$v_i^{eq} = v_i + \frac{v_i^G}{n_i^2} \tag{2.47}$$

By simple substitution of v_i^{eq} for v_i and multiplying the input torque by a constant in the Lyaponov stability proof in section 2.4, it can easily be seen that the proof holds for geared joints.

The robustness arguments of section 2.5 can be extended to the geared joint. The magnitude of the additive rotor parameter errors are multiplied by the same squared gear ratio constant as the nominal rotor parameters. The relative magnitude of the nonlinear term has been reduced by the gear ratio. Since the relative magnitudes of these disturbances with respect to the dominant terms have remained unchanged or been reduced, the robustness arguments from section 2.5 still hold. Likewise, the same friction approximation arguments made for v_i in the direct drive joint case also apply to the v_i^{eq} term.

Some nonlinear geartrain effects will be seen by the torque sensor. Geartrain backlash will show up as a discontinuity in the torque sensor output about zero torque. Its effect on system stability will depend on the duration of the zero torque output while the geartrain unloads and loads, and the magnitude and its rate increase in output torque when the geartrain loads.

Geartrain hysteresis will not be seen by the torque sensor but will effect the joint model. The hysteresis effect implies that more effort is required to drive the geartrain in one direction than in the other. This effect manifests itself in the inability of the joint to be driven back to its initial starting position using the inverse of the control input required to move it away from the starting position. This effect can be considered a bias on the geartrain viscous friction term, v_i^G, that varies with the direction of joint motion. In the geared system, this offset is proportionally reduced by the square of the gear ratio. As a result, although an unmodeled, variable friction term is added by the hysteresis, that unknown friction torque is disadvantaged by the geartrain. In addition, if the hysteresis friction increases total friction it will add the dissipative nature of the system, thus enhancing stability.

2.7 Simulation

The controller described in section 2.4 was simulated to test the stability robustness of the system. A planar, 3 degree-of-freedom robot arm, shown in Fig. 2.2, was used as the plant. Table 2.1 details the parameters of this robot arm. The quantities *α, a, d,* and *θ* are Denavit-Hartenberg robot parameters[58] which describe the robots kinematic configuration. The motor parameters were taken from real brushless DC motors.

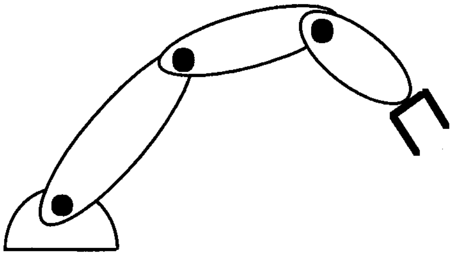


Figure 2.2: Planar 3-DOF robot

Table 2.1: Robot physical parameters

Parameter	**Joint 1**	**Joint 2**	**Joint 3**
α	0	0	0
a	1.0m	0.8m	0.4m
d	0	0	0
θ	θ_1	θ_2	θ_3
link mass	19.4kg	15.1kg	8.0kg
motor mass	5.6kg	4.9kg	4.0kg
link inertia (I_z)	1.6167N-m-s^2	0.8053N-m-s^2	0.1067N-m-s^2
motor inertia (M_{rz})	0.00128N-m-s^2	0.00112N-m-s^2	0.00091N-m-s^2
motor viscous friction (V)	0.00024N-m-s	0.00019N-m-s	0.00016N-m-s
Maximum motor torque	38.3N-m	32.4N-m	26.5N-m

The robot link dynamics were simulated using a set of dynamic equations generated by Robotica[60], a Mathematica based robot kinematics and dynamics generation package. Those equations were converted to a Matlab script and simulated using Matlab and Simulink. The joint simulation was independent from the link simulation. The joint simulation accounted for the effects of the motor and geartrain producing the output torque for the torque sensor and the link simulation. The controller was implemented as a set of state space equations detailed in Eqs. (2.35) and (2.36).

At the start of the simulation, the robot joints were positioned at [-0.8 -0.5 1.3] radians and were not moving. This was the initial starting position for all zero gravity simulations. To create an initial velocity and acceleration condition for the controller, the robot was given a constant joint torque input of [17 5 0.8] N-m for 0.2 sec. During this 0.2 sec period, the controller inputs were connected to the system but its output was not. At 0.2 sec, the constant joint torque was turned off and the controller output connected to the robot.

2.7.1 Simulation 1

This simulation utilized perfect knowledge of M_{rz} and V. The nonlinear term, f_i, was not considered in the plant or controller for this simulation. The simulation was in zero gravity. Figures 2.3-2.5 show the time histories of the joint velocities, joint accelerations, and the motor inertia and friction torques. The controller quickly dissipated the joint kinetic energy. Figure 2.5 shows the difference between the commanded and sensed torque values. This difference results from the transmission loss caused by the inertia and friction of the motor. This loss is what is used by the controller to determine the energy in the motor. The small size of this loss accounts for the large K_τ gain required to quickly dissipate the energy.

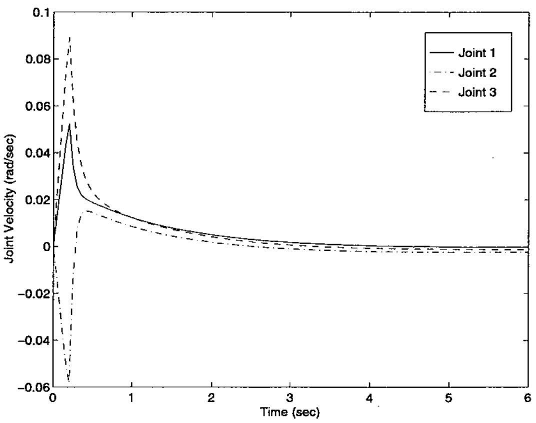


Figure 2.3: Joint velocities: exact joint model

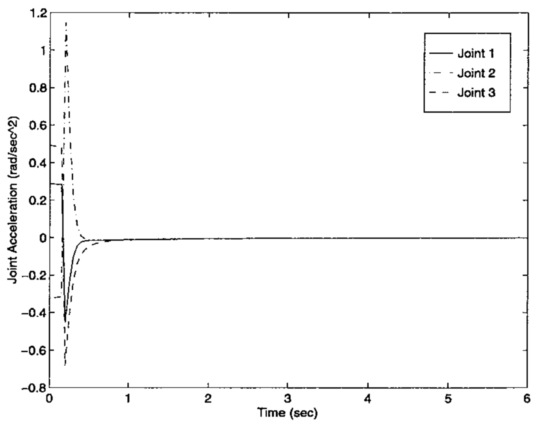


Figure 2.4: Joint accelerations: exact joint model

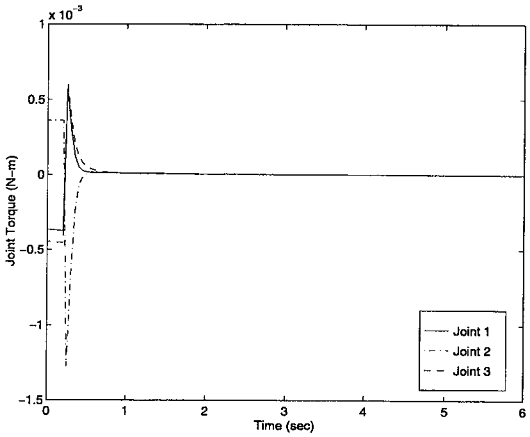


Figure 2.5: Motor inertial and friction torques

2.7.2 Simulation 2

This simulation considers controlling a system that is not exactly known. Figures 2.6-2.8 show the effect of having 10%, 20%, and 40% errors in the parameters of the motor model. The error on the second joint was added in the direction opposite to the first and third joint's errors. The percentage error used was the same for both the motor inertia and the motor friction. The nonlinear term was ignored in both the controller and plant. Although the parameter errors do effect the performance of the controller to a small extent, the system still remains stable. The small change in the system response is interesting. This small change implies that the controller has a high parameter error rejection capability. The error in commanded torque caused by the 20% parameter error can be seen in Fig. 2.9.

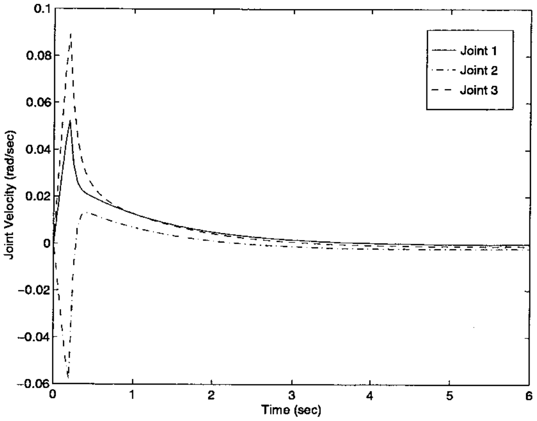


Figure 2.6: Joint velocities: 10% parameter errors

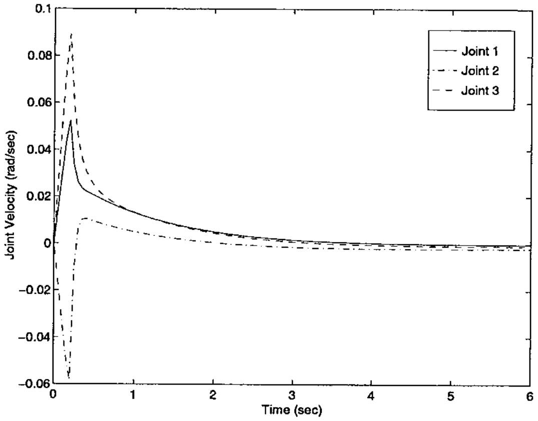


Figure 2.7: Joint velocities: 20% parameter errors

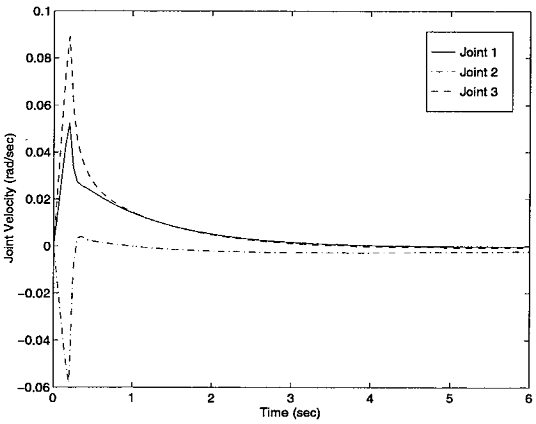


Figure 2.8: Joint velocities: 40% parameter errors

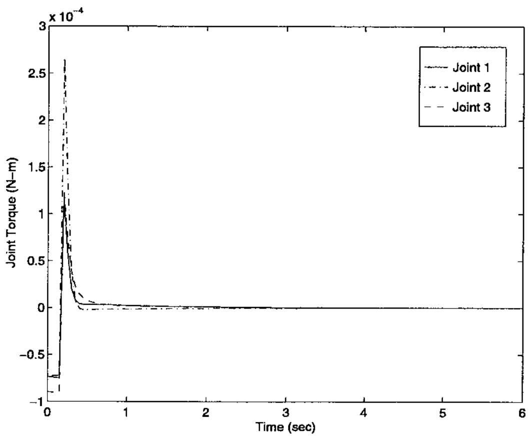


Figure 2.9: Torque error caused by 20% parameter errors

2.7.3 Simulation 3

This simulation considers the nonlinear term, f_i, in the plant model. The controller model, as it will with all simulations, does not include the nonlinear term. For a planar manipulator, this nonlinear term is a function of only the rotor inertia and the joint accelerations of the i-1 preceding joints in the kinematic chain. The nonlinear velocity terms drop out due to the parallel z axes of the link frames. An exact model is assumed in Fig. 2.10 and a model with 20% parameter errors is considered on Fig. 2.11. The inclusion of the nonlinear term in the plant does slow the convergence rate of the controller to the zero kinetic energy state. The torque error caused by ignoring the nonlinear term can be found in Fig. 2.12. The performance of the controller ignoring the nonlinear term shows the stability robustness of the system with regards to this term. The magnitude of the nonlinear term, even in the simple planar model considered, is significant. If the robot has a fixed base, as it did in this simulation, then the first joint has no nonlinear terms.

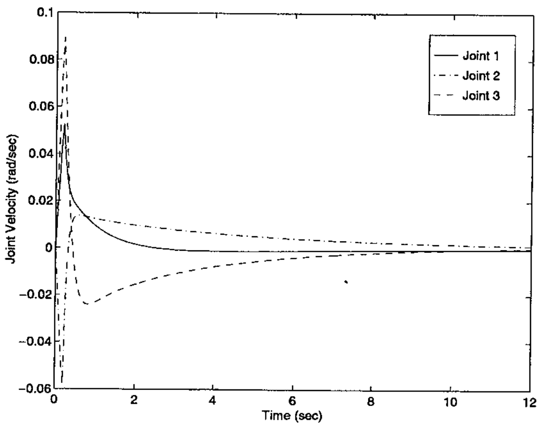


Figure 2.10: Joint velocities: Nonlinear term included in simulation with exact model

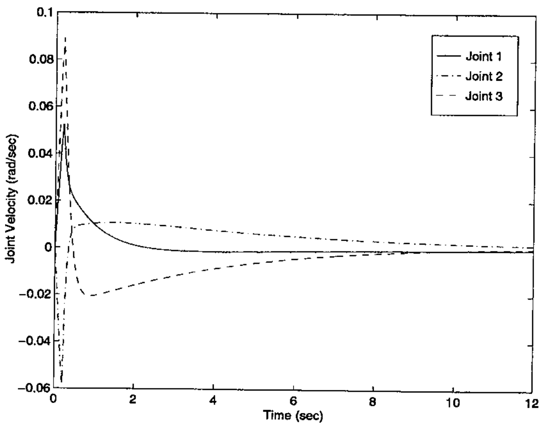


Figure 2.11: Joint velocities: Nonlinear term included with 20% parameter errors

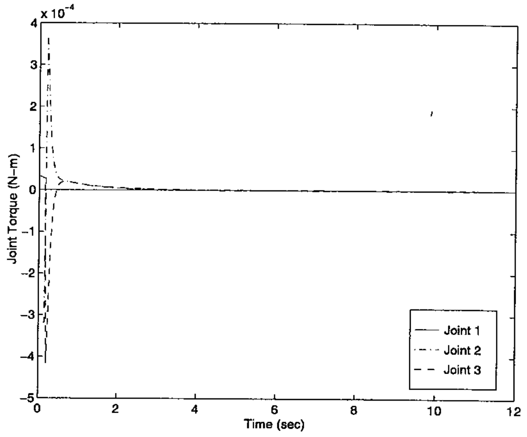


Figure 2.12: Torque error produced by nonlinear term

2.7.4 Simulation 4

This simulation considers a realistic situation of having parameter errors, the nonlinear term, and sensor noise. Figure 2.13 shows the effect of noise on the system. The response of a noise-free system of the same type can be seen in Fig. 2.7. The noise does not decay as kinetic energy decays. As a result, at low energy state, the noise dominates. As the system gains energy due to controller error caused by the noise, the magnitude of the inertial and friction related terms increase and the controller moves the system back towards zero energy. Due to the small size of the dynamic effects that are being considered, the noise levels that produce the signal to noise ratio in Fig. 2.14 are small. The signal to noise ratio and the magnitude of the additive noise can be found in Figs. 2.15 and 2.16, respectively. The signal to noise ratio, by averaging approximately 0db after an initial peak, implies that the magnitude of the signal is very close to the magnitude of the noise in this simulation. Torque sensor data will need to be heavily filtered to give acceptable response at low joint energy levels. To give the results shown in the simulations, the controller gains were high. Lower controller gains should lead to better noise rejection.

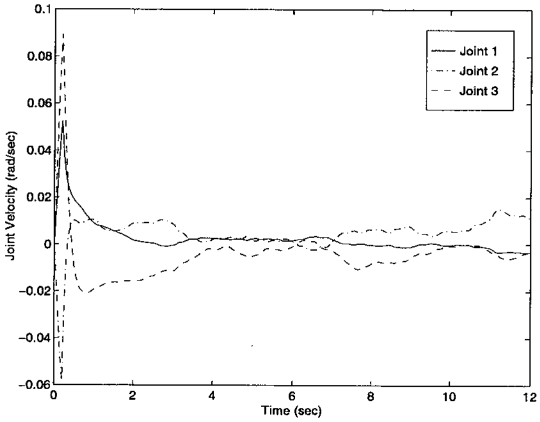


Figure 2.13: Joint velocities: System with noise, 20% parameter errors, and nonlinear term

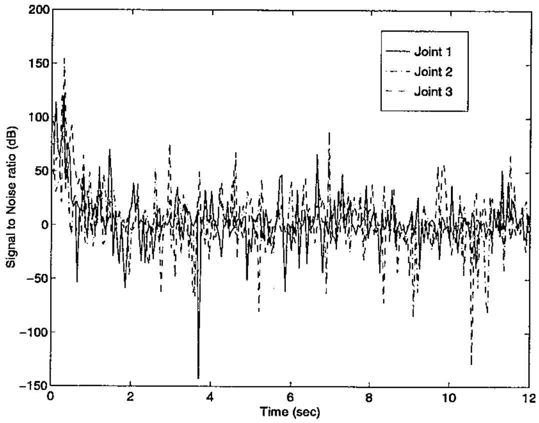


Figure 2.14: Signal to noise ratio

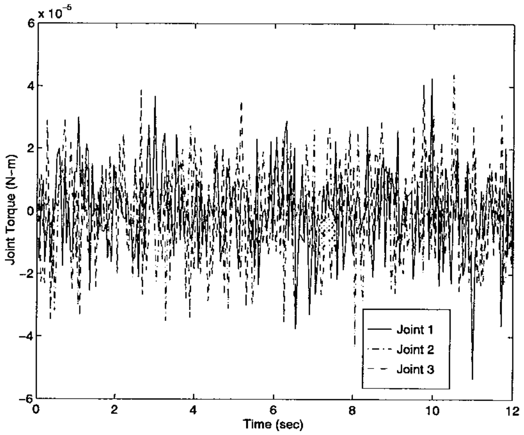


Figure 2.15: Magnitude of noise added to sensed torque

2.7.5 Simulation 5

This simulation looks at the effect of gravity on the controller response. A model with parameter errors and the nonlinear term but without noise is considered. To introduce another nonlinear effect, the motor torque limits, although increased, are not raised enough to keep the motors from saturating. Figures 2.16-2.18 show the controller response to different gravity loadings induced by different starting positions. The same initial starting torque was used in the gravity simulations as was used in the non-gravity simulations. The torques were not designed to compensate for gravity. As a result, the system accelerations are much larger than in the zero gravity case at the end of the 0.2sec startup period. The torque saturation of the joint motors during the simulation shown in Fig. 2.18 can be seen in Fig. 2.19. Although not shown, all three initial positions resulted in some joint motor saturation. To show how the response would be improved if the motors did not remain saturated much of the time, the simulation with initial position [0.8 0.0 0.5] was recalculated with a torque limit four times higher than the previous simulation. The results are shown in Figs. 2.20 and 2.21.

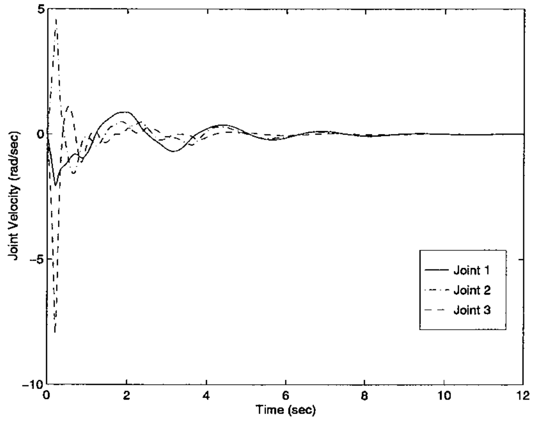


Figure 2.16: Joint velocities - gravity loaded with initial joint position [-0.8 -0.5 1.3]

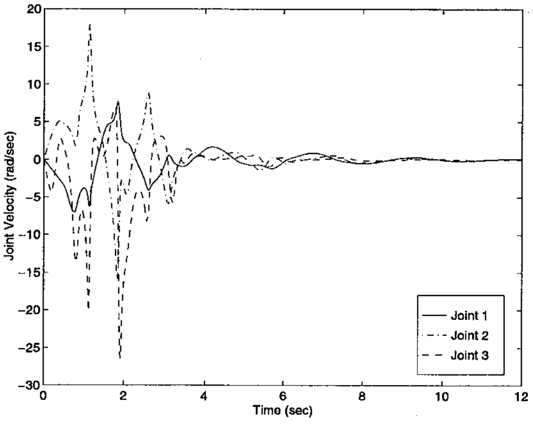


Figure 2.17: Joint velocities: Gravity loaded with initial position [1.2 0.5 -1.3]

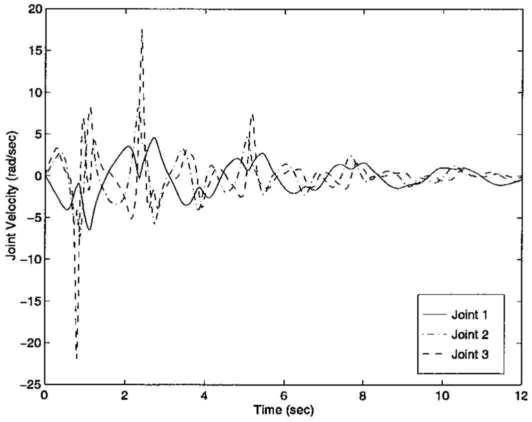


Figure 2.18: Joint velocities - Gravity loading with initial position [0.8 0.0 0.5]

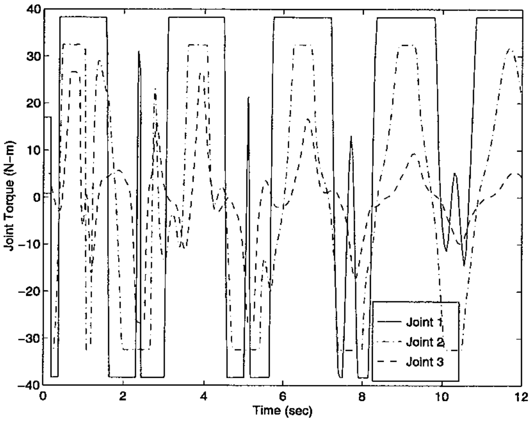


Figure 2.19: Joint torques: Gravity loaded with initial position [0.8 0.0 0.5]

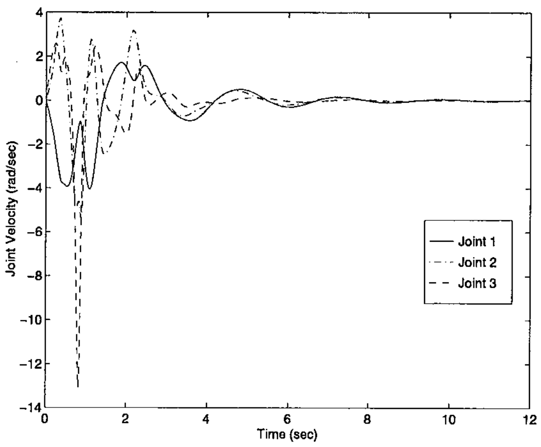


Figure 2.20: Joint velocities: Gravity loaded with initial position [0.8 0.0 0.5] and 4x torque limits

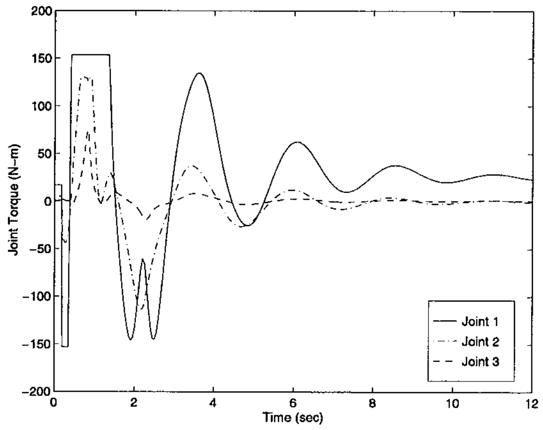


Figure 2.21: Joint Torques: Gravity loaded with initial position [0.8 0.0 0.5] and 4x torque limits

2.7.6 Simulation 6

This simulation details the effect of adding a 100:1 gear ratio into the system. To maintain some equity with the direct drive results presented earlier, the motors in the geared simulations have been reduced to keep the maximum torque output the same as in the direct drive simulations. For realism, these smaller motors also have smaller inertial and friction parameters as shown in Table 2.2. The mass of the smaller motor and the geartrain are assumed to be equal to the mass of the direct drive motor. The gains were changed to reflect the lower motor torque output required by the geared system, the different motor inertial and viscous terms due to the change in motor parameters, and the motor speed increase caused by the gearing. Some of the previous simulations done for a robot with direct drive joints were repeated for a gear drive robot with an ideal geartrain (no geartrain viscous friction, hysteresis, or backlash).

The case using exact model and no nonlinear term is shown in Figs. 2.22 and 2.23. The nonlinear term is then included and some results are shown in Figs. 2.24 and 2.25. The effect of the nonlinear term on the system is less significant in a geared system because the nonlinear term is based entirely on link velocities and accelerations. Errors in the linear joint parameters will also exist in the geared model (see Fig. 2.26). Although these errors will be magnified by the higher motor speeds and accelerations, they have the same relative magnitudes to the nominal parameters as in the direct drive system. The numbers here will be slightly different than in the direct drive simulations because the geared motor parameters are not exactly 1/100th the parameters of the direct drive motors. A realistic system will also have sensor noise. As shown in Figs. 2.27-2.29, the geared simulation will be better able to handle sensor noise in that the noise occurs on the link side of the system and is thus disadvantaged by the gear ratio. Given identical motor parameters, this advantage can be quite dramatic. In this simulation, the geared simulation is able to handle a factor of 10 times higher average noise power and give a similar response to the direct drive simulation even with the smaller motor inertial and friction terms in the geared joint. Gravitational effects can also be handled by the controller as shown in Figs. 2.30 and 2.31. As in the direct drive simulation, the system accelerations are higher due to the initial condition torques inability to overcome gravity. The same gains were used for the gravity simulation as were used for the zero gravity simulation. The motors saturated at the torque limits in both simulations. The torque limits were raised in the second plot to reduce the time in saturation.

Table 2.2: Gear drive motor parameters

	Joint 1	**Joint 2**	**Joint 3**
Motor inertia (M_{rz})	1.6x10^{-7}N-m-s^{2}	1.9x10^{-7}N-m-s^{2}	2.7x10^{-7}N-m-s^{2}
motor viscous friction (V)	2.0x10^{-7}N-m-s	8.8x10^{-8}N-m-s	4.0x10^{-7}N-m-s

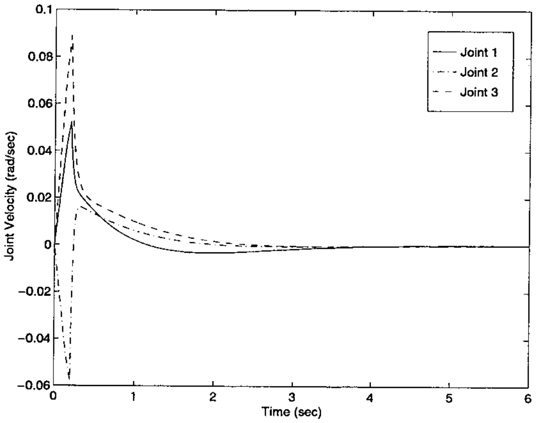


Figure 2.22: Joint velocities: Geared model with exact motor parameters

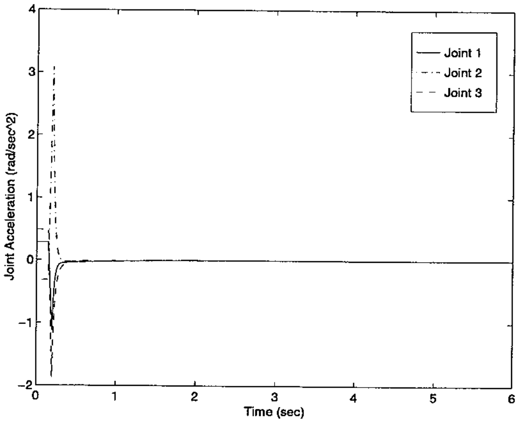


Figure 2.23: Joint accelerations: Geared model with exact motor parameters

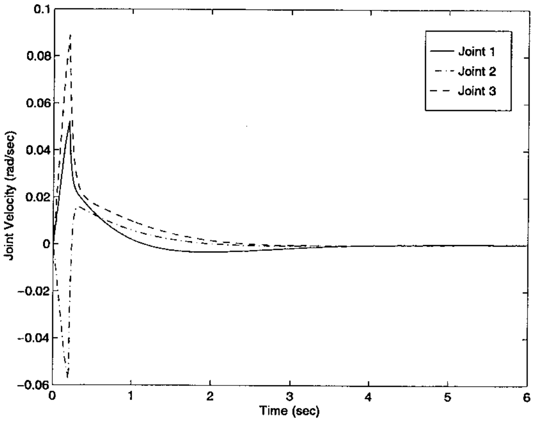


Figure 2.24: Joint velocities: Geared model with nonlinear term

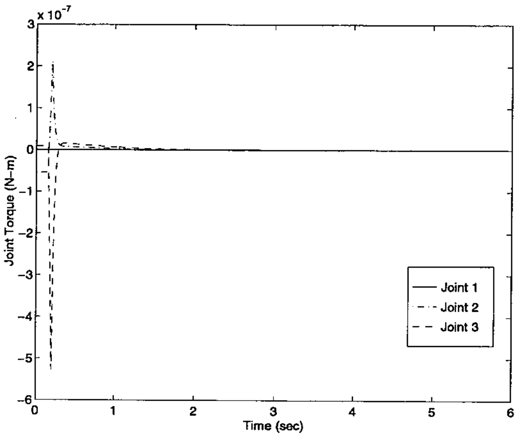


Figure 2.25: Joint torques resulting from nonlinear terms: Geared model

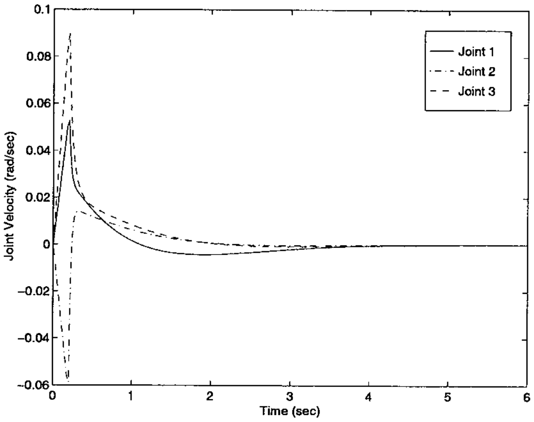


Figure 2.26: Joint velocities: Geared model with nonlinear term and 20% parameter error

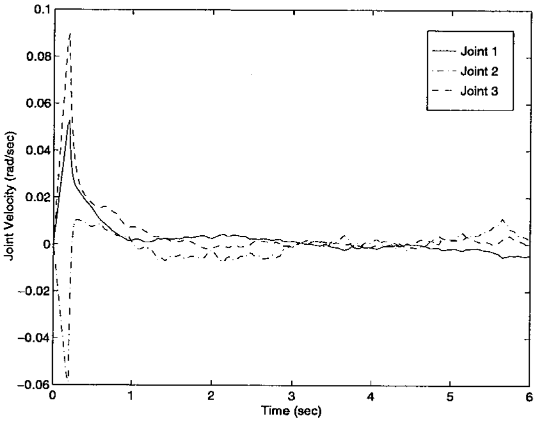


Figure 2.27: Joint velocities: Geared simulation with nonlinear term, 20% parameter error, and noise

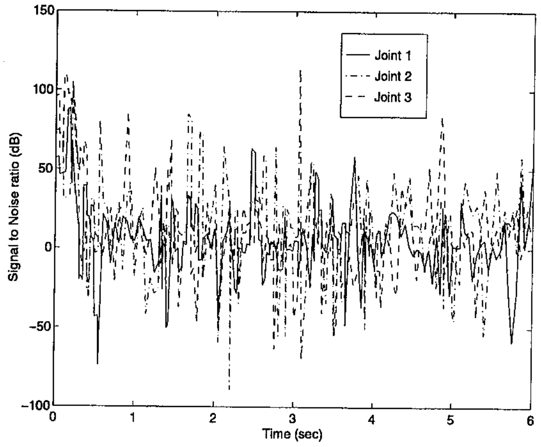


Figure 2.28: Signal to noise ratio

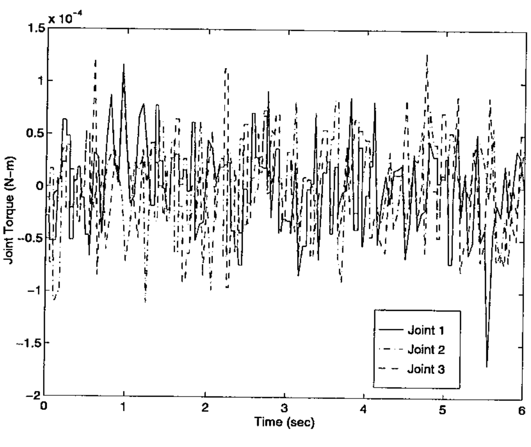


Figure 2.29: System noise

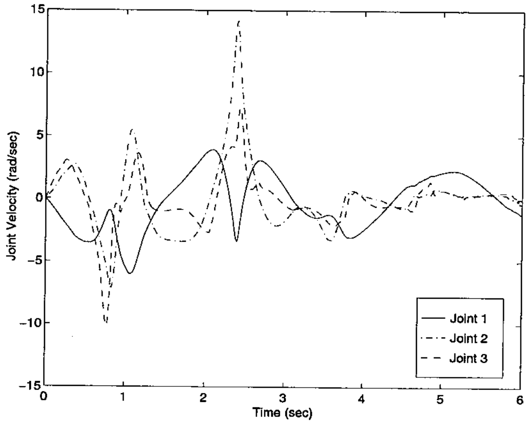


Figure 2.30: Joint velocities: Gravity loaded with nonlinear term and 20% parameter error

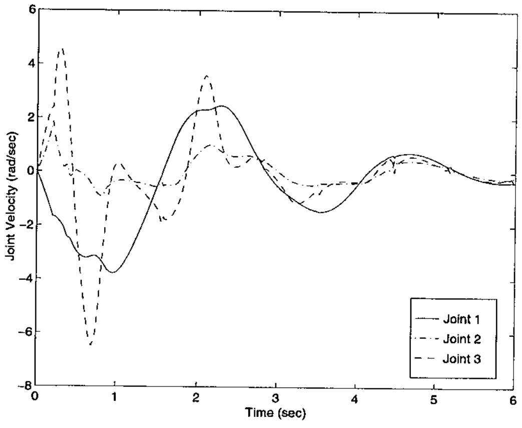


Figure 2.31: Joint velocities: Gravity loaded with nonlinear term, 20% parameter error, and 4x joint torque limits

2.7.7 Simulation 7

The previous simulation dealt with an ideal geartrain. Now, real geartrain effects will be considered. First, geartrain viscous friction will be considered. The geartrain viscous term is selected to be [0.005 0.004 0.002] N-m-sec. The simulation was done without gravity. Adding a geartrain friction term changes the equivalent friction between the motor input and the torque sensor. As a result, the controller gains will be modified for the new system.

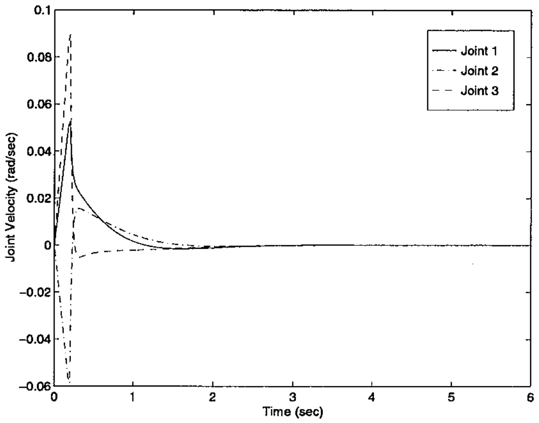


Figure 2.32: Joint velocities: Geartrain viscous friction with nonlinear term, 20% parameter error on all parameters

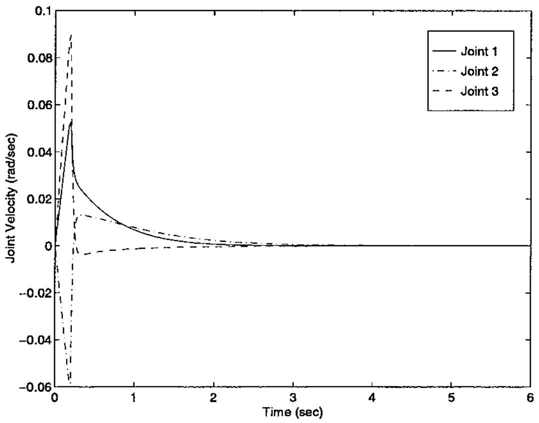


Figure 2.33: Joint velocities: Geartrain with hysteresis, non-linear term, and 20% error on all parameters

Figure 2.32 shows better performance with the geartrain friction than Fig. 2.26 did with ideal geartrain. This simulation accomplished this performance with lower controller gains. These lower gains imply less noise sensitivity and lower power consumption. The geartrain friction enhances the stability of the system by causing energy to dissipate quicker. Another geartrain effect is hysteresis. Hysteresis manifests itself in making the geartrain harder to turn in one direction than in the other. This implies that the geartrain viscous friction is different in different directions. To show the hysteresis effect (see Fig. 2.33), the usage of a constant viscous friction coefficient was replaced by a constant that is half as high as in the negative direction than in the positive direction. The controller model still assumes a constant viscous friction term in both directions. This reduction decreases overall friction, which should result in slightly worse performance.

Hysteresis and viscous friction terms are well behaved terms that decay with the velocity of the system. Backlash is a hard discontinuity in the torque sensor output. This type of nonlinearity can cause problems for most controllers. The stability of the passive dynamic controller detailed here is not guaranteed during backlash.

The model of backlash used in the following simulation was simplistic. The simulation waits for a change of direction in the motor input torque. When the direction change occurs, the torque output is set to zero. The motor is then considered to have a different velocity than the link, but initially its velocity is the link's velocity multiplied by the gear ratio. The motor's velocity will be independent of the link until it moves enough to contact the geartrain again. The gap between geartrain contacts was chosen to be approximately one degree. The motors movement during the time when it is not in contact with the geartrain is represented by Eq. (2.48). When the motor contacts the geartrain, the output torque returns instantaneously to the normal, geared motor torque for the non-backlash system.

$$\ddot{\theta} = M_{rz}^{-1}\tau \tag{2.48}$$

Due to the discontinuous torque outputs, the controller gains were changed for this simulation. The damping gain, D_c, reacted badly to what it perceived as a large velocity change at the discontinuity. As a result, the D_c gain was reduced. The K_τ gains were lowered to prevent large control signal changes at the discontinuity.

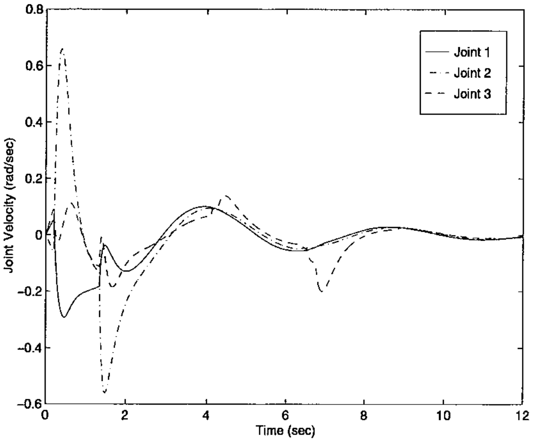


Figure 2.34: Joint velocities: Geared joints with backlash

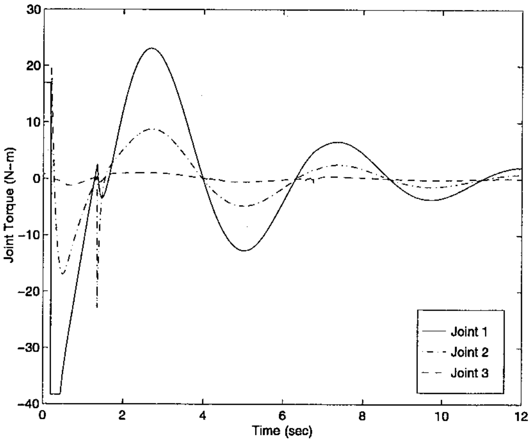


Figure 2.35: Joint torques: Geared joints with backlash

2.7.8 Simulation 8

This simulation deals with a torque bias. The bias in this situation is defined as a constant error in the torque sensor output or in the input command. Either of these situations effects the magnitude of the difference between the torque input and the sensor output which is the interesting quantity for the controller. In some of the previous simulations, errors due to bad joint parameters or ignoring nonlinear terms were considered. These quantities decreased as system acceleration and velocity decreased. The simulations that included noise did have stability problems as the signal to noise ratio decreased. The sensor bias errors can also induce instability as the signal to error ratio decreases. Figures 2.36 and 2.37 show direct drive simulations with constant bias terms of different magnitudes. The differing magnitudes can lead to a steady state error or instability. Gear driven joints will have similar problems. The main difference is that, considering biases of equal magnitudes, the system will be more sensitive to a joint input bias than to a torque sensor bias because the magnitude of the torque sensor bias will be reduced by the gear ratio.

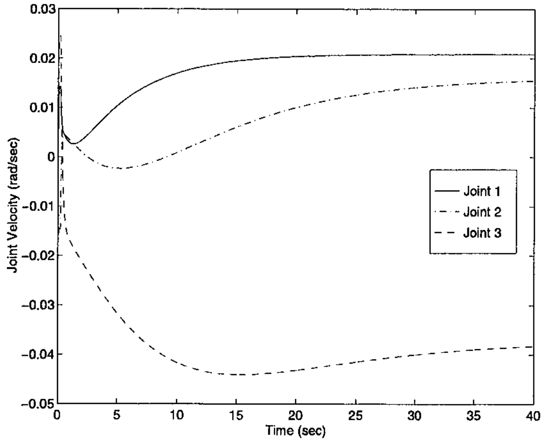


Figure 2.36: Joint velocities: Direct drive joint with torque bias of [5.0 3.0 -6.0]x 10^{-6}N-m

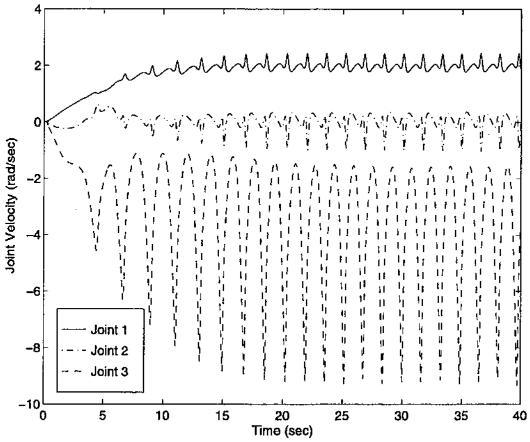


Figure 2.37: Joint velocities: Direct drive joint with torque bias of [5.0 3.0 -6.0]x 10^{-4}N-m

2.8 Summary

A control technique to stabilize a robot system using joint torque feedback and a joint model was presented. A Lyaponov stability proof for this method utilizing the passivity of a dynamic system was presented and the stability robustness of the technique discussed and shown in simulation. Given good joint torque sensor data, the system provides good stabilization performance with large parameter errors and treating nonlinear dynamic effects as a disturbance. If sensor data contains bias errors, noise errors, or backlash, then, depending on the magnitude of the signal to error ratio, the system can become unstable.

Chapter 3 will describe a series of experiments used to confirm the theoretical properties of the virtual passive torque controller. This chapter has concentrated on stability robustness to modeling errors, treating the nonlinear term as a disturbance, and the inclusion of geartrain effects using a simulated 3 degree of freedom planar robot with a simple dynamic model. The goal of these simulations was to prove the concept of the virtual passive controller and indicate possible problems that would be encountered in a real system. In the next chapter, three wrist joints of a 7 degree of freedom revolute manipulator will be controlled using the virtual passive controller. The usage of the virtual passive controller with a position controller for position tracking along with the practical implementation of the control system will be discussed. The goal of the experiments in chapter 3 is not to replicate the simulations in this chapter. The experimental goal is to show stability robustness properties similar to those simulated and explore real-world stability and performance issues.

3. Experimental Verification of Virtual Passive Torque Controller

The purpose of this chapter is to describe the experiments used to determine the performance characteristics of the virtual passive joint torque controller described in chapter 2 on a real system. The experiment hopes to show that, on a system with imprecisely known parameters, the virtual passive torque controller will:

1) In the absence of a joint command, drive the joint velocity to zero stopping any existing joint motion and resist changes from the zero velocity state
2) Improve the position tracking of the position controller by sensing gravitational and dynamic loading
3) Be robust to the nonlinear term it treats as a disturbance
4) Be stable over multiple trajectories with a single controller gain set

Specific tests will be done to show the first three controller characteristics. Any stability problems encountered during the three tests will be discussed.

3.1 Experimental Apparatus

The robot used in the experiment was a Robotics Research 807i manipulator with a Robotics Research type II controller. (The Robotics Research type II controller will be hereafter referred to as the RRC controller). The robot was located at the NASA Langley Research Center in Hampton, VA. The 807i is a 0.8m long, 7 degree of freedom manipulator with an approximately 20lb tip payload capacity. The Devanit-Hartenberg parameters defining the robot may be found in Table 3.1 and a photograph of the robot in Fig. 3.1.

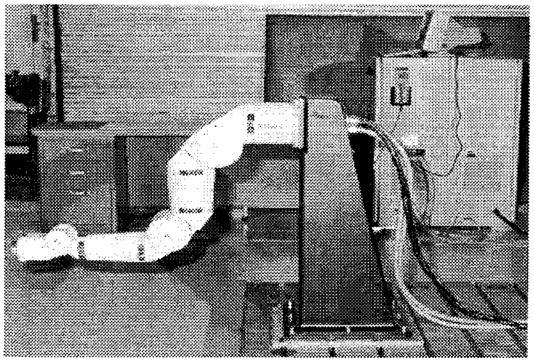


Figure 3.1: Robotics Research 807i

At the beginning of the experiment preparation, the RRC controller was interfaced to a VME card cage via a BIT3 Multibus to VME adapter. This adapter allowed devices in the VME card cage to address a portion of RRC controller's system memory as if it were VME memory. The RRC controller has a memory mapped architecture allowing joint position, velocity, torque, and current to be controlled by writing to memory locations. Likewise, sensed joint position, velocity, and torque can be read from the RRC controller's memory.

Table 3.1: Devanit-Hartenberg Parameters

Joint i	Name	α_{i-1}	a_{i-1}	d_i	θ_i
1	Shoulder roll	0	0	0	θ_1
2	Shoulder pitch	-π/2rad	-85.725mm	0	θ_2
3	Elbow roll	π/2rad	85.725mm	363.22mm	θ_3
4	Elbow pitch	-π/2rad	-71.12mm	0	θ_4
5	Wrist roll	π/2rad	71.12mm	363.22mm	θ_5
6	Wrist pitch	-π/2rad	-49.2mm	0	θ_6
7	Toolplate roll	π/2rad	49.2mm	0	θ_7

In addition to the Multibus to VME adapter, the VME card cage contained a 33Mhz 68040 single board processor, a 25Mhz 68040 single board processor, and assorted memory mapped input/output devices including digital to analog and analog to digital converters. For this experiment only the 33Mhz 68040 and the bus interface were used.

The operating environment chosen for the 68040 board was the VxWorks real-time operating system by Wind River Systems with the ControlShell and Stethoscope products by Real Time Innovations. The VxWorks operating system gave the 68040 the flexibility of an operating system while allowing predictable, real-time synchronization. The ControlShell product provided a graphical programming interface, similar to the Simulink product by Mathworks, which allowed for quick interfacing and reuse of C++ program modules. The ControlShell product also provided a menu system for operation of the system at runtime. The Stethoscope product provided real-time data displays for debugging. The programming environment was hosted on a Sun SparcStation. The 68040 received its programs from the Sun and returned data to the Sun via an ethernet connection.

3.2 Experiment Design

For this experiment, the three wrist joints (joints 5,6, and 7) were chosen to be controlled by the passive controller. The arm joints (joints 1-4) would be controlled by the position controller built into the RRC controller. This decision was made to reduce the time required for the experiment and to limit possible damage to the robot while debugging of the controller. The wrist can be given interesting velocities and accelerations by the arm joints and can be easily subjected to different loading conditions by changing end-effector weights. Using three consecutive joints should disclose problems with controller interactions among joints. It was felt that running the entire arm under passive torque control would not yield significantly more data than the wrist alone. However, in chapters 5 and 7 the robot arm joints will be controlled with the virtual passive controller but will not be analyzed in this depth.

Unfortunately, no detailed joint parameters were available. Without the ability to mount the individual joints in a proper test facility, accurate system identification of small joint parameters through the gearbox would be difficult. It was decided to use rough joint parameters gained from experimental experience. The use of these rough parameters, if successful, should show the robustness of the controller design to parameter error.

In the RRC controller used, there was not a one to one relationship between commanded torque units and sensed torque units. The RRC controller incorporates a torque controller which attempts to overcome gearbox effects. When in torque command mode, this controller cannot be bypassed or altered. Changing to a current command mode will bypass the torque controller, but a current input to torque output map was difficult to achieve due to various gearbox and temperature effects in the system. As a result, it was decided to use the torque command mode with a constant multiplier to convert torque input to torque output. The initial value of this constant was derived by comparing steady state torque commanded and sensed data. As will be discussed later, this parameter was changed to improve system performance. While keeping the RRC controller's torque controllers in the control flow does make the robot act more like a robot with direct drive joints, the joint torque offsets acted upon by the RRC controller's torque controllers produced some problems which had to be overcome by the passive joint controller.

A block diagram of the implemented torque control system can be found in Fig. 3.2. The position controller is a simple PD controller acting on actual and desired position inputs. The output of the position controller is a torque used to bias the sensed torque. The passive controller, perceiving the position controller induced bias as joint movement, attempts to compensate by moving the joint in the direction desired by the position controller.

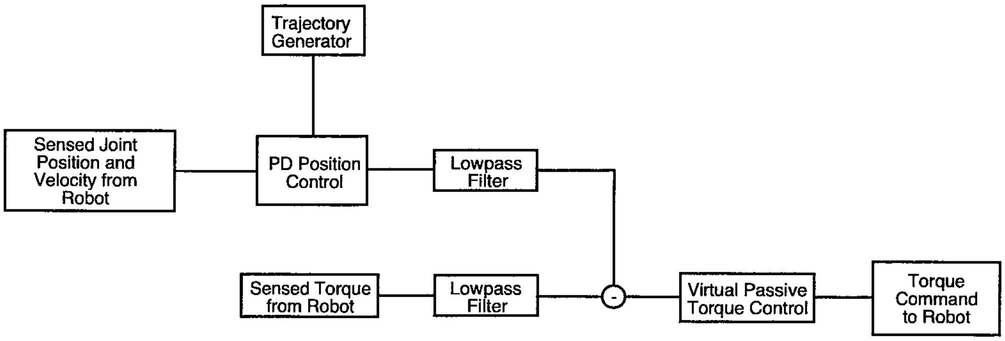


Figure 3.2: Controller block diagram

All the internal torque calculations are done in sensed torque units. The lowpass filters acting on sensed and commanded torques were Butterworth filters with a cutoff frequency of 6Hz. The commanded torque was filtered to remove high frequencies induced by the position controller and to match the phase of the filtered sensed torque. The position controller gains were tuned to give reasonable position tracking performance in the absence of the passive torque controller. Both the position and passive torque controllers for the wrist joints were run on the 68040 board at 200Hz. Position commands for joints 1-4 were generated at 200Hz and sent to the position controllers in the RRC controller. Position trajectories for all joints were created in Matlab and loaded onto the 68040 board. Data was collected and saved in Matlab format for ease of comparison.

To exercise the controller the arm was commanded along 4 joint trajectories. Each trajectory was tested with 3 different end-effector weighting conditions, 0lb, 5lb, and 10lb. Four trajectories were chosen to transition the wrist joint through several loading conditions. Trajectories 1 and 2 involved only wrist joint movements. These trajectories can be seen in Figs. 3.3-3.4. The main difference in these two trajectories is that joint 6, the wrist pitch joint, is heavily loaded in the initial position of trajectory 1 while it is unloaded in the initial position of trajectory 2. In trajectory 2, the load is transferred through the robot structure. Trajectories 3 and 4 have the same wrist joint trajectories as trajectories 1 and 2, respectively. In trajectory 3, joints 1 and 4 were moving. In trajectory 4, joints 2 and 3 were moving. The trajectories of the arm joints are shown in Figs. 3.5-3.7. By moving the four arm joints, the wrist is exposed to different loading conditions and the disturbance of the nonlinear term.

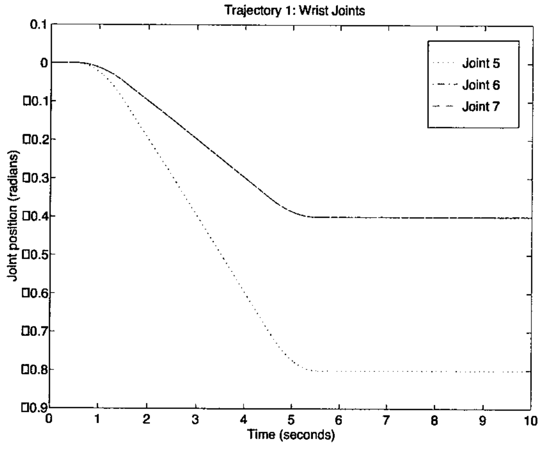


Figure 3.3: Trajectory 1: Wrist Joints

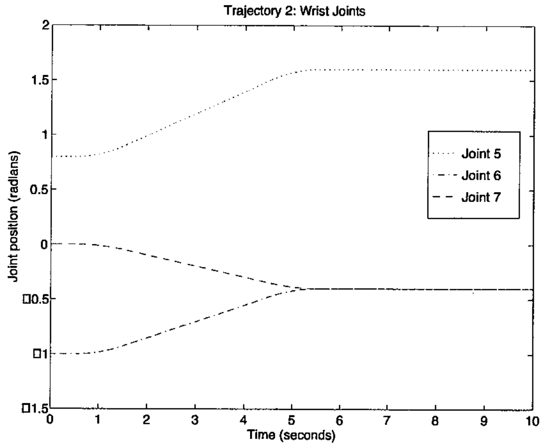


Figure 3.4: Trajectory 2: Wrist Joints

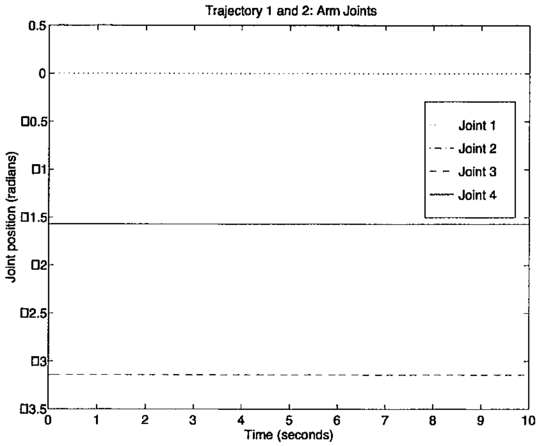


Figure 3.5: Trajectory 1&2: Arm Joints

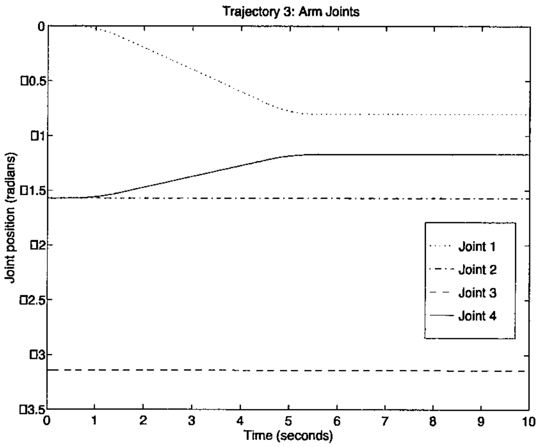


Figure 3.6: Trajectory 3: Arm Joints

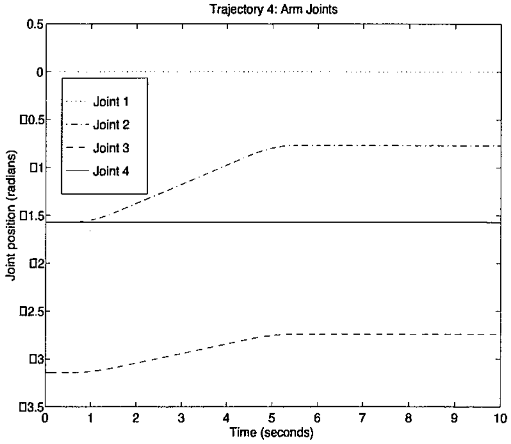


Figure 3.7: Trajectory 4: Arm Joints

The end-effector weights were chosen to produce significantly different joint torques in different weighting conditions. The weights were representative of a load usually carried by an arm of this size and did not put excessive stress on the arm mechanism. An offset of 9 inches from the center of the end-effector allowed joints 5 and 7, the wrist and toolplate roll joints, to be loaded easily.

3.3 Active joint braking

The first property of the controller examined was its ability to drive the velocity of a joint to zero in the absence of a joint command. This is an important mode of operation for fault tolerant applications. As the theory describes, this is done without position information. As a result, the controller cannot hold an absolute position. The resulting control achieves a very highly damped state which will significantly resist a change in joint velocity from the zero velocity state. As will be seen later, this highly damped mode of operation does not impair the position controller when the position controller is operational.

To exercise the passive torque controller, the position controller was allowed to run until three seconds into the trajectory. This activity induced a joint velocity and acceleration into the joints with the passive torque controller. At three seconds, the torque sensor bias generated by the position controller was deactivated for the remainder of the data collection run. Figs. 3.8-3.10 show the response of the wrist joints in all loading conditions during trajectory 1 while Figs. 3.11-3.13 show the same loading conditions in trajectory 2.

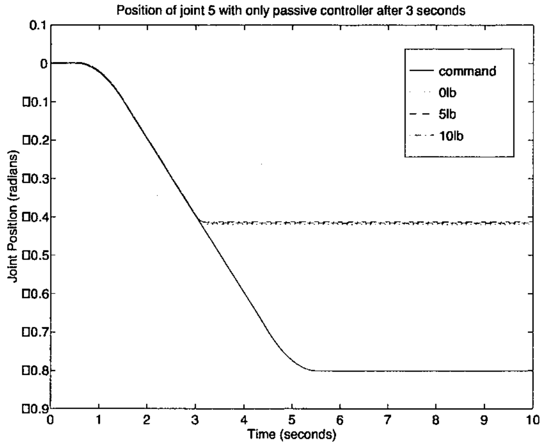


Figure 3.8: Trajectory 1 - Joint 5

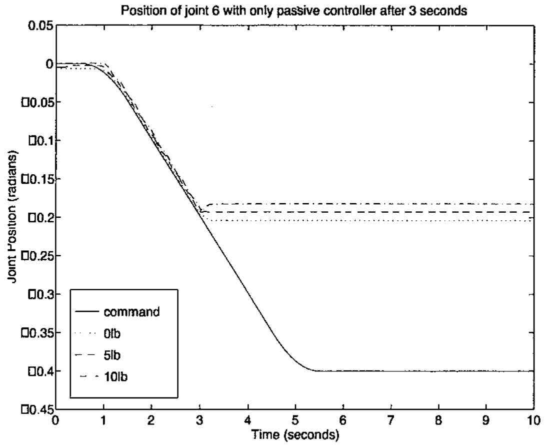


Figure 3.9: Trajectory 1 - Joint 6

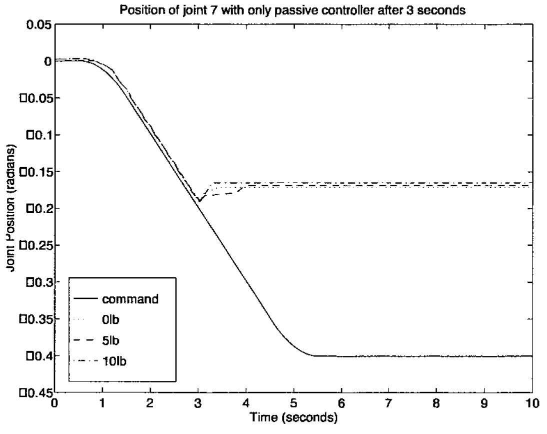


Figure 3.10: Trajectory 1 - Joint 7

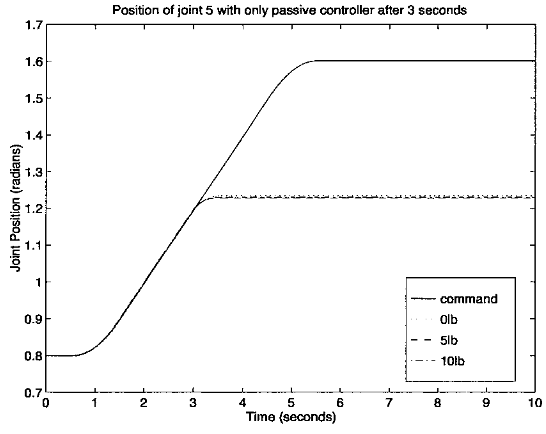


Figure 3.11: Trajectory 2 - Joint 5

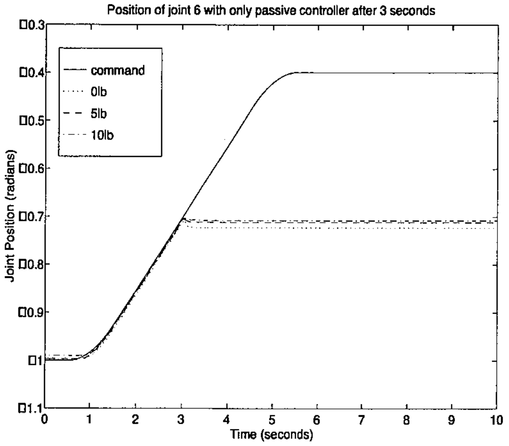


Figure 3.12: Trajectory 2 - Joint 6

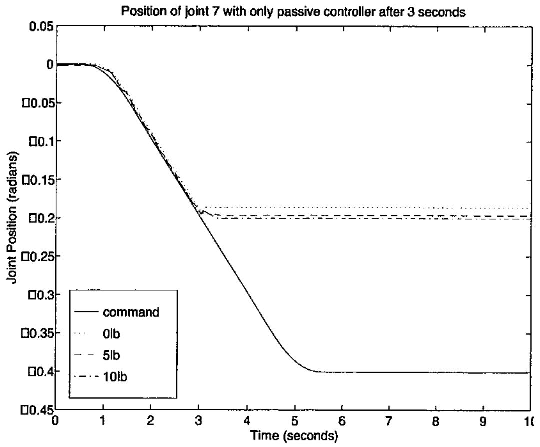


Figure 3.13: Trajectory 2 - Joint 7

The resulting sensed position plots show a smooth, consistent response to the controller change by joint 5. Joint 6 and 7 show some variance in the different loading conditions. The response of joint 6 to trajectory 1, seen in Fig. 3.9, shows the highest variance in final position, +/- 0.0125 radian. Joint 6 is heavily loaded in this configuration and the error grows in the direction of the load with the magnitude of the load. The response of joint 7 to trajectory 1, seen in Fig. 3.10, shows a discontinuity at the time the position controller was deactivated. In this case, the controller suddenly changes position in the direction of loading. This could be due to slightly low passive controller gains or different joint characteristics in different directions. The case for different joint characteristics is made by the response of joint 7 in trajectory 2, seen in Fig. 3.13. In this data set, the loading is similar in magnitude but in a different direction. The resulting discontinuity is less severe.

To show that the effectiveness of the passive torque control in the previous six figures is not a result of high joint friction that exists in the robot, Figs. 3.14-3.16 show the response to having both the passive torque controller and the position controller disabled three seconds into the trajectory. After three seconds, the torque command to all wrist joints was zero. These plots are for the 0lb condition.

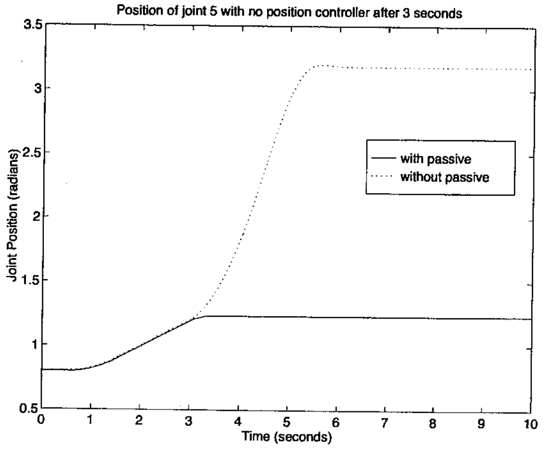


Figure 3.14: Joint 5 with/without control

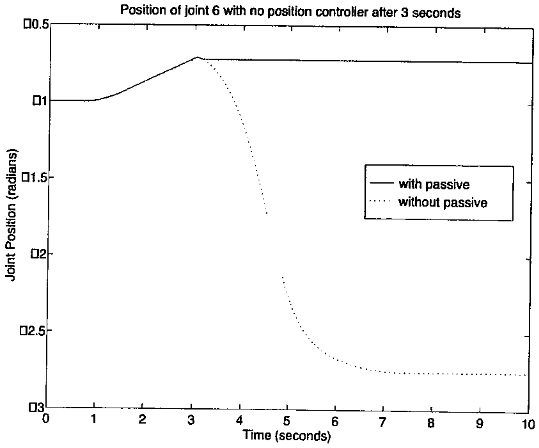


Figure 3.15: Joint 6 with/without control

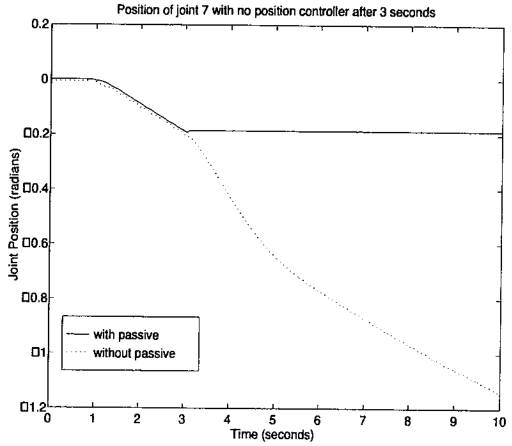


Figure 3.16: Joint 7 with/without control

Joints 5 and 6 reached their steady state condition at the physical joint stop. Joint 7 was still moving at the end of the data collection run. These plots show that the wrist joints, with the low level joint controller running, exhibit the lightly damped properties of direct drive joints. In a perfect implementation, the direct drive assumption could be accurate. Due to joint torque sensor bias, the manipulator exhibits a non-direct drive property. Joint 6 runs rapidly past the gravity equilibrium state to firmly engage the joint stop. This bias makes the torque controller drive to a non-zero torque state in the unloaded condition. This property of joint 6 caused problems in tuning the passive controller.

The sensed torque response to trajectory 1 is shown in Figs. 3.17-3.19. Each joint is shown at the three loading conditions. As expected, the steady state torque increases with the higher loading conditions. When the position controller's torque bias is removed at three seconds, the passive torque controller reacts by driving the joint velocity to zero. The magnitude of this response is related to the magnitude of the load. After the transient response ends, the controller produces a steady state joint torque command resulting in the sensed torque shown. The position controller gains in this test were high and the sensed velocity was not filtered. As a result, the position controller caused most of the oscillations before three seconds.

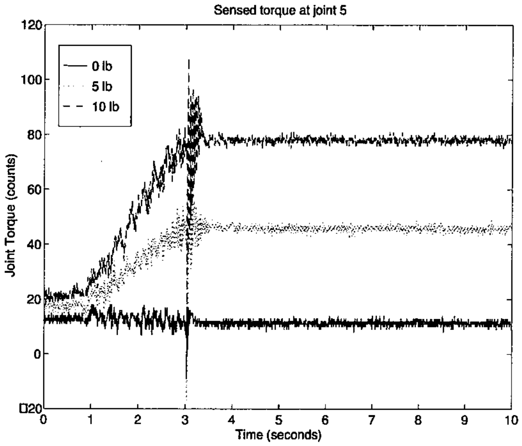


Figure 3.17: Sensed Torque Response - Joint 5

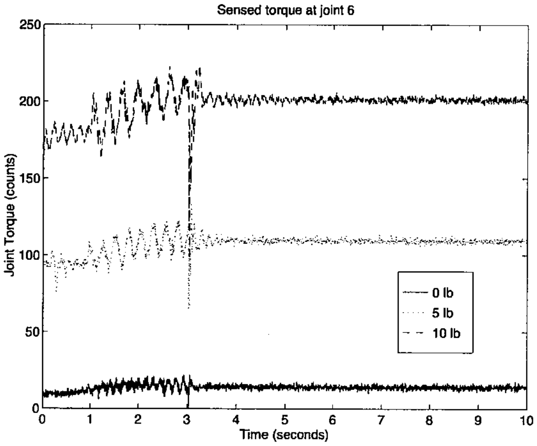


Figure 3.18: Sensed Torque Response - Joint 6

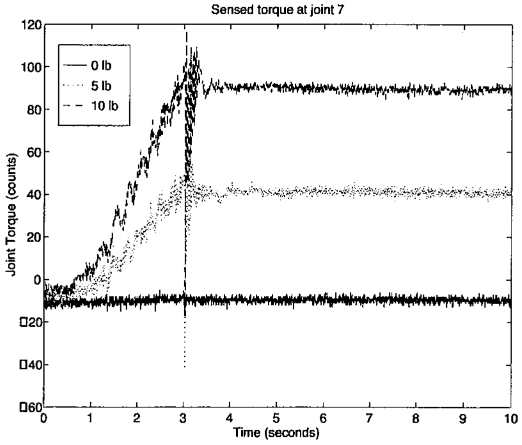


Figure 3.19: Sensed Torque Response - Joint 7

The commanded and sensed torques resulting from trajectory 1 are shown in Figs. 3.20-3.28 for the different weight conditions. All plots are in sensed torque units. Bias in sensed torques have been compensated for in the data..

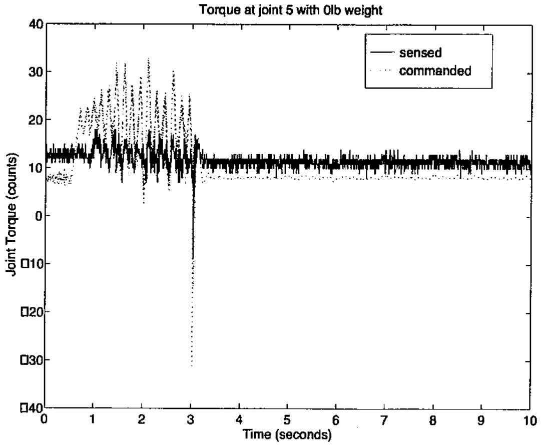


Figure 3.20: Torques - Joint 5 - 0lb

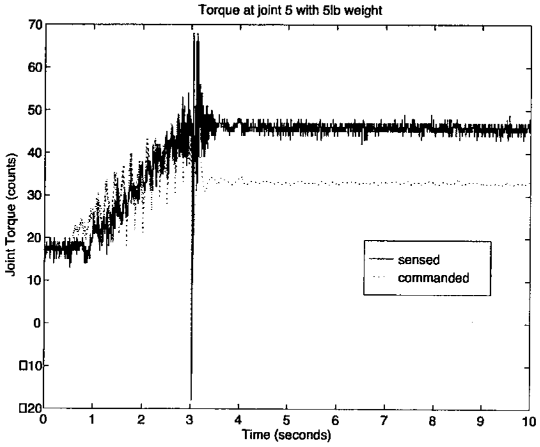


Figure 3.21: Torques - Joint 5 - 5lb

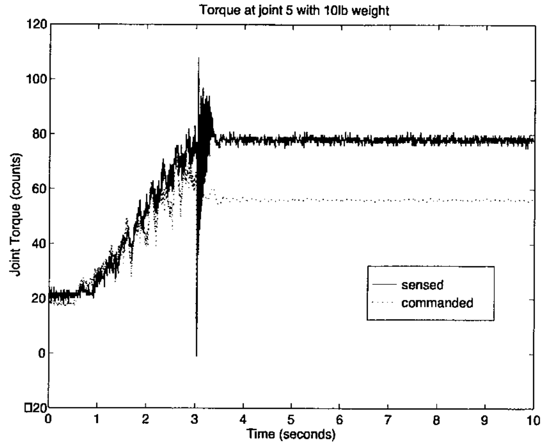


Figure 3.22: Torques - Joint 5 - 10lb

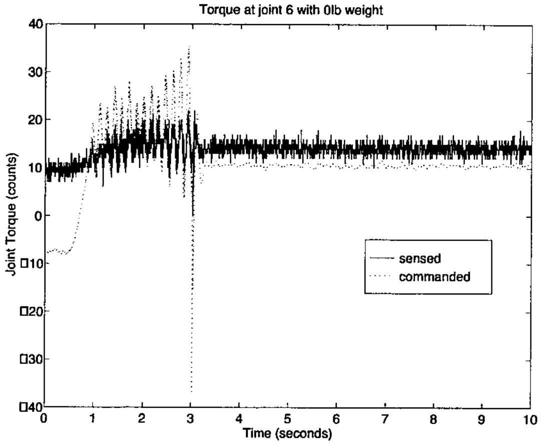


Figure 3.23: Torques - Joint 6 - 0lb

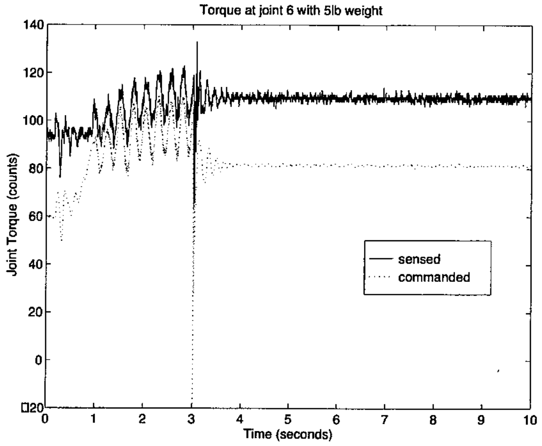


Figure 3.24: Torques - Joint 6 - 5lb

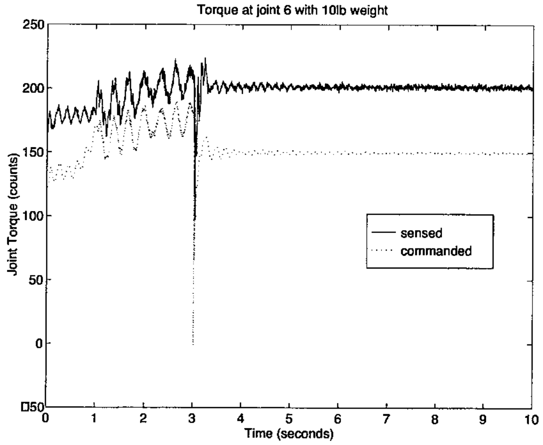


Figure 3.25: Torques - Joint 6 - 10lb

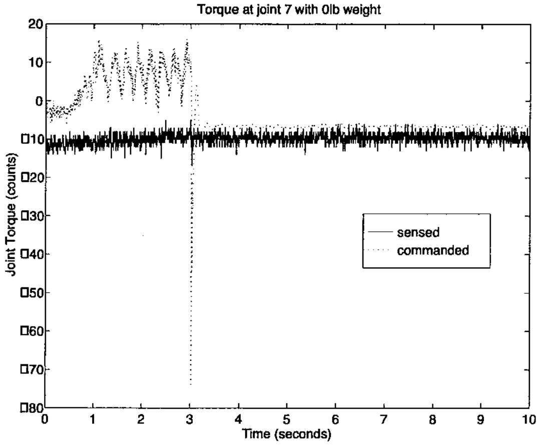


Figure 3.26: Torques - Joint 7 - 0lb

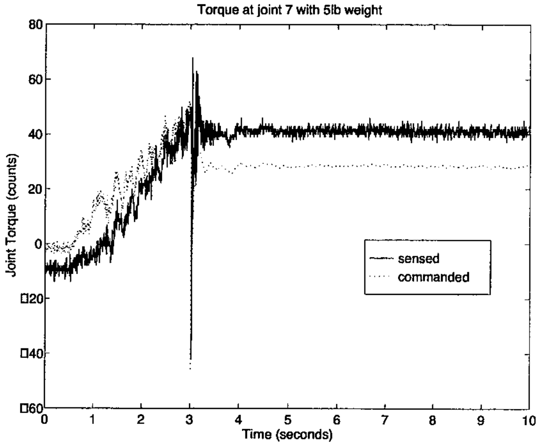


Figure 3.27: Torques - Joint 7 - 5lb

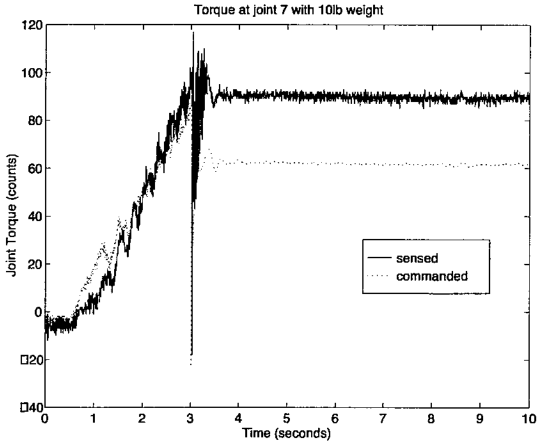


Figure 3.28: Torques - Joint 7 - 10lb

The sensed torque tracks the commanded torque well with some exceptions. In the 0lb weighting cases, the position controller commands much more than is sensed. Without a weight to resist the movement, the torque sensor maintains a low reading. At steady state, all plots develop a constant offset between commanded and sensed torques which is proportional to the loading of the joint. In Figs. 3.20-3.28, the commanded torque always settles to a torque value lower in magnitude than the sensed torque. This difference is caused by a commanded to sensed torque constant that is too low. Lowering the torque constant is equivalent to placing an output gain on the controller by artificially increasing the difference between sensed and commanded torques. The result is more joint torque output at the cost of increased system instability. Without proper choices for the passive controller gain matrices D_c and K_c, the lower than actual torque constant will cause the system to vibrate. Higher output at the cost of increased instability can also be accomplished through the R_τ matrix. While the R_τ matrix is explicitly involved in the controller's A matrix, lowering the torque constant changes the plant, not the controller. Increasing this effective output gain with the commanded to sensed torque constant gives the system designer more flexibility to change system performance without altering the eigenvalues of the system *A* matrix.

3.4 Position Tracking

To show that the passive torque controller enhances position trajectory tracking, trajectories 1 and 2 were allowed to complete, i.e. the position controller was enabled at all times. Test runs were made with the passive torque controller in the loop and with the passive torque controller bypassed. In the bypassed configuration, the position controller's bias torque that was used to alter the sensed torque input to the passive controller is sent directly to the robot as the commanded torque. Figs 3.29-3.34 show the position error between commanded and actual position for trajectory 1 while Figs 3.35-3.40 show the error for trajectory 2. Please note the differences in scale between plots.

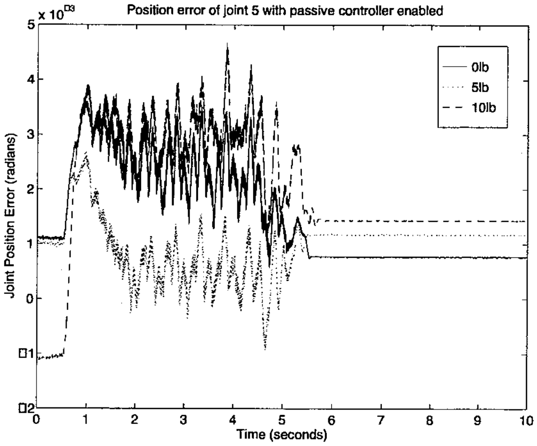


Figure 3.29: Position Error - Joint 5 - Traj. 1 with Passive Controller

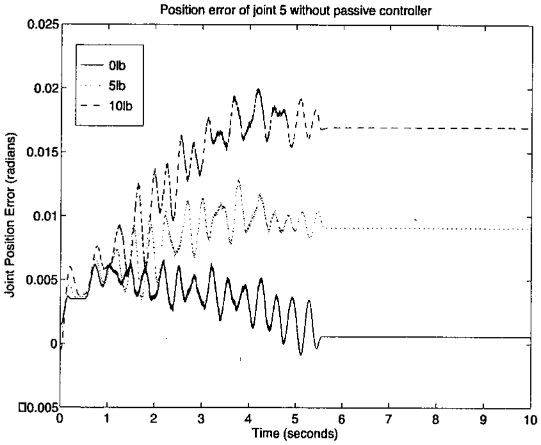
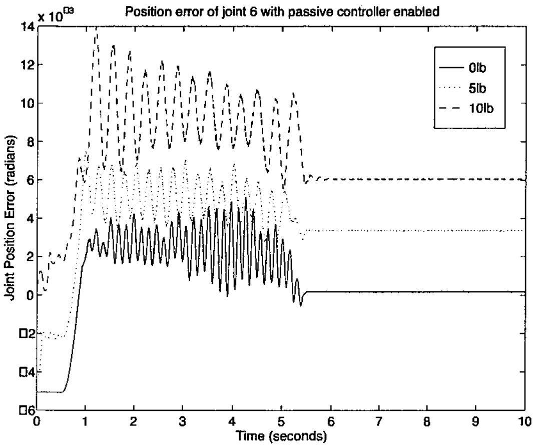


Figure 3.30: Position Error - Joint 5 - Traj. 1 without Passive Controller



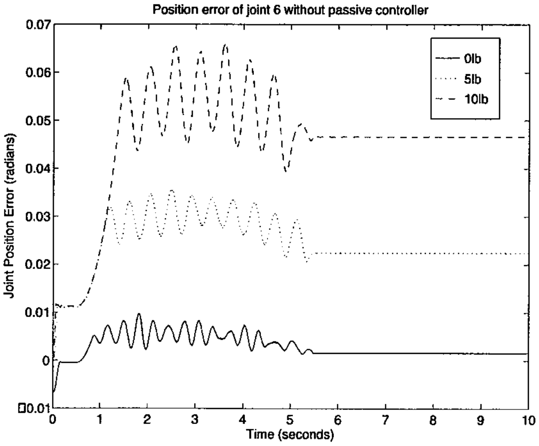
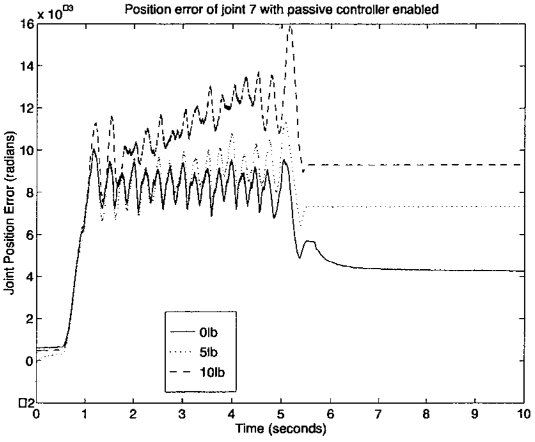


Figure 3.31: Position Error - Joint 6 - Traj. 1 with Passive Controller

Figure 3.32: Position Error - Joint 6 - Traj. 1 without Passive Controller



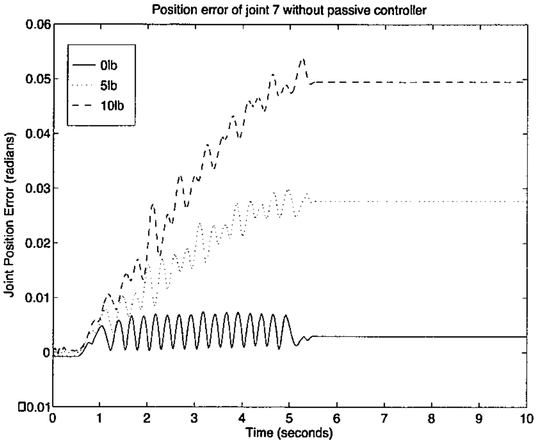


Figure 3.33: Position Error - Joint 7 - Traj. 1 with Passive Controller

Figure 3.34: Position Error - Joint 7 - Traj. 1 without Passive Controller

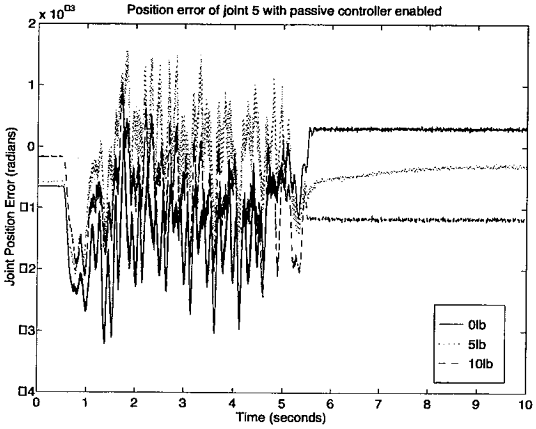


Figure 3.35: Position Error - Joint 5 - Traj. 2 with Passive Controller

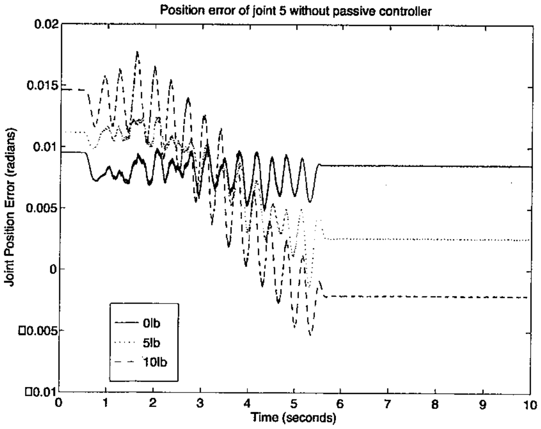


Figure 3.36: Position Error - Joint 5 - Traj. 2 without Passive Controller

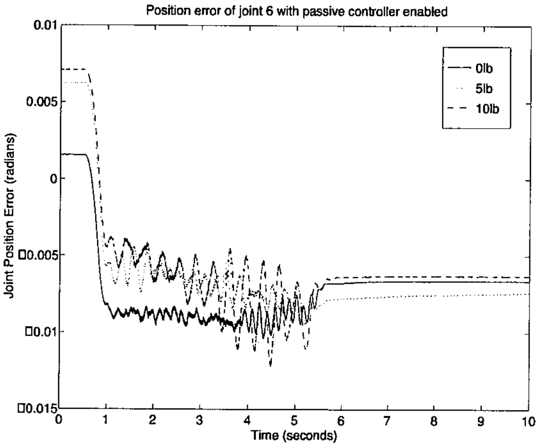


Figure 3.37: Position Error - Joint 6 - Traj. 2 with Passive Controller

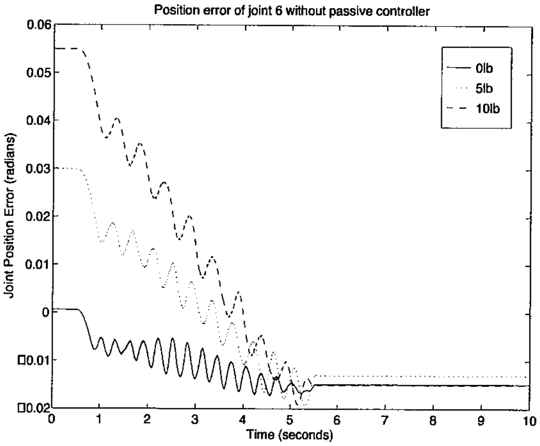


Figure 3.38: Position Error - Joint 6 - Traj. 2 without Passive Controller

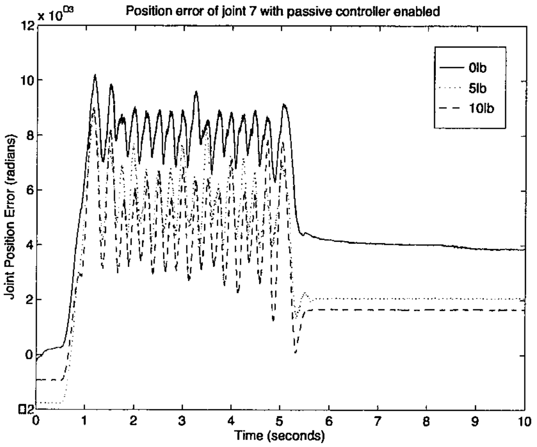


Figure 3.39: Position Error - Joint 7 - Traj. 2 with Passive Controller

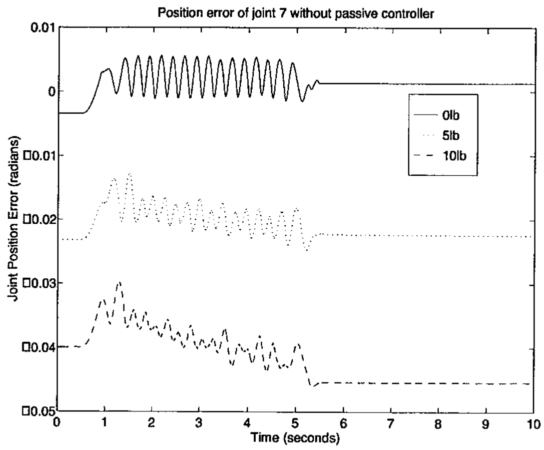


Figure 3.40: Position Error - Joint 7 - Traj. 2 without Passive Controller

To better analyze the errors due to the different controllers and trajectories, the mean and standard deviation of the errors were computed. The results of these computations may be found in Tables 3.2 and 3.3 and Figs. 3.41-3.46.

Table 3.2: Trajectory 1 Statistics

		Mean		Standard Deviation	
Joint	Weight	w/ passive	w/o passive	w/passive	w/o passive
5	0	0.001554	0.002301	0.000911	0.001968
5	5	0.001786	0.008859	0.000771	0.00204
5	10	0.001983	0.014995	0.00114	0.004377
6	0	0.000787	0.002936	0.002079	0.002324
6	5	0.00408	0.024447	0.002288	0.005922
6	10	0.007204	0.045508	0.002519	0.012615
7	0	0.005896	0.003101	0.002392	0.001823
7	5	0.007805	0.021298	0.002507	0.009274
7	10	0.009431	0.037379	0.002945	0.017043

Table 3.3: Trajectory 2 Statistics

		Mean		Standard Deviation	
Joint	Weight	w/ passive	w/o passive	w/passive	w/o passive
5	0	-0.000601	0.008255	0.000966	0.000894
5	5	-0.001085	0.005257	0.000805	0.003517
5	10	-0.00137	0.003187	0.000573	0.006735
6	0	-0.007081	-0.012039	0.002636	0.004669
6	5	-0.006772	-0.003544	0.00376	0.014053
6	10	-0.005812	0.002678	0.003961	0.023715
7	0	0.005568	0.001433	0.002511	0.00199
7	5	0.003798	-0.020853	0.002775	0.002386
7	10	0.003056	-0.041634	0.002311	0.00416

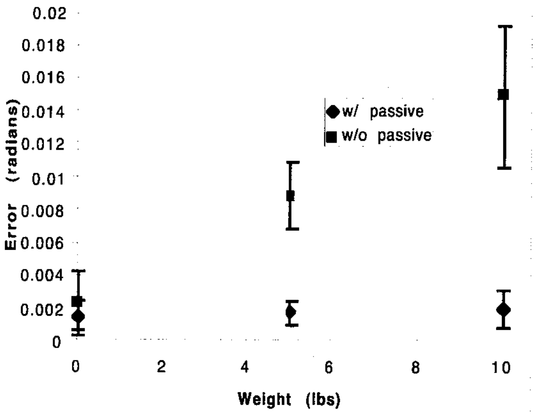


Figure 3.41: Statistics - Joint 5 - Traj. 1

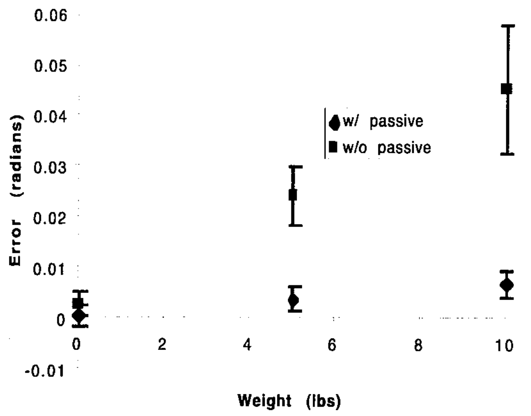


Figure 3.42: Statistics - Joint 6 - Traj. 1

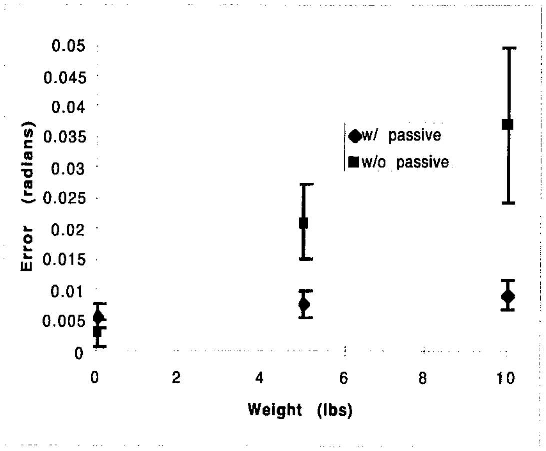


Figure 3.43: Statistics - Joint 7 - Traj. 1

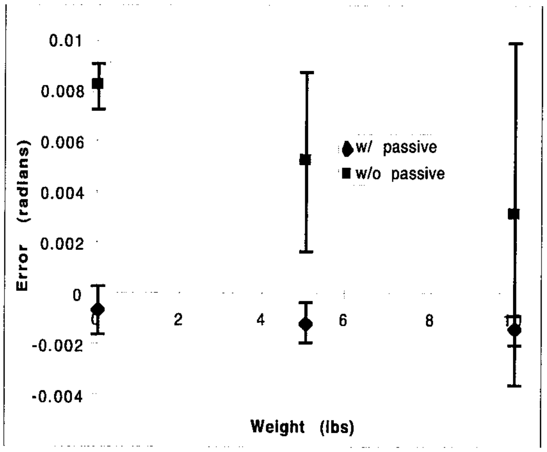


Figure 3.44: Statistics - Joint 5 - Traj. 2

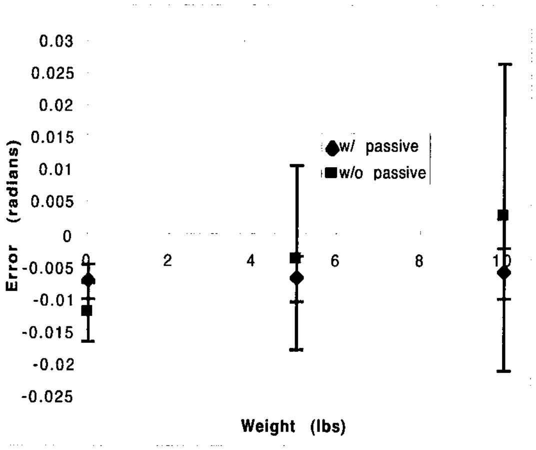


Figure 3.45: Statistics - Joint 6 - Traj. 2

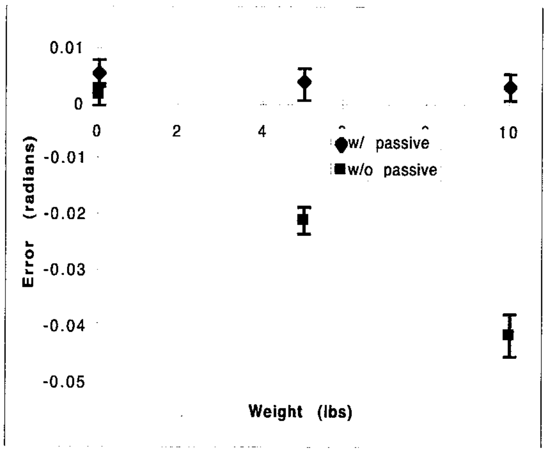


Figure 3.46: Statistics - Joint 7 - Traj. 2

In Figs 3.41-3.46, the data points are the mean values or the position error and the error bars are +/- the standard deviation. At the zero weight conditions, both the position controller only and the position controller with passive controller have similar mean values in most cases and standard deviations in all cases. Similar results were expected because there is little torque loading to cause the passive controller to react. The exception is a difference in the mean value of joint 5 in trajectory 2. This was due to a substantial position error in the initial position that was never overcome by the position controller. In the higher weight conditions, the position controller only configuration exhibits standard deviations which increase with weight implying more variance in error about the mean value. The error mean values also increase in magnitude except in the cases of joint 5 and 6 in trajectory 2 where a zero crossing occurred. Averaging the positive and negative errors lowered the mean error but increased the standard deviation. The cases with the position controller and the passive controller both active maintained fairly constant error means and standard deviations across all weight conditions. This robust performance with unknown loading is one of the strengths of the controller.

3.5 Response to arm joint movement

The following plots show the response to trajectories 3 and 4 where some of the arm joints were moved along with the wrist. The position controller for the wrist joints was disabled at 3 seconds while the trajectories of the arm joints were allowed to continue. These arm trajectories impart another weighting condition on the passive torque controllers and expose them to the nonlinear term shown in the joint model which the controller treats as a disturbance. Figs. 3.47-3.49 show the response to trajectory 3 while Figs. 3.50-3.52 show the response to trajectory 4.

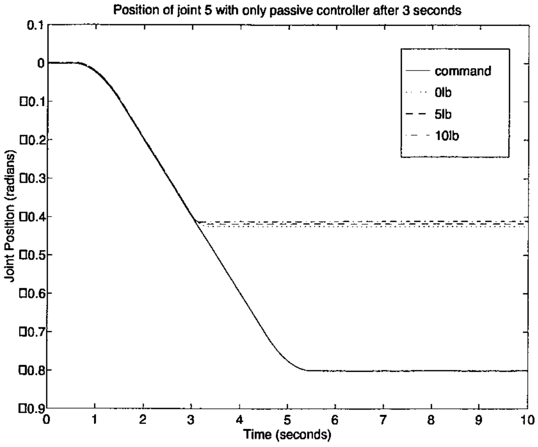


Figure 3.47: Position of Joint 5 - Traj. 3

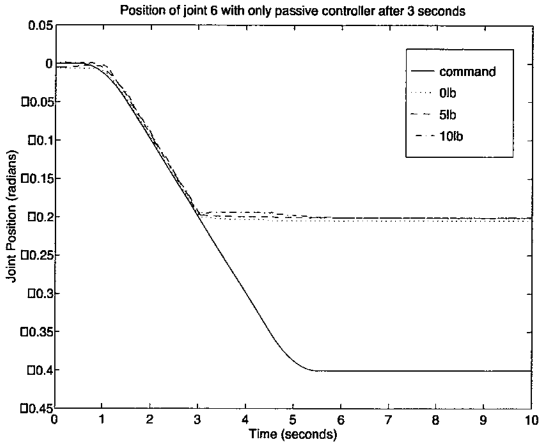


Figure 3.48: Position of Joint 6 - Traj. 3

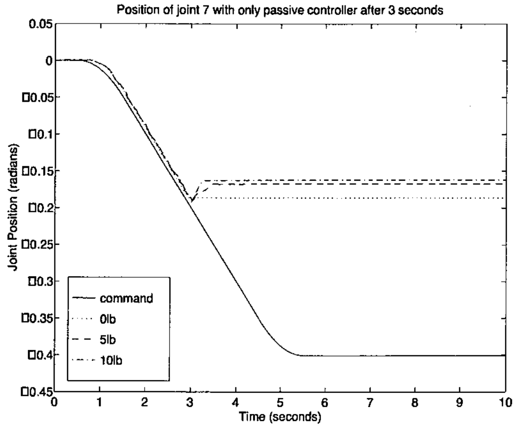


Figure 3.49: Position of Joint 7 - Traj. 3

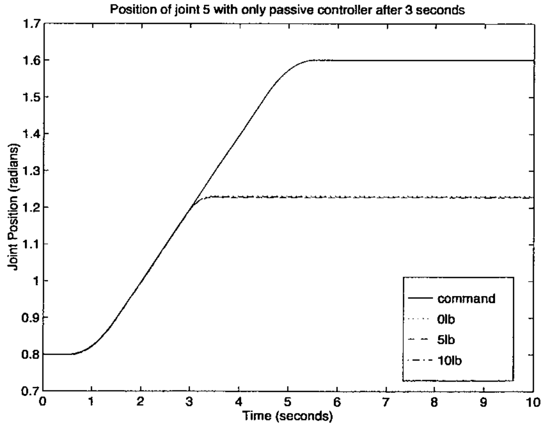


Figure 3.50: Position of Joint 5 - Traj. 4

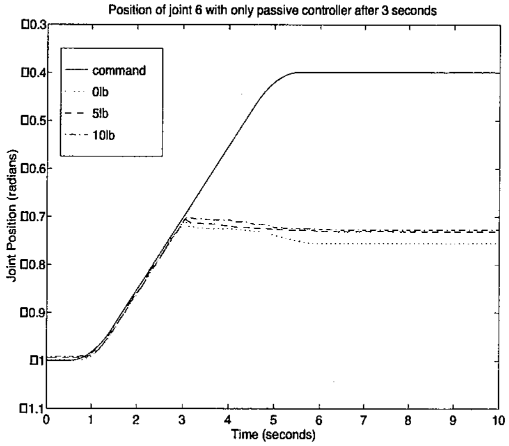


Figure 3.51: Position of Joint 6 - Traj. 4

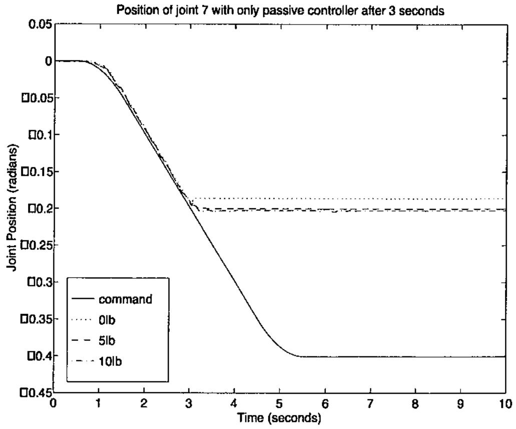


Figure 3.52: Position of Joint 7 - Traj. 4

These plots are similar to those produced by trajectory 1 and 2 where the arm joints were not moved. The different endpoint loading conditions made some minor changes to the steady state value. Since the controller does not utilize position information, a slightly different position response to a different loading condition is expected. The only interesting case is that of joint 6 in trajectory 4, seen in Fig. 3.51. In this case, the load is in the direction of the joint torque sensor bias. Although the passive torque controllers are symmetric, the effect of the joint torque bias is not. As a result, the lower gains forced on the joint torque controller to maintain stability while dealing with the corrupted lower level torque controller in combination with a low joint loading which developed lower sensor feedback for the controller to react to allowed some slippage. While this change is possibly due to the nonlinear term, it is probably due to the dynamic forces caused by decelerating. As the loading increased, the passive torque controller begins to dominate the lower level controller and holds the zero velocity state.

To show a case without position control, the data runs were conducted using trajectories 3 and 4 with no position control on the wrist joints. The passive torque controller was enabled for the entire run. The change in position of the wrist joint from the initial position is shown in Figs. 3.53-3.54. Only the 10lb weighting condition was considered.

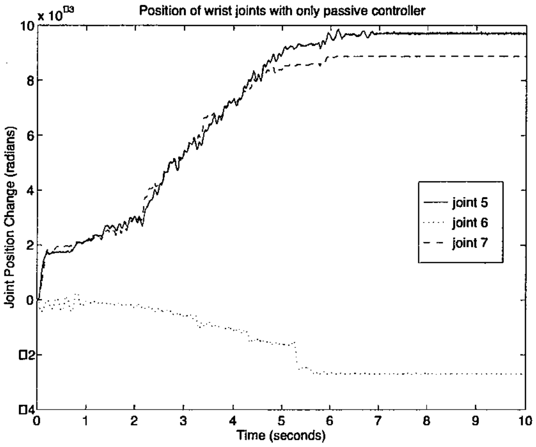


Figure 3.53: Position response to Traj. 3

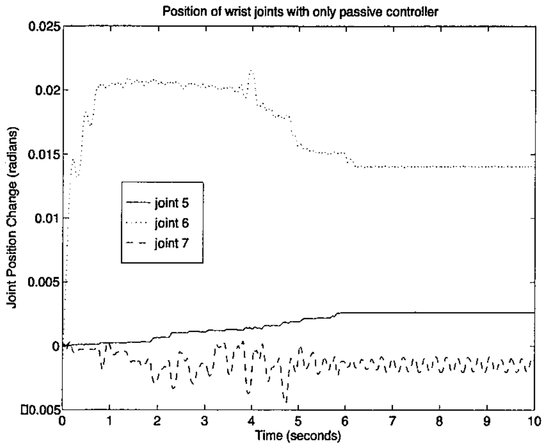


Figure 3.54: Position response to Traj. 4

Slippage in all cases was less than 0.021rad. Joint 6, with its lower passive controller gains and bias problem exhibited the worst performance. The increased slippage in trajectory 4 was due to the more severe loading change. In the response of joint 7 in trajectory 4, a constant, steady state value is not reached. Instead of a constant value, joint 7 settles into an oscillation with a magnitude of approximately 0.001rad. This oscillation is due to using high controller gains to produce a passive torque controller with good braking characteristics in combination with an uncertain system. Lower gains would stop the oscillation, but would allow more slippage. This oscillation continues after the arm trajectory is complete and the other wrist joints have reached a constant steady state. As a result, it cannot be due to the nonlinear term.

These cases show, for the tested joint accelerations and velocities, the controller response is similar to the response without arm joint movement. As a result, the passive controller design is robust to the disturbance caused by the nonlinear term. Due to the geared joint configuration, unknown joint parameters, changes in gravity loading along trajectories, and lower level torque controllers, any error due specifically to the nonlinear term will be difficult to isolate with this experimental apparatus.

3.6 Controller stability

Even with high gains and an uncertain system, the passive joint torque controller has shown good stability over different trajectories and loading conditions. In certain configurations, the loading on the torque controller is such that oscillations can occur. This problem could be solved if good joint braking characteristics were not required. Experience has shown that lower gains, while causing slippage in joint braking, can slightly enhance tracking performance.

Another stability problem occurs in the interaction of the position controller and the passive joint torque controller. At certain loading conditions, these controllers can fight each other and cause joint vibration. Since this vibration is passed through the structure of the robot to the other torque sensors, this vibration can set up a resonance with the other passive torque controllers. An example of this characteristic is shown in Figs. 3.55-3.57.

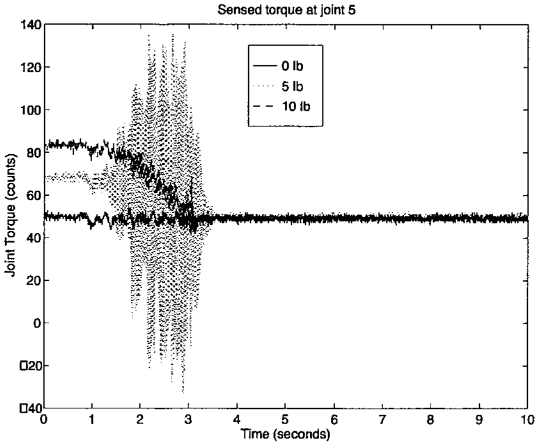


Figure 3.55: Sensed Torque - Joint 5

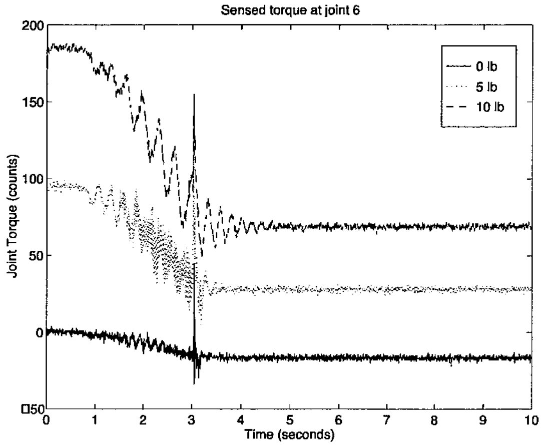


Figure 3.56: Sensed Torque - Joint 6

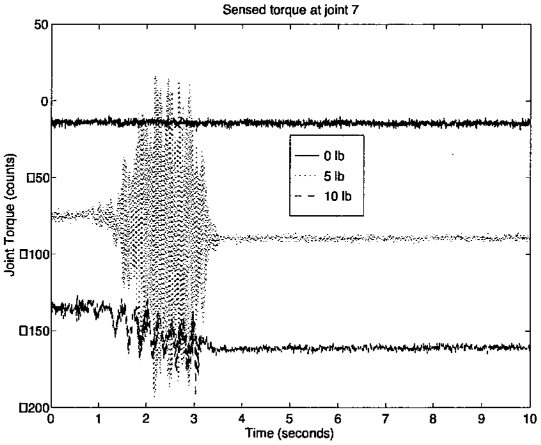


Figure 3.57: Sensed Torque - Joint 7

These plots show the sensed torque response to trajectory 2 with the position controller disabled at three seconds. A large torque vibration occurred in joints 5 and 7 at the 5lb weight condition. Joint 5 and 7 are both roll joints whose axes were becoming more parallel as the trajectory continues. The parallel condition allowed a vibration started in one joint to resonate into the other joint's sensor. This resonance does not occur if the passive controller is bypassed on one of the two joints. The resonance continues if the passive controller on joint 6 is bypassed removing it as a cause. The interesting point of the data is that the resonance occurred at the middle weight condition, not the high or low condition. This can be explained by the relative control authority of the passive controller. At low weights, the passive controller had little authority and the control is mostly done by the position controller. As the weight increases, the passive controller gains more authority. At a moderate weighting condition, the passive controller's control authority is similar to the position controller and they begin to interact with neither controller dominating. At higher weighting conditions, the passive controller begins to dominate the position controller. When the position controller's bias is zeroed at three seconds, the passive controller quickly damps out the resonance. Referring to the position response resulting from these torques in Figs 3.11-3.13, the resonance did not adversely effect position tracking. However, these quick torque changes put undue wear on the drivetrain and can excite modes in robot payloads. Further refining of position controller and passive torque controller gains should alleviate the problem.

3.7 Summary

The passive torque controller was successfully implemented on three consecutive joints in a real robot system. Even with very rough joint parameters, the controller worked as predicted. When the position controller was disabled, the passive torque controllers slowed the joints to a stop and attempted to hold the zero velocity state. When the position controller was enabled, the passive torque controller enhanced position tracking in the presence of unknown loading. At the reasonable robot speeds tested, the controller was robust to the nonlinear term in the joint model. The controller showed good stability in different trajectories and weight conditions. The high passive controller gains did lead to a mild position oscillation in some conditions and an interaction of the position and passive torque controllers that can result in a torque resonance. Better joint parameters, torque sensors, and lower gains should fix these minor stability problems.

Although the virtual passive torque controller has been experimentally shown in this chapter to be able to stabilize a robot joint without position feedback, that feedback is required for position control. In the joint position sensor fault tolerance application, this position feedback cannot come from the primary joint position sensor as it did in this chapter because that sensor has failed. It was found in chapter 1 that without direct sensor redundancy there is no satisfactory method of determining joint position for an extended period in an unstructured environment. Chapter 4 presents a joint specific method of using Cartesian accelerometers to determine joint position without double integration. The virtual passive torque controller will reappear in chapter 5 when it is integrated with the joint specific position determination method to show joint position sensor failure recovery on a real robot system. In chapter 5, and later in chapter 7, the number of joints controlled with the virtual passive torque controller will be increased and the controller implemented on the major arm joints.

4. Joint Specific Position Sensor Fault Tolerance in Robot Systems Using Cartesian Accelerometers

4.1 Introduction

In previous chapters a virtual passive torque controller was presented which does not require position feedback for stabilization. A control system using virtual passive torque control must have joint position feedback available for position control. During failure of the primary joint position sensor this position feedback must come from another source. As discussed in the literature search, currently the only method available for long duration position sensor fault tolerance in unstructured environments is direct sensor redundancy. The use of direct sensor redundancy could affect joint design while an analytically redundant method may not affect the design. This chapter proposes a method using analytical redundancy to obtain joint position from Cartesian accelerometers without integration or inverse kinematics. The proposed method is specific to the joint driving the link to which the accelerometers are attached. The method is based on kinematic analysis of the robot system, known accelerometer placement, and a known acceleration field, such as gravity. This chapter will present a computational algorithm for obtaining joint position from accelerometer measurements. Simulations of a 7 degree of freedom (DOF) robot arm operating in general 6 DOF space will be shown. The effect of sensor bias and noise will be addressed and illustrated. Chapter 5 will detail experimental results.

4.2 Proposed System

The techniques derived in this chapter are applicable to an n joint, rigid link, manipulator able to move in general 6 DOF space. To obtain the general solution for joint position, velocity, and acceleration, each joint will require two triaxial accelerometers, *A1* and *A2*. Figure 4.1 shows an example joint with the required configuration. Although Fig. 4.1 shows two coplanar joints, the equations are applicable to all single rotation joint types.

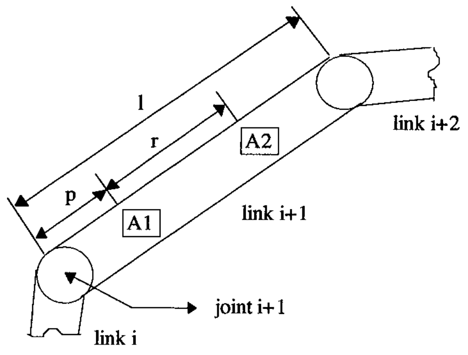


Figure 4.1: Proposed joint configuration

The acceleration of the points where the accelerometers are placed is given by kinematic analysis. For consistency with classical robot notation, the kinematic analysis is presented in Newton-Euler iterative dynamics form [58]. For simplicity, the accelerometer axes are assumed parallel to the axis system of the link to which they are attached. A constant, known rotation could be included in the *A1* and *A2* measurements without changing the analysis. The kinematic equations for the robot are:

$$\omega_{i+1}^{i+1} = R_i^{i+1}\omega_i^i + \dot{\theta}_{i+1}\hat{z} \tag{4.1}$$

$$\dot{\omega}_{i+1}^{i+1} = R_i^{i+1}\dot{\omega}_i^i + R_i^{i+1}\omega_i^i \times \dot{\theta}_i\hat{z} + \ddot{\theta}_{i+1}\hat{z} \tag{4.2}$$

$$a_{i+1}^{i+1} = R_i^{i+1}a_{i+1}^i \tag{4.3}$$

$$A1 = \dot{\omega}_{i+1}^{i+1} \times p + \omega_{i+1}^{i+1} \times \omega_{i+1}^{i+1} \times p + a_{i+1}^{i+1} \tag{4.4}$$

$$A2 = \dot{\omega}_{i+1}^{i+1} \times (r+p) + \omega_{i+1}^{i+1} \times \omega_{i+1}^{i+1} \times (r+p) + a_{i+1}^{i+1} \tag{4.5}$$

$$a_{i+2}^{i+1} = \dot{\omega}_{i+1}^{i+1} \times l + \omega_{i+1}^{i+1} \times \omega_{i+1}^{i+1} \times l + a_{i+1}^{i+1} \tag{4.6}$$

where,

ω_i^i	angular velocity of i^{th} link in i^{th} frame
a_i^i	acceleration of i^{th} link in i^{th} frame
a_{i+1}^i	acceleration of *i+1*th link in i^{th} frame
R_i^{i+1}	rotation matrix from i^{th} frame to *(i+1)*th frame
θ_i	position of i^{th} joint
A1	acceleration of accelerometer A1
A2	acceleration of accelerometer A2
p	vector offset from joint *i+1* to accelerometer A1 in frame *i+1*

r vector offset from accelerometer A1 to accelerometer A2 in frame *i+1*

l vector offset from joint *i+1* to joint *i+2* in frame *i+1*

$\hat{z}$ identity z vector $[0\ 0\ 1]^{\mathrm{T}}$

Given the Devanit-Hartenberg parameter [58], α_i, the 3x3 rotation matrix, R_i^{i+1}, is shown below.

$$R_i^{i+1} = \begin{bmatrix} \cos\theta_{i+1} & \sin\theta_{i+1}\cos\alpha_i & \sin\theta_{i+1}\sin\alpha_i \\ -\sin\theta_{i+1} & \cos\theta_{i+1}\cos\alpha_i & \cos\theta_{i+1}\sin\alpha_i \\ 0 & -\sin\alpha_i & \cos\alpha_i \end{bmatrix} \tag{4.7}$$

4.3 Position solution

The solution method for obtaining joint position from Cartesian accelerometers without integration is based on the kinematic relationships presented in section 4.2. For simplicity, assume that the vector p is ζ, the zero vector $[0\ 0\ 0]^{\mathrm{T}}$. The solution for general p will be presented later. With this assumption, the equation for *A1* becomes:

$$A1 = a_{i+1}^{i+1} \tag{4.8}$$

or

$$A1 = R_i^{i+1} a_{i+1}^{i} \tag{4.9}$$

Assuming that a_{i+1}^{i} is known, Eq. (4.9) can be solved for θ_{i+1} using atan2, the quadrant sensitive inverse tangent function. Only the x and y axes of *A1* will be needed for the calculation. The z axis does not contain information about θ_{i+1} when $p=\zeta$ but will be required for other equations. Let a_{i+1}^{i} and *A1* be expressed by:

$$a_{i+1}^{i} = \begin{bmatrix} \hat{a}_x \\ \hat{a}_y \\ \hat{a}_z \end{bmatrix}, A1 = \begin{bmatrix} A1_x \\ A1_y \\ A1_z \end{bmatrix} \tag{4.10}$$

Solving Eq.(4.9) for $\cos\theta_{i+1}$ and $\sin\theta_{i+1}$ provides:

$$\cos\theta_{i+1} = \frac{1}{\hat{a}_x}(A1_x - \hat{a}_y \sin\theta_{i+1} \cos\alpha_i - \hat{a}_z \sin\theta_{i+1} \sin\alpha_i) \tag{4.11}$$

$$\sin\theta_{i+1} = \frac{(\hat{a}_y \cos\alpha_i + \hat{a}_z \sin\alpha_i)A1_x - A1_y \hat{a}_x}{\hat{a}_x^2 + (\hat{a}_y \cos\alpha_i + \hat{a}_z \sin\alpha_i)^2} \tag{4.12}$$

Dividing Eq.(4.12) by Eq.(4.11) results in:

$$\tan\theta_{i+1} = \frac{(\hat{a}_y \cos\alpha_i + \hat{a}_z \sin\alpha_i)A1_x - A1_y \hat{a}_x}{(\hat{a}_y \cos\alpha_i + \hat{a}_z \sin\alpha_i)A1_y + A1_x \hat{a}_x} \tag{4.13}$$

$$\begin{aligned}\theta_{i+1} &= \tan^{-1}((\hat{a}_y \cos\alpha_i + \hat{a}_z \sin\alpha_i)A1_x - A1_y \hat{a}_x, \\ &(\hat{a}_y \cos\alpha_i + \hat{a}_z \sin\alpha_i)A1_y + A1_x \hat{a}_x)\end{aligned} \tag{4.14}$$

For this method to be effective, there must be a known acceleration field, such as gravity, to excite the accelerometers. The accelerometers must also be of the instrument type that responds to constant acceleration. This field provides a reference for the accelerometer readings.

The link that the accelerometers are attached to cannot be permanently parallel to the acceleration field or the link will act as if the acceleration field does not exist. For example, the acceleration measured by the accelerometers on a link attached to a roll joint whose roll axis is permanently parallel to the acceleration field will be identically zero if $p=\xi$. However, this method will work for that roll joint if its roll axis is being changed by joints earlier in the kinematic chain. At points in the trajectory that the roll axis does line up with the acceleration field, the previous joint position can be used for the small duration of the exact alignment.

4.4 Solution for a_{i+2}^{i+1}

If the trajectories of all the joints earlier in the kinematic chain are known, a_{i+1}^{i}can be calculated using the equations of motion and the position of the $(i+1)^{th}$ joint computed. However, if the trajectory of the $(i+1)^{th}$ joint is unknown due to a sensor failure, then for the $(i+2)^{th}$ joint to be recoverable using the position solution presented in section 4.3, a_{i+2}^{i+1}must be calculated.

The position solution could be differentiated to obtain joint velocity and acceleration. These values could be used to calculate a_{i+2}^{i+1} using the equations of motion. This is impractical because of the noisy nature of accelerometer related quantities. This noise would be magnified by differentiation.

In the ideal situation, *A2* would be attached to the *(i+1)*th frame but placed coincident with the *(i+2)*th joint (i.e. *r=l-p*). In this case, $A2 = a_{i+2}^{i+1}$. In the general case when this placement is not possible, if three triaxial accelerometers were attached to a link, then six relative accelerations could be generated by subtracting one accelerometer measurement from the other two. This would lead to the following six equations:

$$A2 - A1 = \dot{\omega}_{i+1}^{i+1} \times r + \omega_{i+1}^{i+1} \times \omega_{i+1}^{i+1} \times r \tag{4.15}$$

$$A3 - A1 = \dot{\omega}_{i+1}^{i+1} \times r' + \omega_{i+1}^{i+1} \times \omega_{i+1}^{i+1} \times r' \tag{4.16}$$

Where *r'* is the vector from A1 to A3. Although nonlinear in angular velocity, Eqs. (4.15) and (4.16) could be solved numerically for angular acceleration and angular velocity. A fourth acceleration, a_{i+2}^{i+1}, can be calculated by using the resulting angular acceleration and velocity with one of the measured Cartesian accelerations as shown:

$$a_{i+2}^{i+1} = \dot{\omega}_{i+1}^{i+1} \times (l - p) + \omega_{i+1}^{i+1} \times \omega_{i+1}^{i+1} \times (l - p) + A1 \tag{4.17}$$

Since the proposed configuration contains only two triaxial accelerometers, it can only produce three relative accelerations. As a result, the solution above is not applicable to this configuration. However, if the angular acceleration and velocity of the *i*th link is known along with θ_{i+1}, then the equations can be reformatted and a solution is possible.

Equations (1) and (2) for the angular acceleration and velocity of the *(i+1)*th link are substituted into Eq. (4.15) yielding:

$$\begin{aligned} A2 - A1 = {} & (R_i^{i+1}\dot{\omega}_i^i + R_i^{i+1}\omega_i^i \times \dot{\theta}_{i+1}\hat{z} + \ddot{\theta}_{i+1}\hat{z}) \times r + \\ & (R_i^{i+1}\omega_i^i + \dot{\theta}_{i+1}\hat{z}) \times (R_i^{i+1}\omega_i^i + \dot{\theta}_{i+1}\hat{z}) \times r \end{aligned} \tag{4.18}$$

To keep the derivation linear, $\theta_{i+1}, \dot{\theta}_{i+1}$, and $\dot{\theta}_{i+1}^2$ are chosen as the unknowns. Equation (4.18) can be rewritten as,

$$A2 - A1 = BB\ddot{\theta}_{i+1} + (CC + EE + FF)\dot{\theta}_{i+1} + GG\dot{\theta}_{i+1}^2 + DD \tag{4.19}$$

where,

$$BB = \hat{z} \times r \tag{4.20}$$

$$CC = (R_i^{i+1}\omega_i^i \times \hat{z}) \times r \tag{4.21}$$

$$DD = R_i^{i+1}\omega_i^i \times (R_i^{i+1}\omega_i^i \times r) + R_i^{i+1}\dot{\omega}_i^i \times r \tag{4.22}$$

$$EE = \hat{z} \times (R_i^{i+1}\omega_i^i \times r) \tag{4.23}$$

$$FF = R_i^{i+1}\omega_i^i \times (\hat{z} \times r) \tag{4.24}$$

$$GG = \hat{z} \times (\hat{z} \times r) \tag{4.25}$$

The terms *BB,CC,DD,EE,FF,* and *GG* are not dependent on the velocity and acceleration of the $(i+1)^{\text{th}}$ joint. Written in matrix form,

$$A2 - A1 - DD = \begin{bmatrix} CC + EE + FF & BB & GG \end{bmatrix} \begin{bmatrix} \dot{\theta}_{i+1} \\ \ddot{\theta}_{i+1} \\ \dot{\theta}_{i+1}^2 \end{bmatrix} \tag{4.26}$$

or,

$$\tilde{A} = W\Theta \tag{4.27}$$

Calculating the cross products in *W,* let:

$$\Omega = R_i^{i+1}\omega_i^i = \begin{bmatrix} \Omega_x \\ \Omega_y \\ \Omega_z \end{bmatrix} \quad r = \begin{bmatrix} r_x \\ r_y \\ r_z \end{bmatrix} \quad \tilde{A} = \begin{bmatrix} \tilde{A}_x \\ \tilde{A}_y \\ \tilde{A}_z \end{bmatrix} \tag{4.28}$$

Now, *W* becomes:

$$W = \begin{bmatrix} -2\Omega_z r_x & -r_y & -r_x \\ -2\Omega_z r_y & r_x & -r_y \\ 2(\Omega_x r_x + \Omega_y r_y) & 0 & 0 \end{bmatrix} \tag{4.29}$$

Unless $r_x = r_y = 0$, the rank of *W* will be at least 2. Since the vector *r* is a design parameter, non-zero components can be guaranteed. Referring to Eq.(7), it can be easily shown that Ω_z does not depend on θ_{i+1}. As a result, only *W*(3,1) is dependent on joint position. However, due to the *DD* term, all solutions will be dependent on θ_{i+1}.

To solve Eq.(4.27), three cases must be considered. Let ε be some small number, approximately 10^{-3}, used to bypass small elements of *W* that would be zero with perfect measurements. Assuming either r_x or r_y is non-zero, the situations are:

1. $|\Omega_x r_x + \Omega_y r_y| > \varepsilon$ => rank(W)=3
2. $|\Omega_x r_x + \Omega_y r_y| < \varepsilon$ and ($|\Omega_z r_x| > \varepsilon$ or $|\Omega_z r_y| > \varepsilon$)=> rank(W)=2, equations are underdetermined
3. $|\Omega_x r_x + \Omega_y r_y| < \varepsilon$ and $|\Omega_z r_x| < \varepsilon$ and $|\Omega_z r_y| < \varepsilon$ => rank(W)=2, unique solution with no dependence on $\dot{\theta}_{i+1}$

To solve each case, the following techniques can be used:

4.4.1 Case 1

In this case, a solution exists for $\dot{\theta}_{i+1}$ that does not depend on $\ddot{\theta}_{i+1}$ or $\dot{\theta}^2_{i+1}$. As a result, the solution technique will not consider $\dot{\theta}^2_{i+1}$ as an independent variable to be solved. Calculate:

$$\dot{\theta}_{i+1} = \frac{\tilde{A}_z}{2(\Omega_x r_x + \Omega_y r_y)} \tag{4.30}$$

Then, using the results of Eq.(4.30):

$$\begin{bmatrix} -r_y \\ r_x \end{bmatrix} \ddot{\theta}_{i+1} = \begin{bmatrix} \tilde{A}_x + 2\Omega_z r_x \dot{\theta}_{i+1} + r_x(\dot{\theta}_{i+1})^2 \\ \tilde{A}_y + 2\Omega_z r_y \dot{\theta}_{i+1} + r_y(\dot{\theta}_{i+1})^2 \end{bmatrix} \tag{4.31}$$

Note that the calculated value of $\dot{\theta}_{i+1}$ from Eq.(4.30) was squared and used in Eq.(4.31) instead of solving for $\dot{\theta}^2_{i+1}$. If both r_x and r_y are non-zero, Eq.(4.31) is overdetermined and can be solved by a closed-form pseudoinverse.

$$\ddot{\theta}_{i+1} = \frac{1}{r_x^2 + r_y^2} \begin{bmatrix} -r_y & r_x \end{bmatrix} \begin{bmatrix} \tilde{A}_x + 2\Omega_z r_x \dot{\theta}_{i+1} + r_x(\dot{\theta}_{i+1})^2 \\ \tilde{A}_y + 2\Omega_z r_y \dot{\theta}_{i+1} + r_y(\dot{\theta}_{i+1})^2 \end{bmatrix} \tag{4.32}$$

4.4.2 Case 2

In case 2, if $\theta_{i+1}, \dot{\theta}_{i+1}$, and $\dot{\theta}^2_{i+1}$ are considered independent variables, the system is underdetermined, i.e. it has an infinite number for solutions. Unfortunately, the values which the solution is meant to calculate are unique. To deal with this inconsistency, the solution will exploit the relationship between $\dot{\theta}_{i+1}$ and $\dot{\theta}^2_{i+1}$. First, a valid solution for $\theta_{i+1}, \dot{\theta}_{i+1}$, and $\dot{\theta}^2_{i+1}$ is calculated using a closed-from pseudoinverse.

$$\Theta = W(W^T W)^{-1} \begin{bmatrix} \tilde{A}_x \\ \tilde{A}_y \end{bmatrix} \tag{4.33}$$

Where $(W^T W)^{-1}$ is an easily calculated 2x2 matrix inverse with guaranteed existence. Using the value of $\dot{\theta}_{i+1}$ from Eq.(4.33), let,

$$\dot{\theta}_{i+1}^{C} = \dot{\theta}_{i+1} \tag{4.34}$$

now,

$$\begin{bmatrix} \dot{\theta}_{i+1} \\ \ddot{\theta}_{i+1} \end{bmatrix} = \begin{bmatrix} -2\Omega_z r_x + r_x \dot{\theta}_{i+1}^{C} & -r_y \\ -2\Omega_z r_y - r_y \dot{\theta}_{i+1}^{C} & r_x \end{bmatrix}^{-1} \begin{bmatrix} \tilde{A}_x \\ \tilde{A}_y \end{bmatrix} \tag{4.35}$$

Repeat the calculation of Eqs.(4.34) and (4.35) as required. Note that Eq.(4.33) is only calculated once per solution and the $\dot{\theta}_{i+1}$ calculated in Eq.(4.35) is used for the next iteration of Eq.(4.34).

4.4.3 Case 3

In case 3, the solution contains no information on $\dot{\theta}_{i+1}$. As a result, the solution of $\dot{\theta}_{i+1}^{2}$ must be calculated and the solution of $\dot{\theta}_{i+1}$ inferred through the square root relationship. For this case:

$$\begin{bmatrix} \ddot{\theta}_{i+1} \\ \dot{\theta}_{i+1}^{2} \end{bmatrix} = \frac{1}{r_x^2 + r_y^2} \begin{bmatrix} -r_y & r_x \\ -r_x & -r_y \end{bmatrix} \begin{bmatrix} \tilde{A}_x \\ \tilde{A}_y \end{bmatrix} \tag{4.36}$$

Note that this relationship is constant. A possible method to determine the proper sign for $\dot{\theta}_{i+1}$ is use Eqs. (4) and/or (5) to check for the best agreement between ± $\dot{\theta}_{i+1}$ and the measured accelerations. This is problematic in that the measured values in this case do not contain sign information for $\dot{\theta}_{i+1}$. Explicitly, there are two important situations where the accelerometer measurements contain no sign information. They are:

1) $\omega_i = \zeta$

2) If $\omega_i \neq \zeta$ but $\dot{\omega}_i = \zeta$, for the general accelerometer placement *c*, let $b_1 = \dot{\theta}_{i+1}\hat{z} \times (\Omega \times c)$ and $b_2 = \Omega \times (\dot{\theta}_{i+1}\hat{z} \times c)$. If b_1=b_2=0 or b_1+b_2=0, then the accelerometer measurements do not contain sign information.

Because R_i^{i+1} is a homogenous transform, $|\Omega|=|\omega_i|$. In this case, assuming r_x and $r_y >> \varepsilon$, the elements of Ω must be small, implying that the measured accelerations have little or no sign information for $\dot{\theta}_{i+1}$. Several techniques were considered to determine the proper sign. Section 4.6 will detail the best method found.

The discussion of the solution techniques for the three cases has dealt with the general case of r_x and $r_y \neq 0$. If the system is designed such that r_x or r_y is zero, the solution techniques become less complex. The following techniques can be used if $r_x=0$, similar solutions can be shown for $r_y=0$. For all cases of $r_x=0$:

$$\ddot{\theta}_{i+1} = \frac{-\tilde{A}_x}{r_y} \qquad (4.37)$$

Now, calculating $\dot{\theta}_{i+1}$ for each case:

4.4.4 Case 1 for r_x=0

The solution for case 1 is:

$$\dot{\theta}_{i+1} = \frac{\tilde{A}_z}{2\Omega_y r_y} \qquad (4.38)$$

4.4.5 Case 2 for r_x=0

In this case, there exists a quadratic equation in $\dot{\theta}_{i+1}$. Solving with the standard quadratic solution:

$$\dot{\theta}_{i+1} = \frac{\Omega_z r_y \pm \sqrt{r_y(\Omega_z r_y - \tilde{A}_y)}}{r_y} \qquad (4.39)$$

The proper root is selected by checking for consistency with measured accelerations in Eqs. (4.4) and /or (4.5). In contrast to case 3, there is sufficient information in the measured accelerations to determine the proper root. Note that in case 2 for r_x=0 (and similarly for r_y=0), the solution for $\dot{\theta}_{i+1}$ is not iterative as it is in general.

4.4.6 Case 3 for r_x=0

As in the general case 3, the solution has no explicit dependence on $\dot{\theta}_{i+1}$.

$$\dot{\theta}_{i+1}^{2} = \frac{-\tilde{A}_y}{r_y} \tag{4.40}$$

As before, the magnitude of $\dot{\theta}_{i+1}$ is calculated via a square root. As in the general case, determination of the sign is problematic. Refer to Section 4.6 for details. Once a solution for joint velocity and acceleration is found, Eqs. (4.1),(4.2),and (4.6) can be used to calculate a_{i+2}^{i+1}.

4.5 Solution for general *p*

In section 4.3, the special case of $p=\zeta$ was presented. With this assumption, the position calculation from section 4.3 could be applied to provide the position for the a_{i+2}^{i+1} calculation in section 4.4. With non-zero *p*, Eq. (4.8) and its solution in Eq. (4.14) are no longer valid. Now, the solution of θ_{i+1} is dependent on the angular acceleration and velocity of the $(i+1)^{th}$ link. If θ_{i+1} was known, the angular acceleration and velocity of the $(i+1)^{th}$ link calculated in the a_{i+2}^{i+1} solution could be used. Due to the interdependence of the solution methods in the general case, they cannot be used separately as unique solutions.

Three methods were investigated to solve the general problem:

1) Repeated calculation of the position and a_{i+2}^{i+1} solutions.
2) Steepest Descent
3) Nonlinear Least Squares

4.5.1 Repeated Calculation

In the repeated calculation technique, an approximate correction term for *A1* is first calculated using the previous sample's position, velocity, and acceleration of the $(i+1)^{th}$ joint as an approximation of the current value along with the current angular acceleration and velocity of the i^{th} link.

$$(\omega_{i+1}^{i+1})_{approx} = R^{i+1}{}_{i}\omega_i^i + (\dot{\theta}_{i+1})_{approx}\hat{z} \tag{4.41}$$

$$(\dot{\omega}_{i+1}^{i+1})_{approx} = R_i^{i+1}\dot{\omega}_i^i + R_i^{i+1}\omega_i^i \times (\dot{\theta}_{i+1})_{approx}\hat{z} + (\ddot{\theta}_{i+1})_{approx}\hat{z} \tag{4.42}$$

The approximate correction term becomes,

$$Acorr = (\dot{\omega}_{i+1}^{i+1})_{approx} \times p + (\omega_{i+1}^{i+1})_{approx} \times (\omega_{i+1}^{i+1})_{approx} \times p \tag{4.43}$$

$$A1corr = A1 - Acorr \tag{4.44}$$

The $A1corr$ term is used in the position solution from section 4.3 instead of $A1$ to obtain $(\theta_{i+1})_{approx}$. The $(\theta_{i+1})_{approx}$ is then used in R_i^{i+1} to obtain a better solution for the joint acceleration and velocity using the a_{i+2}^{i+1} solution. The process is repeated by calculating a new $Acorr$ using the new approximation of the joint velocity and acceleration instead of the value at the previous time.

4.5.2 Steepest Descent

This method uses the multivariable formulation of the steepest descent method [62] to solve for $\theta, \dot{\theta}, \ddot{\theta}$, and ω. The equations for ω are included to improve the joint velocity solution. The main equation for the steepest descent method is:

$$x(k+1) = x(k) + F'(x(k))^{+}(-F(x(k))) \tag{4.45}$$

Where $F(x(k))$ is the vector valued function to be solved, $F'(x(k))^{+}$ is the pseudoinverse of the gradient of $F(x(k))$, $x(k)$ is the current estimate of the value of x, and $x(k+1)$ is the next estimate of x. In this case, F and F' are:

$$F = \begin{bmatrix} BB\ddot{\theta} + (CC + EE + FF)\dot{\theta} + GG\dot{\theta}^2 + DD + A1 - A2 \\ R_i^{i+1}\omega_i^i + \dot{\theta}\hat{z} - \omega_{i+1}^{i+1} \end{bmatrix} \tag{4.46}$$

$$F' = \begin{bmatrix} \dot{\theta}(dC + dE + dF) + dD & CC + EE + FF + 2\dot{\theta}GG & BB & Z \\ \overline{R}_i^{i+1}\omega_i^i & \hat{z} & \zeta & -I \end{bmatrix} \tag{4.47}$$

Where,

$$\overline{R}_i^{i+1} = \frac{\partial R_i^{i+1}}{\partial \theta_{i+1}} = \begin{bmatrix} -\sin\theta_{i+1} & \cos\theta_{i+1}\cos\alpha_i & \cos\theta_{i+1}\sin\alpha_i \\ -\cos\theta_{i+1} & -\sin\theta_{i+1}\cos\alpha_i & -\sin\theta_{i+1}\sin\alpha_i \\ 0 & 0 & 0 \end{bmatrix} \tag{4.48}$$

$$dC = (\overline{R}_i^{i+1}\omega_i^i \times \hat{z}) \times r \tag{4.49}$$

$$dD = R_i^{i+1}\omega_i^i \times (\overline{R}_i^{i+1}\omega_i^i \times \hat{z}) + \overline{R}_i^{i+1}\omega_i^i \times (R_i^{i+1}\omega_i^i \times \hat{z}) + \overline{R}_i^{i+1}\dot{\omega}_i^i \times r \tag{4.50}$$

$$dE = \hat{z} \times (\overline{R}_i^{i+1}\omega_i^i \times r) \tag{4.51}$$

$$dF = \overline{R}_i^{i+1}\omega_i^i \times (\hat{z} \times r) \tag{4.52}$$

Additionally, Z is the 3x3 zero matrix and I is the 3x3 identity matrix. The vectors $dC, dD, dE,$ and dF are the partial derivatives with respect to θ_{i+1} of $CC, DD, EE,$ and FF respectively. This method is iterated until some desired tolerance on the value of F is achieved or a maximum number of iterations is reached. Unfortunately, due to errors in the angular acceleration and velocity of the i^{th} link and measurement errors in $A1$ and $A2$, the descent may not converge to the desired value even though the tolerance on F is achieved or the tolerance on F may not be achievable. As a result, some logic must be implemented to determine if the result of the descent is better or worse than the seed value. The seed value used for this solution method is one iteration of the repeated calculation method.

4.5.3 Nonlinear Least Squares

The nonlinear least squares (NLS) method is detailed in [63] and its mathematical formulation will not be presented here. Given the vector $G(x)$ which is a function of x, the goal is to minimize the value of $G(x)^T G(x)$. It also requires the gradient of $G(x)$, $G'(x)$. Although the $F(x)$ and $F'(x)$ from the previous section can be used, it was found in simulation that a slightly different formulation gave better results using the NLS technique. Instead of using $A2$-$A1$ to create a relative acceleration, a_{i+1}^{i} was transformed by R_i^{i+1} using the current estimate of θ_{i+1} and subtracted from the measured accelerations to produce six relative accelerations. Although this usage of a_{i+1}^{i} appears to alleviate the need for the second accelerometer in all solution methods, using a single accelerometer was found in simulation to produce unsatisfactory results due to the sensitivity of the solution to errors in the estimate of θ_{i+1}. The quantities $G(x)$ and $G'(x)$ are,

$$G = \begin{bmatrix} BB_p\ddot{\theta} + (CC_p + EE_p + FF_p)\dot{\theta} + GG_p\dot{\theta}^2 + DD_p + R_i^{i+1}a_{i+1}^{i} - A1 \\ BB_{r+p}\ddot{\theta} + (CC_{r+p} + EE_{r+p} + FF_{r+p})\dot{\theta} + GG_{r+p}\dot{\theta}^2 + DD_{r+p} + R_i^{i+1}a_{i+1}^{i} - A2 \\ R_i^{i+1}\omega_i^{i} + \dot{\theta}\hat{z} - \omega_{i+1}^{i+1} \end{bmatrix} \tag{4.53}$$

$$G' = \left[\begin{matrix} \dot{\theta}(dC_p + dE_p + dF_p) + dD_p + \dot{R}_i^{i+1}a_{i+1}^{i} \\ \dot{\theta}(dC_{r+p} + dE_{r+p} + dF_{r+p}) + dD_{r+p} + \dot{R}_i^{i+1}a_{i+1}^{i} \\ \dot{R}_i^{i+1}\omega_i^{i} \end{matrix}\right. \qquad \left.\begin{matrix} CC_p + EE_p + FF_p + 2\dot{\theta}GG_p & BB_p & Z \\ CC_{r+p} + EE_{r+p} + FF_{r+p} + 2\dot{\theta}GG_{r+p} & BB_{r+p} & Z \\ \hat{z} & \zeta & -I \end{matrix}\right] \tag{4.54}$$

4.5.4 Evaluation of solution methods

In simulation, all three solution techniques produced similar responses. As the sampling rate decreases, the steepest descent and NLS techniques produce better results than the repeated calculation technique. However, the repeated calculation technique is much less computationally complex. In real systems, the low computational complexity of the repeated calculation solution will allow sampling rates to be high enough so that its solution will be satisfactory.

4.6 Differential relationship

Until now, the differential relationship between $\theta_{i+1}, \dot{\theta}_{i+1}, \textit{and } \ddot{\theta}_{i+1}$ has been ignored. Utilizing this relationship would lead to a way to handle the cases where there is no sign information for $\dot{\theta}_{i+1}$. One method to exploit the differential relationship is to place an observer at the output of the position solution discussed in section 4.5. The position solution is modeled as a linear system expressed by

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx \end{aligned} \tag{4.55}$$

with state-space matrices:

$$x = \begin{bmatrix} \tilde{\theta}_{i+1} \\ \dot{\tilde{\theta}}_{i+1} \end{bmatrix}, \quad u = \begin{bmatrix} \theta_{i+1} \\ \ddot{\theta}_{i+1} \end{bmatrix}, \quad y = \dot{\theta}_{i+1}$$

$$A = \begin{bmatrix} -K_v & 1 \\ -K_k & 0 \end{bmatrix} \; B = \begin{bmatrix} K_v & 0 \\ K_k & 1 \end{bmatrix} \; C = \begin{bmatrix} 0 & 1 \end{bmatrix}$$

The states are joint position $\tilde{\theta}_{i+1}$ and velocity $\dot{\tilde{\theta}}_{i+1}$, the inputs are calculated joint position θ_{i+1} and acceleration $\ddot{\theta}_{i+1}$ from the solution in section 4.5, and the output is the filtered joint velocity, $\dot{\tilde{\theta}}_{i+1}$. The K_k scalar in Eq.(4.55) is used to compensate for possible calculated acceleration bias due to accelerometer sensor error. Without the K_k term, the system is an unstable double integrator and a bias in the calculated acceleration can lead to unbounded joint velocity. The K_k term can cause oscillations in steady state. The K_v term adds damping and can cancel out the oscillations. Equation (4.55) is used to smooth the calculated joint velocity, using the concept of an observer such as the Kalman filter. In the observer design, the calculated joint velocity is used as the observer input to correct the velocity state. For a real system, the observer gains could be selected using Kalman filter design techniques for error rejection.

Although the velocity observer was proposed as a method for dealing with the lack of sign information for the solution of $\dot{\theta}_{i+1}$, it has another use. In the presence of accelerometer bias and noise, the method for determining the sign of the solution for $\dot{\theta}_{i+1}$ suggested in section 4.5 is no longer the best choice even when information on the sign of the $\dot{\theta}_{i+1}$ solution is available. That method suggested comparing the calculated acceleration using both $\dot{\theta}_{i+1}$ solutions with a measured accelerometer output. A more robust method is to pick the solution for $\dot{\theta}_{i+1}$ that is closest to the previous output of the velocity observer. If neither solution is closer than some reasonable bound on the change in $\dot{\theta}_{i+1}$ over one sample period, use the previous velocity observer output as the current value of $\dot{\theta}_{i+1}$. If the calculated velocity from section 4.5 is not satisfactory due to sensor bias and noise, the plant in Eq. (4.55) may be used alone without the calculated joint velocity and observer formulation.

4.7 Response to accelerometer bias and noise

Using the position solution method from section 4.5, an error in *Acorr*, accelerometer bias, or accelerometer noise can be considered to be an additive offset. Given the form of the $\tan^{-1}$ function, that additive offset will lead to a bounded offset in θ_{i+1}. That joint offset, ϕ_{i+1}, will depend on the size of the bias, ξ, and the value of joint position. Although the ϕ_{i+1} is not constant, it is bounded. The resulting joint offset will be largest around θ_{i+1}=0 and will decay as θ_{i+1} approaches ±π.

$$\tan(\theta_{i+1} + \phi_{i+1}) = \tan(\theta_{i+1}) + \xi \tag{4.56}$$

$$\phi_{i+1} = \tan^{-1}\left(\frac{\xi}{\tan^2\theta_{i+1} + \xi\tan\theta_{i+1} + 1}\right) \tag{4.57}$$

An accelerometer bias will also lead to a bounded offset in the joint acceleration calculation. Considering the special case of r_x=0, the joint acceleration offset, $\ddot{\phi}_{i+1}$, is:

$$\ddot{\phi}_{i+1} = \frac{-\tilde{A}_{xerror}}{r_y} \tag{4.58}$$

Where $\tilde{A}_{xerror}$ is the difference between the measured and actual relative Cartesian acceleration along the x-axis. In the general case, $\ddot{\phi}_{i+1}$ also depends on $\dot{\theta}_{i+1}$. Again, examining the technique for the simplified case of r_x=0, the solution for the joint velocity offset, $\dot{\phi}_{i+1}$, for cases 1-3 is given in Eq. (4.59) through (4.61), respectively.

$$\dot{\phi}_{i+1} = \frac{\tilde{A}_{zerror}}{2\Omega_y r_y} \tag{4.59}$$

$$\dot{\phi}_{i+1} = -\Omega_z \pm \sqrt{\frac{-A_{yerror}}{r_y} + \Omega_z^2} \qquad (4.60)$$

$$\dot{\phi}_{i+1} = \pm\sqrt{\frac{-A_{yerror}}{r_y}} \qquad (4.61)$$

The error bounds in Eqs. (4.59-4.61) assume perfect information about Ω_z. Cumulative errors from failed joints previous in the kinematic chain will effect this assumption. Although cumulative errors in Ω_z will affect the magnitude of the error, the error will be bounded. It should be noted that the kinematic parameter r_y is involved with the effect of the error in all cases. Enlarging the parameter will lead to better error robustness.

Examining the A matrix in Eq. (4.55), choosing positive definite, diagonal K_k and K_v matrices is a sufficient condition for the observer to be asymptotically stable. With this choice of gains along with the error bound on the observer input from Eqs. (4.59-4.61), $\dot{\theta}_{i+1}$ is bounded leading to a bounded $\ddot{\phi}_{i+1}$ in the general case where both r_x and r_y are non-zero.

Further examination of Eq. (4.55) gives a steady state bound on the joint velocity offset, $\dot{\theta}_{i+1}^{ss}$, and joint position offset, θ_{i+1}^{ss}, due to the observer's plant dynamics. Let steady state be defined as:

$$\vec{0} = Ax + Bu \qquad (4.62)$$

Solving Eq.(4.59) for x leads to:

$$\dot{\theta}_{i+1}^{ss} = \frac{K_v^{i+1}}{K_k^{i+1}}\ddot{\phi}_{i+1} \qquad (4.63)$$

$$\theta_{i+1}^{ss} = \theta_{i+1} + \frac{1}{K_k^{i+1}}\ddot{\phi}_{i+1} \qquad (4.64)$$

where $K_v^{i+1}$and K_k^{i+1} are the $i+1^{th}$ elements of the diagonal of the diagonal observer gain matrices. In Eq. (4.63), it appears that the joint velocity offset can be made close to zero by choosing K_v^{i+1} very small and/or K_k^{i+1} very large. Choosing either gain in this manner can lead to oscillations depending on the size of $\ddot{\phi}_{i+1}$. As a result, the steady state assumption and Eq. (4.63) are invalid. Some adjustments can be made to $K_v^{i+1}$and K_k^{i+1} without affecting the steady state assumption giving the system designer some flexibility.

As stated earlier, the main advantage of the proposed method is that direct integration is not used to obtain position or velocity from acceleration. Integration can lead to large, unbounded errors due to bias and noise.

4.8 Simulation

Simulations were conducted to show the performance of the proposed solution technique. A Robotics Research 807i manipulator was simulated using Matlab and Simulink numerical simulation software. The 807i is a 0.8m long robot with seven revolute joints. The full 7 DOF configuration was implemented and tested in general 6 DOF space. Although position sensor recovery was simulated successfully on several other joints, joints 3 and 4 are presented in the simulation results. Joints 1 and 2 were actuated to provide an angular acceleration and velocity excitation for the 3rd and 4th frames, thus forcing solution cases 1 and 2 to be used along with case 3. Joints later in the kinematic chain have no effect on the solution (possible induced vibrations will be considered noise). Accelerometers were placed at reasonable locations along the robot links, i.e. the case of non-zero p and $r+p\neq l$ was simulated. Depending on the joint, either r_x or r_y were designed to be zero to use the simplified solution method. The commanded joint trajectory was a smooth trajectory lasting 1 second starting from [1 0 0 0 0 0 0] radians and ending at [0.5 -0.4 0.2 -0.25 -0.2 0.1 0.6] radians. Total simulation time was 7 seconds to show steady state performance. Kalman filter design techniques were used to determine the observer gains. The calculated position was not used to drive the robot simulation.

To show the performance of the method in the presence of constant accelerometer bias, a random bias was calculated at the start of the simulation and added to the accelerometer output. For the following simulations, the bias had a mean of 0.007m/s^2 and standard deviation of 0.024m/s^2. In comparison to raw accelerometer outputs in a 9.81m/s^2 gravity field, these bias values are small. However, as shown in Fig. 4.2, the bias is a substantial portion of the relative acceleration between accelerometers on the same link of a small robot moving along a reasonable trajectory. In steady state, the relative acceleration should be zero along all axes without bias.

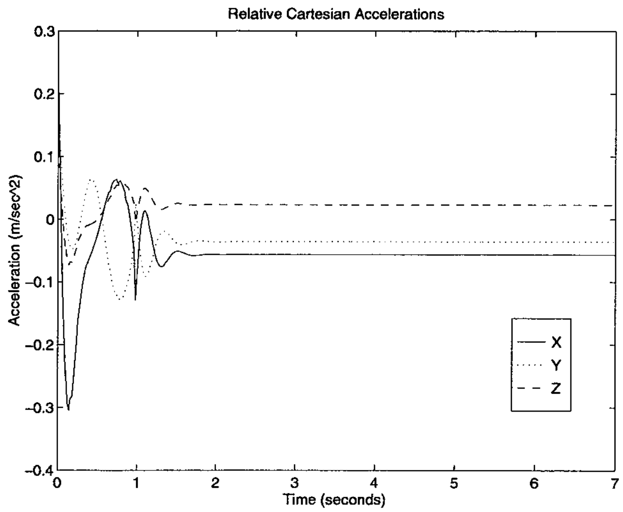


Figure 4.2: Relative acceleration of link 3 with accelerometer bias

The calculated joint position, velocity, and acceleration with accelerometer bias are shown in Figs. 4.3-4.5. Figures 4.4 and 4.5 show the velocity and acceleration of joint 4 which are affected by errors from the calculations for joint 3.

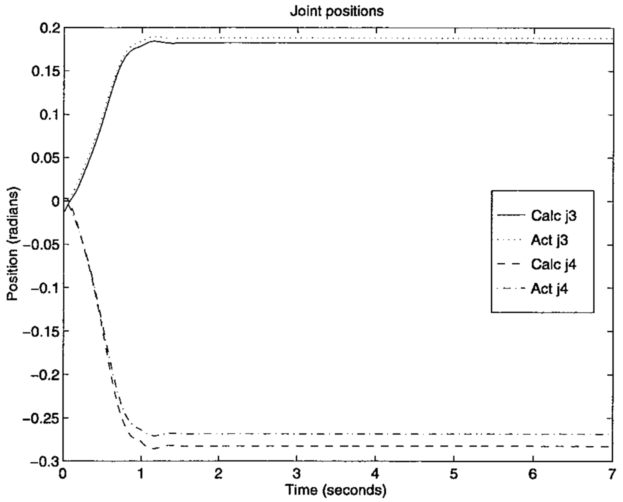


Figure 4.3: Calculated and actual positions with accelerometer bias

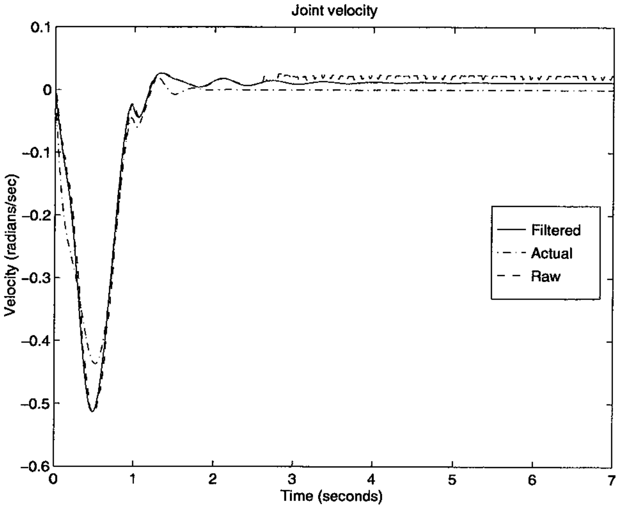


Figure 4.4: Calculated, observed, and actual velocity of joint 4 with accelerometer bias

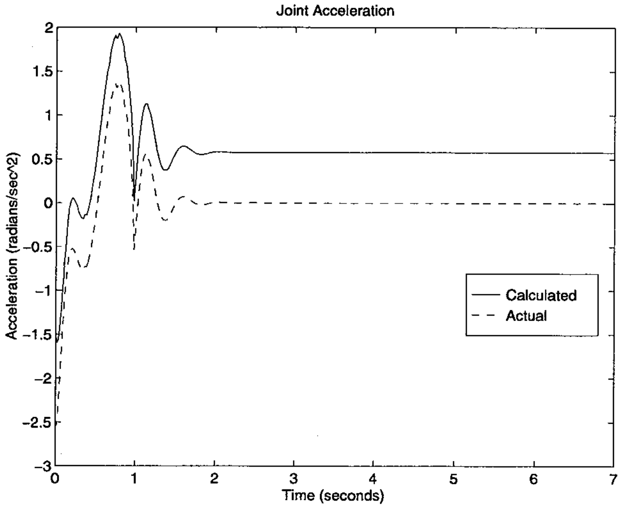


Figure 4.5: Calculated and actual acceleration of joint 4 with accelerometer bias

In the next simulation, sensor noise was added to the accelerometers along with the bias error. The added noise had a peak to peak amplitude of ~0.16m/s^2. This additive noise further contaminated the relative accelerations. The velocity and acceleration calculations in this simulation were dominated by noise. Even with the noisy velocity and acceleration, the position error shown in Fig. 4.6 is acceptable.

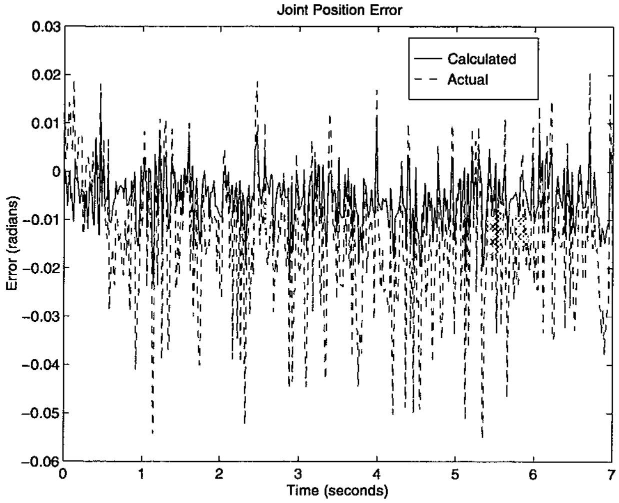


Figure 4.6: Position error with bias and noise

To show the advantage of this technique over integration based methods, the same accelerometer data which produced Fig. 4.6 was used in an integration based technique. The integration based technique used double integration to obtain Cartesian position of the end of the link attached to joint 4 from Cartesian accelerometer readings. The resulting Cartesian position is then compared to the Cartesian position calculated from forward kinematics[58] using the joint positions computed with the new methods presented earlier. As shown in Figs. 4.7 and 4.8, the solution method presented results in a bounded Cartesian position error, while the integration based solution resulted in an unbounded Cartesian error. Note that the measurements are in meters and the different scales on the plots. The results shown for the integration solution assumed that the gravity bias could be removed from the accelerometer data. During the failure of a position sensor, accurate removal would not be possible.

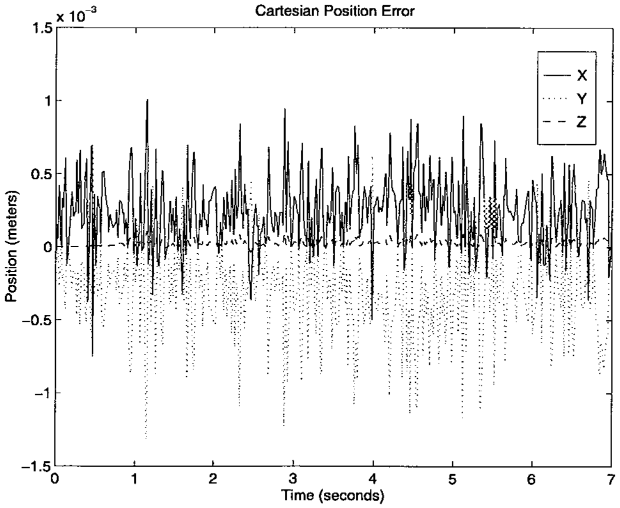


Figure 4.7: Error in Cartesian Position of Frame 4: Proposed Method

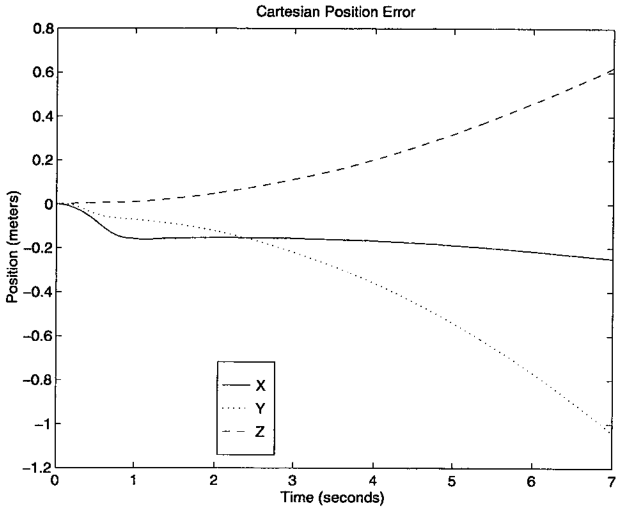


Figure 4.8: Error in Cartesian Position of Frame 4: Integration Method

Errors in the initial position will result in a bias on the integrated position solution. Other than changing the initial *Acorr* in the proposed solution method, it will have no effect.

4.9 Summary

This chapter has presented a technique to obtain joint position for a robot manipulator using Cartesian accelerometer information. A mathematical derivation for the technique based on kinematic analysis is shown and confirmed in simulation. The technique is valid for *n* DOF systems, handles general trajectories, allows reasonable accelerometer placement, and is not computationally complex. It has been shown in simulation to have bounded error performance due to bounded system measurement errors. With the advent of small, inexpensive, accurate accelerometers, the number of accelerometers required by this method is achievable. The technique allows placement of accelerometers in locations reasonable for retrofitting into existing robot designs.

This chapter has developed a theoretical foundation for using accelerometers to determine the position of a specific robot joint. However, the accuracy of this position using real sensor data and the usefulness of the calculated position in controlling a real joint are not discussed. In chapter 5, three joints of a real manipulator will be instrumented with accelerometers and practical implementation issues for this position determination method will be discussed. Experiments in chapter 5 will show the performance of this position determination method in both FDI and as part of a feedback control system in failure recovery.

5. Experimental Results: Joint Specific Position Sensor Fault Tolerance

The purpose of this chapter is to describe the experiments used to determine the performance of the joint specific position fault tolerance theory developed in chapter 4 on a real robot.

5.1 Experimental setup

The experimental apparatus used was based on the system used for the virtual passive torque sensor experiments in chapter 3. The Robotics Research 807i manipulator and its control system were modified as follows:

1. Communication between the primary robot control processor, a 33Mhz 68040 based computer, and a 25MHz 68040 based computer, to be referred to as the accelerometer processor, was enabled across the VME backplane.
2. Drivers for 32 lines of 16-bit analog to digital I/O (A/D) were coded and debugged.
3. Accelerometer mounts were designed, fabricated, and installed.
4. Electrical wiring and connectors for the accelerometers were installed.
5. Eighteen single axis accelerometers making up six triaxial accelerometers were installed on the mounts and electrically connected
6. MATLAB simulation code was ported to C++ to run on the accelerometer processor.

The accelerometer mounts can be seen in Fig. 5.1 and their placement on the manipulator in Figs. 5.2 and 5.3. The accelerometer mounts were designed to attach to existing screw holes in the manipulator. The mounts are attached to the shoulder pitch, elbow roll, and elbow pitch joints, joints 2, 3, and 4 respectively. Depending on the joint to which they are attached, they create an offset along either the local frame's x or y axis. This single offset, in addition to resulting in simple accelerometer mounts, allows for the use of the least complex solution method developed. It should be noted that while the accelerometer mounting structures shown are large and could easily be damaged in a real environment, they are only lab prototypes designed to require the smallest modification to the existing robot structure and use the accelerometers available. Smaller accelerometers are available which could be incorporated into the robot links.

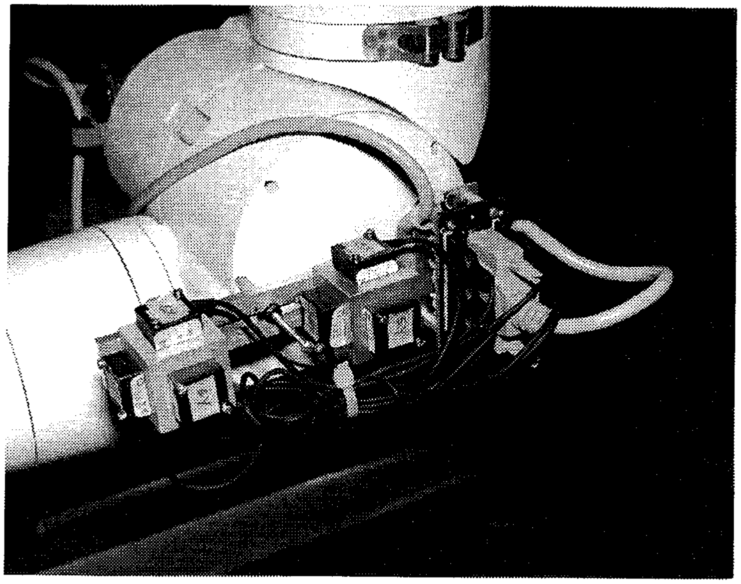


Figure 5.1: Accelerometer mounts

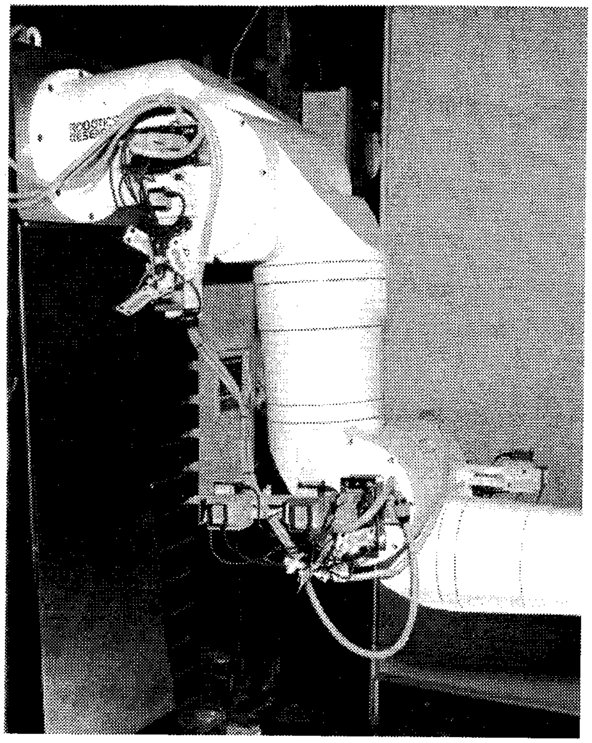


Figure 5.2: Robot side view

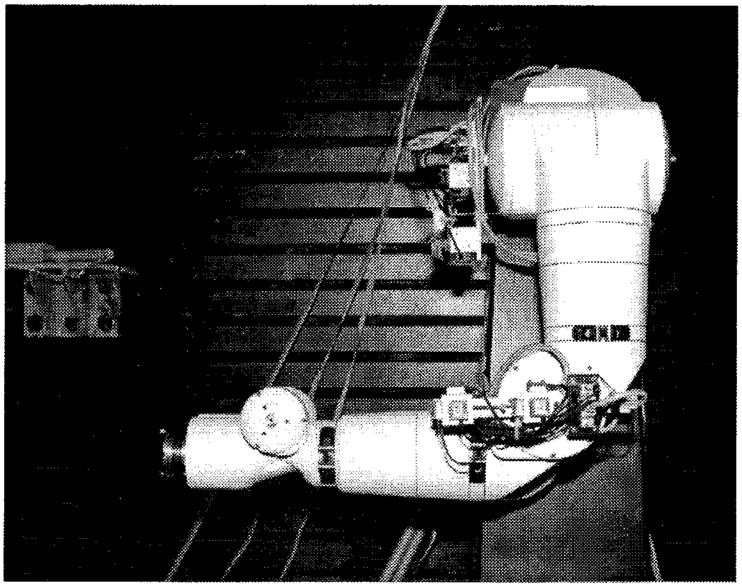


Figure 5.3: Robot front view with elbow roll rotated -π/2 from home position

The main problem encountered in the experimental setup was electromagnetic noise. The Robotics Research controller was the main source of this noise. When operational, the controller contaminated the local ground and radiated electromagnetic interference, especially when the PWM motor drives were enabled. As a result, the analog signals from the accelerometers were corrupted by up to 100mV noise spikes in the initial electrical configuration. The original configuration used single ended accelerometer signals (all signals used a common ground), some shielded and some unshielded cables, a single 40Hz analog anti-aliasing filter per signal, and one single ended A/D channel per signal. The configuration was refined to use differential accelerometer signals, as much shielded cable as possible, two 40Hz analog filters (one for each side of the differential signal), and a differential A/D channel (using two A/D channels per signal). This new electrical configuration eliminated most of the interference due to radiated noise. The contaminated ground could not be fixed. The Robotics Research controller's ground was connected to the VME card cages ground through the BIT3 bus-to-bus interface. With the refined electrical configuration, if the bus-to-bus interface cable was disconnected, the noise was less that 0.5mV peak-to-peak with the robot motors enabled. This noise level approached the noise specification for the 16-bit A/D board. When the cable was connected, the noise approached 8mV peak-to-peak. Without significantly changing the way robot communication was done, this problem could not be solved. A 10Hz digital filter was added to the system which cut the noise down to 3mV peak-to-peak. If the accelerometers chosen had been the traditional non-amplified type with approximately 40mV/G response, this noise would have been unacceptable. However, the accelerometers used have built-in amplification. Over their ±2G range, the accelerometer's response was 1V/G. This made the resulting noise ±0.0015G.

The final electrical configuration with its differential signals limited the scope of the experiment. Although three joints were instrumented with two triaxial accelerometers each for joint position determination, only a total of five triaxial accelerometers could be measured with the 32 A/D lines available. As a result, the experiments were limited to two simultaneous failures instead of the planned three.

Another problem encountered was variable accelerometer bias. Although the effect of noise and constant bias was simulated, the effect of noise and variable bias was not. This error type was experienced when two accelerometers, each with slightly different response characteristics, were used to calculate the relative acceleration required for joint velocity and acceleration calculations. Each accelerometer has a different zero offset and acceleration coefficient to convert volts to m/s^2. Although an automatic calibration program was developed which used linear regression to determine these parameters, it did involve some error. This error in relative acceleration, along with the noise being in phase or out of phase between accelerometers, caused a problem with relative acceleration calculations. To help alleviate the problem in steady state, a digital bias filter was included. This filter, a 0.1Hz low pass filter, attempted to determine the value of the difference between the signals caused by error without responding to the real difference caused by dynamic effects.

The noise and bias problems encountered in the system forced a change to the planned control system. The determination of joint velocity has always been problematic with this solution method. Although fair velocity magnitudes were available, the size of the relative acceleration error made sign determination unreliable. With the noise and bias encountered, the sign problem was too much for the velocity observer to correct. As a result, the velocity feedback part of the observer was eliminated and the differential model only used along with the calculated joint position and acceleration to determine joint velocity. In addition to better velocity response in this situation, removing the velocity feedback had the advantages of reducing the overall calculations required to determine velocity and reducing the number of accelerometers that would be required for some joint configurations. The reduction in accelerometers is dependent on the sensor configuration. This property was not taken advantage of during this experiment because the decision to use the model only velocity formulation was made after the accelerometers were installed. The disadvantage of removing the velocity observer component is that accurate velocity calculations are more dependent on tuning of model stiffness and damping terms to allow good tracking with minimal oscillation.

The required computational loading along with the sampling rate set by the 40Hz anti-aliasing filters determined the update rate of the accelerometer processor. The final configuration allowed accelerometer sampling and related filtering to execute at 100Hz and the position, velocity, and acceleration calculation to execute at 75Hz assuming two simultaneous joint calculations. Interrupt driven accelerometer sampling and better coding could increase the update rate.

5.2 Experimental Goals

The goal of this experiment is to show acceptable position tracking performance using the proposed solution method. This performance will be shown for both single and multiple joint failures. Two modes of operation will be considered. In the FDI (Failure Detection and Identification) mode, the joints will be considered unfailed and servoed using the Robotics Research position controller provided by the manufacturer. In this mode, the accelerometers will be read and the algorithm will attempt to calculate the joint position, velocity, and acceleration without feedback from the "failed" joint. This configuration will show the ability of the solution method to track actual measurements. In practice, the FDI mode could be used to determine if a joint position sensor has failed. In fault recovery mode, the calculated position and velocity data will be used along with the virtual passive torque control method developed in chapter 2 to servo the joint. In practice, the fault recovery mode would be enabled when a joint position sensor was determined to have failed.

Before the experimental results are discussed, it should be understood that the proposed method is a fault tolerant control method which uses traditional sensors for a non-traditional purpose. The method is not advocated as a primary control method. In practice, the position tracking performance of an accelerometer cannot match that of an optical encoder, resolver, or even a potentiometer. As a result, the servo performance will not be as good as one using traditional joint sensors and a basic control algorithm, much less a control method using accurate dynamic models. The goal is to show stable servo performance with adequate position response to make continued operation during failure an option with this method.

5.3 FDI Mode

In FDI mode, the joints are all servoed using measured position information but some joints are considered "failed" by the algorithm running on the accelerometer processor. Although the algorithm does not use the position or velocity feedback from the "failed" joints, it is available to the rest of the system and is presented here for comparison. This FDI mode is useful to detect joint position sensor failures. One joint at a time could be considered "failed" and its position calculated and compared to the actual measurement. Given a reasonable error bound, this method could be effective. Considering multiple joints "failed" simultaneously would have less relevance to FDI, but the response does help indicate the position tracking performance that can be expected when the fault recovery mode is enabled.

5.3.1 Single joint "failures"

The first joint considered is the shoulder pitch joint. It is expected to show the best performance of the instrumented joints because it has the least interesting dynamic effects on accelerometers attached to the joint. With this experiment's fixed base, only the velocity and acceleration for the shoulder roll joint contribute to the dynamic effects along with the shoulder pitch's own movements.

For all following plots, the Home position is defined as [0 -π/2 -π -π/2] radians for the first four joints. The three wrist joints will be fixed at [0 0 0] for all trajectories. Although most trajectories start from at or near the Home position, this is a convenience for comparison only. With exceptions that will be discussed and shown in the experimental results, this method has been tested over multiple trajectories with acceptable response. To better calibrate the accelerometers with the linear calibration method used, the calibration took place at points near the Home position. As the robot moves farther from the region around Home, calibration errors will increase. This problem could be resolved with a more sophisticated calibration algorithm.

The first data set involved moving from Home to [0 1.27 -π -π/2] in 1.5 seconds. This trajectory and all trajectories presented with a duration of the 1.5 seconds are too fast for a robot in an unstructured environment where fault tolerance is usually required. Fast trajectories were tested to examine the method's sensitivity to dynamic effects. Figures 5.4-5.6 show the position, velocity, and acceleration response.

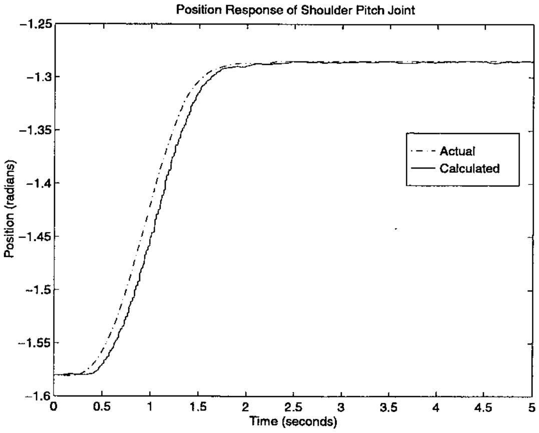


Figure 5.4: Position response of shoulder pitch joint to trajectory ending at [0 1.27 -π -π/2]

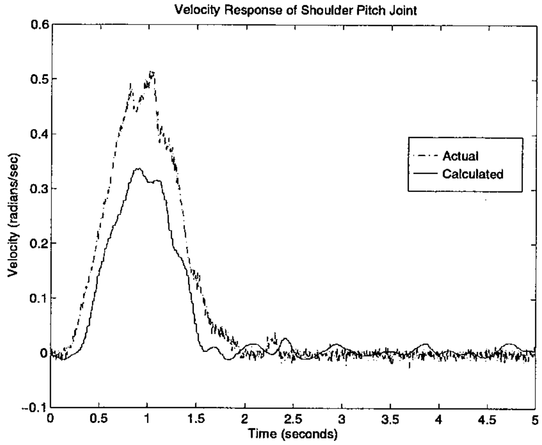


Figure 5.5: Velocity response of shoulder pitch joint to trajectory ending at [0 1.27 -π -π/2]

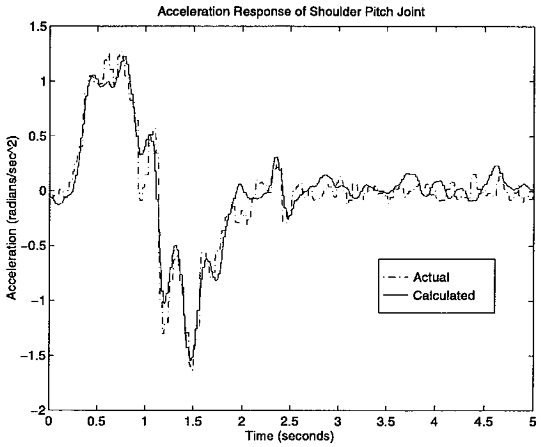


Figure 5.6: Acceleration response of shoulder pitch joint to trajectory ending at [0 1.27 -π -π/2]

In Fig. 5.4, the effects of the accelerometer response and filters are seen in the delay between the measured position and the calculated position. Part of this error is also due to the error in the magnitude of the calculated velocity which is used to form the correction term for the position calculation. The velocity error in Fig. 5.5 is due to the stiffness and damping terms in the differential model used to calculate velocity from the calculated position and acceleration. Decreasing the damping term will cause a damped oscillation in steady state but will allow for better tracking during movement. The joint acceleration tracks the "measured" acceleration well in Fig. 5.6 seemingly without the delay shown in the position tracking. This is because the joint acceleration is not directly measured in this system. The measured velocity is numerically differentiated and filtered by a lowpass filter with the same cutoff frequency as the accelerometer lowpass filters resulting in a similar delay.

The next data set shows the response from Home to [0.5 1.27 -π -π/2] in 1.5 seconds. This trajectory actuated the shoulder roll joint, the only other joint that can affect the shoulder pitch calculations. The result can be seen in Figs. 5.7-5.9.

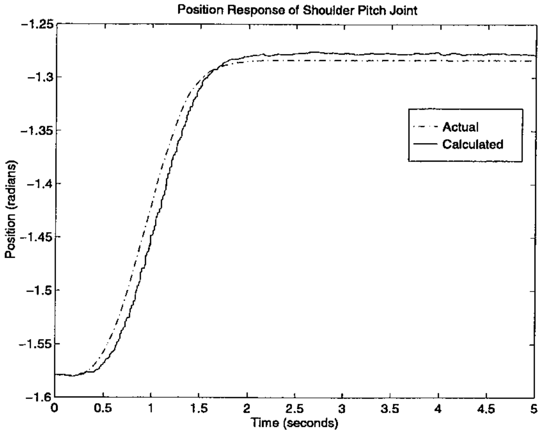


Figure 5.7: Position response of shoulder pitch joint to trajectory ending at [0.5 1.27 -π -π/2]

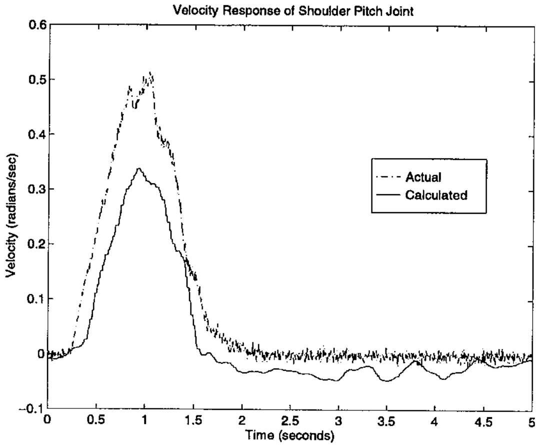


Figure 5.8: Velocity response of shoulder pitch joint to trajectory ending at [0.5 1.27 -π -π/2]

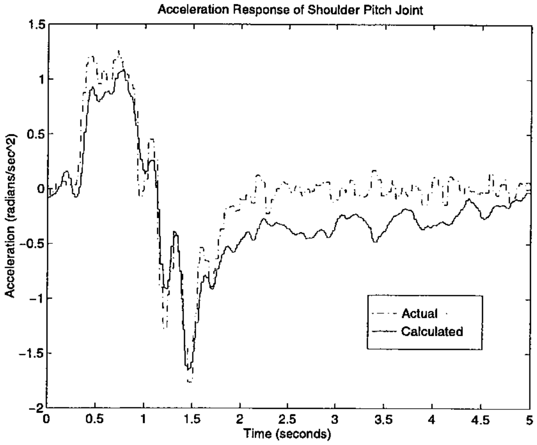


Figure 5.9: Acceleration response of shoulder pitch joint to trajectory ending at [0.5 1.27 -π -π/2]

As the robot moved farther from the Home position, the calibration error increased resulting in a larger steady state position error in Fig. 5.7 than in Fig. 5.4. The bias between the accelerometers also changed with position. The reduction of the steady state bias by the bias filter can be seen in the acceleration in Fig. 5.9.

Next a "failure" of the elbow roll joint is considered. The following plots will show one of the most significant problems with this solution method, its sensitivity to noise is dependent on the accelerometers' position in the gravity field. Figures 5.10-5.12 show the response to moving from Home to [0 -π/2 -3.5 -π/2] in 1.5 seconds. Although all calculated values track well, the noise on the position calculation in Fig. 5.10 is high. Figures 5.13 and 5.14 show the position response for trajectories from [0.1 -π/2 -π

-π/2] to [0.1 -π/2 -3.5 -π/2] and [0.5 -π/2 -π -π/2] to [0.5 -π/2 -3.5 -π/2] respectively. As the shoulder roll joint moves putting more gravity acceleration on the elbow roll's x and y accelerometer axes, the axes used in the position calculation, the signal to noise ratio increases and the calculated position becomes less noisy.

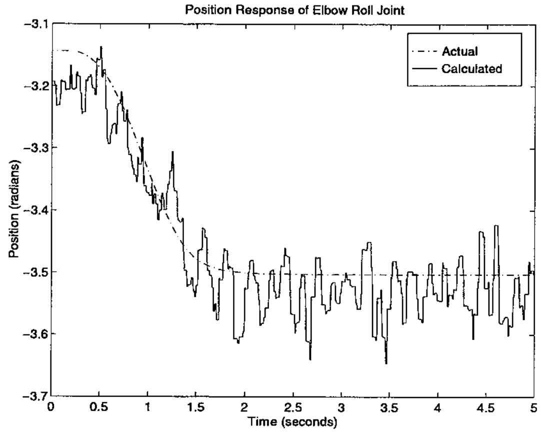


Figure 5.10: Position response of elbow roll joint to trajectory ending at [0 -π/2 -3.5 -π/2]

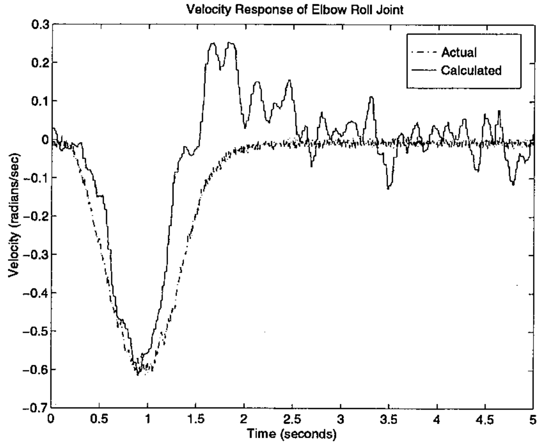


Figure 5.11: Velocity response of elbow roll joint to trajectory ending at [0 -π/2 -3.5 -π/2]

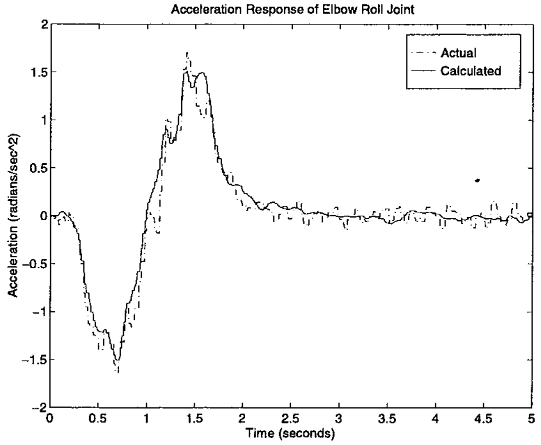


Figure 5.12: Acceleration response of elbow roll joint to trajectory ending at [0 -π/2 -3.5 -π/2]

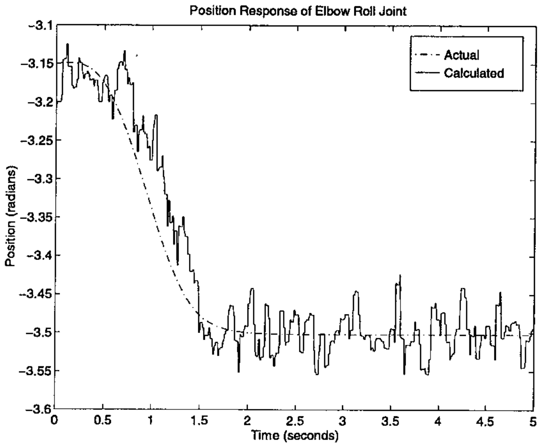


Figure 5.13: Position response of elbow roll joint to trajectory ending at [0.1 -π/2 -3.5 -π/2]

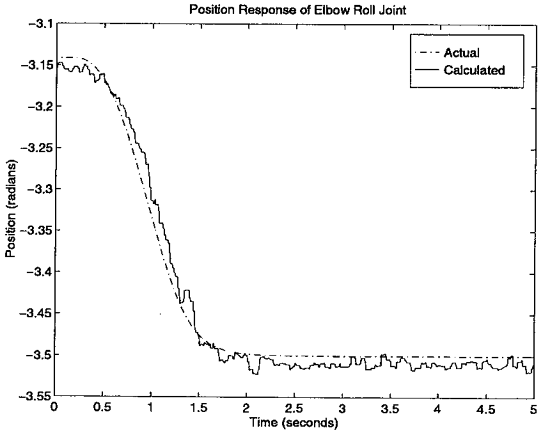


Figure 5.14: Position response of elbow roll joint to trajectory ending at [0.5 -π/2 -3.5 -π/2]

The next trajectory shows the effect of moving the shoulder roll and shoulder pitch joints on the elbow roll calculations. This trajectory runs from [0.3 -π/2 -π -π/2] to [0.5 -1.27 -3.5 -π/2] in 1.5 seconds. The results are shown in Figs. 5.15-5.17. The position response shows some errors of up to 0.12 radian during the trajectory and the acceleration response shows errors of up to 1.7 rad/s^2. These large errors with the shoulder joints moving shows an interaction between position and acceleration errors which is not seen when the shoulder joints are stationary. As shown earlier, the joint position calculation in this region is susceptible to noise. The calculated position term forms a rotation matrix which transforms the angular acceleration and velocity from the shoulder joints to the local frame. These terms are used in the joint acceleration calculation. Errors in these terms cause errors in the joint acceleration calculation which is then used to determine the next joint position. If the shoulder joints are not moving, they have no angular velocity and acceleration. As a result, the position error does not affect the acceleration calculation, as can be seen in the acceleration tracking in Fig. 5.12. Small synchronization errors between measured position, velocity, and Cartesian acceleration can also contribute to the error.

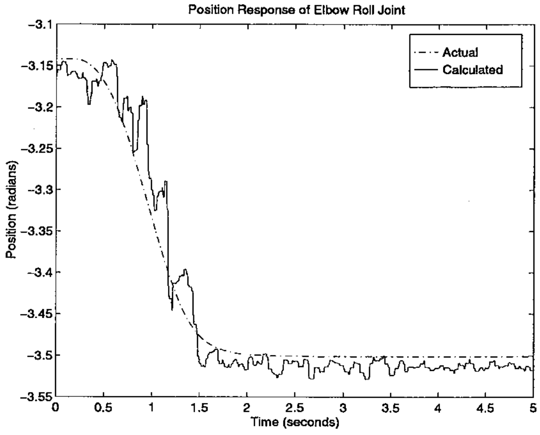


Figure 5.15: Position response of elbow roll joint to trajectory ending at [0.5 -1.27 -3.5 -π/2]

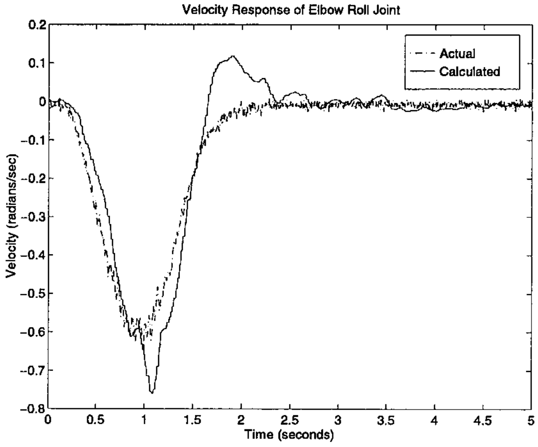


Figure 5.16: Velocity response of elbow roll joint to trajectory ending at [0.5 -1.27 -3.5 -π/2]

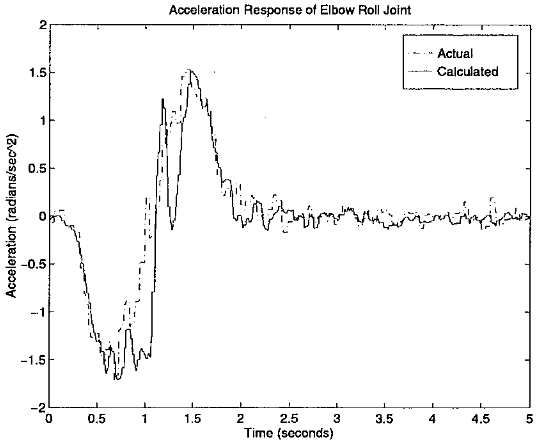


Figure 5.17: Acceleration response of elbow roll joint to trajectory ending at [0.5 -1.27 -3.5 -π/2]

The last single joint “failure” considered is the elbow pitch joint. The first trajectory is an elbow pitch joint only move from Home to [0 -π/2 -π -1.27] in 1.5 seconds. The results are shown in Figs. 5.18-5.20. The performance is similar to the shoulder pitch performance when the shoulder roll was fixed. In the Home configuration, the accelerometer axes on the elbow pitch joints are rotated π/2 radians from the shoulder pitch accelerometers. The performance shown by the elbow pitch joint when horizontal in this configuration reflects the shoulder pitch performance when horizontal and vice-versa (π/2 radians is vertical for the shoulder pitch joint in the Home configuration).

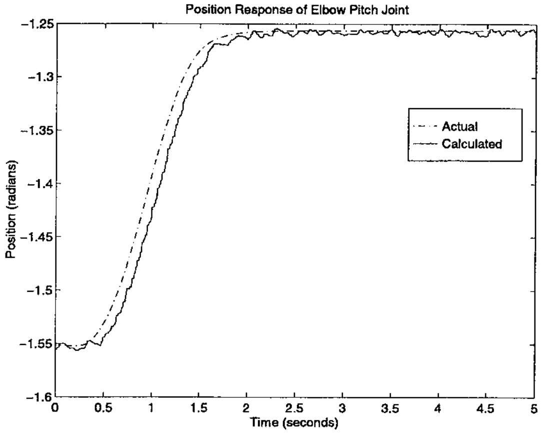


Figure 5.18: Position response of elbow pitch joint to trajectory ending at [0 -π/2 -π -1.27]

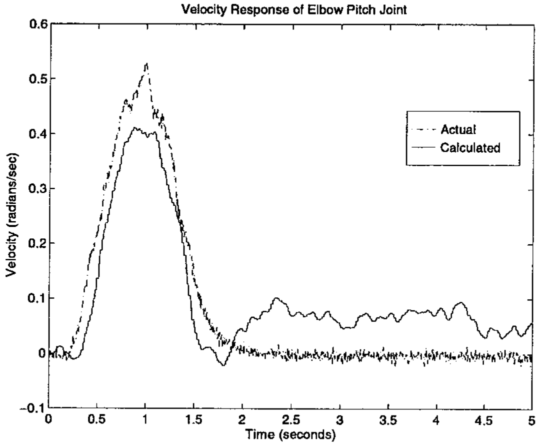


Figure 5.19: Velocity response of elbow pitch joint to trajectory ending at [0 -π/2 -π -1.27]

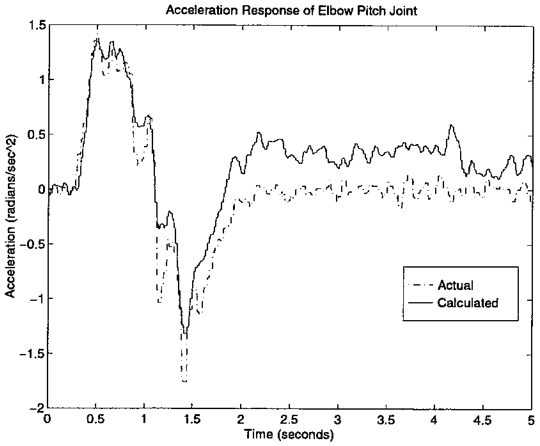


Figure 5.20: Acceleration response of elbow roll joint to trajectory ending at [0 -π/2 -π -1.27]

The next trajectory actuated all four joints. It ran from Home to [0.5 -1.27 -3.5 -1.27] in 1.5 seconds. The results are in Figs. 5.21-5.23. Note that although there is more tracking error than in Figs. 5.18-5.20, the error is not as severe as the elbow roll's error in Figs. 5.15-5.17. This is because of the elbow pitch accelerometers' orientation to the gravity field. If the shoulder roll position was around ±π/2 instead of around zero, the noise sensitivity of the elbow roll and pitch joints would be reversed.

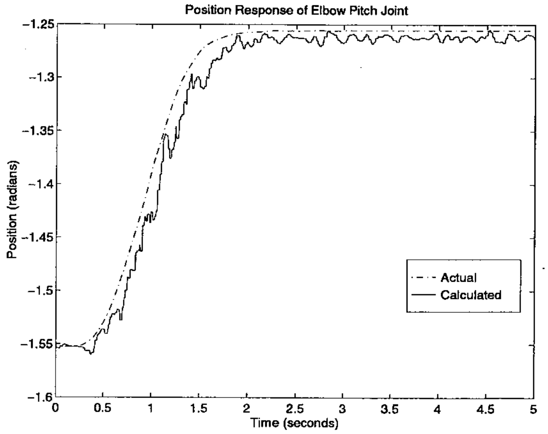


Figure 5.21: Position response of elbow pitch joint to trajectory ending at [0.5 -1.27 -3.5 -1.27]

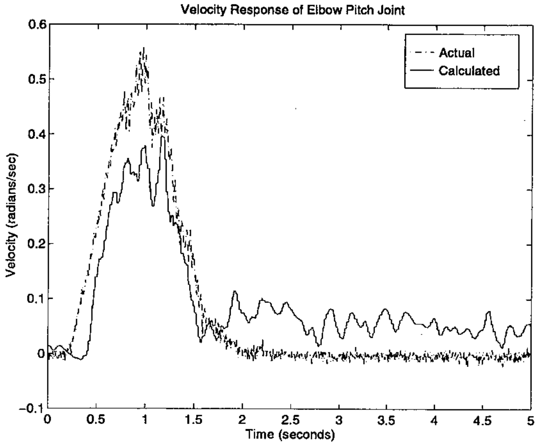


Figure 5.22: Velocity response of elbow pitch joint to trajectory ending at [0.5 -1.27 -3.5 -1.27]

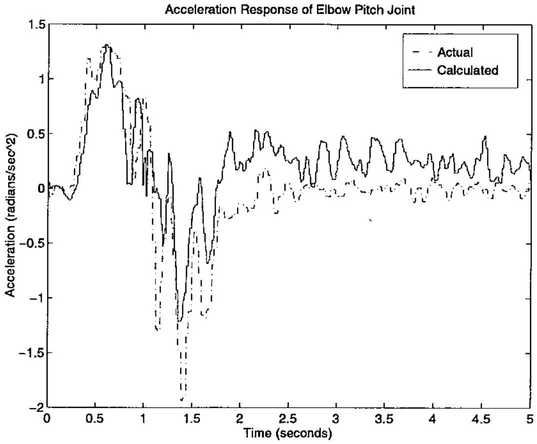


Figure 5.23: Acceleration response of elbow roll joint to trajectory ending at [0.5 -1.27 -3.5 -1.27]

5.3.2 Multiple joint "failures"

Multiple joint "failures" are interesting in that they show the effect of accumulating errors in joint position, velocity, and acceleration calculations. Since this method assumes a serial link manipulator and uses information from joints earlier in the kinematic chain, any error in a previous joint will contribute to the error in the current joint.

The first of two multiple joint "failure" cases considered is the simultaneous failure of the shoulder pitch and elbow roll joints. Figures 5.24-5.26 show the response of the elbow roll joint to the trajectory from

[0.3 -π/2 -π -π/2] to [0.5 -1.27 -3.5 -π/2] in 1.5 seconds. The response to this trajectory using measured data from the shoulder roll joint can be seen in Figs. 5.15-5.17. The multiple joint failure case appears slightly better than the single joint failure case. This implies that noise or bias was less severe during this run or a better accelerometer calibration was used. The plots for the response of the shoulder pitch joint are not presented as they are similar to the plots in Figs. 5.7-5.9. Using the proposed solution method, failures of joints later in the kinematic chain do not effect the calculations of joints before them.

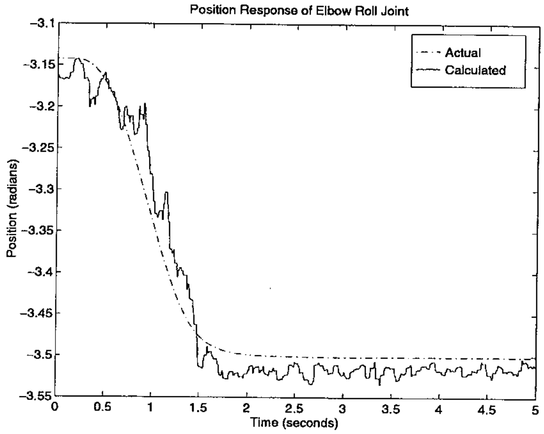


Figure 5.24: Position response of elbow roll joint to trajectory ending at [0.5 -1.27 -3.5 -π/2]

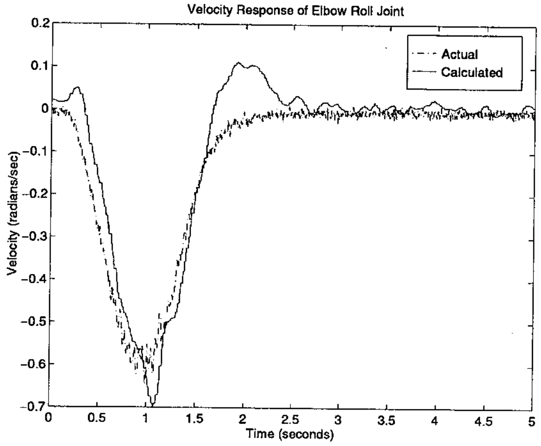


Figure 5.25: Velocity response of elbow roll joint to trajectory ending at [0.5 -1.27 -3.5 -π/2]

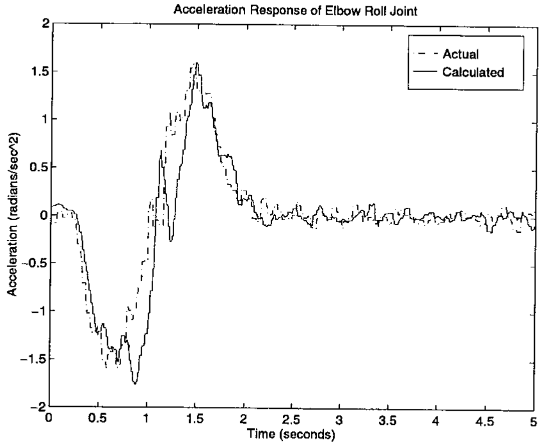


Figure 5.26: Acceleration response of elbow roll joint to trajectory ending at [0.5 -1.27 -3.5 -π/2]

If the tracking performance must be improved the trajectory can be slowed. Figure 5.27 shows the position response to the same trajectory executed in 4.0 seconds. The lower speed reduces the coupling of position and acceleration errors. To show that other trajectories are possible in this multiple failure mode, the position response resulting from executing the trajectory from [0.3 -π/2 -π -π/2] to [0.5 -1.77 -2.7 -π/2] in 1.0 seconds is shown in Fig. 5.28. The increased transient error is due to the higher speed of the trajectory and the steady state error due to accelerometer calibration.

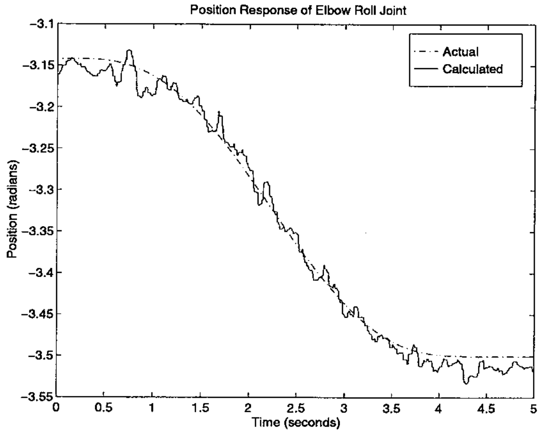


Figure 5.27: Position response of elbow roll joint to trajectory ending at [0.5 -1.27 -3.5 -π/2] in 4.0 seconds

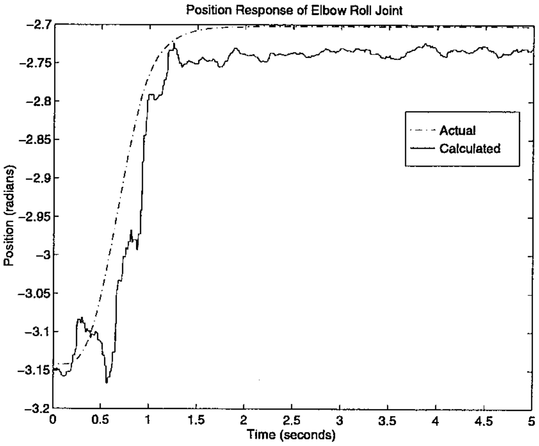


Figure 5.28: Position response of elbow roll joint to trajectory ending at [0.5 -1.77 -2.7 -π/2] in 1.0 second

The second multiple joint failure considered involves the shoulder pitch and elbow pitch joints. Figures 5.29-5.31 show the response of the elbow pitch joint to the trajectory from Home to [0.5 -1.27 -3.5 -1.27] in 1.5 seconds. The same trajectory is shown in Figs. 5.21-5.23 with the shoulder pitch joint assumed not failed. Similar tracking performance is shown to the non-failed condition.

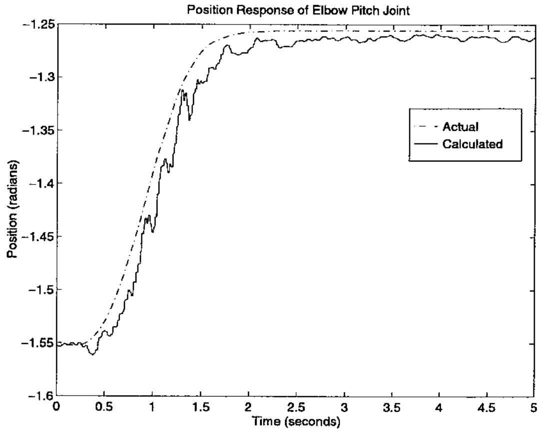


Figure 5.29: Position response of elbow pitch joint to trajectory ending at [0.5 -1.27 -3.5 -1.27]

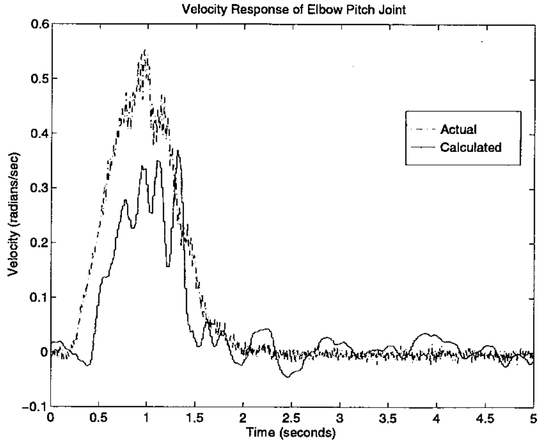


Figure 5.30: Velocity response of elbow pitch joint to trajectory ending at [0.5 -1.27 -3.5 -1.27]

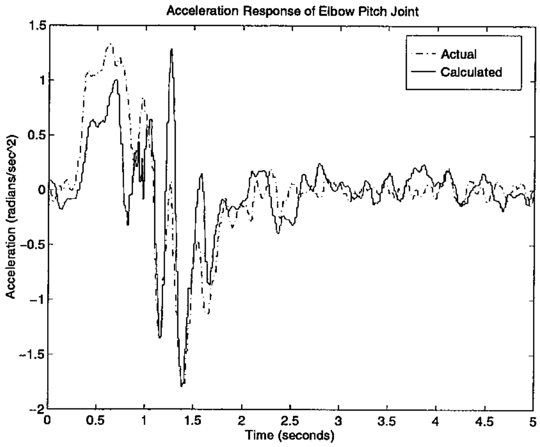


Figure 5.31: Acceleration response of elbow roll joint to trajectory ending at [0.5 -1.27 -3.5 -1.27]

To show the effects of speeding up and slowing down this trajectory, refer to Figs. 5.33 and 5.35 which show the position response to trajectory from Home to [0.5 -1.27 -3.5 -1.27] executed in 0.75 and 4.0 seconds, respectively. Since the response of the shoulder pitch joint to these trajectories has not been presented, its position response is included in Figs. 5.32 and 5.34. The tracking improved in the slower case but degraded in faster case. It should be noted that the improvement and degradation were not as severe as in the case of the elbow roll.

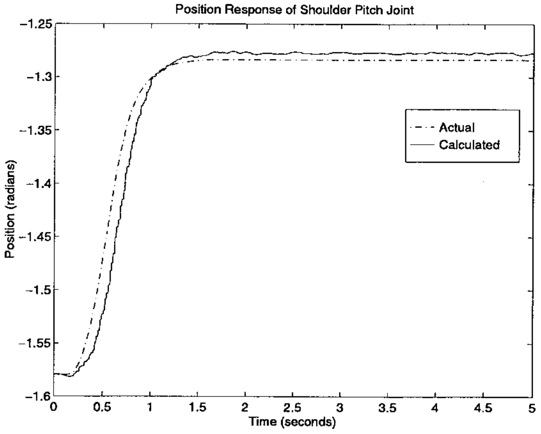


Figure 5.32: Position response of shoulder pitch joint to trajectory ending at [0.5 -1.27 -3.5 -1.27] in 0.75 seconds

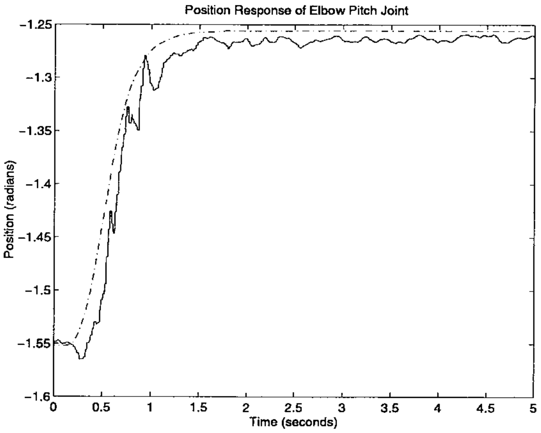


Figure 5.33: Position response of elbow pitch joint to trajectory ending at [0.5 -1.27 -3.5 -1.27] in 0.75 seconds

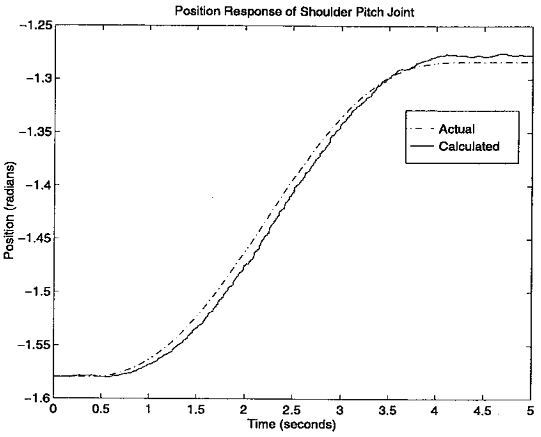


Figure 5.34: Position response of shoulder pitch joint to trajectory ending at [0.5 -1.27 -3.5 -1.27] in 4.0 seconds.

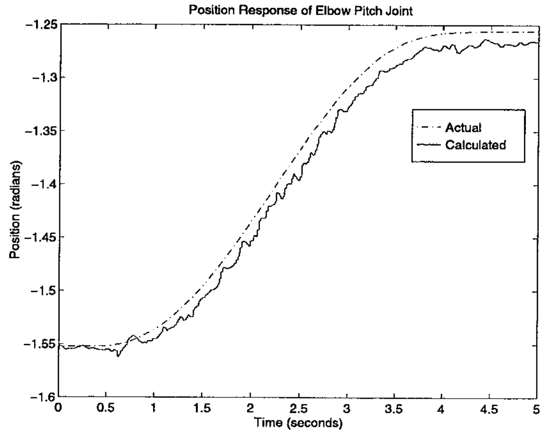


Figure 5.35: Position response of elbow pitch joint to trajectory ending at [0.5 -1.27 -3.5 -1.27] in 4.0 seconds

To show response to different trajectories, trajectories from Home to [1 -1 -4 -1] and from Home to [-0.5 -1.77 -2.6 -1.17] in 1.5 seconds were executed. The position responses of the shoulder pitch and elbow pitch joints are found in Figs. 5.36-5.39. The steady state tracking error of the shoulder pitch joint in Fig. 5.36 is due to calibration error.

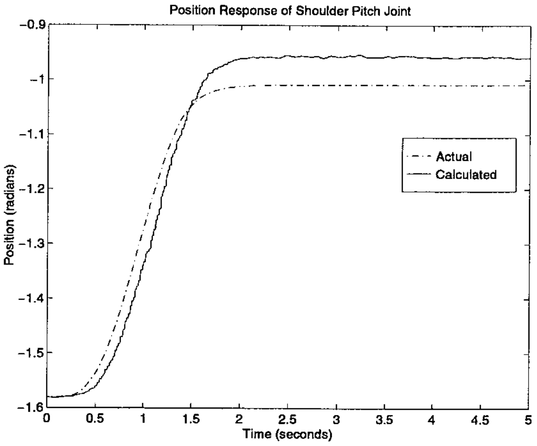


Figure 5.36: Position response of shoulder pitch joint to trajectory ending at [1 -1 -4 -1]

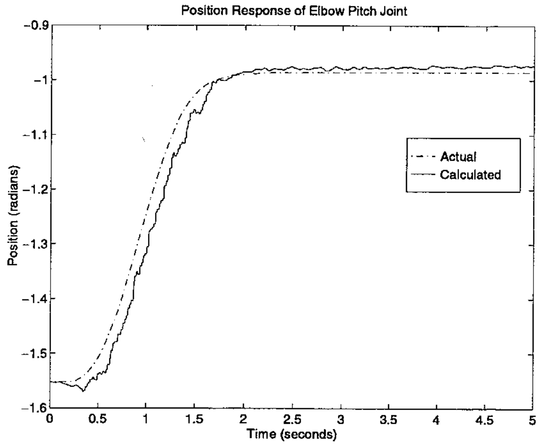


Figure 5.37: Position response of elbow pitch joint to trajectory ending at [1 -1 -4 -1]

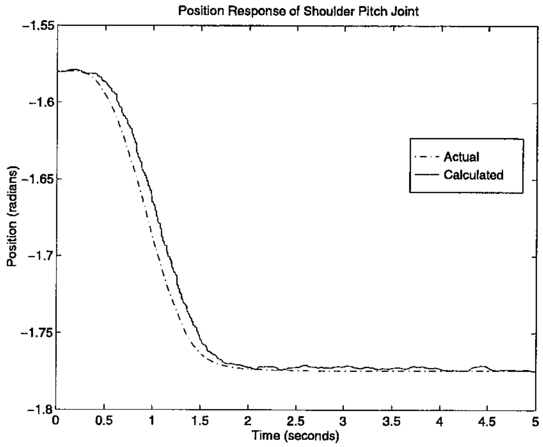


Figure 5.38: Position response of shoulder pitch joint to trajectory ending at [-0.5 -1.77 -2.6 -1.17]

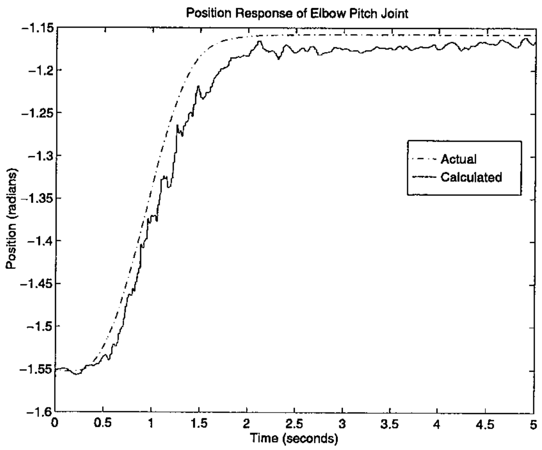


Figure 5.39: Position response of elbow pitch joint to trajectory ending at [-0.5 -1.77 -2.6 -1.17]

Although not shown, the multiple failure of the elbow roll and elbow pitch joints is calculable with this method. The poor performance on the elbow roll in some configurations would degrade the elbow pitch joint's performance. For practical situations, trajectories in this multiple failure mode would need to be slowed, the trajectories limited to areas where both the elbow pitch and elbow roll joints were not as susceptible to noise, or only one joint moved at a time. This problem is not unique to the elbow roll joint. All joints will have configurations where their responses are more susceptible to noise. These configurations can be predicted by examining the accelerometers' orientation with respect to gravity. If both the x and y axes have small signal to noise ratios due to their orientation, tracking will be less accurate.

5.4 Failure recovery

The previous sections have concentrated on performance in the FDI mode where the data calculated from the accelerometer outputs was not used to servo the joints. In this section, the calculated data will be used as feedback for the joint servo loops. The goal of this section is to show that the calculated data can be used to stably control one or two failed joints. A secondary goal is to match the servo performance provided by the Robotics Research position controller provided with the system. The Robotics Research controller uses a simple PD controller with gravity offset in addition to its low level torque controller for position control. It is not an optimal controller but is acceptable for general use. Since the proposed position feedback method is not optimal, the resulting control cannot be expected to track perfectly. The goal is to make it acceptable for emergency use.

Due to the noise and delay evident on some of the calculated data, a position controller with very high gains would lead to unacceptable performance due to vibration or instability. In addition, since a gravity offset algorithm is dependent on position, any position error would lead to a torque offset error. This situation is an good case to use the virtual passive torque controller detailed in chapters 2 and 3. The virtual passive controller will provide both gravity and dynamic compensation so the position gains can be lowered. The virtual passive controller can also be tuned to provide adjustable joint damping to trade tracking error for vibration suppression. The use of the virtual passive torque controller assumes that although the joint position and velocity sensors have failed, the joint torque sensor is operational.

This application will show the stability of virtual passive control on the major arm joints of a robot. In chapter 3, a detailed analysis of the properties of the control technique was done concentrating on the wrist joints. In this section, the shoulder pitch and elbow pitch joints will be controlled by a virtual passive based controller. The position controller will be a simple PD based controller which uses the calculated position and velocity with the trajectory command to offset the measured torque used by the virtual passive controller. This offset appears to the virtual passive controller as movement. The virtual passive controller moves the joint in the desired direction to dissipate the energy. Results for the failure of elbow roll joint are not presented because a real joint torque sensor failure was discovered after the experiment began.

5.4.1 Single joint failure recovery

The first single failure considered is the shoulder pitch joint. The trajectory from Home to [0.5 1.27 -π -π/2] in 1.5 seconds was executed and the position response shown in Fig. 5.40. The response to this trajectory in FDI mode can be found in Figs. 5.7-5.9. The measured joint position is given for comparison but was not used by the controller. This joint failure showed a very good FDI response and the calculated position continued to track the measured position well during failure recovery. The small overshoot hump in the trajectory is adjustable. The delay, actually servo error, between commanded and actual trajectory can be decreased by increasing the position gains at the expense increasing the overshoot. This tradeoff is characteristic of PD based controllers. The size of this overshoot is also determined by the calculated position tracking error. If the error is such that the controller believes that it must continue along at a high velocity instead of slowing to a stop, the controller will overshoot the proper stopping point. When the calculated position comes close enough to the desired position, the controller will slow the joint and attempt to correct the overshoot but, at this point, the overshoot has already occurred. The offset between commanded and actual at the beginning of the trajectory is a result of using a real controller to move to the starting point before the trajectory was run. For reference, the same trajectory executed by the Robotics Research controller is shown in Fig. 5.41. The characteristic of the Robotics Research controller is this it trades a "delay" due to servo error for a more critically damped performance.

The gains for the failure recovery case were tuned to give similar "delay" due to servo error at this trajectory duration. Slightly different starting points are the result of using different controllers to move to the starting point form different positions.

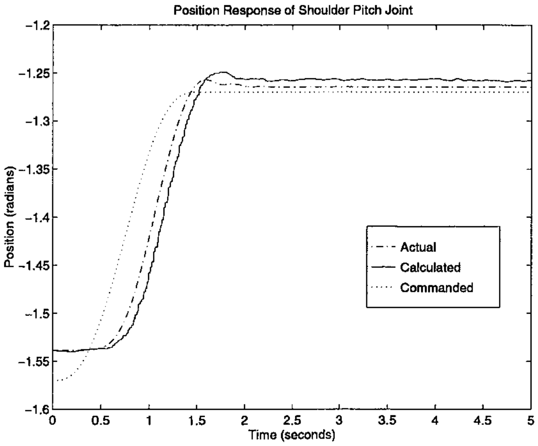


Figure 5.40: Position response of shoulder pitch joint to trajectory ending at to [0.5 1.27 -π -π/2]

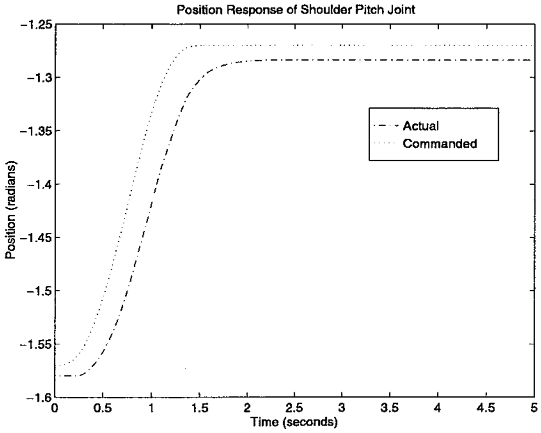


Figure 5.41: Position response of shoulder pitch joint to trajectory ending at to [0.5 1.27 -π -π/2] using Robotics Research controller

The second single joint failure considered is the elbow pitch joint. This joint is a possibly the best candidate for fault tolerance using this method for this robot or any other single elbow, articulated design. In a single elbow articulated robot, even a kinematically redundant one with joint offsets like the Robotics Research robot used, the elbow pitch is the only joint which significantly affects the distance between the shoulder center and the wrist center. If this joint fails, the robots utility is greatly reduced. Adding redundant position sensors to shoulder joints is fairly easy due to the joints usually large size and lack of a weight penalty in dynamic performance assuming a fixed base. However, the elbow pitch joint is usually one of the most optimized designs in robot construction. The elbow pitch joint must carry significant loads but must be small enough to keep the shoulder joint sizes realistic. The light weight of the accelerometers used in this method along with not requiring a change in current joint designs makes this method desirable.

The first trajectory considered is a elbow pitch only move from Home to [0 -π/2 -π -1.27] in 1.5 seconds. The overshoot shown in Fig. 5.42 is due to the interaction of low position gains and the virtual passive controller. This is a worst case move in the gravity field for the virtual passive controller to compensate. It should be noted that the Robotics Research controller response shown in Figs. 5.43 and 5.45 also overshot. Higher gains on either the virtual passive controller or the PD position controller could reduce the overshot due to gravity but with the noisy position data both would cause vibration. A combination of low position controller gains and higher virtual passive controller gains would correct the overshoot but would increase the servo error "delay" during the trajectory. The second trajectory actuated all four joints from Home to [0.5 -1.27 -3.5 -1.27] in 1.5 seconds. The response in Fig. 5.44 shows that even with noisier calculated position data the position still tracked smoothly. If better steady state performance is required, a PID controller can be used with the usual tradeoff in transient performance.

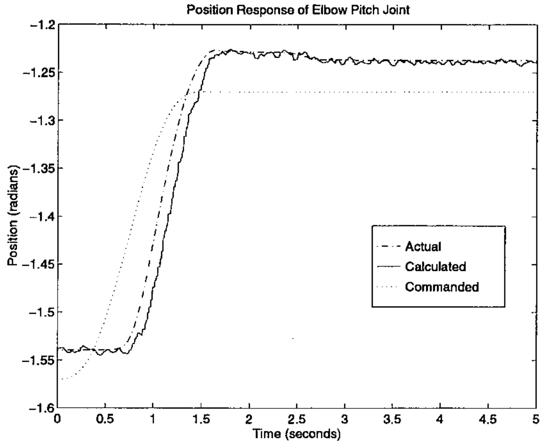


Figure 5.42: Position response of elbow pitch joint to trajectory ending at to [0 -π/2 -π -1.27]

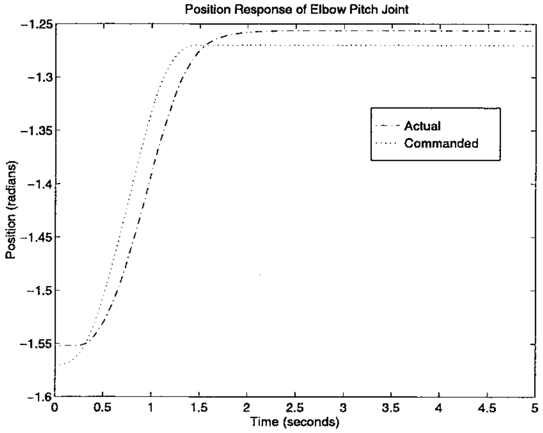


Figure 5.43: Position response of elbow pitch joint to trajectory ending at to [0 -π/2 -π -1.27] using Robotics Research controller

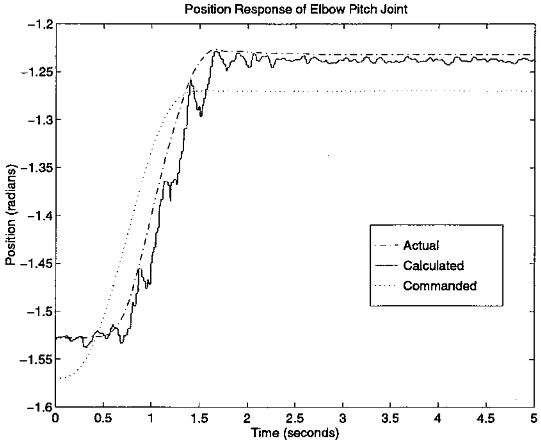


Figure 5.44: Position response of elbow pitch joint to trajectory ending at to [0.5 -1.27 -3.5 -1.27]

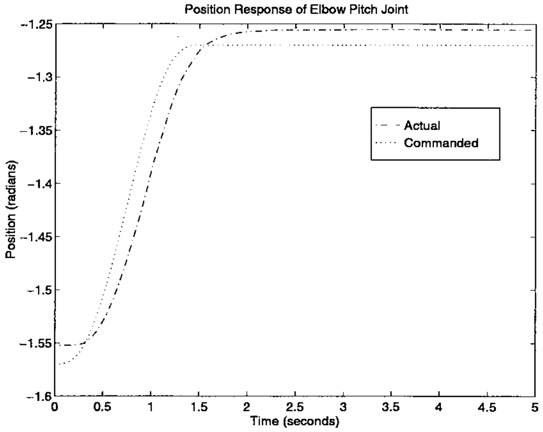


Figure 5.45: Position response of elbow pitch joint to trajectory ending at to [0.5 -1.27 -3.5 -1.27] using Robotics Research controller

The last trajectory for the failure of only the elbow pitch joint presented is from Home to [1 -1 -4 -1] in 1.5 seconds. This is a very extreme move for a robot to make in normal operation. It covers a significant region of the workspace at average speeds of up to 0.5 rad/s. The resulting plot shown in Fig. 5.46 shows a significant overshoot. As speeds continue to increase, this overshoot will continue to increase with this gain set. Gains can be lowered to lessen overshoot with the penalty of degrading performance at lower speeds. To further explain the relationship between calculated position error and overshoot, it can be seen that the point where the calculated position crosses the commanded position at approximately 1.4 seconds is approximately the point of maximum overshoot on the actual joint trajectory. The steady state performance on this trajectory is better because the elbow joint is not as heavily gravity loaded and, ironically, because the overshoot is large. The energy caused by the position controller in attempting to correct the overshoot is sufficient to overcome the stiffness of the passive controller and better approach the desired position. It should be noted that the calculated position continues to track the measured position during the overshoot.

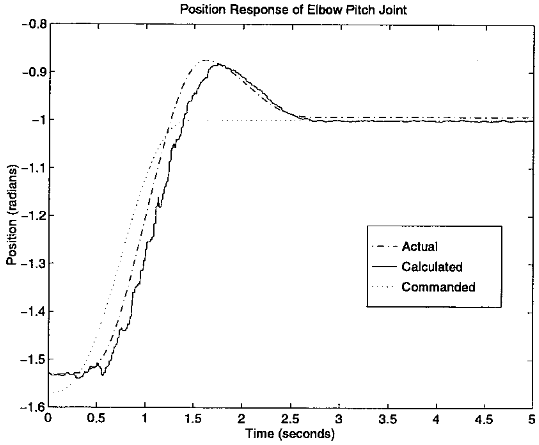


Figure 5.46: Position response of elbow pitch joint to trajectory ending at to [1 -1 -4 -1]

5.4.2 Multiple joint failure recovery

As in the FDI case of multiple joint "failure", calculation errors in earlier joints will effect the calculations required by joints later in the kinematic chain. During multiple joint fault recovery, the control system will have to compensate for servo errors and possible control interaction between joints. The multiple failure case considered is the simultaneous failure of the shoulder pitch and elbow pitch joints. To show the results of any controller interaction, position responses of both joints will be provided for all trajectories in this section.

The first trajectory considered is from Home to [0.5 -1.27 -3.5 -1.27] in 1.5 seconds. The response shown in Figs. 5.47-5.48 show similar response to the single failure cases.

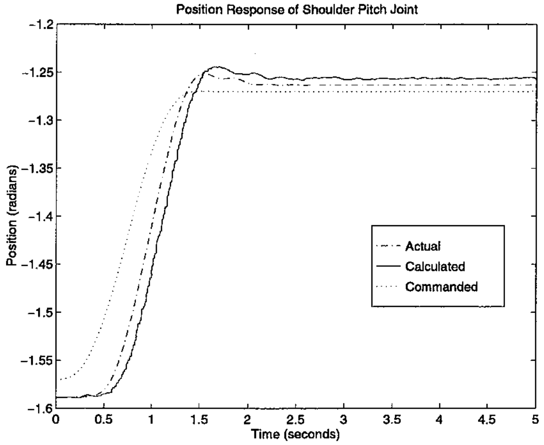


Figure 5.47: Position response of shoulder pitch joint to trajectory ending at [0.5 -1.27 -3.5 -1.27] in 1.5 seconds.

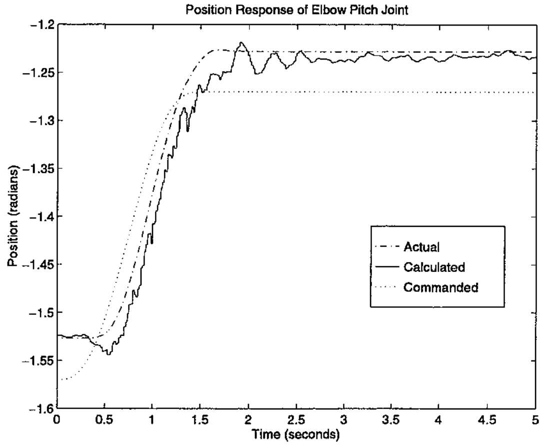


Figure 5.48: Position response of elbow pitch joint to trajectory ending at [0.5 -1.27 -3.5 -1.27] in 1.5 seconds

To show the effects of increasing and decreasing trajectory execution times, the same trajectory is executed in 0.75 seconds and 4.0 seconds. The results of these trajectories are shown in Figs 5.49-5.50 and Figs 5.51-5.52, respectively. In the faster trajectory, performance is similar to the 1.5 second long trajectory with some increased overshoot. The slower trajectory shows the stiffness of the virtual passive controller. The shoulder pitch joint which tracked the faster trajectory well undershot the desired trajectory while the elbow pitch which overshot in the faster trajectory tracked the desired trajectory in steady state. With the slower trajectory, the position controller did not have as much authority as it had in the faster trajectories because the difference between commanded and calculated positions did not grow as rapidly. As a result, the stiffness of the virtual passive controller did not allow either joint to move as much. The stiffness also increased the "delay" due to servo error.

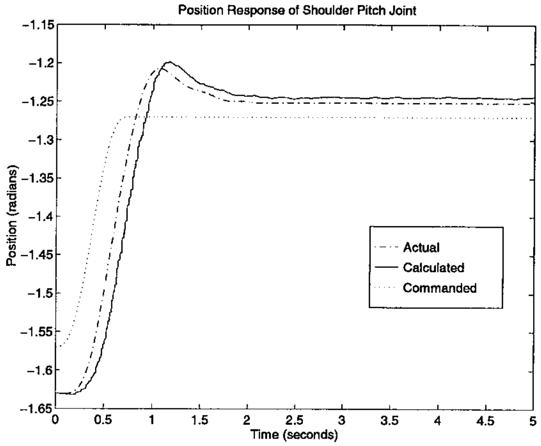


Figure 5.49: Position response of shoulder pitch joint to trajectory ending at [0.5 -1.27 -3.5 -1.27] in 0.75 seconds.

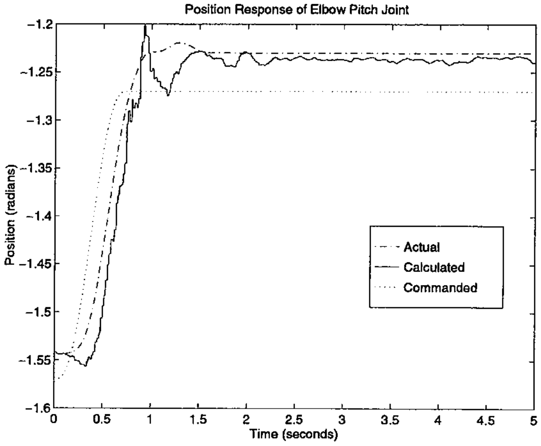


Figure 5.50: Position response of elbow pitch joint to trajectory ending at [0.5 -1.27 -3.5 -1.27] in 0.75 seconds

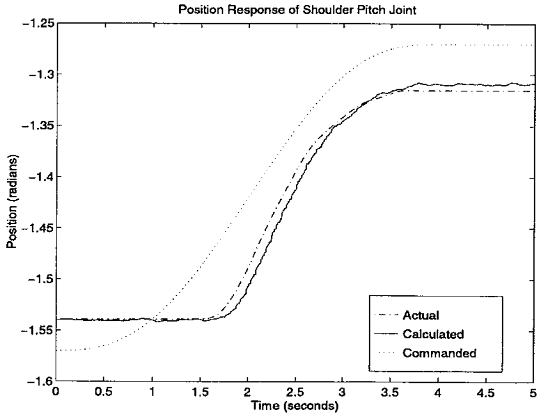


Figure 5.51: Position response of shoulder pitch joint to trajectory ending at [0.5 -1.27 -3.5 -1.27] in 4.0 seconds.

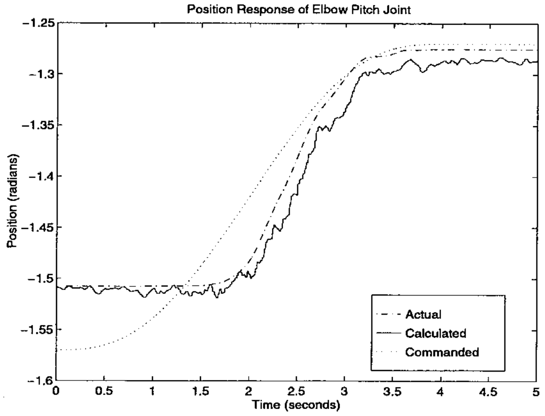
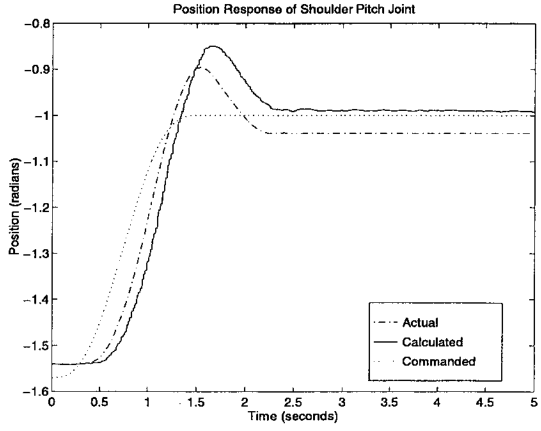


Figure 5.52: Position response of elbow pitch joint to trajectory ending at [0.5 -1.27 -3.5 -1.27] in 4.0 seconds

The next trajectory is the extreme case of moving from Home to [1 -1 -4 -1] in 1.5 seconds shown earlier. In this case Figs. 5.53-5.54 show that both joints have some overshoot with the shoulder pitch's overshoot being the largest. The shoulder pitch's overshoot served to cushion the response to the elbow pitch resulting in a smaller overshoot than in the single joint failure case. The steady state error between the calculated and measured shoulder pitch joint position is again due to calibration error.



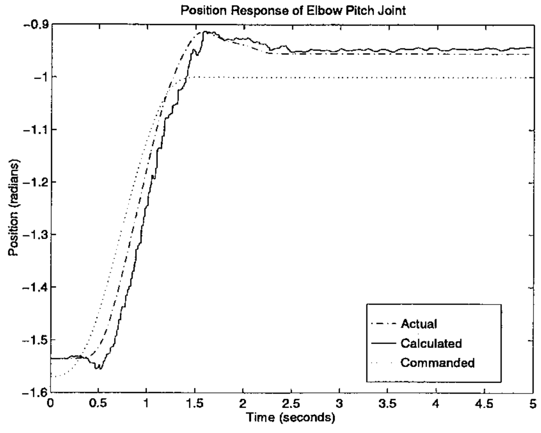
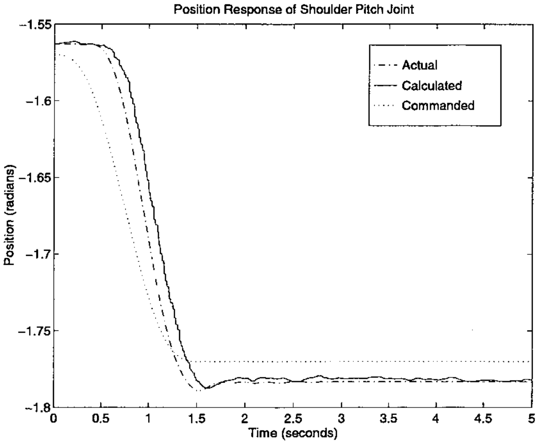


Figure 5.53: Position response of shoulder pitch joint to trajectory ending at [1 -1 -4 -1]

Figure 5.54: Position response of elbow pitch joint to trajectory ending at [1 -1 -4 -1]

The final trajectory considered is from Home to [-0.5 -1.77 -2.6 -1.17] in 1.5 seconds. This trajectory puts the arm in a configuration opposite to previous trajectories but the response shown in Figs. 5.55-5.56 is similar.



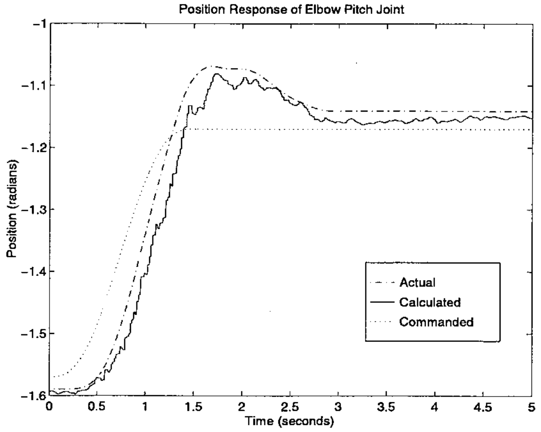


Figure 5.55: Position response of shoulder pitch joint to trajectory ending at [-0.5 -1.77 -2.6 -1.17]

Figure 5.56: Position response of elbow pitch joint to trajectory ending at [-0.5 -1.77 -2.6 -1.17]

To show the performance of the passive controller with actual measured joint feedback, Figs. 5.57-5.58 show the response of the passive controller to the trajectory shown in Figs. 5.53-5.54 with the same gains used during joint failure. If the position gains are increased, the performance shown in Figs. 5.59-5.60 can be realized. This controllers dynamic performance is better than the Robotics Research controller's response shown in Figs. 5.61-5.62 when the higher gains are used. These results show that the proposed control method gives the designer a choice of how to deal with the transition from operational joints to failed joints. The designer can elect to have tracking error similar to a standard PD controller with gravity compensation during normal performance and transition to stable performance during failure without changing controller gains. However, if the designer wants to obtain the benefit of better tracking with higher gains during normal operation it can be obtained by increasing the gains during normal operation and decreasing them during failure without changing controller structure. The stiffness of the virtual passive controller will resist most of the movement that would normally be caused by the gain reduction.

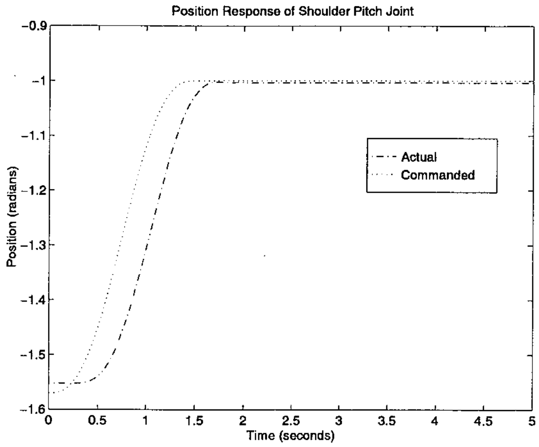


Figure 5.57: Position response of shoulder pitch joint using passive controller with low gains

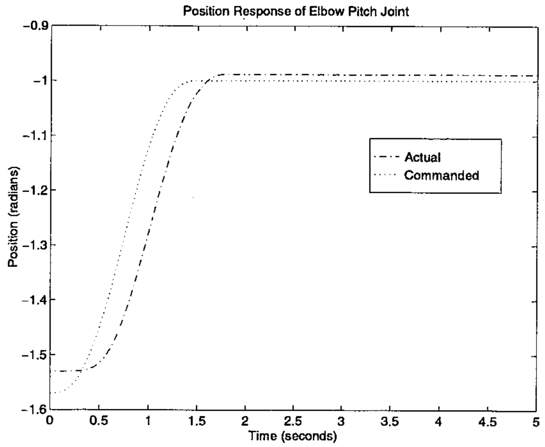


Figure 5.58: Position response of elbow pitch joint using passive controller with low gains

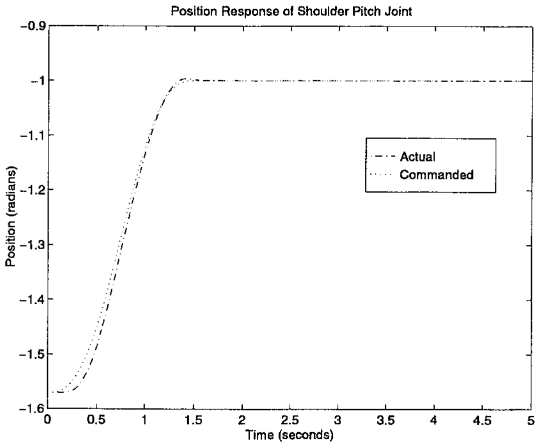


Figure 5.59: Position response of shoulder pitch joint using passive controller with high gains

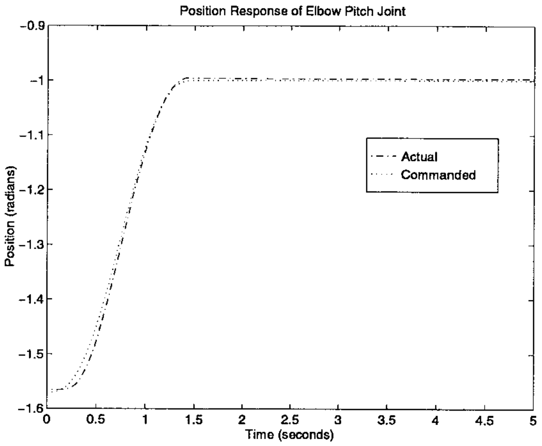


Figure 5.60: Position response of elbow pitch joint using passive controller with high gains

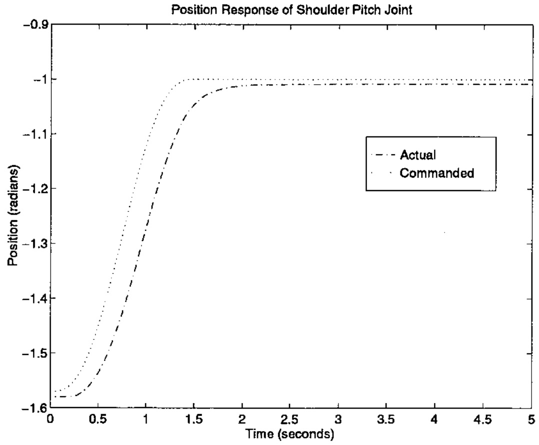


Figure 5.61: Position response of shoulder pitch joint using Robotics Research controller

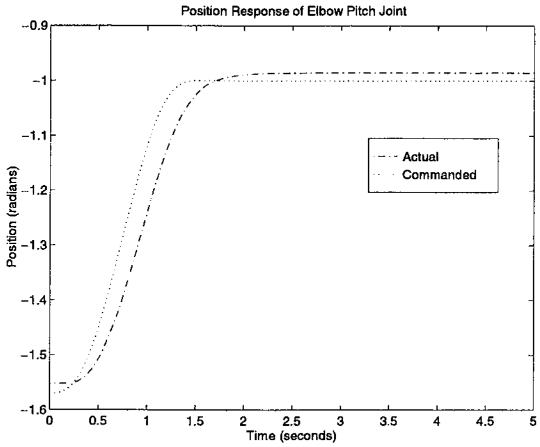


Figure 5.62: Position response of elbow pitch joint using Robotics Research controller

5.5 Summary

The proposed method for calculating joint positions from accelerometer data performed acceptably in the FDI mode in which the joints were servoed using measured joint data. Problems with increased noise susceptibility in certain configurations were shown. In failure recovery mode, the proposed position calculation method combined with the virtual passive torque controller described earlier showed stable servo performance which in some cases approached that of the Robotics Research baseline controller. Errors due to position controller overshoot and passive controller stiffness were discussed. All results were obtained in the presence of significant noise, bias, and with a simple accelerometer calibration. The resulting servo performance does not approach the optimal performance that can be obtained with joint position sensor, accurate models, and/or higher gains. However, the resulting performance, if carefully used, is adequate for continued operation during failure.

This chapter has shown experimentally the usefulness of the joint specific position determination technique and discussed some of its problems. One problem is that each joint that needs to be made fault tolerant must be instrumented. Instrumenting all robot joints when probably only one or two will fail during a critical task would not be cost effective. However, since which joint position sensors will fail cannot be predicted, all joints necessary for critical task completion must be instrumented. Chapter 6 presents a system wide position determination method which offers a reduction in the number of accelerometers required to determine the position of an interchangeable set of joints at the cost of higher computational complexity.

6. System Wide Joint Position Sensor Fault Tolerance in Robot Systems Using Cartesian Accelerometers

6.1 Introduction

The application of accelerometers in fault tolerance was expanded to joint position sensor fault tolerance in chapter 4 of this thesis. That joint specific method used two triaxial accelerometers to determine the position of a robot joint without integration or inverse kinematics. While useful to protect a particular joint sensor, the number of accelerometers required to protect multiple joints in the robot arm can be impractical. This chapter proposes a system wide approach to position sensor fault tolerance. A combination of end-effector mounted and link mounted accelerometers can be used to determine the position of multiple joints in a robot system. The accelerometers can be located in a manner to provide fault tolerance to any joint in the robot system.

6.2 Kinematic Equations

The proposed solution method is designed for an *n* revolute joint, rigid link robot system with *m* triaxial accelerometers mounted at various locations on the robot. A robot of this kinematic configuration is governed by the following kinematic equations:

$$\omega_{i+1}^{i+1} = R^{i+1}{}_{i}\omega_i^i + \dot{\theta}_{i+1}\hat{z} \tag{6.1}$$

$$\dot{\omega}_{i+1}^{i+1} = R_i^{i+1}\dot{\omega}_i^i + R_i^{i+1}\omega_i^i \times \dot{\theta}_i\hat{z} + \ddot{\theta}_{i+1}\hat{z} \tag{6.2}$$

$$a_{i+1}^{i+1} = R_i^{i+1}(\dot{\omega}_i^i \times l_i + \omega_i^i \times \omega_i^i \times l_i + a_i^i) \tag{6.3}$$

$$Am_{i+1}^{i+1} = \dot{\omega}_{i+1}^{i+1} \times pm_{i+1} + \omega_{i+1}^{i+1} \times \omega_{i+1}^{i+1} \times pm_{i+1} + a_{i+1}^{i+1} \tag{6.4}$$

where

ω_i^i angular velocity of i^{th} link in i^{th} frame

a_i^i acceleration of i^{th} link in i^{th} frame

R_i^{i+1} rotation matrix from i^{th} frame to $(i+1)^{th}$ frame

θ_i position of i^{th} joint

Am_{i+1}^{i+1} acceleration of accelerometer Am_{i+1} in the $(i+1)^{th}$ frame

pm_{i+1} vector offset from joint $i+1$ to accelerometer Am_{i+1}^{i+1}

l_i vector offset from joint i to joint $i+1$ in the frame i

$\hat{z}$ identity z vector $[0\ 0\ 1]^{\mathrm{T}}$

Given the Devanit-Hartenberg parameter [58], α_i, the 3x3 rotation matrix, R_i^{i+1}, is shown below.

$$R_i^{i+1} = \begin{bmatrix} \cos\theta_{i+1} & \sin\theta_{i+1}\cos\alpha_i & \sin\theta_{i+1}\sin\alpha_i \\ -\sin\theta_{i+1} & \cos\theta_{i+1}\cos\alpha_i & \cos\theta_{i+1}\sin\alpha_i \\ 0 & -\sin\alpha_i & \cos\alpha_i \end{bmatrix} \tag{6.5}$$

6.3 Solution Method

Each accelerometer contains information on the position, velocity, and acceleration of joints preceding the accelerometer in the kinematic chain. End-effector mounted accelerometers contain information about all joints. However, this information is mixed together in a nonlinear fashion. Except for accelerometers attached to the first joint, there is no single acceleration that is, in general, dependent on solely the position, velocity, or acceleration of the i^{th} joint. In addition, components of the acceleration due to certain joints can dominate due to kinematic configuration or trajectory. For certain configurations, the end-effector pose does not uniquely determine the joint trajectory of the robot. As a result, only utilizing end-effector mounted accelerometers is not sufficient for determining joint trajectories. Distributing accelerometers along the arm in addition to end-effector mounted accelerometers can help alleviate these problems by obtaining information at points closer to joints of interest.

The proposed solution method utilizes the distributed Cartesian accelerometers in conjunction with any working joint position sensors to recover from a position sensor failure. This is done by calculating a knot point, a point in the trajectory with position, velocity, and acceleration, that will make the measured accelerations match the accelerations determined by Eq. (6.4). In Eq. (6.4), the data from the operational joint sensors is used to the extent possible, leaving only terms involving the failed joints,

$$Q(\theta_j, \dot{\theta}_j, \ddot{\theta}_j) = A, \quad j \in b \tag{6.6}$$

where,

b is the set of failed joints

A is the vector of accelerometer measurements $[\ Am_1^1 \ldots Am_m^m\ ]^{\mathrm{T}}$

Q is a set of equations of type (6.4) corresponding to A

If r is the number of failed joints, then the resulting system has $3m$ nonlinear equations (one equation per axis, three axes per accelerometer) in $3r$ (position, velocity, and acceleration of failed joints) unknowns. If this system is solvable, then the knot points for all r joints can be determined and the joints controlled.

A minimum criteria for solution is that there be at least as many equations as unknowns, i.e. $m \geq r$. If this condition is not satisfied, then the minimum error solution will be one in the least squares sense. Due to the complexity of the equations, this condition is optimistic. As r approaches m, the system becomes more ill-conditioned. A better measure of predicted system performance will be discussed later.

The method discussed also relies on the existence of a known acceleration field, usually gravity. This field provides a known, constant excitation to the system accelerometers. The solution method can be applied to systems without such a field, but positions will be relative, not absolute, and drift due to small errors is likely. The accelerometers must be of the instrument type that can detect constant accelerations.

The solution method requires a computational technique for solving a system of nonlinear equations. To be practical, the solution must be calculable in real-time. Real time in this sense implies updates at a rate fast enough for stable joint control. This requirement limits the available solution techniques. The solution technique must also be robust to sensor noise and bias. The convergence of the technique to a good solution must be predictable. Traditional nonlinear solution techniques can be applied to this problem. Some, such as steepest descent[62], were found to be sensitive to sensor error. Others, such as nonlinear least squares[63], gave excellent results but were too time consuming for real-time implementation. A new method, continuation minimax, is developed as a compromise between solution accuracy and suitability for real-time implementation.

6.4 Continuation Minimax

This method is a combination of the continuation method[64,65] and the minimax method[66,67]. The classical continuation method is based on using sequential linear programming to optimize a nonlinear function based on nonlinear criteria. In this formulation, F is the function to be optimized, P is the vector of optimization parameters, P_a is the current approximation of the optimal solution, J_c is the gradient of the constraint function, c, and J_F is the gradient of the optimization function. Placed into a linear programming problem:

$$\text{Minimize } \Delta F = J_F^T \Delta P \qquad (6.7)$$

subject to

$$J_c \Delta P \le -c(P_a)\Delta\alpha \tag{6.8}$$

The quantity α ranges from 0 to 1 in increments of $\Delta\alpha$. The step size controls the accuracy of the approximation of the optimum *P,* i.e. P^*. The next approximation of P^*, which is $P_a+\Delta P$, leads to a better approximation of the optimal *F*, i.e. F^*.

The minimax solution technique detailed in [66] also uses linear programming. The object of the minimax solution is to find the maximum of the minimum error, γ, between the calculated constraint equations and the desired value of the constraints:

$$\text{Minimize } \gamma \tag{6.9}$$

subject to

$$J_c \Delta P - \gamma I_{3m} \le c(P_a) \tag{6.10}$$

$$-\Lambda_a \le \Delta P_i \le \Lambda_a \tag{6.11}$$

Where I_{3m} is the $3m$x$3m$ identity matrix. The parameter Λ_a controls the amount of change allowed in ΔP. The bound is heuristically changed depending on a comparison of the decrease in the function *c* and the linearized version. If the ratio of the two is small:

$$c(P_a + \Delta P) - c(P_a) \le 0.25 J_c \Delta P \tag{6.12}$$

then $\Lambda_{a+1}=\Lambda_a/4$. If the ratio is large:

$$c(P_a + \Delta P) - c(P_a) \ge 0.75 J_c \Delta P \tag{6.13}$$

then $\Lambda_{a+1}=2\Lambda_a$, otherwise $\Lambda_{a+1}=\Lambda_a$. The minimax solution is iterated until a desired value of γ is achieved or only a minimal change in γ is produced.

The proposed computational procedure, continuation minimax, is a combination of both techniques. The rationale behind the combination is to capitalize on the predictable solution time, determined by $\Delta\alpha$, of the continuation method and the ability of the minimax algorithm to find the minimal error between the true $c(P^*)$ and the approximated $c(P_a)$.

In linear programming form, the continuation minimax formulation is:

$$\text{Minimize } F = \sum_{i=1}^{m} \gamma_i \tag{6.14}$$

subject to:

$$J_c \Delta P - diag(\gamma_1 .. \gamma_m) \leq \alpha c(P_a) \tag{6.15}$$

$$-J_c \Delta P - diag(\gamma_1 .. \gamma_m) \geq -\alpha c(P_a) \tag{6.16}$$

$$-\Lambda_a \leq \Delta P_i \leq \Lambda_a \tag{6.17}$$

$$\gamma_i \geq 0 \tag{6.18}$$

The bound Λ_a is adjusted as in the minimax formulation.

Reasons for some of the differences between existing methods and the new continuation minimax formulation are:

- The function F was linear, so it was used instead of a linearized version
- The quantity α ranges from 0 to 1 in increments $\Delta\alpha$ as before, but is used explicitly in the iteration to allow for the combination of methods
- The summation of several γ_i was chosen over only γ for better performance during sensor error. With low sensor error, a formulation using only γ is also possible and would further decrease computational requirements.
- An upper and lower bound, Eqs. (6.15) and (6.16), are required because ideally γ_i=0 $\forall$i. An upper or lower bound alone would not reduce to this condition at the ideal case.

For specific application to the position sensor fault tolerance problem, c=$Q(P_a)$-A, P_a is $[\theta_a \;\; \dot{\theta}_a \;\; \ddot{\theta}_a]$ (the joint position, velocity, and acceleration), and J_c is the gradient of c with respect P.

The continuation minimax method has an accuracy and an execution time that is controlled by the choice of $\Delta\alpha$. Other solution techniques, under certain conditions, may obtain a more optimal solution quicker. The primary advantage of the continuation minimax solution technique is its ability to give a solution to the problem in a predictable execution time. This predictability is a requirement for real-time implementation.

6.5 Gradient Calculation

Most nonlinear solution methods require a function gradient for rapid convergence. In this application, the continuation minimax solution method requires J_c, the gradient of c. The gradient of c is equal to the gradient of Q. The closed-form solution for J_c for an n link manipulator becomes more complex quickly as n grows. The gradient could be computed numerically using finite differences[68]. Finite differences are not as accurate as a closed-form solution. The approach taken here is to calculate the gradient recursively. The system equations are designed to be calculated recursively, so the proper application of partial derivatives and the chain rule will lead to the desired outcome.

The desired gradient has the form:

$$\frac{\partial Q}{\partial P_a} = \begin{bmatrix} \frac{\partial Am_1}{\partial \theta_1} & \cdots & \frac{\partial Am_1}{\partial \theta_n} & \frac{\partial Am_1}{\partial \dot{\theta}_1} & \cdots & \frac{\partial Am_1}{\partial \dot{\theta}_n} & \frac{\partial Am_1}{\partial \ddot{\theta}_1} & \cdots & \frac{\partial Am_1}{\partial \ddot{\theta}_n} \\ \vdots & & \vdots & \vdots & & \vdots & \vdots & & \vdots \\ \frac{\partial Am_m}{\partial \theta_1} & \cdots & \frac{\partial Am_m}{\partial \theta_n} & \frac{\partial Am_m}{\partial \dot{\theta}_1} & \cdots & \frac{\partial Am_m}{\partial \dot{\theta}_n} & \frac{\partial Am_m}{\partial \ddot{\theta}_1} & \cdots & \frac{\partial Am_m}{\partial \ddot{\theta}_n} \end{bmatrix} \tag{6.19}$$

For simplicity, the measurement Am_i is always in the i^{th} frame. Now,

$$\begin{aligned} \frac{\partial Am_{i+1}}{\partial \theta_k} = {} & \frac{\partial \dot{\omega}_{i+1}^{i+1}}{\partial \theta_k} \times pm_{i+1} + \frac{\partial \omega_{i+1}^{i+1}}{\partial \theta_k} \times \omega_{i+1}^{i+1} \times pm_{i+1} \\ & + \omega_{i+1}^{i+1} \times \frac{\partial \omega_{i+1}^{i+1}}{\partial \theta_k} \times pm_{i+1} + \frac{\partial a_{i+1}^{i+1}}{\partial \theta_k} \end{aligned} \tag{6.20}$$

$$\begin{aligned} \frac{\partial Am_{i+1}}{\partial \dot{\theta}_k} = {} & \frac{\partial \dot{\omega}_{i+1}^{i+1}}{\partial \dot{\theta}_k} \times pm_{i+1} + \frac{\partial \omega_{i+1}^{i+1}}{\partial \dot{\theta}_k} \times \omega_{i+1}^{i+1} \times pm_{i+1} \\ & + \omega_{i+1}^{i+1} \times \frac{\partial \omega_{i+1}^{i+1}}{\partial \dot{\theta}_k} \times pm_{i+1} + \frac{\partial a_{i+1}^{i+1}}{\partial \dot{\theta}_k} \end{aligned} \tag{6.21}$$

$$\frac{\partial Am_{i+1}}{\partial \theta_k} = \frac{\partial \dot{\omega}_{i+1}^{i+1}}{\partial \dot{\theta}_k} \times pm_{i+1} + \frac{\partial a_{i+1}^{i+1}}{\partial \theta_k} \tag{6.22}$$

The partial derivative will be zero for all quantities of joint *k* where $k>i+1$. This a consequence of the later joints not affecting the previous links.

Equations (6.20),(6.21),and (6.22) are not recursive themselves, but the equations for the partials they depend on are recursive. The equations require that Eqs. (6.1) through (6.3) be calculated for all *n* links to determine the angular velocities, angular accelerations, and Cartesian accelerations. The partial derivatives can then be calculated in the following manner:

$$\frac{\partial \omega_{i+1}^{i+1}}{\partial \theta_k} = \begin{cases} R_i^{i+1} \dfrac{\partial \omega_i^i}{\partial \theta_k} & \mathrm{k < i + 1} \\ \dot{R}_k^{k+1} \omega_k^k & \mathrm{k = i + 1} \\ \bar{0} & \mathrm{k > i + 1} \end{cases} \tag{6.23}$$

$$\frac{\partial \omega_{i+1}^{i+1}}{\partial \dot{\theta}_k} = \begin{cases} R_i^{i+1} \dfrac{\partial \omega_i^i}{\partial \dot{\theta}_k} & \mathrm{k < i + 1} \\ \hat{z} & \mathrm{k = i + 1} \\ \bar{0} & \mathrm{k > i + 1} \end{cases} \tag{6.24}$$

$$\frac{\partial \omega_{i+1}^{i+1}}{\partial \ddot{\theta}_k} = \bar{0} \qquad \forall \mathrm{k} \tag{6.25}$$

$$\frac{\partial \dot{\omega}_{i+1}^{i+1}}{\partial \theta_k} = \begin{cases} R_i^{i+1} \dfrac{\partial \dot{\omega}_i^i}{\partial \theta_k} + R_i^{i+1} \dfrac{\partial \omega_i^i}{\partial \theta_k} \times \dot{\theta}_k \hat{z} & \mathrm{k < i + 1} \\ \dot{R}_i^{i+1} \dot{\omega}_i^i + \dot{R}_i^{i+1} \omega_i^i \times \dot{\theta}_k \hat{z} & \mathrm{k = i + 1} \\ \bar{0} & \mathrm{k > i + 1} \end{cases} \tag{6.26}$$

$$\frac{\partial \dot{\omega}_{i+1}^{i+1}}{\partial \dot{\theta}_k} = \begin{cases} R_i^{i+1} \dfrac{\partial \dot{\omega}_i^i}{\partial \dot{\theta}_k} + R_i^{i+1} \dfrac{\partial \omega_i^i}{\partial \dot{\theta}_k} \times \dot{\theta}_k \hat{z} & \mathrm{k < i + 1} \\ R_i^{i+1} \omega_i^i \times \hat{z} & \mathrm{k = i + 1} \\ \bar{0} & \mathrm{k > i + 1} \end{cases} \tag{6.27}$$

$$\frac{\partial \dot{\omega}_{i+1}^{i+1}}{\partial \ddot{\theta}_k} = \begin{cases} R_i^{i+1} \dfrac{\partial \dot{\omega}_i^i}{\partial \ddot{\theta}_k} & \mathrm{k < i + 1} \\ \hat{z} & \mathrm{k = i + 1} \\ \vec{0} & \mathrm{k > i + 1} \end{cases} \tag{6.28}$$

$$\frac{\partial a_{i+1}^{i+1}}{\partial \theta_k} = \begin{cases} R_i^{i+1} \begin{pmatrix} \dfrac{\partial \dot{\omega}_i^i}{\partial \theta_k} \times l_i + \dfrac{\partial \omega_i^i}{\partial \theta_k} \times \omega_i^i \times l_i + \\ \omega_i^i \times \dfrac{\partial \omega_i^i}{\partial \theta_k} \times l_i + \dfrac{\partial a_i^i}{\partial \theta_k} \end{pmatrix} & \mathrm{k < i + 1} \\ \dot{R}_i^{i+1} \left(\dot{\omega}_i^i \times l_i + \omega_i^i \times \omega_i^i \times l_i + a_i^i \right) & \mathrm{k = i + 1} \\ \vec{0} & \mathrm{k > i + 1} \end{cases} \tag{6.29}$$

$$\frac{\partial a_{i+1}^{i+1}}{\partial \dot{\theta}_k} = \begin{cases} R_i^{i+1} \begin{pmatrix} \dfrac{\partial \dot{\omega}_i^i}{\partial \dot{\theta}_k} \times l_i + \dfrac{\partial \omega_i^i}{\partial \dot{\theta}_k} \times \omega_i^i \times l_i + \\ \omega_i^i \times \dfrac{\partial \omega_i^i}{\partial \dot{\theta}_k} \times l_i + \dfrac{\partial a_i^i}{\partial \dot{\theta}_k} \end{pmatrix} & \mathrm{k < i + 1} \\ \vec{0} & \mathrm{k \geq i + 1} \end{cases} \tag{6.30}$$

$$\frac{\partial a_{i+1}^{i+1}}{\partial \ddot{\theta}_k} = \begin{cases} R_i^{i+1} \left(\dfrac{\partial \dot{\omega}_i^i}{\partial \dot{\theta}_k} \times l_i + \dfrac{\partial a_i^i}{\partial \dot{\theta}_k} \right) & \mathrm{k < i + 1} \\ \vec{0} & \mathrm{k \geq i + 1} \end{cases} \tag{6.31}$$

where,

$$\dot{R}_i^{i+1} = \begin{bmatrix} -\sin\theta_{i+1} & \cos\theta_{i+1}\cos\alpha_i & \cos\theta_{i+1}\sin\alpha_i \\ -\cos\theta_{i+1} & -\sin\theta_{i+1}\cos\alpha_i & -\sin\theta_{i+1}\sin\alpha_i \\ 0 & 0 & 0 \end{bmatrix} \tag{6.32}$$

is the derivative of R_i^{i+1} with respect to θ_{i+1}.

Using Eqns. (6.20) through (6.32), the elements of the gradient can be calculated. The previous equations have been verified against numerically calculated gradients of Q.

6.6 Simulation

Simulations were conducted to verify the solution method presented in this chapter. A Robotics Research 807i manipulator was simulated using Matlab and Simulink numerical simulation software. The 807i is a 0.8m long robot with seven revolute joints. The full 7 DOF configuration, as shown in Fig. 6.1, was implemented and tested in general 6 DOF space. Two triaxial accelerometers were attached to the end-effector and one triaxial accelerometer was attached to joints 2, 4, and 6.

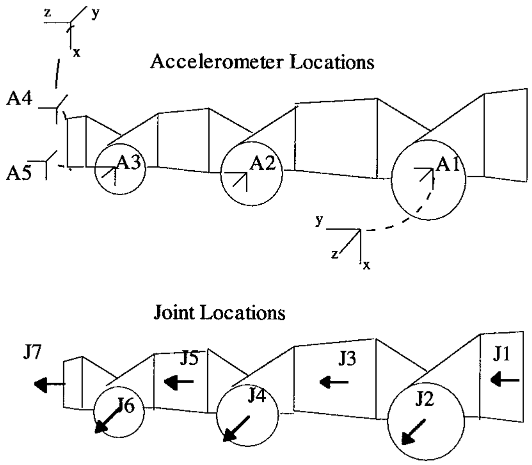


Figure 6.1: Robot diagram

Trajectories were generated using a standard 3-5-3 trajectory generation algorithm[58]. The desired trajectory starting and ending positions were [1.1 0.2 -0.3 0.4 -1.5 0.6 0.7] radians and [0.5 -0.3 0.0 0.2 -0.2 0.0 1.4] radians respectively. The trajectory was to be completed in 1.0 second followed by 0.7 second of zero velocity and acceleration command. This rather fast move would show the resilience of the method to dynamic effects. The calculated positions were not used as feedback to the position controller. A very low tracking accuracy controller was used in the system to show that precise, smooth trajectory following is not required by the solution method. This low precision controller accounted for the need for a 0.7 second settling period after the command was completed.

6.6.1 Performance Tests

To verify the method, a nonlinear least squares algorithm was used with ideal sensor data. Joints 2, 3, 5, and 6 were assumed to have failed joint sensors. The maximum position error produced was $8\text{x}10^{-7}$ radians. Assuming ideal data, this maximum error could be further reduced by changing the tolerance of the algorithm. To maintain the tolerances that this algorithm produced, a large number of function and gradient calls were required in addition to the overhead of the solution method. The subroutine calls required are plotted in Fig. 6.2. Although the number of subroutine calls is lower if the desired solution tolerance is lowered, the number of calls is unpredictable. During error spikes, the tolerance may not be achievable, causing difficulty for the solution algorithm unless properly handled.

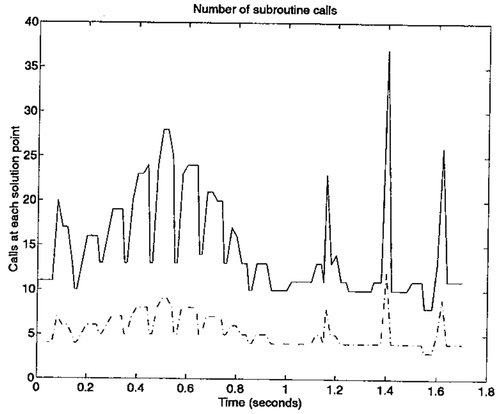


Figure 6.2: Number for function and gradient evaluations for Nonlinear Least Squares simulation. (Number of function evaluations is solid line)

Now, The response of the continuation minimax is analyzed with realistic sensor conditions. Accelerometers are susceptible to several error types: bias, noise, and placement error. The bias and noise can be caused by temperature, off axis accelerations, amplifier noise, vibration, or many other causes. Since the accelerometers must be placed at different points on the arm, small errors between their specified and actual positions are possible.

The primary usage of this technique is to continue operation after sensor failure without external calibration. Between the time a sensor failure occurs and when it is detected, an error will accumulate. All nonlinear optimization methods can converge to an "optimal" solution that is not the desired solution. While it is maintained that given a position reasonably close to the actual position the optimal solution will be the desired solution, higher numbers of failures will lead to a smaller convergence range. In [64] and [65], the basic continuation method has been shown not to require a very good seed value to converge to the desired optimum.

Figures 6.3 through 6.5 show the error caused individually by sensor bias, noise, and placement error. Three joints, 2,5, and 6, were assumed failed. The following plots include an error of [-0.1 -0.08 0.1] radian in the initial joint position of the failed joints. The accelerometer bias varied from axis to axis with magnitudes of -0.2973m/s^2 to 0.2949m/s^2. The accelerometer error due to noise ranged in magnitude from

-0.4039m/s^2 to 0.4285 m/s^2. Accelerometer placement error ranged from 50mm to -50mm. All simulations used $\Delta\alpha$=0.333333. This choice of $\Delta\alpha$ implies 3 function and gradient calls per solution.

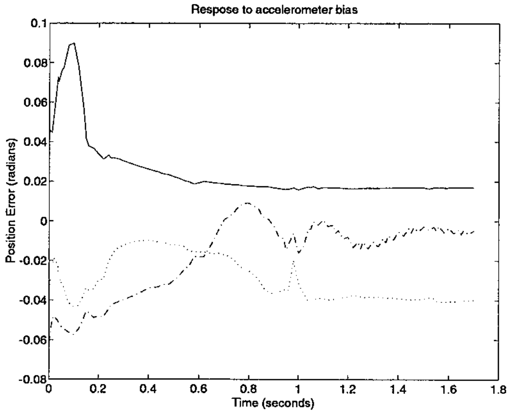


Figure 6.3: Response to accelerometer bias (Joint 2 error is solid line, Joint 5 is dash-dot, Joint 6 is dotted)

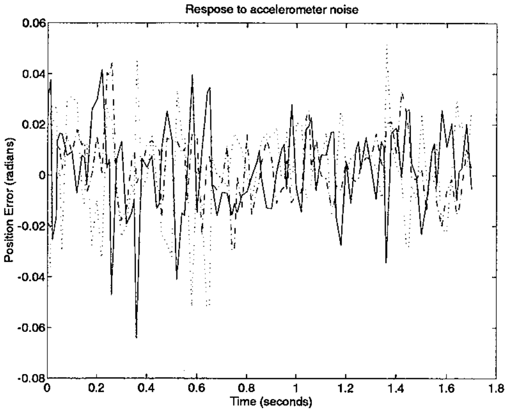


Figure 6.4: Response to accelerometer noise (Joint 2 error is solid line, Joint 5 is dash-dot, Joint 6 is dotted)

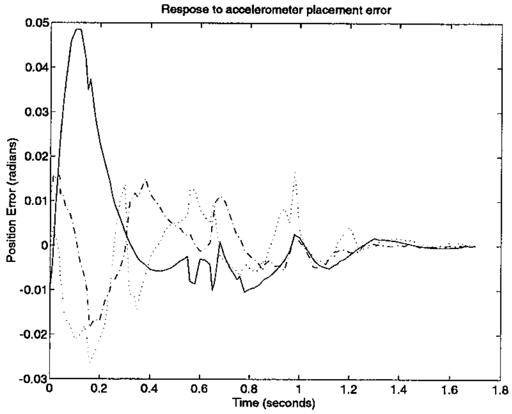


Figure 6.5: Response to accelerometer placement error (Joint 2 error is solid line, Joint 5 is dash-dot, Joint 6 is dotted)

To show the effect of a combination of sensor errors, Fig. 6.6 shows the response to sensor bias, noise, and placement errors. For variety, the same trajectory was used but joints 1,4, and 7 were assumed failed. An initial position error of [0.1 0.12 -0.05] radian was used for the failed joints. The same accelerometer bias and placement errors were used as in earlier simulations along with similar noise amplitude. It should be noted that the accelerometer configuration need not be altered to provide fault tolerance for the different set of failed joints.

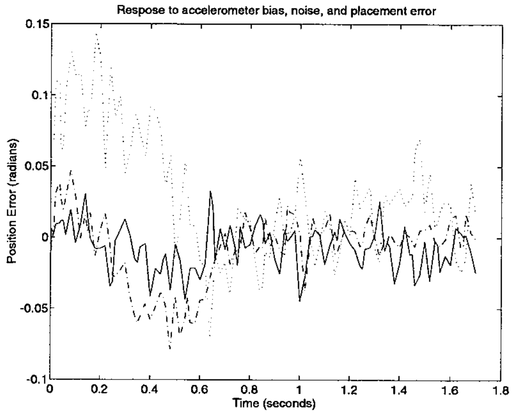


Figure 6.6: Response to accelerometer bias, noise, and placement error (Joint 1 error is solid line, Joint 4 is dash-dot, Joint 7 is dotted)

As with all sensor based methods, better data will produce better results.

6.6.2 Solution existence

Given a specific set of operational joint sensors, accelerometers, and a joint trajectory, the joint positions may or may not be recoverable. As discussed earlier, simple comparison of the number of equations and unknowns is not sufficient. Along certain trajectories, equations may not be unique due to accelerometer alignment or other numerical problems. Even when a certain situation is solvable in the ideal case, it may be too sensitive to sensor error to yield a reasonable solution. It may also be too sensitive to small parameter changes for the chosen solution method to converge quickly, if ever.

The proposed method to determine whether a sensor configuration and joint trajectory are solvable is inspection of the condition number of the gradient used by the solution. This usage is based on the application of condition numbers in linear algebra[69].

Ideally, a proposed trajectory and configuration is evaluated off-line. If acceptable condition numbers are produced along the trajectory, the trajectory will be given to the robot to execute. The main problem is determining what constitutes an acceptable condition number. Figure 6.7 shows what occurs to the gradient's condition number as failures occur. All plots assumed noise free accelerometers in ideal locations but did include the sensor bias used in previous simulations.

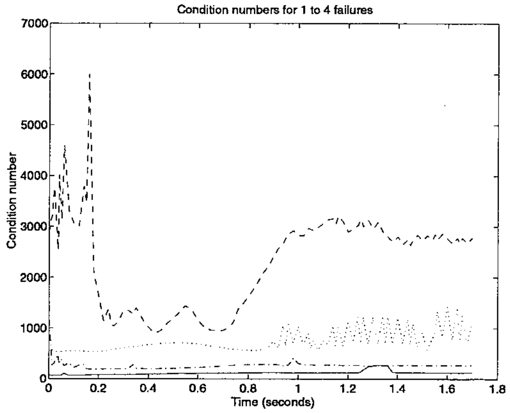


Figure 6.7: Condition number of gradient matrix with 1,2,3, and 4 failures. (1 failure is solid, 2 failures is dash-dot, 3 failures is dotted, 4 failures is dashed)

The error of joint 2 associated with each condition number is show in Fig. 6.8.

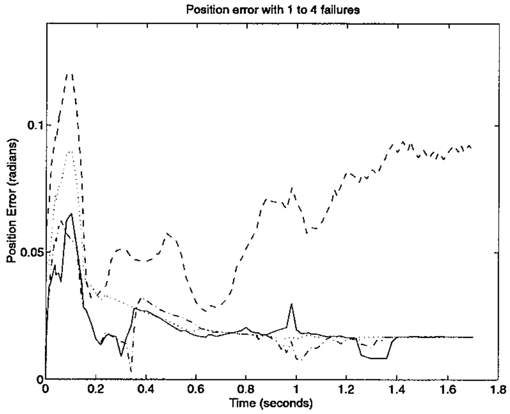


Figure 6.8: Error in joint 2 during 1,2,3, and 4 failures. (1 failure is solid, 2 failures is dash-dot, 3 failures is dotted, 4 failures is dashed)

Examining Figs. 6.7 and 6.8, the position error of joint 2 is relatively small for 1,2, and 3 failures. These failures have gradient condition numbers of <1500. With 4 failures, the error is still acceptable, but is becoming significantly worse. While the condition number between 0.2 and 0.7 seconds is low, it maintains an average close to 3000. Although not shown in the figures, five failures leads to an unacceptable position error which peaks at -1.4 radians. The condition number during 5 failures has values above 10^6 and an average above 10^5. It is suggested that an average condition number of below 2000 be used as a cutoff for trajectory solvability. This number can be relaxed if more robust solution techniques (at the cost of computational complexity) and better accelerometers are utilized. Further familiarization with a specific group of trajectories can also allow the designer to be confident with higher condition numbers in the area of known solvable trajectories.

As seen in the previous plots, for systems with reasonable condition numbers, the proposed solution method shows bounded error response to bounded sensor errors. The same cannot be stated for integration based techniques. A classical technique for end effector tracking uses double integration of accelerometer readings to obtain Cartesian position. Sensor error will accumulate quickly in this technique. As a result, the method can only be applied for short periods. The proposed technique can be applied indefinitely.

There are techniques that can be used to enhance the solvability of the equations. These include:

- Engage joint brakes to eliminate joint velocity and acceleration components. This can be used to improve the initial position estimate before executing a controlled move.
- Lower desired velocities and accelerations of the system. Low energies imply lower contribution of non-acceleration field components on the system

- Take into account the condition number of the gradient as a penalty function for designing the trajectory thereby producing a more robust trajectory for the solution method.
- Check for noise and bias in the sensor data during periods where all sensors are operational. Filter coefficients identified during this period could be used to improve the accelerometer data during sensor failure.

6.7 Summary

This chapter has discussed a method using Cartesian accelerometers to recover joint position sensor information lost during failure. A solution technique, continuation minimax, was proposed to solve the required nonlinear equations in real-time. A recursive technique to calculate the function gradient required for solving the nonlinear equations was detailed. Simulations were presented showing the response of the technique to non-ideal sensor data. It was shown that a set of accelerometers can protect against different sets of sensor failures throughout the robot system. By not utilizing integration in the solution method, this method neutralizes the concern of accumulating error in using accelerometers to produce position related information.

This chapter has developed a theoretical foundation for using accelerometers to determine the position of an interchangeable set of robot joints using a system wide approach. However, the accuracy of these positions using real sensor data and the usefulness of the calculated positions in controlling real joints are not discussed. In chapter 7, three joints of a real manipulator and its end-effector will be instrumented with accelerometers and practical implementation issues for this position determination method will be discussed. Experiments in chapter 7 will show the performance of this position determination method in both FDI and as part of a feedback control system in failure recovery.

7. Experimental Results - System Wide Joint Position Sensor Fault Tolerance

The purpose of this chapter is to describe the experiments used to determine the performance of the system wide joint position sensor fault tolerance theory developed in chapter 6 on a real robot.

7.1 Experimental Setup

The experimental apparatus was based on the system used for the joint specific position sensor fault tolerance experiments detailed in chapter 5. The Robotics Research 807i manipulator and its control system were modified from that configuration as follows:

1. A 33Mhz 68040 based computer, to be referred to as the optimization processor, was added to the system and communication enabled between it and the two existing processors.
2. Accelerometer mounts for the robot end-effector were designed, fabricated, and installed.
3. Accelerometers were moved to new locations for this experiment and appropriate cable modifications made.
4. MATLAB simulation code was ported to C++ to run on the optimization processor.

The end effector accelerometer mount can be seen in Fig. 7.1. Two triaxial accelerometers were mounted to the end-effector. The joint accelerometer mounts attached to the shoulder pitch, elbow roll, and elbow pitch joints from the joint specific experiment were used in this experiment. Each joint mount had one triaxial accelerometer attached instead of the two triaxial accelerometers used in the joint specific experiment. These joint placements are not ideal. They did allow for utilization of existing mounts. Placing an accelerometer on the wrist pitch joint instead of the elbow roll joint would have been a better choice but there were no mounting points for that joint. The resulting system was instrumented with a total of five triaxial accelerometers. These mounts are impractical for real world use. They are meant only for experimental use. Given the small size of current accelerometers, the accelerometers could be better incorporated into the robot structure.

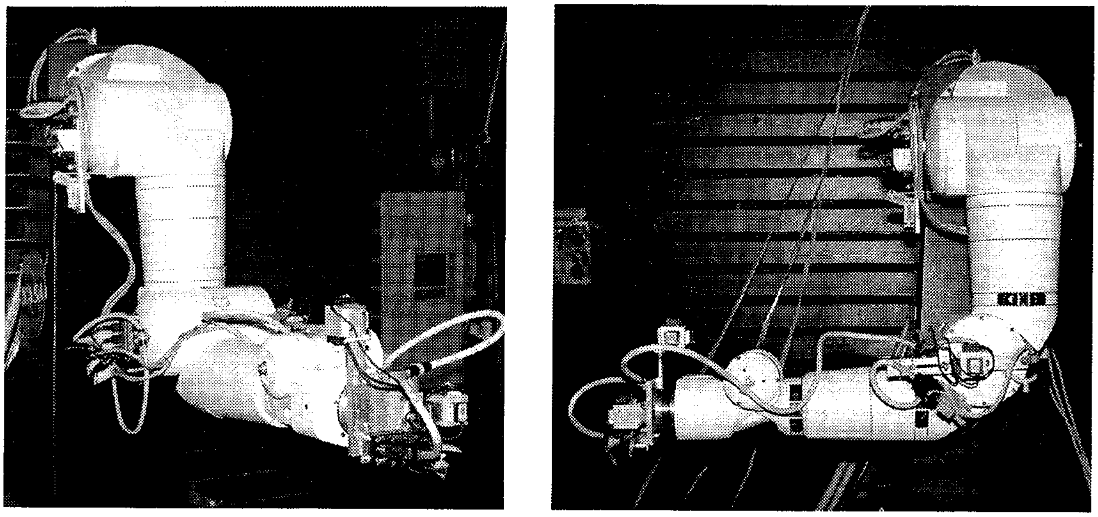


Figure 7.1: Robot with accelerometer mounts

The accelerometer noise and calibration problems encountered in the previous experiment were unchanged. As a result, the algorithm used in this experiment had to deal with significant sensor noise, calibration error, and delay caused by accelerometers and filters. Since this method does not rely on relative accelerations to calculate joint velocity and acceleration as the previous method did, the variable bias between accelerometers was not as much of a problem.

The increased computational complexity of this method over the joint by joint method required the inclusion of a third processor and modification of the algorithm. In the theoretical discussion, the continuation minimax method was presented as a compromise between a robust, highly accurate nonlinear optimization method such as nonlinear least squares and a less computationally complex method such as steepest descent. A recursive gradient calculation method was also presented for use by the optimization method. Unfortunately, the continuation method combined with the required function and gradient calculations was only calculable with the third processor used at 3Hz assuming three simultaneous joint failures. This update rate is unacceptable for real time control. To increase the update rate, the following changes were made:

1. Instead of calculating the gradient multiple times during one cycle, a single gradient calculation was done over four cycles. Due to the modular nature of the gradient calculation method, this was accomplished by doing the calculations in the following order:
   First cycle: Partial derivatives of angular velocity
   Second cycle: Partial derivatives of angular acceleration
   Third cycle: Partial derivatives of linear acceleration of the joint frame
   Fourth cycle: Partial derivatives of calculated accelerometer output and form gradient

(The partial derivatives of the calculated accelerometer outputs are not recursive and use the previous partial derivatives which are recursive and more computationally complex.)

Calculating the gradient at a lower speed assumed that the robot will not be moving fast enough to make the gradient change significantly over the four cycles.

2. The continuation minimax method was not used. The linear programming code available was not fast enough for a real-time implementation. Like most robust linear programming algorithms, it was optimized to accurately solve large programs, not to quickly solve small programs. Better code and faster processors could make this method practical. Instead of continuation minimax, the steepest descent method was used.
3. In the steepest descent method, a singular value decomposition (SVD) algorithm was used to form the least squares solution for the descent step. Although a new function evaluation was performed and the difference between the measured and calculated acceleration used with the last SVD calculation to form the next descent step, the SVD was not calculated each iteration. The SVD was calculated on the cycle after a new gradient was formed. The next gradient calculation began the cycle after the new SVD was calculated. As a result, a new SVD of the gradient was available every fifth cycle.
4. To decrease computational workload, the joint velocity was not calculated using the optimization algorithm. The differential model developed in chapter 4 which utilized joint position and acceleration calculations to determine joint velocity was used. Since the optimization algorithm will not produce a joint velocity calculation, the model only formulation instead of the observer formulation was used.

These changes resulted in a system that could provide a position update at 60Hz with three simultaneous joint failures. This rate of calculation resulted in a new SVD of the gradient at 12Hz.

7.2 Experimental Goals

The goal of this experiment is to show acceptable position tracking performance using the proposed solution method. Various combinations of failed joints will be considered to show the ability of the algorithm to recover from up to three simultaneous failures with the installed sensor package. Two modes of operation will be discussed. In the FDI (Failure Detection and Identification) mode, the joints will be considered unfailed and servoed using the Robotics Research position controller provided by the manufacturer. In this mode, the accelerometers will be read and the algorithm will attempt to calculate the joint position, velocity, and acceleration without feedback from the "failed" joint. This configuration will show the ability of the solution method to track actual measurements. In practice, the FDI mode could be used to determine if a joint position sensor has failed. In fault recovery mode, the calculated position and velocity data will be used along with the virtual passive torque control method developed in earlier chapters to servo the joint. In practice, the fault recovery mode would be enabled when a joint position sensor was determined to have failed.

Before the experimental results are discussed, it should be understood that the proposed method is a fault tolerant control method which uses traditional sensors for a non-traditional purpose. The method is not advocated as a primary control method. In practice, the position tracking performance of an accelerometer cannot match that of an optical encoder, resolver, or even a potentiometer. As a result, the servo performance will not be as good as one using traditional joint sensors and a basic control algorithm, much less a control method using accurate dynamic models. The goal is to show stable servo performance with adequate position response to make continued operation during failure an option with this method.

7.3 FDI Mode

In FDI mode, the joints are all servoed using measured position information but some joints are considered "failed" by the algorithm running on the optimization processor. Although the algorithm does not use the position or velocity feedback from the "failed" joints, it is available to the rest of the system and is presented here for comparison. This FDI mode is useful in detecting joint position sensor failures. One joint at a time could be considered "failed" and its position calculated and compared to the actual measurement. Given a reasonable error bound, this method could be effective. Considering multiple joints "failed" simultaneously would have less relevance to FDI, but the response does help indicate the position tracking performance that can be expected when the fault recovery mode is enabled.

7.3.1 Single joint "failures"

In the first set of plots one joint will be "failed" and the other joints will be operating normally but not moving. This will show the tracking performance of the algorithm on a real system without dealing with the dynamic effects of joints other than the failed joint. This condition is the best case for the system to operate. Failure of each of the seven robot joints will be considered. Although joint velocity and acceleration are calculated, due to the large number of plots required to show individual joint performance, only joint position plots will be presented. Figures 7.2 and 7.3 show the response of the shoulder roll and shoulder pitch joints. All trajectories begin in the Home position, [0 -π/2 -π -π/2 0 0 0] radians. The delay between actual and calculated is due to accelerometer response, filter delays, and communication delays.

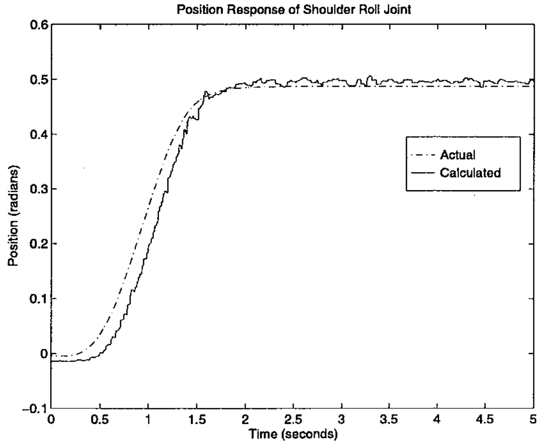


Figure 7.2: Position response of shoulder roll joint to trajectory ending at 0.5

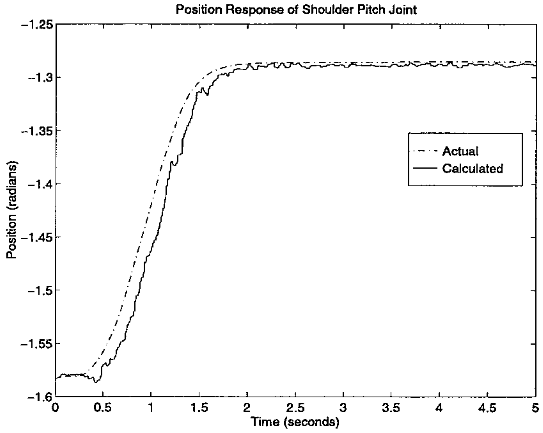


Figure 7.3: Position response of shoulder pitch joint to trajectory ending at -1.27

The elbow roll joint is considered next. As in the previous method, this joint is problematic in the home position because it is parallel to gravity. In this configuration, the signal to noise ratio is very low. As a result, the position response shown in Fig. 7.4 does not track well. If the shoulder roll joint is moved to 0.5 radians, the position response improves as shown in Fig. 7.5. It should be noted that all joints will have configurations where the signal to noise ratio is low. This property should be considered in trajectory planning.

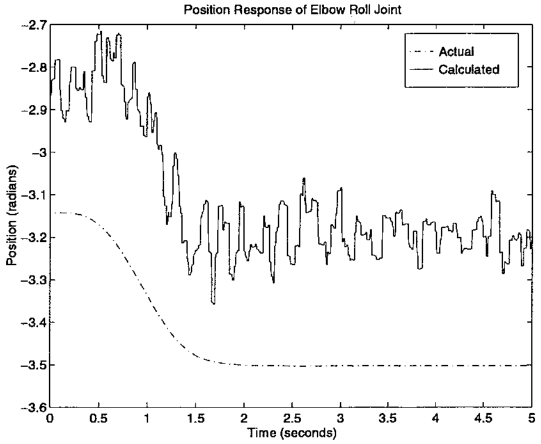


Figure 7.4: Position response of elbow roll joint to trajectory ending at -3.5

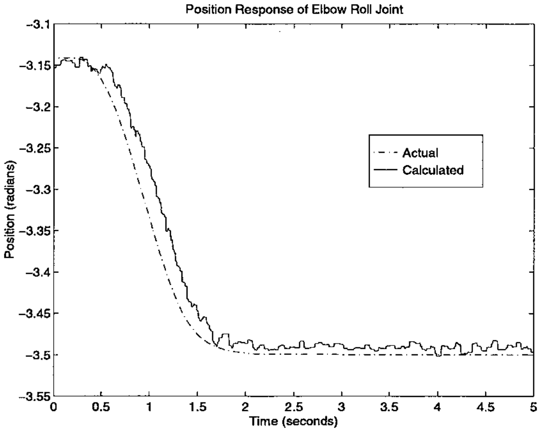


Figure 7.5: Position response of elbow roll joint to trajectory ending at -3.5 with shoulder roll at 0.5

The problematic nature of this joint lends itself to a comparison of the response of the continuation minimax method and the steepest descent method. In a real failure situation, the exact position of the joint will not be known precisely when any fault recovery algorithm is activated. One of the properties of the continuation method, on which continuation minimax is based, is that it has good convergence properties with an large error in initial conditions. Figures 7.6 and 7.7 show the response of continuation minimax and steepest descent with a large initial error of 1 radian. The robot was not moving during this test. The "failure" occurs at one second. Before one second, the 1 radian error between calculated and measured is a result of inducing the position error with the fault recovery algorithm inactive. For a fair comparison, both algorithms were run at the same update rate, 3Hz, and allowed the same number of gradient and function evaluations, 3 per cycle. The continuation minimax shows a slightly better position response than the steepest descent method.

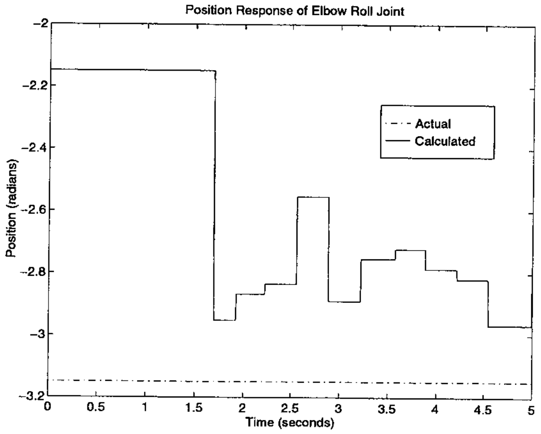


Figure 7.6: Position response of elbow roll joint with continuation minimax and startup error

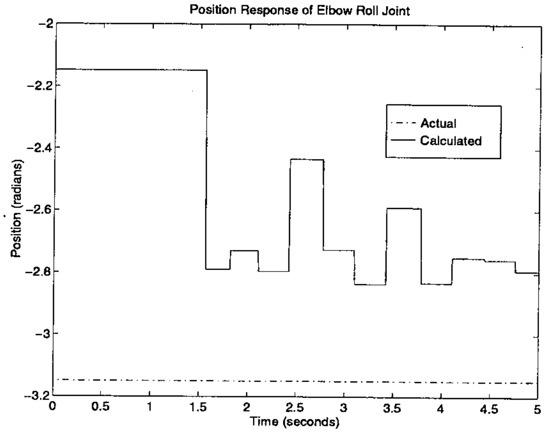


Figure 7.7: Position response of elbow roll joint with steepest descent and startup error

Figure 7.8 shows the response of the steepest descent method run at 60Hz with a SVD calculation at 12Hz. Its response is similar to the slower steepest descent calculation. It should be noted that even with the large initial error, the steepest descent method converged.

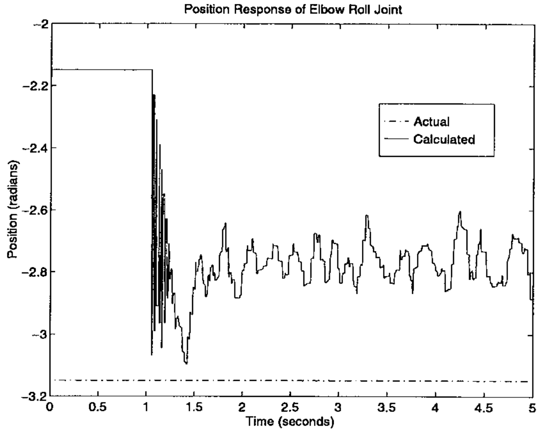


Figure 7.8: Position response of elbow roll joint with steepest descent and startup error calculated at 60Hz

Although the continuation minimax method has some theoretical advantages in robustness to initial error, convergence, and accuracy, these advantages are outweighed in this system by its low maximum update rate.

Returning to the discussion of single joint failures, Figs 7.9-7.12 show the position response of the elbow pitch, wrist roll, wrist pitch, and toolplate roll joints, respectively. All four joints tracked well.

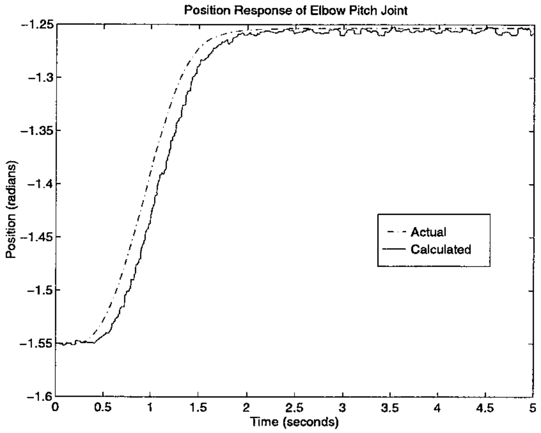


Figure 7.9: Position response of elbow pitch joint to trajectory ending at -1.27

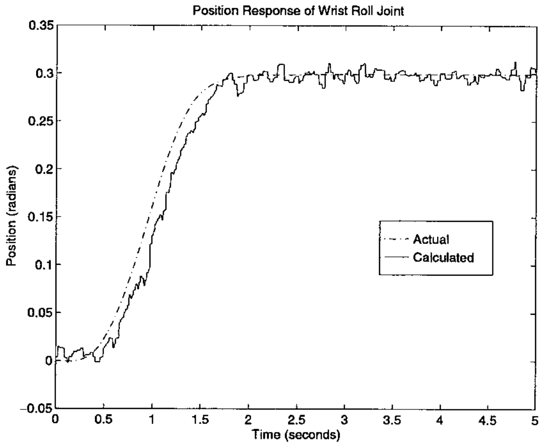


Figure 7.10: Position response of wrist roll joint to trajectory ending at -0.3

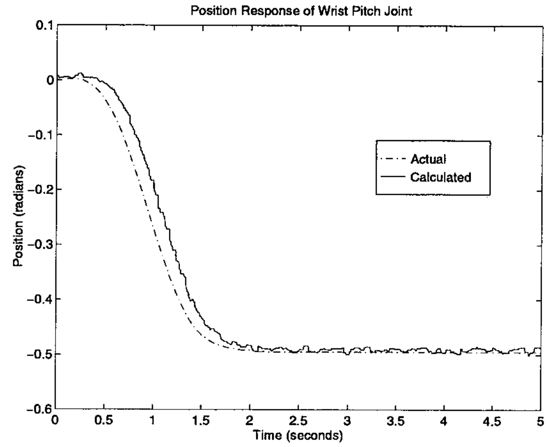


Figure 7.11: Position response of wrist pitch joint to trajectory ending at -0.5

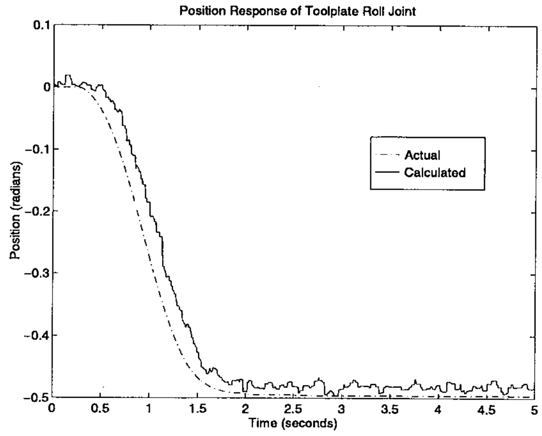


Figure 7.12: Position response of toolplate roll joint to trajectory ending at -0.5

Unlike the joint specific method in chapters 4 and 5, the joint calculations in this method depend on the configuration of joint before and after the "failed" joint in the kinematic chain. The next set of plots shows the response to single joint failures along the trajectory from Home to [0.5 -1.27 -3.5 -1.27 0.3 -0.5 -0.5] radians executed in 1.5 seconds. All robot joints are moving in this trajectory. The position responses can be seen in Figs. 7.13-7.19. Position tracking did deteriorate some from the previous case. Some of this error is due to small synchronization errors between calculated and measured joint values. These errors could be reduced with a higher sample rate or by taking more care in filter selection and phase analysis. Fig 7.15 shows how the tracking of the elbow roll joint improves as its signal to noise ratio increases.

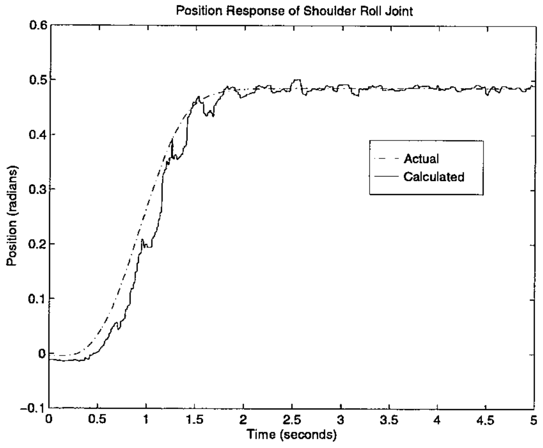


Figure 7.13: Position response of shoulder roll joint to trajectory ending at [0 -1.27 -3.5 -1.27 0.3 -0.5 -0.5] in 1.5 sec

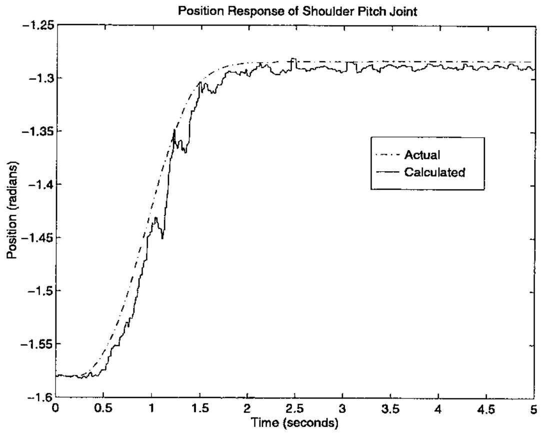


Figure 7.14: Position response of shoulder pitch joint to trajectory ending at [0 -1.27 -3.5 -1.27 0.3 -0.5 -0.5] in 1.5 sec

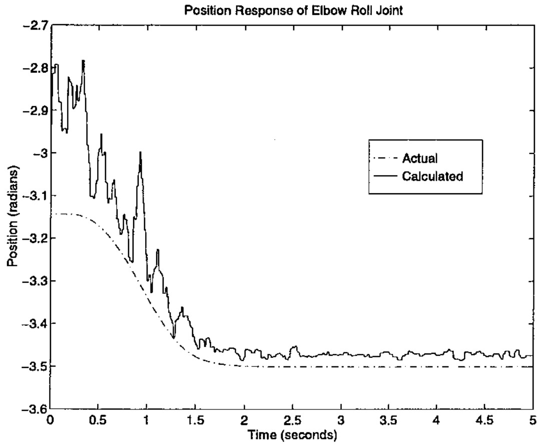


Figure 7.15: Position response of elbow roll joint to trajectory ending [0 -1.27 -3.5 -1.27 0.3 -0.5 -0.5] in 1.5 sec

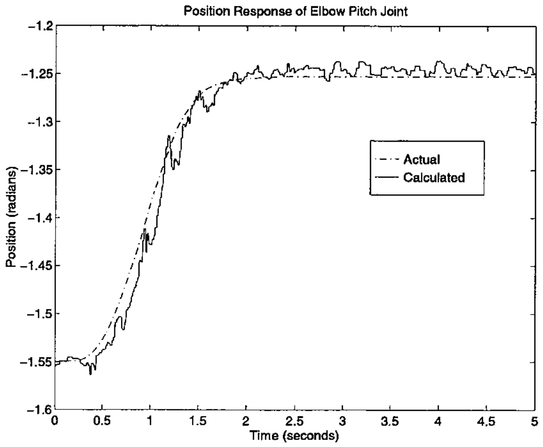


Figure 7.16: Position response of elbow pitch joint to trajectory ending at [0 -1.27 -3.5 -1.27 0.3 -0.5 -0.5] in 1.5 sec.

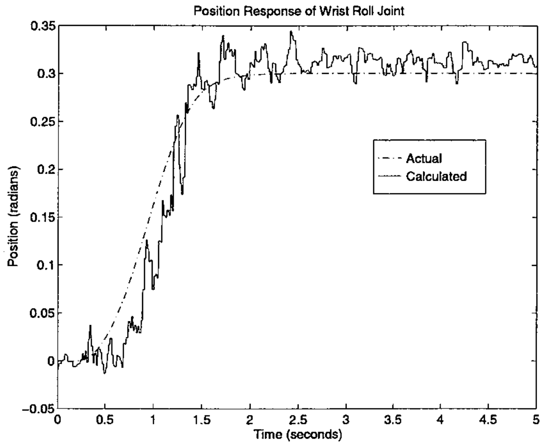


Figure 7.17: Position response of wrist roll joint to trajectory ending at [0 -1.27 -3.5 -1.27 0.3 -0.5 -0.5] in 1.5 sec.

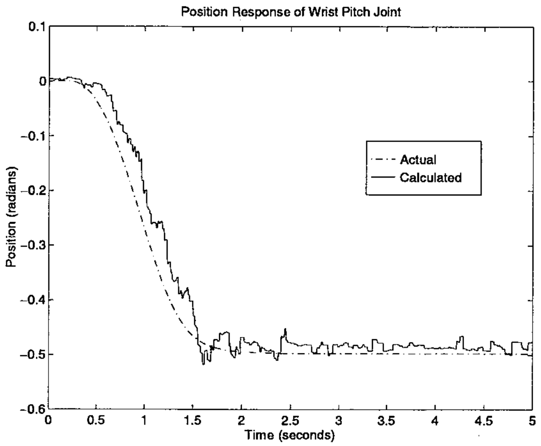


Figure 7.18: Position response of wrist pitch joint to trajectory ending at [0 -1.27 -3.5 -1.27 0.3 -0.5 -0.5] in 1.5 sec.

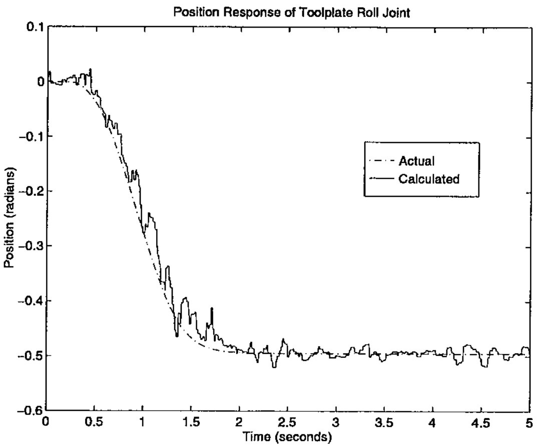


Figure 7.19: Position response of toolplate roll joint to trajectory ending at [0 -1.27 -3.5 -1.27 0.3 -0.5 -0.5] in 1.5 sec

Other trajectories were run with similar results. The main points that should be considered in designing trajectories are:

1. All joints can have configurations where the sensors that contain information about them are parallel to gravity and their signal to noise ratio is low causing errors in the position calculation. Given a particular accelerometer configuration, some joints may be more susceptible to this condition than others.
2. Although gradient condition numbers for the single failure case with this accelerometer configuration are normally low, it should be noted that there could be configurations where the condition number is high and the gradient is not well behaved.
3. Since the gradient is updated at a slow rate there will be a practical limit on the speed which the arm can move and the position solution track the actual position.

It should be noted that the position for all joints were determined with the same algorithm and accelerometer configuration. This flexibility is the main advantage of this method. Although the calculated position may not be as accurate as the joint specific method and the existence of the solution is not guaranteed, fewer accelerometers are required to provide fault tolerance to all joints in the robot.

7.3.2 Multiple joint "failures"

Multiple joint failure is an interesting case because it attempts to extract information about multiple joints from the accelerometer readings. The ability of the algorithm to separate the effects of the position and acceleration of each joint depends on the which joints have failed, the number of failed joints, the optimization algorithm used, and the robot trajectory.

The first case considered is the two failure case. The purpose of presenting this case is for comparison with the two joint failure case in the joint specific experiments in chapter 5. The shoulder pitch and elbow pitch joints are "failed" and the robot moved along a trajectory from Home to [0.5 -1.27 -3.5 -1.27 0 0 0] radians executed in 1.5 seconds. Figures 7.20 and 7.21 show the position response of the failed joints.

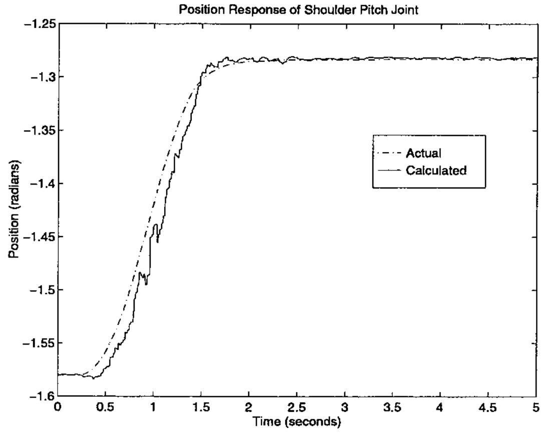


Figure 7.20: Position response of shoulder pitch joint to trajectory ending at [0 1.27 -3.5 -1.27 0 0 0] in 1.5 sec.

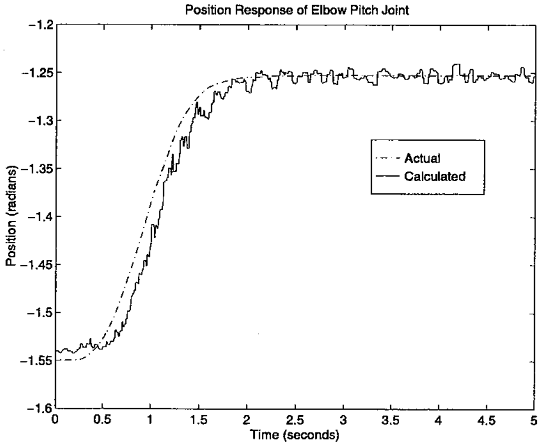


Figure 7.21: Position response of elbow pitch joint to trajectory ending at [0 1.27 -3.5 -1.27 0 0 0] in 1.5 sec

Since this solution method is dependent on the trajectory of all joints, the trajectory from Home to [0.5 -1.27 -3.5 -1.27 0.3 -0.5 -0.5] radians executed in 1.5 seconds to examine the effect of moving the wrist joints on trajectory tracking. Figures 7.22 and 7.23 show the position response of the failed joints. The position tracking response is similar to the case where the wrist joints were not moving.

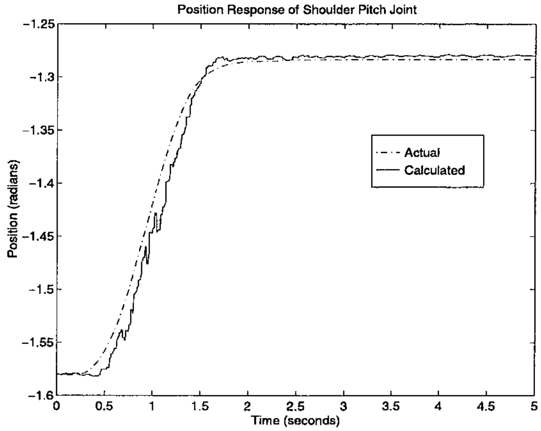


Figure 7.22: Position response of shoulder pitch joint to trajectory ending at [0 1.27 -3.5 -1.27 0.3 -0.5 -0.5] in 1.5 sec

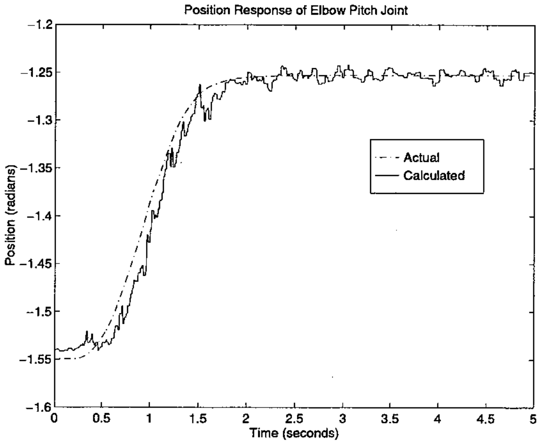


Figure 7.23: Position response of shoulder pitch joint to trajectory ending at [0 1.27 -3.5 -1.27 0.3 -0.5 -0.5] in 1.5 sec

A more interesting case involves three joint failures. Although the accelerometer configuration in this experiment is theoretically capable of recovering from more failures, the three joint failure case is numerically well behaved in most configurations. It would be difficult to justify the continued operation of a robot with four or more failures of any type except in extreme circumstances.

To consider all possible three joint failures and show position tracking on different trajectories would be impractical. As a compromise, six sets of three failures along one trajectory were considered to examine the response to different failure patterns. Afterwards, the response to several trajectories will be considered but with only two sets of failed joints. The joint failure sets are:

For the single trajectory:

Failure set 1: Shoulder roll, elbow pitch, and toolplate roll

Failure set 2: Shoulder pitch, wrist roll, and wrist pitch

Failure set 3: Shoulder pitch, elbow pitch, and wrist roll

Failure set 4: Shoulder roll, elbow roll, and wrist pitch

Failure set 5: Shoulder pitch, elbow pitch, and toolplate roll

Failure set 6: Elbow pitch, wrist pitch, and toolplate toll

For multiple trajectories:

Failure set A: Shoulder roll, elbow pitch, and toolplate roll

Failure set B: Shoulder pitch, elbow pitch, and wrist pitch

For consistency with earlier results, the trajectory chosen for the single trajectory analysis is Home to [0.5 -1.27 -3.5 -1.27 0.3 -0.5 -0.5] radians executed in 1.5 seconds. Figures 7.24-7.26 show the response using failure set 5.

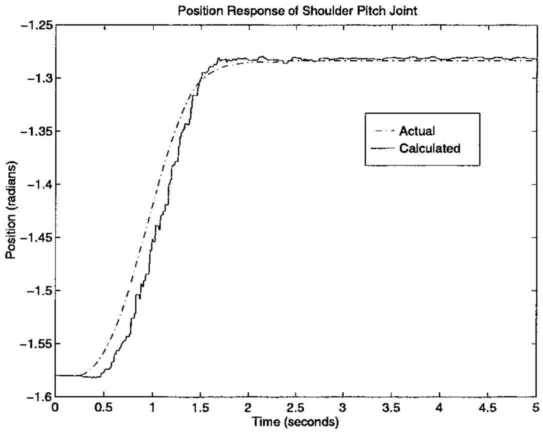


Figure 7.24: Position response of shoulder pitch joint in failure set 5

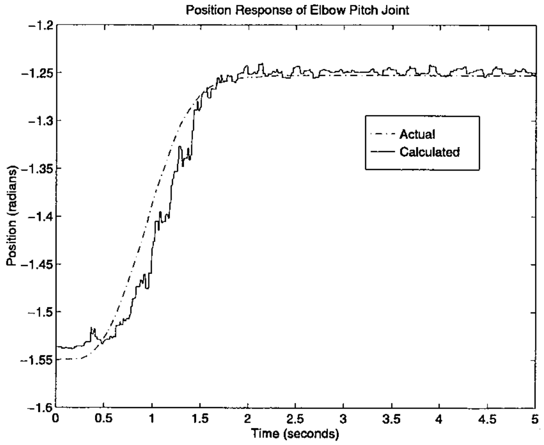


Figure 7.25: Position response of elbow pitch joint in failure set 5

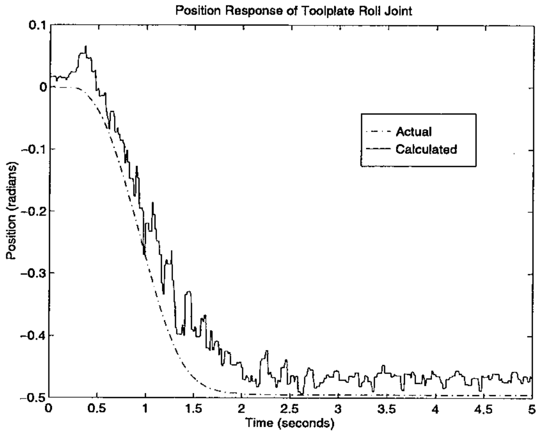


Figure 7.26: Position response of toolplate roll joint in failure set 5

Failure sets 1-4 and 6 showed similar position response with the following exceptions. Since the trajectory began with the elbow roll joint parallel to the gravity gradient, position tracking of the elbow roll joint in failure set 4 was not acceptable until the joint became less parallel and the signal to noise ratio decreased. Unfortunately, in the current accelerometer configuration, all accelerometers have a low signal to noise ratio for the elbow roll calculation in the Home configuration. This situation could be corrected by not mounting all triaxial accelerometers orthogonal to the joint frames. The wrist roll joint did not exhibit good steady state performance in either failure set 2 or 3. An example of this performance is shown in Fig. 7.27. Although not shown, the velocity and acceleration tracked well but contained an offset. During this same trajectory, with only the wrist roll joint failed, the trajectory tracking shown in Fig. 7.17 was acceptable. The additional failures in combination with accelerometer errors led the solution technique to find an optimum that, while close to the desired, did not give the expected results. Adding an accelerometer attached to the wrist pitch joint would improve performance. It should be noted that errors in these joints did not lead to errors of corresponding magnitude in other failed joints.

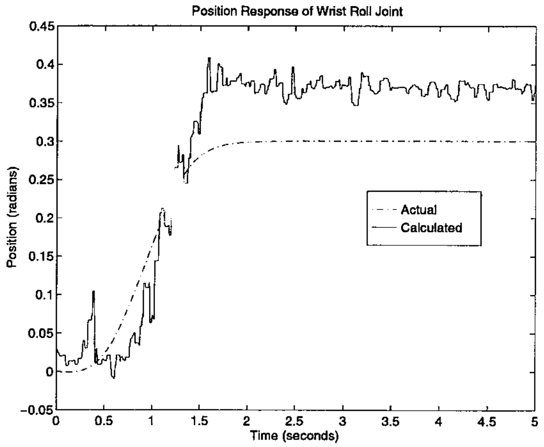


Figure 7.27: Position response of wrist roll joint in failure set 2

Different trajectories using failure sets A and B will now be considered. The first trajectory is a move from Home to [1 -1 -4 -1 -0.5 -0.7 -0.7] executed in 1.5 sec. Figures 7.28-7.30 show the results using failure set A while Fig. 7.31-7.33 show the response to failure set B. Calibration error is responsible for the steady state error. The final configuration is outside the area used by the automatic calibration routine. The smaller calibration area was chosen to give better results close to the Home position. In Fig. 7.30, there is a large tracking error at about 1.5 seconds into the trajectory. This error will be important in the failure recovery case.

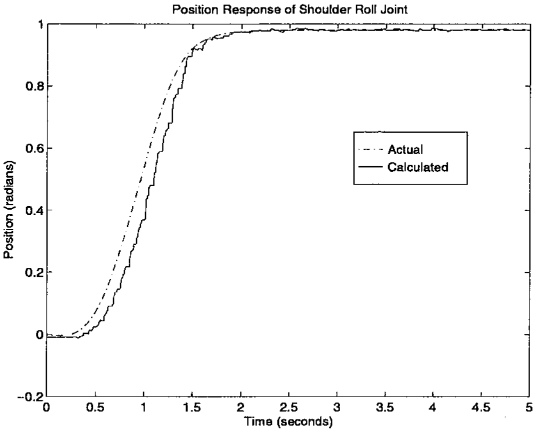


Figure 7.28: Position response of shoulder roll joint in failure set A

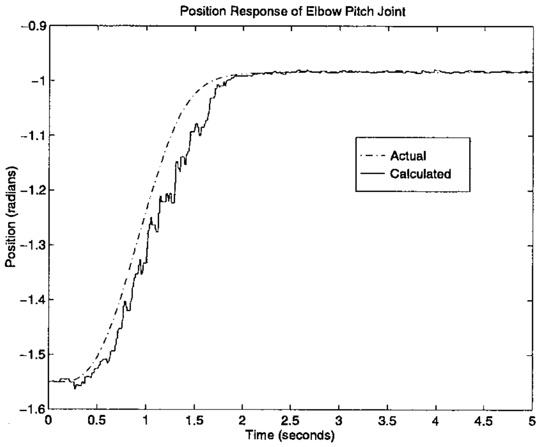


Figure 7.29: Position response of elbow pitch joint in failure set A

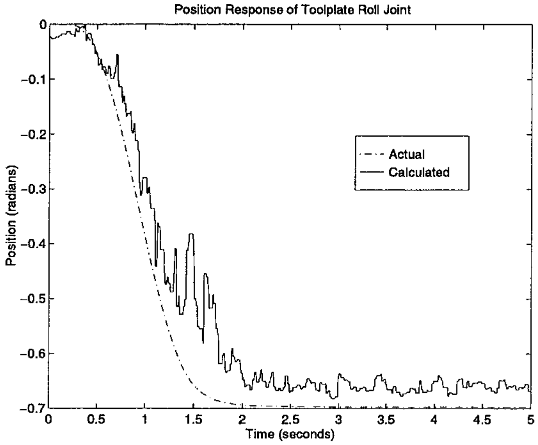


Figure 7.30: Position response of toolplate roll joint in failure set A

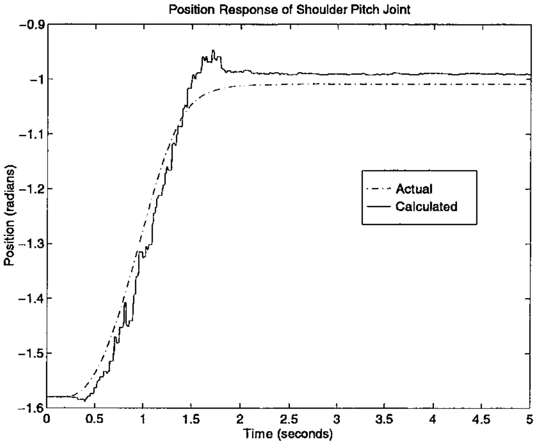


Figure 7.31: Position response of shoulder pitch joint in failure set B

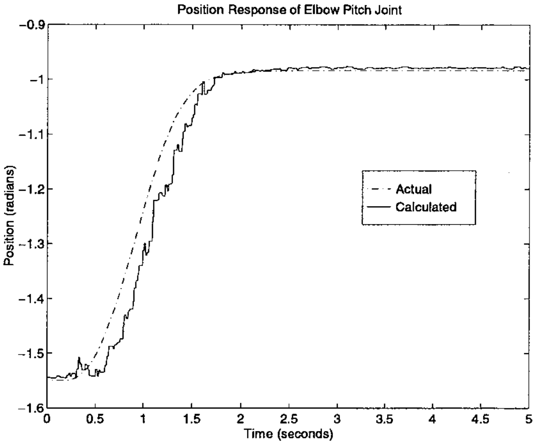


Figure 7.32: Position response of elbow pitch joint in failure set B

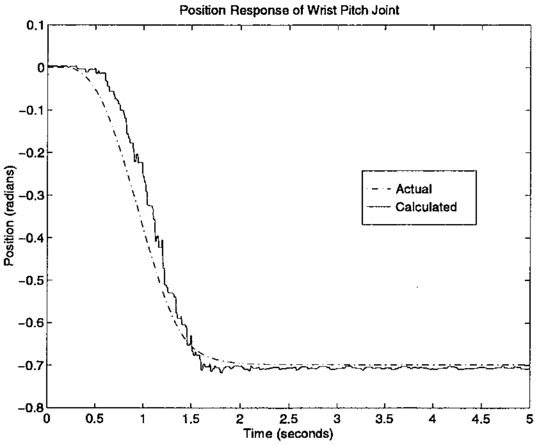


Figure 7.33: Position response of wrist pitch joint in failure set B

Slowing the trajectory to 4.0 sec improves performance. Figures 7.34-7.39 show the response of both failures sets to the slower trajectory.

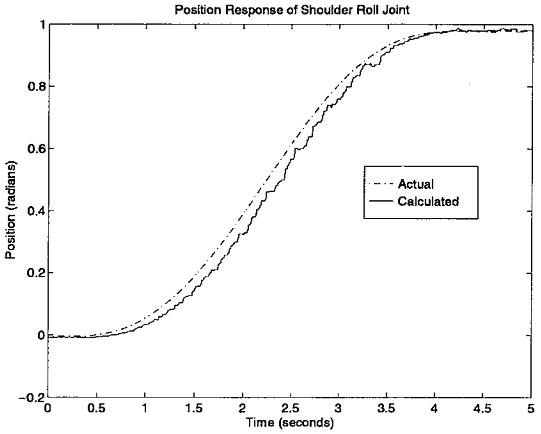


Figure 7.34: Position response of shoulder roll joint in failure set A

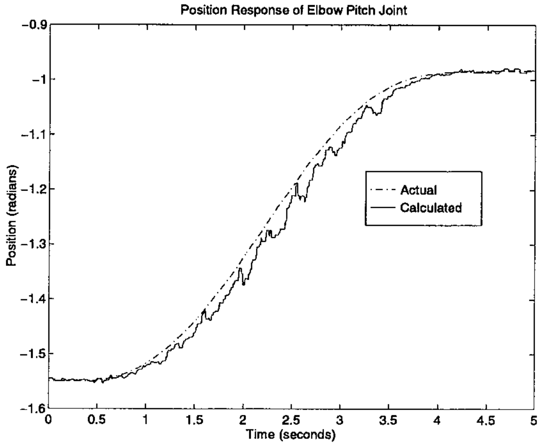


Figure 7.35: Position response of elbow pitch joint in failure set A

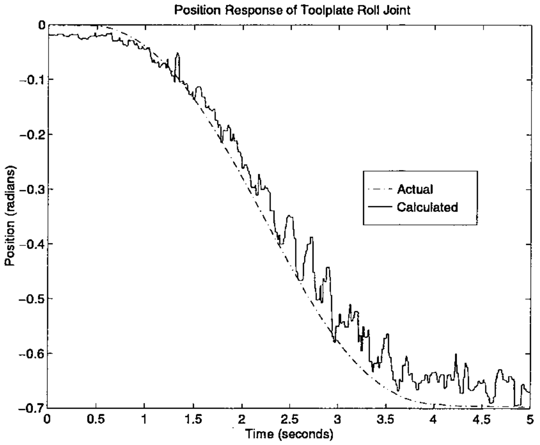


Figure 7.36: Position response of toolplate roll joint in failure set A

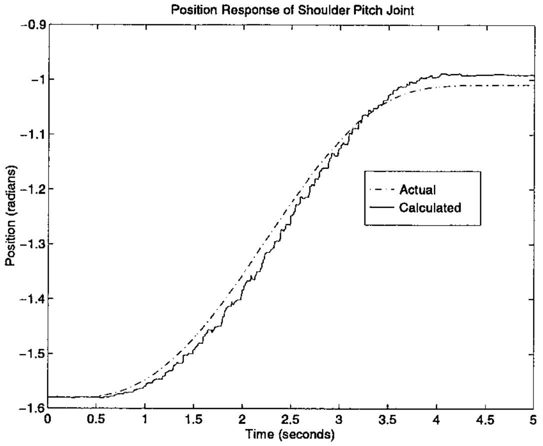
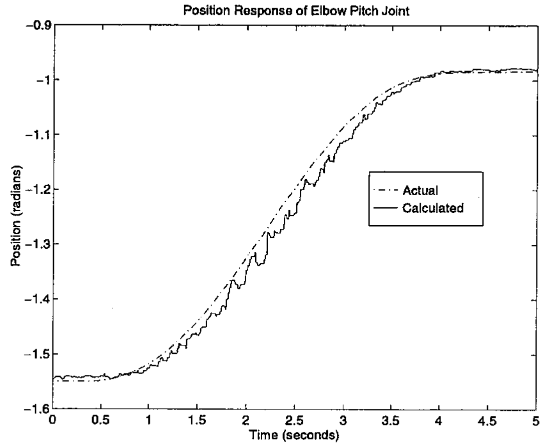


Figure 7.37: Position response of shoulder pitch joint in failure set B



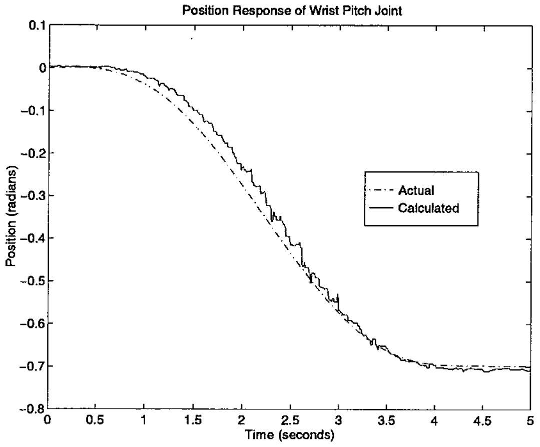
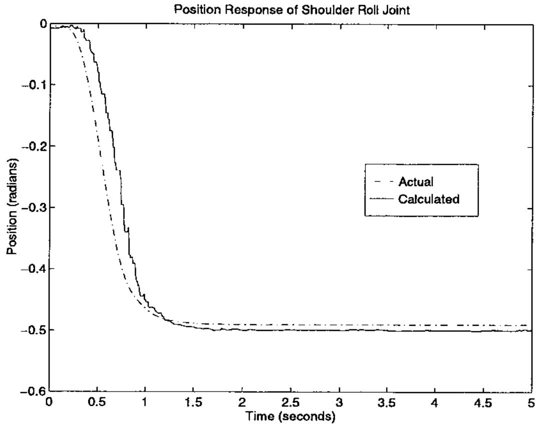


Figure 7.38: Position response of elbow pitch joint in failure set B

Figure 7.39: Position response of wrist pitch joint in failure set B

The final trajectory considered is from Home to [-0.5 -1.77 -2.6 -1.17 0.5 -0.3 0.5] executed in 0.75 sec. This trajectory moves the robot in a quadrant opposite to the trajectories discussed earlier. It also shows the effect of high speed on a shorter motion than that considered in the previous trajectory. As in the previous trajectory, the rate of change of the data is at the edge of the optimization algorithms ability at 60Hz. Figures 7.40-7.42 show the results using failure set A while Fig. 7.43-7.45 show the response to failure set B. Failure set A yields better results than failure set B. Although not presented, slowing the trajectory improves tracking. This case points out that the performance of one set of failures along a trajectory does not necessarily predict the performance of another set containing the same number of failures along the same trajectory.



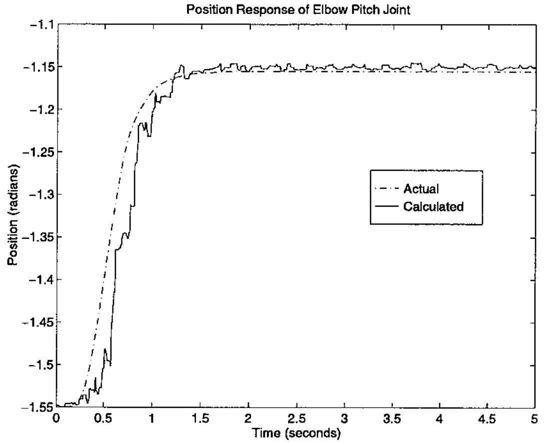


Figure 7.40: Position response of shoulder roll joint in failure set A

Figure 7.41: Position response of elbow pitch joint in failure set A

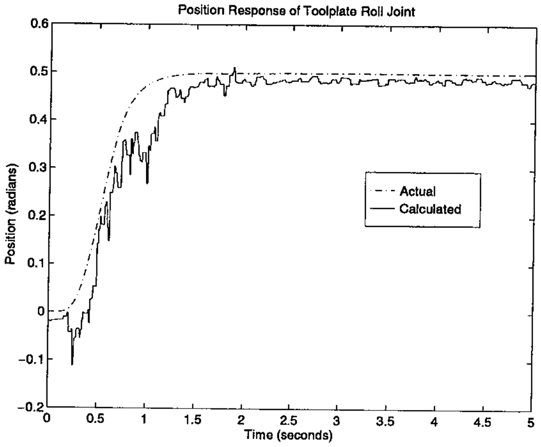


Figure 7.42: Position response of toolplate roll joint in failure set A

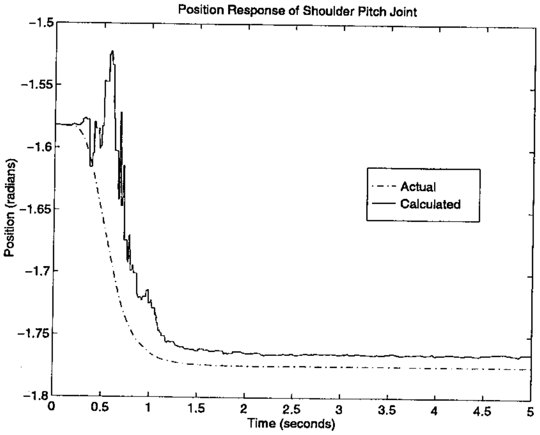


Figure 7.43: Position response of shoulder pitch joint in failure set B

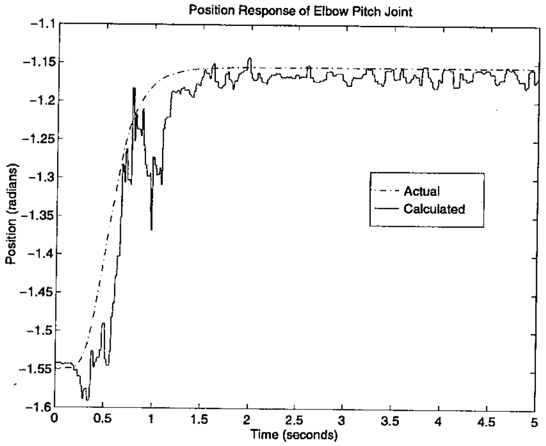


Figure 7.44: Position response of elbow pitch joint in failure set B

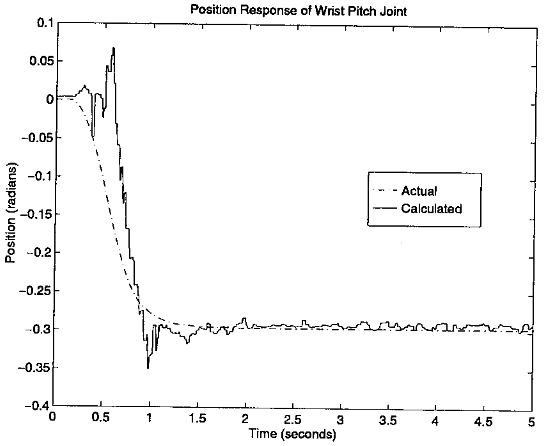


Figure 7.45: Position response of wrist pitch joint in failure set B

The trajectories in this section were arbitrarily chosen. No attempt to analyze the trajectories and choose trajectories with better performance by minimizing the condition number of the gradient was made. In the real system, performance problems due to the manipulators position in the gravity gradient and calibration error were more significant than the gradient's condition number when dealing with the number of failures discussed.

7.4 Failure recovery

The previous sections have concentrated on performance in the FDI mode where the data calculated from the accelerometer outputs was not used as feedback to position the joints. In this section, the calculated data will be used as feedback for the joint servo loops. The goal of this section is to show that the calculated data can be used to stably control up to three failed joints. Since the proposed position feedback method is not optimal, the resulting control cannot be expected to track perfectly. The goal is to make it acceptable for use during failure.

Due to the noise and delay evident on some of the calculated data, a position controller with very high gains would lead to unacceptable performance due to vibration or instability. In addition, since a gravity offset algorithm is dependent on position, any position error would lead to a torque offset error. This situation is an good case to use the virtual passive torque controller detailed in other chapters. The virtual passive controller will provide both gravity and dynamic compensation so the position gains can be lowered. The virtual passive controller can also be tuned to provide adjustable joint damping to trade tracking error for vibration suppression. The use of the virtual passive torque controller assumes that although the joint position and velocity sensors have failed, the joint torque sensor is operational. The position controller will be a simple PD based controller which uses the calculated position and velocity with the trajectory command to offset the measured torque used by the virtual passive controller. This offset appears to the virtual passive controller as movement. The virtual passive controller moves the joint in the desired direction to dissipate the energy.

This application will show the stability of virtual passive control on most of the robot. In chapter 3, a detailed analysis of the properties of the control technique was done concentrating on the wrist joints. In the joint specific accelerometer experiments in chapter 5, the shoulder pitch and elbow pitch joints were controlled by a virtual passive based controller. In this section, all joints of the robot were controlled using the virtual passive control method except the elbow roll joint. The torque sensor for the elbow roll joint in the robot used for this experiment was inoperative. During the following tests, the remaining six joints were always controlled by the passive torque controller with PD position gains suitable for operating with position and velocity feedback from the proposed solution method. When a joint was not failed it used the joint position and velocity feedback from the joint sensors.

7.4.1 Single joint failure recovery

The trajectory considered for single joint failure recovery is from Home to [0.5 -1.27 -3.5 -1.27 0.3 -0.5 -0.5] radians in 1.5 sec. The FDI mode results for this trajectory can be found in Figs. 7.13-7.19. In this trajectory, one joint at a time was failed and the other six joints servoed normally using joint feedback.

Figures 7.46-7.51 show the response to individual joint failures. Due to the real failure of a joint torque sensor, results for the elbow roll joint were not obtained. Although not shown, the response to moving only the failed joint, the FDI response for which is shown in Figs. 7.2-7.3 and 7.9-7.12, is slightly better due to better position tracking performance. Similarly to the previous experiments in failure recovery for the joint specific case the position servo performance is hampered by system delay and low position gains due to noisy position feedback. It should be noted that even though the final position did not meet the exact setpoint, the optimization routine accurately determined the joint position. The wrist roll joint has the worst calculated position of the joints and, as a result, the worst position tracking.

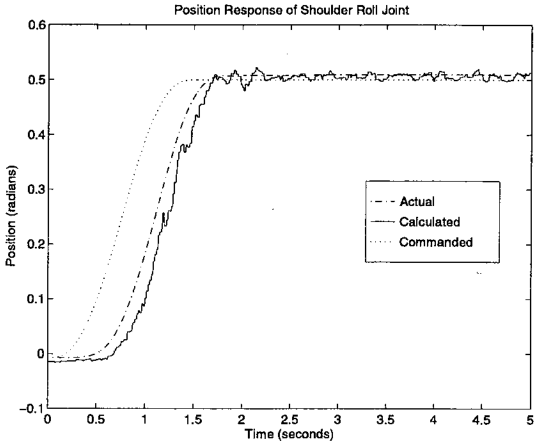


Figure 7.46: Position response of shoulder roll joint to trajectory ending at [0 1.27 -3.5 -1.27 0.3 -0.5 -0.5] in 1.5 sec

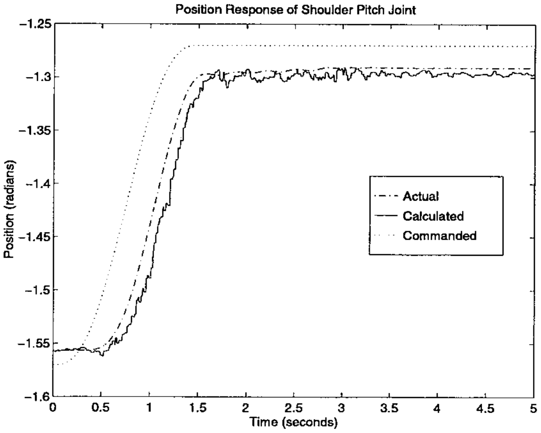


Figure 7.47: Position response of shoulder pitch joint to trajectory ending at [0 1.27 -3.5 -1.27 0.3 -0.5 -0.5] in 1.5 sec

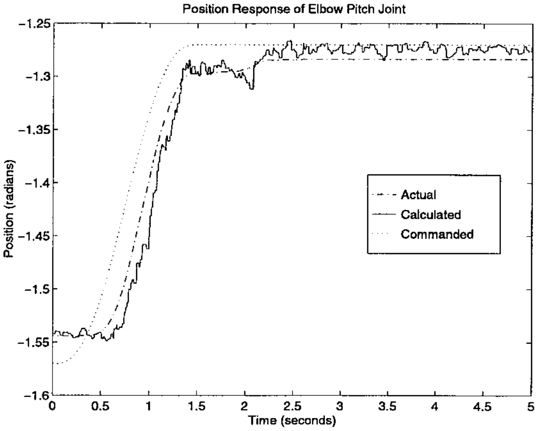


Figure 7.48: Position response of elbow pitch joint to trajectory ending at [0 1.27 -3.5 -1.27 0.3 -0.5 -0.5] in 1.5 sec

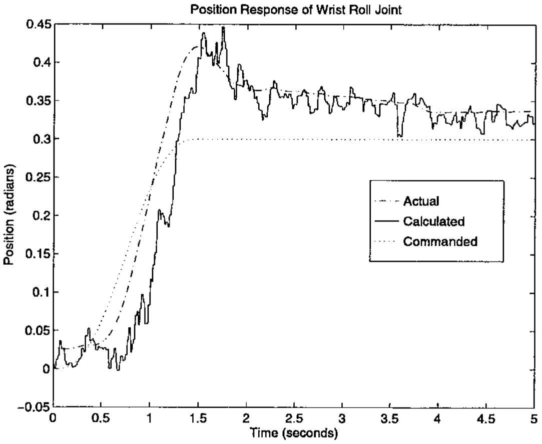


Figure 7.49: Position response of wrist roll joint to trajectory ending at [0 1.27 -3.5 -1.27 0.3 -0.5 -0.5] in 1.5 sec

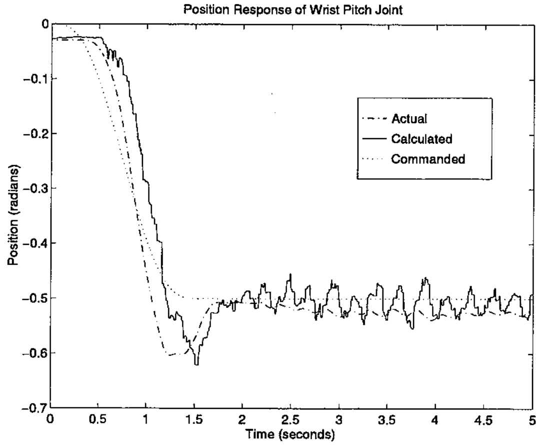


Figure 7.50: Position response of wrist pitch joint to trajectory ending at [0 1.27 -3.5 -1.27 0.3 -0.5 -0.5] in 1.5 sec

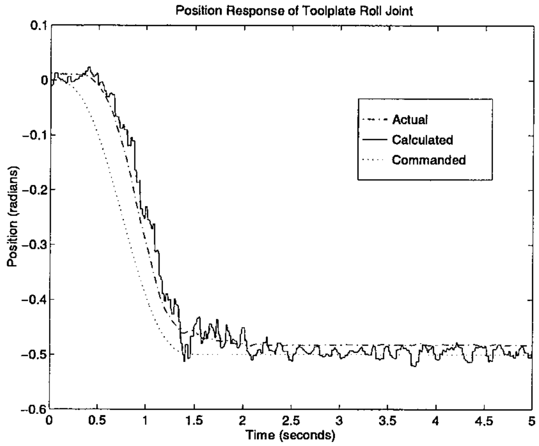


Figure 7.51: Position response of toolplate roll joint to trajectory ending at [0 1.27 -3.5 -1.27 0.3 -0.5 -0.5] in 1.5 sec

Examining Fig. 7.50, it can be seen that the joint had a small oscillation in steady state. This occurred because in the final configuration the joint was under less gravity loading than in the original configuration. This lower gravity loading results in less control authority in the passive controller. At this point, the position controller begins to dominate and the noise in the calculated position induces an oscillation. The tracking performance using the lower gains can be found in Fig. 7.52. Alternatively, the torque control gains could be increased thus increasing damping but reducing position response.

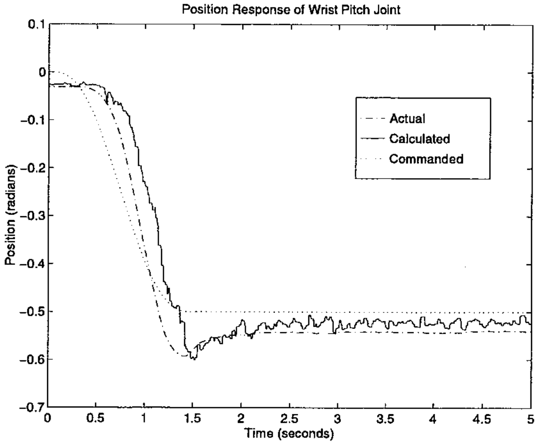


Figure 7.52: Position response of wrist pitch joint with lower position gains

7.4.2 Multiple joint failure recovery

For comparison with the joint specific experiments in chapter 5, the two joint failure involving the shoulder pitch and elbow pitch joints is now considered. Figures 7.53-7.54 show the closed loop response to the commanded trajectory from Home to [0.5 -1.27 -3.5 -1.27 0 0 0] radians in 1.5 sec, the same trajectory used in the joint specific failure recovery analysis. The FDI response to this trajectory is shown in Figs. 7.20-7.21. Since this solution method depends on the trajectories of all joints, the trajectory from Home to [0.5 -1.27 -3.5 -1.27 0.5 -0.5 -0.5] radians in 1.5 sec is also considered. The response to this trajectory can be found in Figs. 7.55-7.56. These trajectories show slightly worse performance than the joint specific case. This lesser performance is due to using accelerometers that are farther away from the failed joints and lower position gains to compensate for noisier position calculations.

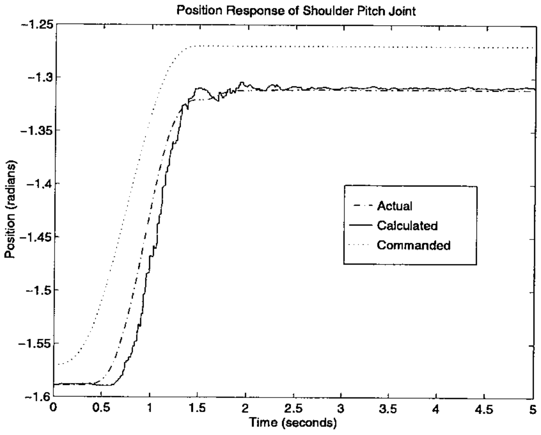


Figure 7.53: Position response of shoulder pitch joint to trajectory ending at [0 -1.27 -3.5 -1.27 0 0 0] in 1.5 sec

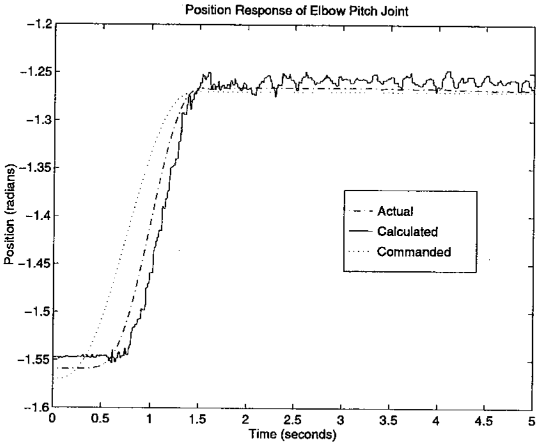


Figure 7.54: Position response of elbow pitch joint to trajectory ending at [0 -1.27 -3.5 -1.27 0 0 0] in 1.5 sec

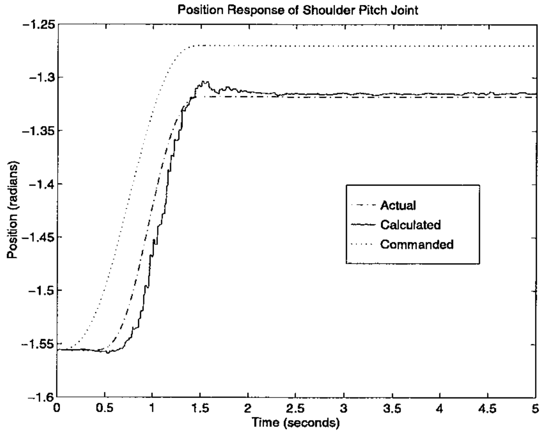


Figure 7.55: Position response of shoulder pitch joint to trajectory ending at [0 -1.27 -3.5 -1.27 0.3 -0.5 -0.5] in 1.5 sec

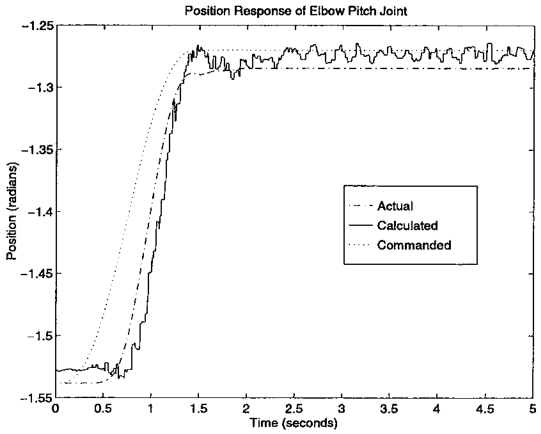


Figure 7.56: Position response of elbow pitch joint to trajectory ending at [0 -1.27 -3.5 -1.27 0.3 -0.5 -0.5] in 1.5 sec

The next failure class involves the simultaneous failure of three joints. Because of the large number of combinations of failed joints and possible trajectories, the following plots will attempt to summarize the effects seen while using closed loop control. The response to using failure set 6 from the multiple joint FDI section on a trajectory from Home to [0.5 -1.27 -3.5 -1.27 0.3 -0.5 -0.5] radians in 1.5 sec can be seen in Figs. 7.57-7.59. Most failures result in good tracking performance on this trajectory and other trajectories whose rate of change was slower than this trajectory's. However, in the cases discussed in the FDI section where the optimization solution did not track well, the position tracked the erroneous position calculation.

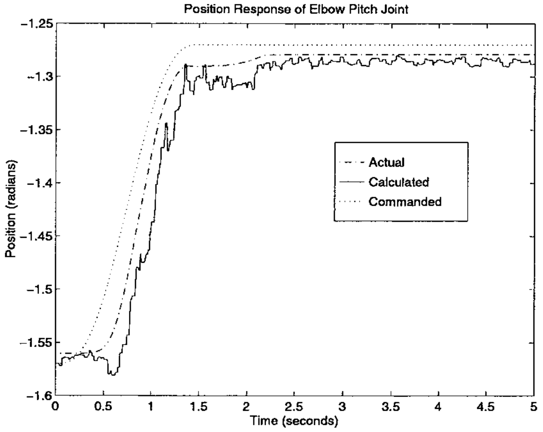


Figure 7.57: Position response of elbow pitch joint in failure set 6

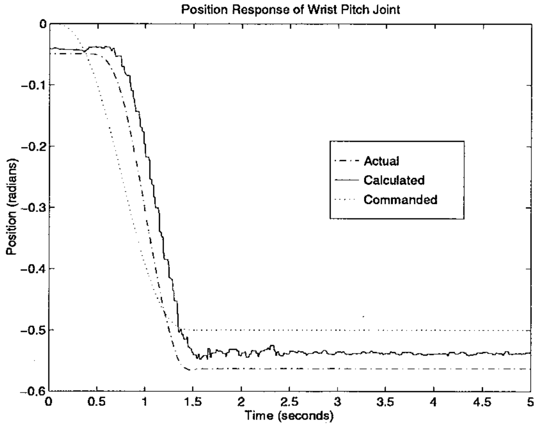


Figure 7.58: Position response of wrist pitch joint in failure set 6

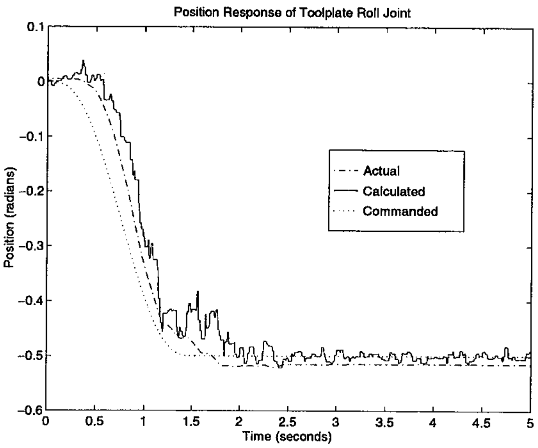


Figure 7.59: Position response of toolplate roll joint in failure set 6

The most severe problem encountered in failure recovery occurred during the trajectory from Home to [1 -1 -4 -1 -0.5 -0.7 -0.7] in 1.5 sec with failure set A. During this trajectory, the toolplate roll joint failed to track the actual trajectory resulting in a runaway. The position calculation error seen in the FDI response in Fig. 7.30 caused the position controller to send out too large a torque command. Although the virtual passive torque controller could possibly be tuned to remain stable during the torque command offset caused by this position error, the resulting joint would be too stiff and highly damped for practical position tracking response. This situation shows that although the virtual passive torque control is inherently stable, that stability property is not guaranteed when the sensed torque is offset to allow for position tracking. If the trajectory is slowed to complete in 4 sec, it completes satisfactorily. Figures 7.60-7.62 show the response in the slower trajectory. If the toolplate roll joint is not considered failed, the trajectory completes satisfactorily in 1.5 sec. Figures 7.63-7.64 show the response of the two failed joints.

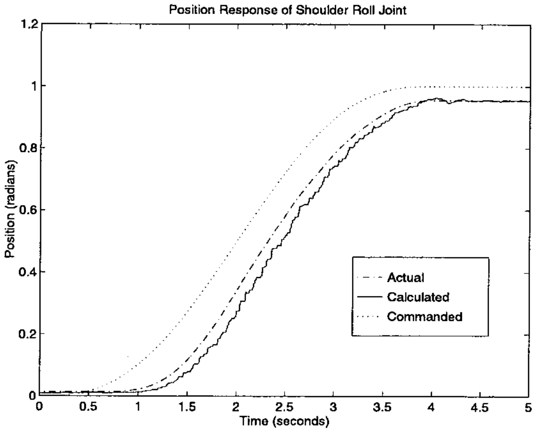


Figure 7.60: Position response of shoulder roll joint in failure set A

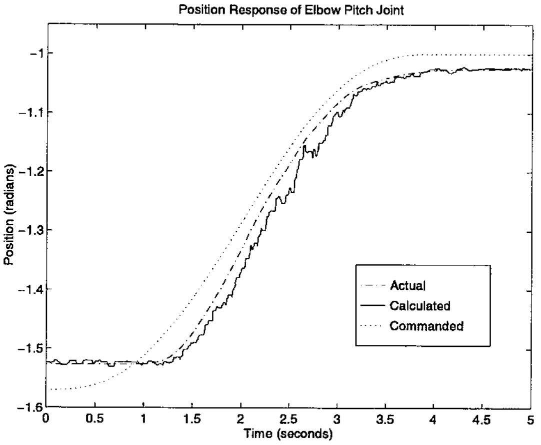


Figure 7.61: Position response of elbow pitch joint in failure set A

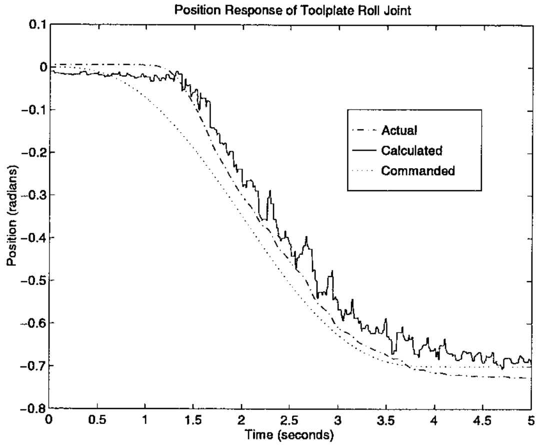


Figure 7.62: Position response of shoulder roll joint in failure set A

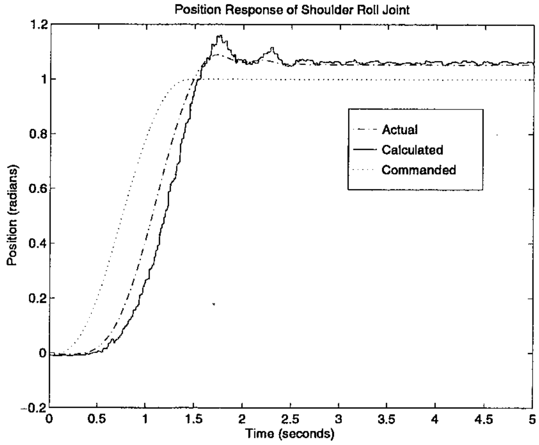


Figure 7.63: Position response of shoulder roll joint with toolplate roll unfailed

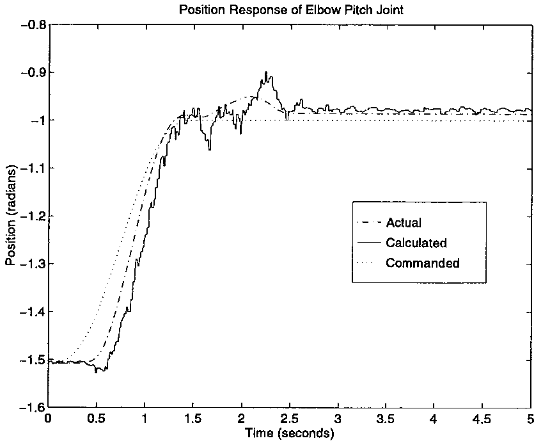


Figure 7.64: Position response of elbow pitch joint with toolplate roll unfailed

The trajectory from Home to [1 -1 -4 -1 -0.5 -0.7 -0.7] in 1.5 sec with failure set B will be considered next. The robot was able to complete this high speed trajectory satisfactorily with failure set B. A trajectory of this speed with this failure set causes overshoot in the control. The overshoot is corrected by the controller and steady state performance is satisfactory. The position response to this trajectory can be seen in Fig. 7.65-7.67. Overshoot can be lowered by either reducing the gains or slowing the trajectory. The results of slowing the trajectory to complete in 4.0 sec. are shown in Fig. 7.68-7.70. If the trajectories to be performed by the robot are not very fast, the position gains should be increased to improve slower trajectory tracking where overshoot in faster trajectories is not a concern.

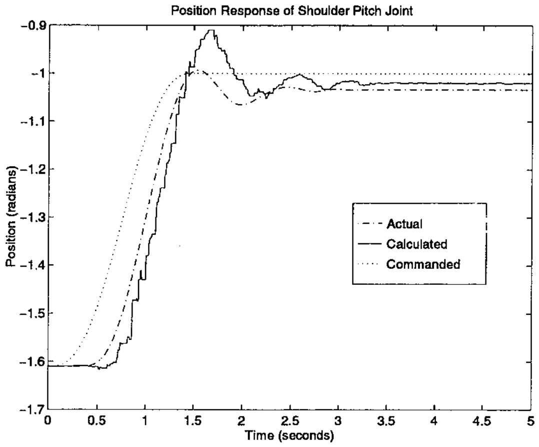


Figure 7.65: Position response of shoulder pitch joint in failure set B to 1.5 sec trajectory

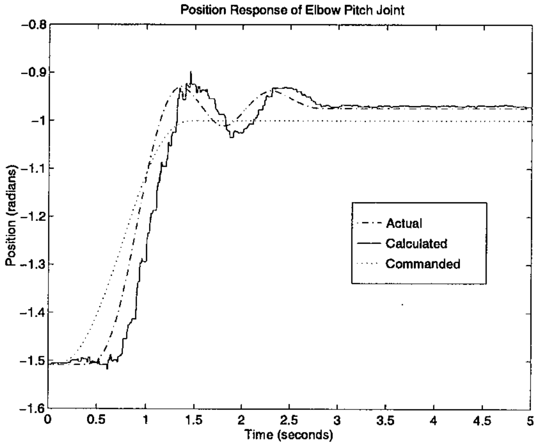


Figure 7.66: Position response of elbow pitch joint in failure set B to 1.5 sec trajectory

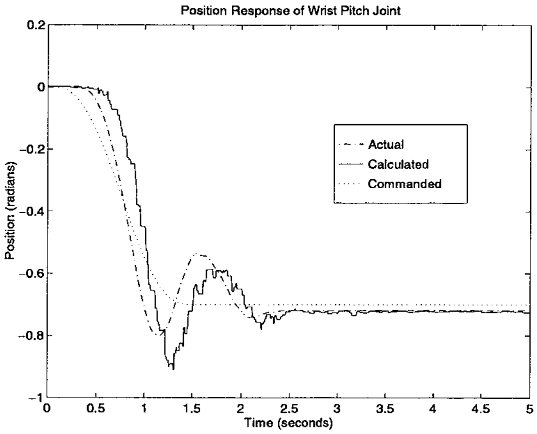


Figure 7.67: Position response of wrist pitch joint in failure set B to 1.5 sec trajectory

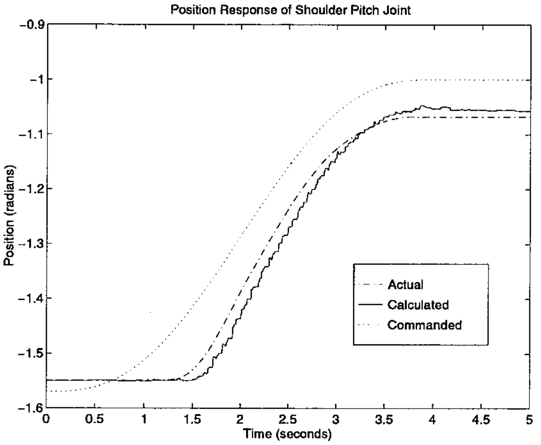


Figure 7.68: Position response of shoulder pitch joint in failure set B to 4.0 sec trajectory

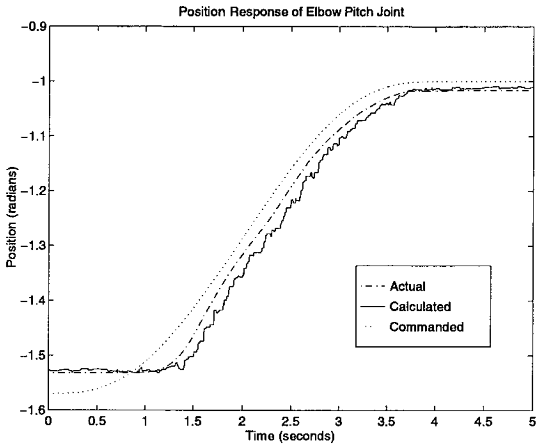


Figure 7.69: Position response of elbow pitch joint in failure set B to 4.0 sec trajectory

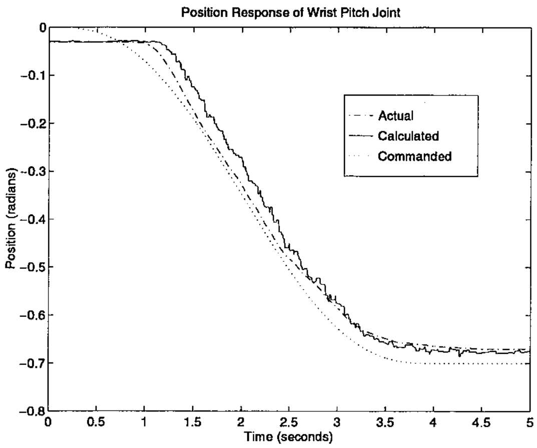


Figure 7.70: Position response of wrist pitch joint in failure set B to 4.0 sec trajectory

Other trajectories were successfully completed with three failures and showed similar error characteristics. To summarize, if used carefully, this combination of an optimization solution and virtual passive torque control can produce satisfactory tracking performance. The problems encountered were:

1. Instability at high speeds with more than two failed joints with certain failure sets and trajectories.
2. Position overshoot in fast trajectories if position gains are increased to improve slower trajectory tracking. Large errors in slow trajectories occur when position gains are lowered to reduce overshoot in fast trajectories.
3. Oscillation if position gains are high enough to dominate torque control when there is little torque feedback. If torque control gains are increased to increase damping to reduce oscillations position tracking performance is reduced.

7.5 Summary

This chapter has presented experimental results of using the system-wide accelerometer position fault tolerance technique. The technique examined differs from the theoretical development by using steepest descent and singular value decomposition instead of continuation minimax as an optimization technique to reduce computational complexity. In addition, the differential velocity model developed in chapter 4 was used to replace the velocity calculation in the optimization to reduce complexity. Results were presented using this technique in FDI and fault recovery modes. This system wide technique performed satisfactorily in most situations with up to three simultaneous joint failures. The results show that the same sensor configuration can determine joint positions of different sets of failed joints. The tracking results were of similar performance to those found in the joint specific accelerometer case. The primary advantage of the system-wide solution method is that it requires fewer accelerometers than the joint specific case to add fault tolerance to all system joint position sensors. The disadvantages include higher computational complexity and less robust convergence properties than the joint specific method

8. Conclusions and Suggestions for Further Work

The goal of this dissertation was to develop a method for position sensor fault tolerance which would not require extensive modification of existing robot designs. Currently, the only acceptable method for long term position sensor fault tolerance in unknown environments is a backup position sensor built into the robot joint. The new method should be designed to work reliably in unstructured environments for an extended duration. Such a method would have applications in space, nuclear, and underwater applications. The sensors chosen to be used in the development of this method were joint torque sensors and Cartesian accelerometers. Torque sensors were chosen because they are commonly available in advanced robots of the type used in the target applications. Cartesian accelerometers were chosen for their availability, small size, low power consumption, and flexibility in placement.

A new torque controller that utilized joint torque feedback, virtual passive control techniques, and a simple joint model was developed. This controller has four interesting properties: it is Lyaponov stable, it does not rely on position feedback for stability, it automatically compensates for unknown end-effector loads and link dynamics, and can act as an active joint brake. As a part of a robot with working sensors, it improves tracking performance by compensating for unmodelled loads and dynamics. It reduces computational complexity because it replaces complex, model based torque computations with sensors and a simple linear controller. The stability of the controller is robust to parameter errors in the joint model. The application of the controller to an articulated robot system with gear driven joints instead of a direct drive, SCARA type arm is new for a torque controller used for dynamic compensation. The experiments discovered some interactions between the position controller and the torque controller that can affect stability. This instability can be corrected by tuning the system gains.

As part of a robot with joint position sensor failures, the controller allows the position control gains to be reduced by compensating for dynamic and static joint loading. The controller gains can be tuned to adjust joint stiffness and damping. This combination of lower gains and adjustable dynamics allow joint position control to be accomplished stably with imperfect joint position feedback. During failure, the automatic transition from a controller that assists joint position performance to an active joint brake by removing the position control inputs gives the controller an elegant safety mode for transition from unfailed operation to failed operation. If the control designer desires, the position control gains can be tuned to allow for a seamless transition from failed operation to unfailed operation at the cost of reduced position tracking performance during normal operation.

Although the in depth analysis of controller performance was done only on three consecutive wrist joints, in later experiments six of the seven joints of a Robotics Research 807i manipulator were controlled by the virtual passive joint controller. The exception being a joint with a non-working torque sensor. In addition, the joint model used in the studies was rough. Further work should include analysis of performance on all joints of a robot with accurate joint models. The true effect of ignoring the nonlinear joint dynamic terms and modeling errors on a real system could be investigated on such a system. No trajectories examined in the experiments involved contact with the environment. Although, as a part of tuning the controller, the end-effector was repeatedly given impulsive forces and was tuned to be stable during these interactions, no real, prolonged contact of the end-effector or the robot links with the environment was studied. Future work should include contact and contact transitions along with the interaction of the controller with various classical end-effector force control techniques.

Position sensor fault tolerance implies a method for position feedback external to the failed position sensor. While the virtual passive torque controller along with a joint torque sensor forms a stable joint without position feedback, for the joint to servo to a position, position feedback must be available to the controller. Two methods to obtain position feedback from Cartesian accelerometers were developed. The main difference between these methods and earlier attempts is that sensed Cartesian acceleration is not integrated to obtain position. In addition, neither method requires a dynamic model of the robot. Both methods use multiple triaxial Cartesian accelerometers, arm kinematics, and a known gravity or acceleration field to obtain position.

The first method is a joint specific method which relies on two triaxial Cartesian accelerometers attached to the link driven by the joint whose position sensor is to be made fault tolerant. This method allows arbitrary placement of accelerometers on the link. However, mathematical simplifications are given for placements which are easily realizable on most manipulators. An iterative solution technique is presented to solve for joint position, velocity, and acceleration. A joint velocity observer is also presented to improve the accuracy of the calculation. Three joints of the Robotics Research manipulator were instrumented with accelerometers. Up to two joints were failed simultaneously and their performance in FDI and fault recovery mode using the virtual passive torque controller was analyzed. The position tracking of the method and solution technique on this real system was adequate for FDI and fault recovery in most situations. The main problem encountered was the low signal to noise ratio of the accelerometers in certain configuration due to the orientation of the joints and sensors to the gravity field.

Since it was only convenient to instrument three joints on this robot, none of which were wrist joints, and only two joints could be failed simultaneously due to A/D limitations, an obvious area for future work is the instrument more joints and fail more joints simultaneously. Even with this limitation, the experiments successfully proved that the position sensors of two of the most important joints in an articulated robot design, the shoulder and elbow pitch joints, can be made fault tolerant with this method. A better insulated wiring and A/D system could be developed that would reduce accelerometer noise and provide better position response. If a cleaner accelerometer signal was available, the system delay caused by filters could be reduced and further experiments performed with a faster position update rate to analyze the improved system response. No experiments conducted involved environmental contact by the robot. Although the filters used have a low enough cutoff frequency to mask out most if the impulse acceleration caused by contact, the effect of a grating contact with imparts a long term, low frequency, high amplitude vibration is unknown. As a result, further experiments should include contact as well as non-contact trajectories. Trajectory generation to avoid arm configurations where accelerometers are parallel to the gravity gradient and have a low signal to noise ratio should also be explored. The simulations and experiments assumed that no information other than torque was available from the failed joint. However, some joint designs include both position and velocity sensors. Experiments should be performed assuming a failed position sensor but using the feedback from a working velocity sensor to improve performance. Although not explored, the solution method should work, probably with lower performance, if there are one or more working joints and no failed joints between the joint with the failed sensor and the dual triaxial accelerometers. Experiments should be conducted to confirm this hypothesis.

The second accelerometer based method is a system wide joint position fault tolerance method. This method involves the use of accelerometers mounted on the end effector and other places along the arm with an optimization algorithm to obtain the position of the joints with failed position sensors. The advantage of the method over the joint specific accelerometer based method is that it requires fewer accelerometers to provide fault tolerance to all joint position sensors. A new optimization method, continuation minimax, was presented as a compromise between robustness and computational complexity. A recursive formulation for the gradient required by optimization methods was also shown. On the computer system used in the experiments, the continuation minimax based method was impractical due to its low update rate. In addition, multiple calculations of the gradient during one iteration overloaded the processor at a low sample rate. As a result, the gradient calculation was broken up over multiple sample periods and the steepest descent method used for optimization. With these changes, joint FDI and fault recovery were demonstrated on up to three failed joints at a time. With the exception of certain fast trajectories and joint configurations which some joints and the accelerometers which contained information about those joints were parallel to the gravity vector, the method produced satisfactory performance for use during a failure.

Further work on the system wide accelerometer method should include a faster computer so the costs and benefits of using the continuation minimax method could be better understood. Different sensor configurations which involve breaking up the triaxial configuration, not mounting all accelerometers orthogonal to the joint coordinate systems, and different placement of the accelerometers should be considered and their effect experimentally shown. Specifically, an accelerometer should be mounted on the wrist pitch joint in this experimental apparatus. During the experiments, it was found that the condition number analysis proposed to determine the solvability of the system given a trajectory and set of failed joints, while successfully showing which combinations would not work, was not conclusive evidence that a trajectory would complete satisfactorily. A new measure combining the condition number, accelerometer error bounds, and the position of accelerometers relative to the gravity vector should be developed. Such a measure could be used in trajectory generation to produce better performance by avoiding problematic regions. The accelerometer configuration used in the experiment was designed to protect the system from multiple, simultaneous joint failures. However, the need to continue operation after only a single joint failure is a more practical goal. Other accelerometer configurations with fewer accelerometers should be tested for fault tolerance to a single joint position sensor failure.

Another possibility for further work in both accelerometer based methods is the more complete analysis of lower speed trajectories. To show the resilience of the methods to dynamic effects and explore the performance with real computational constraints, most trajectories considered are too fast for practical use on a robot with a position sensor failure. Better analysis of the performance possible by tuning controller gains for lower speed trajectories should be explored. Most near term applications of robotics in space, nuclear, and underwater environments involve teleoperation. During teleoperation, the robot is moving slowly and the operator is part of the system feedback loop. The combination of tuning the gains for lower speed operation and allowing the operator to compensate for imperfect position tracking performance may make this fault tolerant control method more practical. In teleoperation, continued, stable operation of the arm during failure is more important that accuracy at high speed.

This dissertation has experimentally shown that either accelerometer based method in conjunction with the virtual passive joint controller can stably servo the joint without position feedback. Again, the methods proposed are backup methods to be used during failure of the sensors required by the primary control method. The accelerometer based methods' performance cannot be fairly compared with controllers utilizing accurate position feedback and perfect dynamic models. However, if accurate position feedback and reasonable joint parameters are available, the virtual passive joint controller design can be compared with classical methods such as computed torque.

This dissertation has successfully developed and experimentally verified methods that allow continued operation of a robot joint after the failure of its joint position sensor using joint torque sensors and Cartesian accelerometers. The methods are practical in cost, integrability into an existing robot system, and computational complexity. If used on moderate speed trajectories, the resulting system is capable of continued operation after the failure of multiple joint sensors.

In conclusion, this dissertation contains the following research contributions:

1. The design, development, and experimental testing of a virtual passive joint torque controller which improves position tracking performance during normal joint operation without a complex dynamic model and can stabilize a robot joint without position feedback.
2. The design, development, and experimental testing of a joint specific method of determining joint position using Cartesian accelerometers without integration for FDI and the integration of the joint

specific position determination method and the virtual passive torque control method for failure recovery.

3. The design, development, and experimental testing of a system wide method of determining joint position using Cartesian accelerometers without integration for FDI and the integration of the system wide position determination method and the virtual passive torque control method for failure recovery.

9. References

[1] Patton, R. Frank, P, and Clark, R., eds., Fault Diagnosis in Dynamic Systems, New York: Prentice Hall, 1989.

[2] Clark, R.N., Fosth, D.C., and Walton, V.M., "Detection of instrument malfunctions in control systems", IEEE Transactions on Aerospace and Electronic Systems, AES-11, pp. 465-473, 1975.

[3] Deckert, J.C., Desai, M.N., Deyst, J.J., and Wilisky, A.S., "DFBW sensor failure identification using analytic redundancy", IEEE Transactions on Automatic Control, AC-22, pp. 795-809, 1977.

[4] Kitamura, M., "Detection of sensor failures in nuclear plant using analytic redundancy", Transactions of the American Nuclear Society, vol. 34, pp. 581-583, 1980.

[5] Frank, P.M., "Fault Diagnosis in Dynamic Systems Using Analytical and Knowledge-based Redundancy- A Survey and Some New Results", Automatica, vol. 24, no. 3, pp. 459-474, 1990.

[6] Frank, P.M., "Enhancement of robustness in observer-based fault detection", International Journal of Control, vol. 59, no. 4, pp. 955-981, 1994.

[7] Emanmi-Naeini, A., Akhter, M.M., and Rock, S.M., "Effect of Model Uncertainty on Failure Detection: The Threshold Selector", IEEE Transactions on Automatic Control, vol. 33, no. 12, pp. 1106-1115, 1988.

[8] Saif, M. and Guan, Y., "A New Approach to Robust Fault Detection and Identification", IEEE Transactions on Aerospace and Electronic Systems, vol. 29, no. 3, pp. 685-695, 1993.

[9] Gao, Z. and Antsaklis, P.J., "Stability of the pseudo-inverse method for reconfigurable control systems", International Journal of Control, vol. 53, no.3, pp. 717-729, 1991.

[10] Jiang, J., "Design of reconfigurable control systems using eigenstructure assignments", International Journal of Control, vol. 59, no.2, pp. 395-410, 1994.

[11] Looze, P.L., Weiss, J.L., Eterno, J.S., and Barrett, N.M., "An Automatic Redesign Approach for Restructurable Control Systems", IEEE Control Systems Magazine, pp. 16-22, May 1985.

[12] Huang, C.Y. and Stengel, R.F., "Restructurable Control Using Proportional-Integral Implicit Model Following", Journal of Guidance, Navigation, and Control, vol. 13, no. 2, pp. 303-309, 1990.

[13] Wu, N.E. and Chen, T., "Failure Sensitivity and Robustness in Reconfigurable Flight Control Systems", Proceedings of the National Aerospace and Electronics Conference (NAECON 94), pp. 548-555, 1994.

[14] Horak, D.T., "Failure Detection in Dynamic Systems with Modeling Errors", Journal of Guidance, Control, and Dynamics, vol. 11, no. 6, pp. 508-516, 1987.

[15] Visinsky, M.L., Walker, I.D., and Cavallaro, J.R., "New Dynamic Model-Based Fault Detection Thresholds for Robot Manipulators", 1994 IEEE International Conference on Robotics and Automation, pp. 1388-1395, 1994.

[16] Freyermuth, B., "An Approach to Model Based Fault Diagnosis of Industrial Robots", 1991 IEEE International Conference on Robotics and Automation, pp. 1350-1356, 1991.

[17] Visinsky, M.L., Walker, I.D., and Cavallaro, J.R., "Layered Dynamic Fault Detection and Tolerance for Robots", 1993 IEEE International Conference on Robotics and Automation, pp. 180-187, 1993.

[18] Wu, E.C., Hwang, J.C., and Chladek, J.T., "Fault-tolerant joint development for the Space Shuttle remote manipulator system: analysis and experiment", IEEE Transactions on Robotics and Automation, vol. 9, no. 5, pp. 675-684, 1993.

[19] Bergerman, M. and Xu, Y., "Robust Control of Underactuated Manipulators: Analysis and Implementation", Carnegie Mellon University Technical Report, CMU-RI-TR-94-12, 1994.

[20] Pradeep, A.K., Yoder, P.J., Mukundan, R., and Schilling, R.J., "Crippled Motion in Robots", IEEE Transactions on Aerospace and Electronic Systems, vol. 24, no. 1, pp. 2-13, 1988.

[21] Paredis, C.J.J., Au, W.K.F., and Khosla, P.K., "Kinematic Design of Fault Tolerant Manipulators", Computers and Electrical Engineering, vol. 20, no. 3, pp. 211-220, 1994.

[22] Maciejewski, A.A., "Fault Tolerant Properties of Kinematically Redundant Manipulators", 1990 IEEE International Conference on Robotics and Automation, vol. 1, pp. 638-642, 1990.

[23] Lewis, C.L. and Maciejewski, A.A., "Dexterity Optimization of Kinematically Redundant Manipulators in the Presence of Joint Failures", Computers and Electrical Engineering, vol. 20, no. 3, pp. 273-288, 1994.

[24] Paredis, C.J.J., and Khosla, P.K., "Global Trajectory Planning for Fault Tolerant Manipulators", 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems, vol. 2, pp. 428-434, 1995.

[25] Ting, Y., Tosunoglu, S., and Tesar, D., "A Control Structure for Fault-Tolerant Operation of Robotic Manipulators", 1993 IEEE International Conference on Robotics and Automation, vol. 1, pp. 684-690, 1993.

[26] Ting, Y., Tosunoglu, S., and Fernandez, B., "Control Algorithms for Fault-Tolerant Robots", 1994 IEEE International Conference on Robotics and Automation, vol. 1, pp. 910-915, 1994.

[27] Ting, Y., Tosunoglu, S., and Freeman, R., "Torque Redistribution Method for Fault Recovery in Redundant Serial Manipulators", 1994 IEEE International Conference on Robotics and Automation, vol. 2, pp. 1396-1401, 1994.

[28] Paredis, C.J.J., Brown, H.B., and Khosla, P.K., "A Rapidly Deployable Manipulator System", 1996 IEEE International Conference on Robotics and Automation, vol. 2, pp. 1434-1439, 1996.

[29] Schmitz, D., Khosla, P., and Kanade, T., "The CMU Reconfigurable Modular Manipulator System", Proceedings of the International Symposium and Exposition on Robots, pp. 473-488, 1988.

[30] Kelmar, L., and Khosla, P.K., "Automatic Generation of Forward and Inverse Kinematics for a Reconfigurable Modular Manipulator System", Journal of Robotic Systems, vol. 7, no. 4, pp. 599-619, 1990.

[31] Stewart, D.B., Schmitz, D.E., and Khosla, P.K., "Implementing Real-time Robotics Systems Using CHIMERA II", 1990 IEEE International Conference on Robotics and Automation, vol. 1, pp. 598-603, 1990.

[32] Gertz, M., Stewart, D., and Khosla, P., "A Software Architecture-based Human-machine Interface for Reconfigurable Sensor-based Control Systems", Proceedings of the 8th IEEE International Symposium on Intelligent Control, pp. 75-80, 1993.

[33] Visinsky, M.L., Carvallaro, J.R., and Walker, I.D., "Expert System Framework for Fault Detection and Fault Tolerance in Robotics", Computers and Electrical Engineering, vol. 20, no. 5, pp. 421-435, 1994.

[34] Abdallah, C., Dawson, D., Dorato, P., and Jamshidi, M., "Survey of Robust Control for Rigid Robots", IEEE Control Systems Magazine, vol. 11, no. 2, pp. 24-30, 1991.

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

[36] Qu, Z., and Dawson, D.M., Robust Tracking Control of Robot Manipulators, New York: IEEE Press, 1996.

[37] Kosuge, K., Takeuchi, H., and Furita, K., "Motion Control of a Robot Arm Using Joint Torque Sensors", IEEE Transactions on Robotics and Automation, vol. 6, no. 2, pp. 258-263, 1990.

[38] Imura, J., Yokohohji, Y., and Yoshikawa, T., "Robust Control of Robot Manipulators Based on Joint Torque Sensor Information", International Journal of Robotics Research, vol. 13, no. 5, pp. 434-442, 1994.

[39] Padgaonkar, A.J., Krieger, K.W., and King, A.I., "Measurements of Angular Acceleration of a Rigid Body Using Linear Accelerometers", Transactions of the ASME, September 1975, pp. 552-556, 1975.

[40] Nusholtz, G.S., "Geometric Methods in Determining Rigid-body Dynamics", Experimental Mechanics, vol. 33, no. 2, pp. 153-158, 1993.

[41] Pisacane, V.L., and Moore, R.C., eds, Fundamentals of Space Systems, London: Oxford University Press, 1994.

[42] Kotnik, P.T., Yurkovich, S., and Ozguner, U., "Acceleration Feedback Control for a Flexible Manipulator Arm", 1988 IEEE Conference on Robotics and Automation, vol. 1., pp. 322-23, 1988.

[43] Khorami, F., and Jain, S., "Experimental results on an inner/outer loop controller for a two-link flexible manipulator", 1992 IEEE Conference on Robotics and Automation, vol. 1., pp. 742-747, 1992.

[44] Khorami, F., and Zheng, S., "Vibration control of flexible-link manipulators", 1990 American Control Conference, vol. 1, pp. 175-80, 1990.

[45] Milford, R.I., Asokanthan, S.F., and Gilmore, D.B., "Experimental On-line Identification of an Elastic Robot Manipulator", Journal of Electrical and Electronics Engineering, vol. 13, no. 4, pp. 321-7, 1993.

[46] Bennekers, B., and Teal, H., "Robot Runaway Protection System", 1988 IEEE Conference on Robotics and Automation, vol. 3, pp. 1474-76, 1988.

[47] Baerveldt, A.-J. and Vischer, D., "A runaway protection system for robots based on acceleration measurements", Robot Control 1991 (SYROCO '91). Selected Papers from the 3rd IFAC/IFIP/IMACS Symposium, pp. 531-535, 1991.

[48] Zimmerman, M. and Sulzer, W., "High bandwidth orientation measurement and control based on complementary filtering", Robot Control 1991 (SYROCO '91). Selected Papers from the 3rd IFAC/IFIP/IMACS Symposium, pp. 525-530, 1991.

[49] Howe, R.D., and Cutkosky, M.R., "Sensing skin acceleration for slip and texture perception", 1989 IEEE Conference on Robotics and Automation, vol. 1., pp. 145-50, 1989.

[50] Koivo, A.J., Fundamentals for Control of Robotic Manipulators, New York: John Wiley and Sons, 1989.

[51] Bortoff, S.A. and Spong, M.W., "Pseudolinearization of the Acrobot using Spline Functions", Proceedings of the 31st Conference on Decision and Control, vol. 1, pp. 593-598, 1992.

[52] Luh, J.Y., Walker, M., and Paul, R.P., "Resolved acceleration control for mechanical manipulators", IEEE Transactions on Automatic Control, vol. 25, pp. 468-474, 1980.

[53] Hung, J.Y., Gao, W., and Hung, J.C., "Variable Structure Control: A Survey," IEEE Transactions on Industrial Electronics, vol.40, no. 1, pp.2-22, 1993.

[54] Siacrd, P. and Wen, J., "Application of a Passivity Based Control Methodology for Flexible Joint Robots to a Simplified Space Shuttle RMS", American Control Conference, vol. 2, pp. 1690-1694, 1992.

[55] Berghuis, H. and Nijmeijer, H., "A Passivity Approach to Controller-Observer Design for Robots", IEEE Transactions on Robotics and Automation, vol. 9, no. 6, 1993.

[56] De Santis, A, and Lanari, L., "Stabilization and control of a flexible structure continuum model", Proceedings of 32nd Conference on Decision and Control, vol. 4, pp. 3210-3215, 1993.

[57] Juang, J., Wu, S., Phan, M., and Longman, R.W., "Passive Dynamic Controllers for Nonlinear Mechanical Systems", AIAA Journal of Guidance, Control, and Dynamics, vol. 16, no. 5, pp. 845-851, 1993.

[58] Craig, J., Introduction to Robotics: Mechanics and Control, 2nd edition, Reading: Addison Wesley, 1989.

[59] Oh, H., Vadali, S.R., and Junkins, J.L., "Use of the Work-Energy Rate Principle for Designing Feedback Control Laws", AIAA Journal of Guidance, Control, and Dynamics, vol. 15, no. 1, pp. 275-277, 1992.

[60] Nethery, J., "Robotica: A structured environment for computer aided design and analysis of robots," Master's thesis, University of Illinois, Urbana, IL, 1993.

[61] Chladek, J.T., "Fault Tolerance for Space Based Manipulator Mechanisms and Control System", First International Symposium On Measurement and Control in Robotics, 1990.

[62] Press, W.H., Flannery, B.P., Teukolsky, S.A., and Vetterling, W.T., Numerical Recipes in C, Cambridge: Cambridge University Press, 1988.

[63] More, J.J., "The Levenberg-Marquardt Algorithm: Implementation and Theory", Numerical Analysis, ed. G.A. Watson, Springer-Verlag, pp. 105-116, 1977.

[64] Richter, S.L. and DeCarlo, R.A., "Continuation Methods: Theory and Applications", IEEE Transactions on Automatic Control, vol. AC-28, no. 6, pp. 660-665, 1983.

[65] Horta, L.G., Juang, J-N., and Junkins, J.L., "A Sequential Linear Optimization Approach for Controller Design", Journal of Guidance, Control, and Dynamics, vol. 9, no. 8, pp. 699-703, 1986.

[66] Madsen, K., "An Algorithm for Minimax Solution of Overdetermined Systems of Non-linear Equations", Journal of the Institute of Mathematics and Applications, vol. 16, pp. 321-328, 1975.

[67] Bandler, J.W., Kellerman, W., and Madsen, K., "A Superlinearly Convergent Minimax Algorithm for Microwave Circuit Design", IEEE Transactions on Microwave Theory and Techniques, vol. MTT-33, no. 12, pp. 1519-1530, 1985.

[68] Gerald, C.F. and Wheatly, P.O., Applied Numerical Analysis, 5th edition, Reading: Addison-Wesley, 1994.

[69] Strang, G., Linear Algebra and its Applications, 3rd edition, San Diego: Harcourt Brace Jovanovich, 1988.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE January 1997	3. REPORT TYPE AND DATES COVERED Technical Memorandum

4. TITLE AND SUBTITLE

Robot Position Sensor Fault Tolerance

5. FUNDING NUMBERS

519-30-21-04

6. AUTHOR(S)

Hal A. Aldridge

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

NASA Langley Research Center
Hampton, VA 23681-0001

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Washington, DC 20546-0001

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

NASA TM 110314

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION / AVAILABILITY STATEMENT

Unclassified-Unlimited
Subject Category 31

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

Robot systems in critical applications, such as those in space and nuclear environments, must be able to operate during component failure to complete important tasks. One failure mode that has received little attention is the failure of joint position sensors. Current fault tolerant designs require the addition of directly redundant position sensors which can affect joint design. A new method is proposed that utilizes analytical redundancy to allow for continued operation during joint position sensor failure. Joint torque sensors are used with a virtual passive torque controller to make the robot joint stable without position feedback and improve position tracking performance in the presence of unknown link dynamics and end-effector loading. Two Cartesian accelerometer based methods are proposed to determine the position of the joint. The joint specific position determination method utilizes two triaxial accelerometers attached to the link driven by the joint with the failed position sensor. The joint specific method is not computationally complex and the position error is bounded. The system wide position determination method utilizes accelerometers distributed on different robot links and the end-effector to determine the position of sets of multiple joints. The system wide method requires fewer accelerometers than the joint specific method to make all joint position sensors fault tolerant but is more computationally complex and has lower convergence properties. Experiments were conducted on a laboratory manipulator. Both position determination methods were shown to track the actual position satisfactorily. A controller using the position determination methods and the virtual passive torque controller was able to servo the joints to a desired position during position sensor failure.

14. SUBJECT TERMS

Robot joint control; Fault tolerance; Torque sensor; Accelerometer

15. NUMBER OF PAGES

168

16. PRICE CODE

A08

17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT