ROBUST NONLINEAR TRACKING CONTROL

OF

ROBOTIC MANIPULATORS

Thesis submitted for the degree of
Doctor of Philosophy
at the University of Leicester

by

Liqun Yao B.Sc. (China)
Department of Engineering
University of Leicester

August 1999
To Weidong and Bingyuan
Statement of Originality

The accompanying thesis, submitted for the degree of Doctor of Philosophy, entitled 'Robust nonlinear tracking control for robotic manipulators', is based on work undertaken by the author in the Department of Engineering at the University of Leicester during the period between October 1993 and March 1999. As far as the author is aware, all of the work described within is original unless otherwise acknowledged in the text or in the references. None of the work has been submitted for any other degree in this or any other university.

Liqun Yao

August 1999
Robust Nonlinear Tracking Control Of
Robotic Manipulators

Abstract

This thesis has discussed the development and implementation of robust nonlinear tracking control for a parallel and serial topology Tetrahedral robot (Tetrabot), although the theoretical control strategy presented is applicable to any manipulator tracking problem. The design of robust tracking controllers involves deriving a tracking law for uncertain dynamic systems, such that the actual positions closely track desired trajectories. Two new schemes, a robust sliding mode control and a Lyapunov-based robust tracking control, have been presented for uncertain dynamical systems in the presence of model uncertainty and disturbances. The foci of this study are the concepts and techniques of robust nonlinear tracking control with a bias toward industrial applications.

The Tetrabot system structure, hardware, software and the results of implementation on the three degree of freedom parallel geometry have been studied. In order to implement robust tracking control laws, the Tetrabot system software has been further developed. Most importantly, the results of implementation of a nonlinear tracking controller on the Tetrabot rig facility are also studied. To demonstrate the performance attainable by this control strategy, the trajectory involved movement across the primary working volume to the end-effect point which is the largest distance possible and involved the continuous motion; such a motion will invoke a wide range of the possible nonlinear dynamic representations. The proposed control strategy is robust to variations in robot loading. The experimental results obtained for the closed-loop response indicate that compensation, which employs explicit off-line parameter estimation, can improve tracking accuracy significantly. Using the robust tracking controllers, the position errors were smaller than those obtained using the original PID controllers. The robust tracking controller showed excellent results.
Acknowledgements

My thanks go first to my supervisor Dr. S. K. Spurgeon, whose patience and extraordinary insight supported me throughout the course of this research. Working under her supervision has been surely a nice experience.

The implementation work described in this thesis was done by the help of many people of the Robotics Research Project. My special thanks go to Professor I. Postlethwaite for much support in both the equipment and some finance. Many thanks go to A. Willby for much help in both hardware and software problems. My thanks also go to Dr W. Feng who introduced me to the Tetrabot. I would further like to thank Dr. X. Y. Lu and Dr D. W. Gu for many useful discussions.

I would like to thank all the members of the control Group who have made working in the group a memorable and pleasant experience especially Ghassan, Tim, Jens, Chris, Alex, Song Ju, Aamer, Bag, Fu Wah, Mark.

My time at Leicester of University has been enriched by the pleasant company of my husband Weidong and my son Bingyuan throughout tough time.

I would like to thank my mother and my parents-in-law for their support and encouragement. I would particularly thank my father for the guidance and encouragement he gave me when I was still a child.

Leicester
August 1999
## Contents

1 Introduction 1

1.1 Contributions ................................................................. 1
1.2 Thesis Overview ............................................................. 5

2 Robust Tracking Control Methods 9

2.1 Introduction ................................................................. 9
2.2 Overview of Robust Tracking Control ................................. 9
2.3 Robust Tracking Control Problems ...................................... 13
2.4 Robust Tracking via Dynamic Sliding Mode Control ............... 14
   2.4.1 Sliding Mode .......................................................... 15
   2.4.2 Output Tracking for the Uncertain System ....................... 16
   2.4.3 Robust Output Tracking Control Scheme ....................... 20
2.5 Lyapunov Theory-based Robust Tracking Control ................... 27
   2.5.1 Lyapunov Theory .................................................... 27
   2.5.2 Uncertain System ................................................... 28
   2.5.3 Lyapunov Functions for Tracking ................................ 32
4.4.1 Trajectory Generation for the Tetrabot ............................................ 56
4.4.2 Linear Function with Parabolic Blends (LFPB) for Tetrabot Trajectory Implementation ................................. 56
4.4.3 Joint Interpolated Motion of Tetrabot ............................................... 59
4.5 Tetrabot Dynamics ............................................................................................ 61
4.5.1 Tetrabot Dynamic Model ....................................................................... 61
4.5.2 Inverse Dynamics and Newton-Euler Equations ............................... 65
4.5.3 Drive Dynamics ...................................................................................... 68
4.6 Linear Control of the Tetrabot ...................................................................... 69
4.6.1 Actuator Dynamics ................................................................................ 70
4.6.2 Set-point Tracking Control .................................................................... 70
4.7 Concluding Remarks ......................................................................................... 73

5 Robust Tracking via Sliding Mode Control for Flexible Joint Manipulators 74

5.1 Introduction ...................................................................................................... 74
5.2 Elastic Joint Manipulator Dynamic Model ...................................................... 75
5.3 Robust Output Tracking Control Scheme ..................................................... 80
5.4 Application to a Two-Link Flexible Joint Manipulator ................................ 85
5.4.1 Output Tracking of Uncertain System via Feedback Linearisation (FLC) .................................................. 87
5.4.2 Output Tracking via Robust Sliding Mode Control (RSMC) ... 90
5.5 Concluding Remarks ......................................................................................... 95
6 Robust Tracking via Dynamical Sliding Mode Control Schemes for Modified Flexible Joint Manipulators

6.1 Introduction ................................................................. 105
6.2 Modified Elastic Joint Manipulator Dynamic Model .................... 106
6.3 Robust Output Tracking Control Scheme .................................. 109
6.4 Application to a Two-link Flexible Joint Robot ............................... 112
   6.4.1 Output Tracking of Uncertain System via Dynamic Feedback Linearisation (DFLC) .............. 113
   6.4.2 Robust Tracking via Dynamic Sliding Mode Control (RDSMC) ............................................. 116
6.5 Concluding Remarks .......................................................... 121

7 Robust Tracking via Sliding Mode Control for Rigid Robotic Manipulators

7.1 Introduction ................................................................. 127
7.2 Rigid Manipulator Dynamical Model ....................................... 127
7.3 Robust Sliding Mode Control Schemes ..................................... 130
7.4 Application to Three DOF Tetrabot Robot ................................. 133
   7.4.1 The Tetrabot Model .................................................... 133
   7.4.2 Output Tracking of Uncertain System via Feedback Linearisation (FLC) ......................... 137
   7.4.3 Simulation Results for FLC ............................................. 138
   7.4.4 Output Tracking via Robust Sliding Mode Control (RSMC) ............................................ 139
   7.4.5 Simulation Results DRSMC and CRSMC ........................................ 140
8 Lyapunov-based Robust Nonlinear Tracking Control Schemes for Rigid Manipulators 153

8.1 Introduction ............................................................... 153

8.2 Uncertain Mechanical System ................................................ 153

8.3 Development of Nonlinear Tracking Controller for the Tetrabot ............. 154

8.4 Application to a Three DOF Model of Tetrabot ................................. 159

8.4.1 Open-loop Simulation of the Tetrabot ....................................... 160

8.4.2 Numerical Simulation Results .................................................. 162

8.5 Concluding Remarks ....................................................... 164

9 Nonlinear Tracking Controller Applied to the Tetrabot Apparatus 167

9.1 Introduction ............................................................... 167

9.2 Experimental Set-up ........................................................ 168

9.2.1 Computer-controlled Tetrabot System Structure .......................... 168

9.2.2 Tetrabot Hardware ........................................................ 168

9.2.3 Computer Control and Data Collection ................................. 171

9.2.4 Tetrabot Software ........................................................ 171

9.3 Software Test Tools ........................................................ 178

9.3.1 Controller Test Tools ....................................................... 179

9.3.2 Controller Calibration Test Tools .......................................... 179

9.3.3 Input/Output Ports Test Tools ............................................. 180
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.3.4 Analogue I/O Board Test Tools</td>
<td>180</td>
</tr>
<tr>
<td>9.3.5 Motor Servo Amplifiers Test Tools</td>
<td>180</td>
</tr>
<tr>
<td>9.4 Theoretical Design Versus Practical Application</td>
<td>181</td>
</tr>
<tr>
<td>9.5 Experimental Method</td>
<td>182</td>
</tr>
<tr>
<td>9.5.1 Voltage Controlled DC Motor</td>
<td>182</td>
</tr>
<tr>
<td>9.5.2 Control Law for Experimentation</td>
<td>183</td>
</tr>
<tr>
<td>9.5.3 Controller Software Interface</td>
<td>186</td>
</tr>
<tr>
<td>9.6 Experimental Results and Discussions</td>
<td>186</td>
</tr>
<tr>
<td>9.6.1 Numerical Simulation Results</td>
<td>186</td>
</tr>
<tr>
<td>9.6.2 Experimental Results and Discussions</td>
<td>188</td>
</tr>
<tr>
<td>9.7 Concluding Remarks</td>
<td>190</td>
</tr>
<tr>
<td>10 Conclusions and Further Studies</td>
<td>194</td>
</tr>
<tr>
<td>10.1 Conclusions</td>
<td>194</td>
</tr>
<tr>
<td>10.2 Further Studies</td>
<td>198</td>
</tr>
<tr>
<td>A Tetrabot Parameters</td>
<td>202</td>
</tr>
<tr>
<td>A.1 Geometric Measures (Figure 4.2)</td>
<td>202</td>
</tr>
<tr>
<td>A.2 Masses, Moments of Inertia</td>
<td>202</td>
</tr>
<tr>
<td>A.3 Driving System Parameters</td>
<td>203</td>
</tr>
<tr>
<td>B Tetrabot Control Software in C</td>
<td>204</td>
</tr>
<tr>
<td>B.1 The Header Files of the Tetrabot Control software (Table B.1)</td>
<td>204</td>
</tr>
</tbody>
</table>
B.2 Tetrabot Controller Software (Tables B.2 and B.3) ........................................... 204

C Tetrabot Control Functions ............................................................... 207

C.1 Tetrabot Controller Functions (Tables C.1 - C.10) .......................... 207

References ................................................................................. 217
List of Figures

4.1 Tetrabot Manipulator Mechanism .............................................................. 48
4.2 Tetrabot Kinematics and Geometry ............................................................ 50
4.3 The force structure of the Tetrabot .............................................................. 62
4.4 The block diagram for the Tetrabot position control ................................. 70
4.5 (a) The PID controller for Tetrabot with $K_p=1$, $K_i=1$, $K_d=1$ and $m_p=0$ Kg 72
4.6 (b) The PID controller for Tetrabot with $K_p=1$, $K_i=4.5$, $K_d=1$ and $m_p=0$ Kg .............................................................. 72
4.7 (c) The PID controller for Tetrabot with $K_p=1$, $K_i=1$, $K_d=1$ and $m_p=6$ Kg 72
4.8 (d) The PID controller for Tetrabot with $K_p=1$, $K_i=4.5$, $K_d=1$ and $m_p=6$ Kg ...................................................................................... 73

5.1 a. The actual trajectories with the $FLC$; b. The error between the desired 
and actual trajectories; c. The desired trajectories; d. The torques. $A = 
0.0876$. No uncertainty. .................................................................................. 96

5.2 a. The actual trajectories with the $FLC$; b. The error between the desired 
and actual trajectories; c. The desired trajectories; d. The torques. $A = 
1.0$. No uncertainty. ...................................................................................... 96
5.3  a. The errors between the desired and actual trajectories with different $A$ using a $FLC$ with the nominal system for the link one. b. The errors between the desired and actual trajectories with different $A$ using a $FLC$ with the nominal system for the link two. ......................................................... 97

5.4  The uniformly distributed random disturbance. .............................................. 97

5.5  a. The actual trajectories with the $FLC$; b. The error between the desired and actual trajectories; c. The desired trajectories; d. The torques. $A = 0.0876$. Uncertain system. ................................................................. 98

5.6  a. The actual trajectories with the $FLC$; b. The error between the desired and actual trajectories; c. The desired trajectories; d. The torques. $A = 1.0$. Uncertain system. ................................................................. 98

5.7  a. The errors between the desired and actual trajectories with different $A$ using the $FLC$ with uncertain system for the link one. b. The errors between the desired and actual trajectories with different $A$ using the $FLC$ with uncertain system for the link two. ......................................................... 99

5.8  a. The actual trajectories with the $RSMC$; b. The error between the desired and actual trajectories; c. The sliding surfaces; d. The torques. $A = 0.0873$. Nominal system. ................................................................. 100

5.9  a. The actual trajectories with the $RSMC$; b. The error between the desired and actual trajectories; c. The sliding surfaces; d. The torques. $A = 1.0$. Nominal system. ................................................................. 100

5.10 a. The errors between the desired and actual trajectories with different $A$ using the $SMC$ with nominal system for the link one. b. The errors between the desired and actual trajectories with different $A$ using the $SMC$ with nominal system for the link two. ......................................................... 101

5.11 a. The actual trajectories with the $RSMC$; b. The error between the desired and actual trajectories; c. The sliding surfaces; d. The torques. $A = 0.0873$. Uncertain system. ................................................................. 102
5.12 a. The actual trajectories with the RSMC; b. The error between the
desired and actual trajectories; c. The sliding surfaces; d. The torques.
$A = 1.0$. Uncertain system. ................................................................. 102

5.13 a. The errors between the desired and actual trajectories with different $A$
using the RSMC with uncertain system for the link one. b. The errors be­
tween the desired and actual trajectories with different $A$ using the RSMC
with uncertain system for the link two. ............................................. 103

6.1 The uniformly distributed random disturbance. .............................. 122

6.2 a. The actual trajectories with the DFLC; b. The error between the
desired and actual trajectories; c. The desired trajectories; d. The torques.
$A = 0.0876$. Uncertain system. .......................................................... 123

6.3 a. The actual trajectories with the DFLC; b. The error between the
desired and actual trajectories; c. The desired trajectories; d. The torques.
$A = 1.0$. Uncertain system. ............................................................... 123

6.4 a. The errors between the desired and actual trajectories with different $A$
using the DFLC with uncertain system for the link one. b. The errors be­tween the desired and actual trajectories with different $A$ using the FLC
with uncertain system for the link two. ............................................. 124

6.5 a. The actual trajectories with the RDSMC; b. The error between the
desired and actual trajectories; c. The sliding surfaces; d. The torques.
$A = 0.0873$. Uncertain system. .......................................................... 125

6.6 a. The actual trajectories with the RDSMC; b. The error between the
desired and actual trajectories; c. The sliding surfaces; d. The torques.
$A = 1.0$. Uncertain system. ............................................................... 125
6.7  
   a. The errors between the desired and actual trajectories with different 
   $A$ using the $RDSMC$ with uncertain system for the link one.  
   b. The errors between the desired and actual trajectories with different $A$ using 
   the $DRSMC$ with uncertain system for the link two.  

7.1 The desired trajectories for Robust Sliding Mode Control.  

7.2 (1). The rod tracking errors and the motor torques with FLC at the lower 
        bound $m=3$Kg and (2). The upper bound $m=6$Kg. Uncertain system.  

7.3 (1). The rod tracking errors with DRSMC at the lower bound $m=3$Kg and 
        (2). The upper bound $m=6$Kg. No uncertainty.  

7.4 (1). The motor torques with the DRSMC at the lower bound $m=3$Kg and 
        (2). The upper bound $m=6$Kg. No uncertainty.  

7.5 (1). The sliding surfaces with the DRSMC at the lower bound $m=3$Kg and 
        (2). The upper bound $m=6$Kg. No uncertainty.  

7.6 The uncertainty in the Tetrabot system.  

7.7 (1). The rod tracking errors with DRSMC at the lower bound $m=3$Kg and 
        (2). The upper bound $m=6$Kg. Uncertain system.  

7.8 (1). The motor torques with the DRSMC at the lower bound $m=3$Kg and 
        (2). The upper bound $m=6$Kg. Uncertain system.  

7.9 (1). The sliding surfaces with the DRSMC at the lower bound $m=3$Kg and 
        (2). The upper bound $m=6$Kg. Uncertain system.  

7.10 (1). The sliding reachability functions with DRSMC and (2). The sliding 
          reachability functions with CRSMC. Uncertain system.  

7.11 (1). The rod tracking errors with CRSMC at the lower bound $m=3$Kg and 
          (2). The upper bound $m=6$Kg. No uncertainty.  

xii
7.12 (1). The motor torques with the CRSMC at the lower bound \( m=3\text{Kg} \) and 
(2). The upper bound \( m=6\text{Kg} \). No uncertainty. ......................... 150

7.13 (1). The sliding surfaces with the CRSMC at the lower bound \( m=3\text{Kg} \) and 
(2). The upper bound \( m=6\text{Kg} \). No uncertainty. ......................... 151

7.14 (1). The rod tracking errors with CRSMC at the lower bound \( m=3\text{Kg} \) and 
\( \epsilon = 0.01 \) (2). The upper bound \( m=6\text{Kg} \). Uncertain system. .......... 151

7.15 (1). The motor torques with the CRSMC at the lower bound \( m=3\text{Kg} \) and 
\( \epsilon = 0.01 \) (2). The upper bound \( m=6\text{Kg} \). Uncertain system. .......... 152

7.16 (1). The sliding surfaces with the CRSMC at the lower bound \( m=3\text{Kg} \) and 
\( \epsilon = 0.01 \) (2). The upper bound \( m=6\text{Kg} \). Uncertain system. .......... 152

8.1 The desired trajectories for the nonlinear tracking controller. ............ 164

8.2 (a) The rod trajectory errors and (b) the motor torques with the payload 
at the lower bound \( m = 3\text{Kg} \). .................................................. 165

8.3 (a) The rod trajectory errors and (b) the motor torques with the payload 
at the upper bound \( m = 6\text{Kg} \). .................................................. 165

9.1 The Tetrabot Controller Architecture ................................................. 169

9.2 The Process Scheduling ........................................................................ 173

9.3 The Tetrabot System Startup ............................................................. 174

9.4 The Tetrabot enable Function ............................................................ 175

9.5 The Tetrabot demo_mode Function .................................................... 176

9.6 The Tetrabot run_mode Function ....................................................... 177

9.7 The Tetrabot Controller Software Interface. ...................................... 187
9.8  (a) The rod tracking errors with the payload at the lower bound $m=3$Kg
and (b) The upper bound $m=6$Kg. ............................................. 189

9.9  The desired joint trajectories. .............................................. 191

9.10 The rod tracking errors with PID and NTC. ......................... 192
List of Tables

6.1 DFLC and RDSMC controllers maximum tracking errors for two links ... 122

7.1 The FLC, DRSMC and CRSMC controllers maximum tracking errors and
torques for three rods ([1]:No uncertainty; [2]:Uncertain system) ............... 144

8.1 Tracking controller parameters ............................................................. 160

8.2 Inertia Matrix of Tetrabot (Open-Loop) ................................................. 161

8.3 Tracking Controllers Parameters ......................................................... 162

9.1 The NTC and PID controllers maximum tracking errors for three rods ... 193

B.1 The header files of Tetrabot control software ........................................ 204

B.2 The Tetrabot control software – 1 ....................................................... 205

B.3 The Tetrabot control software – 2 ....................................................... 206

C.1 The functions – 1 .................................................................................. 207

C.2 The functions – 2 .................................................................................. 208

C.3 The functions – 3 .................................................................................. 209

C.4 The functions – 4 .................................................................................. 210

C.5 The functions – 5 .................................................................................. 211
C.6 The functions — 6 ............................................... 212
C.7 The functions — 7 ............................................... 213
C.8 The functions — 8 ............................................... 214
C.9 The functions — 9 ............................................... 215
C.10 The functions — 10 .......................................... 216
Chapter 1

Introduction

1.1 Contributions

In complex industrial situations robots are usually a nonlinear multivariable system, with inevitable model uncertainties, which are required to operate with variable payloads over a given workspace at high speeds in a constrained environment. High performance specifications in these applications cannot be achieved using conventional point to point control strategies. Advanced control strategies are required. The structural flexibility of robot motion, at high-accuracy and high-speed operation, becomes very important.

Robot manipulators have a finite number of degrees of freedom. A reduced-order model, which is still highly nonlinear and complex, is typically used for the purpose of simulation and controller design. The control design needs to be based upon a relatively simple but accurate model, and also robust to modelling errors and uncertainties. The basis for control design and stability is a dynamic model that captures prominent features of the system under consideration. To account for the uncertainty, the robot mathematical model uses the notion of uncertainty. Possible uncertainties include unknown parameters, unknown functions, disturbances and unmodelled dynamics. For robotic manipulators, it is appropriate and convenient to model and to interpret uncertainties as deterministic. The robotic system parameters and payload can be viewed as structured uncertainties. From a control point of view, modelling inaccuracies can be classified into two major kinds of uncertainties: structured uncertainties and unstructured uncertainties. Unstructured uncertainties include friction, disturbances, and unmodelled dynamics.
Due to the nonlinear nature of the robot system, nonlinear robust tracking control is the best and the optimal method for achieving robustness. Compared with the linear-based controls, the nonlinear controls to be designed have two major advantages. First, nonlinear controls can guarantee global stability, while linear controls usually ensure local stability. Second, nonlinear controls can achieve better performance while using less control energy. Determination of successful controls for general nonlinear systems remains the most challenging problem in control theory. Research on robust tracking control for robot manipulators has made significant progress over the last few years. The development of effective robust tracking controllers represents an important step towards the versatile application of high-speed and high-precision robots.

The tracking control problem for a robot manipulator is an important research topic since many of the tasks performed by a robot involve path tracking and high speed robots demand complicated controllers to achieve high accuracy. In recent years, there have been many papers grappling with this control problem. Many are based on Lyapunov theory [7, 16, 20, 66, 6, 111]; some are based on variable structure controllers [84, 110], feedback linearisation [41, 83, 55]; $H_\infty$ control [104, 33, 60]; and adaptive controllers [59, 81]. Some are composed of a few approaches to achieve the best tracking control. Robust control of nonlinear robotic systems in the presence of uncertainties is discussed using Lyapunov techniques in [86]. A Lyapunov function that is highly dependent on the specific structure of the equations of motion of rigid robots is used. Dynamic sliding mode control is a new approach that can be used to control nonlinear systems which are in differential I-O form and which may not be dynamic feedback linearisable. Dynamic sliding mode control combines the advantage of dynamic feedback control and sliding mode techniques in the controller [50, 51].

The performance of the Lyapunov theory-based tracking control law considered here will be seen to achieve accurate tracking, despite the presence of parameter uncertainty. A class of uncertain dynamical systems is considered which is described by ordinary differential equations. For a class of desired state motions, controllers are obtained which ensure that the controlled system can asymptotically track the desired motion to within any desired degree of accuracy. These controllers are utilised to obtain tracking controllers for a general class of uncertain mechanical systems in Corless [16] and Corless and Leitmann.
Robust tracking control design for flexible joint manipulators has attracted much attention in the last few years. The main reason is that the elastic joint effects must be taken into account in both modelling and controller design in order to achieve high tracking performance. Sliding mode control design is recognised as robust with respect to system uncertainties in both theoretical research and application \[50, 51, 26, 65, 96\]. It combines the advantage of feedback linearisation control and sliding mode techniques in the controller design phase while simultaneously asymptotically linearising the nonlinear system.

Application of dynamical sliding mode controllers to the regulation and tracking of flexible joint manipulators was undertaken by Sira-Ramirez et al \[73\]. They treated the asymptotic output tracking problem from the perspective of dynamical feedback linearisation and dynamical variable structure control but did not consider the model uncertainty, the parametric uncertainty and the external uncertainties.

Robust sliding mode control schemes for uncertain nonlinear systems were introduced by Lu & Spurgeon \[53\]. They were proposed for multiple input-output systems with additive uncertainties. The stability of systems is generalised to the case of uniform ultimate boundedness control of systems. This is used to prove the stability of the overall closed-loop systems. The uncertain system with appropriately chosen sliding mode control is shown to be ultimately bounded if the zero dynamics of the nominal system are uniformly asymptotically stable \[53\].

In this thesis, two nonlinear robust tracking approaches are considered; robust tracking via sliding mode control techniques and via the Lyapunov technique. First, the robust output tracking problem for flexible joint and rigid manipulators is treated from the perspective of feedback linearisation and robust sliding mode control \[53\]. The robust controllers are derived based on the Fliess' generalised controller canonical form (GCCF) \[36, 37\]. The GCCF is straightforward to obtain from the system of differential equations describing the manipulator dynamics. To control manipulators, the following uncertainties must be dealt with: the weight of the object is unknown, the friction and other parameters in the manipulator dynamics may be uncertain and external uncertainties may be present.
Chapter 1. Introduction

Robust tracking controller design approaches for nonlinear control systems usually require the evaluation of upper bounds on nonlinear quantities of the process. The usefulness of robustness criteria is determined by the accuracy of the robustness bounds. The goal is to design and assess a robust tracking controller using the sliding mode technique to deal with uncertainties for flexible joint and rigid robots with $n$ controlled degrees of freedom. The robust tracking problem is also treated from the perspective of a Lyapunov-theory-based tracking control law [16, 20] to achieve accurate robot tracking despite the presence of parameter uncertainty. The robotic manipulators are assumed to be modelled by a set of differential equations and, hence, the dynamics of the robotic manipulator is governed by the set of differential equations. The uncertainties are characterised deterministically by certain structure conditions and known bounding functions. The control problem considered involves robustly tracking a desired reference trajectory to within any desired degree of accuracy. The aim is to develop and implement the Lyapunov tracking controller to deal with uncertainties for real-time robots.

The GEC Tetrabot (Tetrahedral robot) is an experimental assembly robot with a novel three DOF parallel and three DOF serial geometric configuration. This leads to a six DOF robot capable of very precise and rapid movements over a working volume of 1 m$^3$. The original design of the Tetrabot incorporates a control strategy based on the assumption that the motion of each of the six joints is decoupled and results in a primary control system for the Tetrabot consisting of linear PID controllers to control the position of each joint [28]. However, the Tetrabot is a nonlinear multivariable system, with inevitable model uncertainties, which is required to operate with variable payloads over a given workspace at high speeds. To meet such high performance specifications, advanced robust tracking control strategies are required.

This thesis considers the development and implementation of robust tracking controllers for the parallel and serial topology Tetrabot, although the theoretical control strategy presented is applicable to any manipulator tracking problem. A Tetrabot model is developed and discussed. This accurate reduced-order model, which is highly nonlinear and complex, is used to design robust tracking controllers. Simulation studies are used to parameterise the control for a Lyapunov theory-based tracking control and to evaluate the upper bounds for a robust sliding mode tracking controller. The robust sliding mode
controller is applied to a two DOF manipulator with flexible joint and a three DOF rigid manipulator. Both simulation results show excellent robust performance. Most importantly, the results of implementation on the GEC Tetrabot are described.

1.2 Thesis Overview

The Chapters 1 and Chapter 2 consider robust tracking control problems and formulate new robust tracking schemes. The remainder of the report documents the application of these new results to control problems associated with a flexible joint robot and an experimental assembly robot.

Chapter 1: Introduction introduces background and fundamental theory and previews the material in the subsequent chapters.

Chapter 2: Overview of Tracking Control Methods discusses robust tracking control strategies used in controlling robotic manipulators. Two methods of robust tracking control are considered. One is based on the deterministic control of uncertain systems: the Lyapunov Control design technique is considered. The outstanding feature of these controllers is their excellent robustness and invariance properties; Another is based on the sliding mode control of uncertain systems which shows the tracking error is ultimately bounded if the zero dynamics of the nominal system are uniformly asymptotically stable. The tracking control problem of a manipulator can be divided into two subproblems: the motion planning subproblem and the motion control subproblem. The motion controller is designed to ensure the actual positions $q(t)$ closely track the desired trajectories $q^*_R(t)$. The nonlinear tracking controllers will be designed using Lyapunov theory-based tracking control techniques and robust sliding mode control techniques.

Chapter 3: Dynamics of Robotic Systems presents a brief discussion of the dynamics of robotic systems. These include the motion of the rigid robotic manipulator, joint flexibility issues and actuator dynamics. The discussion introduces generic expressions that are used to derive useful properties of the dynamics and to develop bounding functions for the unknowns, which will be useful for controller design and analysis.
Chapter 4: Analysis of the Tetrabot System introduces the Tetrabot mechanism: a three degree of freedom positioning mechanism to which is attached a three degree of freedom wrist. The forward kinematics solution computes the forward kinematics from the joint space (base coordinate) to the intermediate task space, then to the toolplate. The inverse kinematics solution involves deriving the joint coordinates for any specified position and orientation of point toolplate referred to a base coordinate frame with origin located and fixed with respect to the support structure. A Linear Function with Parabolic Blends (LFPB) trajectory has been used for trajectory interpolation. The time optimum control algorithm is used. The Tetrabot dynamics include inverse dynamics and drive dynamics. The system model has been modelled as a three DOF device with the wrist mechanism and radial arm combined together as a single rigid body. Finally the linear set-point tracking control has been simulated on the Tetrabot model.

Chapter 5: Robust Tracking via Sliding Mode Control for Flexible Joint Manipulators is presented for the robust tracking control of robotic manipulators via sliding modes in the presence of model uncertainty and disturbances. The control scheme addresses the following problem: given the extent of parametric uncertainty and the external uncertainties, design a nonlinear sliding mode controller to achieve robust tracking precision. The explicit robustness guarantees provided by the methodology are demonstrated using flexible joint manipulator models. The methodology is compared with the traditional feedback linearisation. Robust tracking controllers will be given a case study which relates to flexible joint manipulators. The controllers are applied to a two DOF manipulator with flexible joints. Simulation results show excellent robust performance.

Chapter 6: Robust Tracking via Sliding Mode Control for Rigid Manipulators discuss the redesign of robust tracking control schemes in Chapter 5 for rigid manipulators in the presence of model uncertainty and disturbances to achieve robust tracking. The robustness guarantees provided by the methodology are demonstrated using rigid manipulator models. The methodology is compared with a traditional feedback linearisation and a discontinuous robust sliding mode control. In practice, application of robust sliding mode is more important. The relative performance is studied by simulation of a three DOF rigid robot - a Tetrabot. The robust sliding mode controller shows excellent robustness.
Chapter 7: Robust Tracking via Dynamic Sliding Mode Control for Flexible Joint Manipulators is presented for the robust output tracking problem of modified elastic joint manipulators that are treated from the perspective of dynamical feedback linearisation and robust dynamical sliding mode control. In this chapter, by using the motor position as the system output, the system now becomes relative degree two and the GCCF involve second order time derivatives of the control input torque. It will show a dynamical feedback control scheme results. The controller is applied to a two DOF manipulator with a flexible joint. For robot systems with dynamical sliding mode control, the same closed loop poles are chosen as in the sliding mode controller but better performance is achieved.

Chapter 8: Lyapunov-based Robust Nonlinear Tracking for Rigid Manipulators details Lyapunov-based nonlinear tracking control schemes for the rigid manipulators and applies the theory to the Tetrabot. M Corless and G Leitmann have developed a class of uncertain dynamical systems described by ordinary differential equations and characterised by certain structural conditions and known bounding functions. For a feasible class of desired state motions they present a class of controllers which assure that the controlled system can asymptotically track the desired motion to any desired degree of accuracy. Various classes of controllers are presented. The design of all these controllers is based on Lyapunov theory. This study considers the design of the nonlinear tracking controller, and simulates the controller on the Tetrabot mathematical model developed in Chapter 4. The theoretical nonlinear tracking controller will be applied to the Tetrabot in Chapter 9.

Chapter 9: Nonlinear Tracking Controller Applied to the Tetrabot Apparatus describes the implementation of the nonlinear tracking controller and overviews the system structure, hardware and software of the Tetrabot. It details the software test tools including the controller, the controller calibration, the interpolation, I/O and the motor servo amplifiers. A controller is developed and tuned using these software test tools. Theoretical analysis and computer simulations of the nonlinear tracking controller are important but not sufficient, because the value and applicability of a nonlinear tracking controller lie in its actual hardware implementation. The results of the implementation of a nonlinear tracking controller on the Tetrabot rig facility are described. The proposed
control strategy is robust to variations in robot loading. The experimental results obtained for the closed-loop response indicate that compensation, which employs explicit off-line parameter estimation, can improve tracking accuracy significantly. The performance of the tracking controller in real-time is good, but not as good as predicted by the numerical simulation studies. The fact that the simulation model has yet to be validated might account for the differences.

Chapter 10: Conclusions and Further Studies summarises the contributions of the thesis together with recommendations for future work.
Chapter 2

Robust Tracking Control Methods

2.1 Introduction

This chapter discusses robust tracking control strategies used in controlling robotic manipulators. Section 2.2 contains an overview of the robust tracking control methods that have been used for the robotic manipulators. The robust tracking problem has commonly been tackled using linear controls such as computed torque control, PD control and PID control. Compared with these linear-based controls, the nonlinear controls can guarantee global stability, while linear controls usually ensure local stability, and can achieve better performance while using less control energy. In Section 2.3, the robust tracking problem for robotic manipulators is outlined. In Sections 2.5 and 2.4, two recently developed schemes, a Lyapunov-based robust tracking control and a robust sliding mode control, are presented for robotic manipulators in the presence of model uncertainty and disturbances.

2.2 Overview of Robust Tracking Control

Research on robust tracking control of robotic manipulators has made significant progress over the last few years. The literature on robust control of robotic manipulators is extremely rich. A comprehensive survey of robust control designs for nonlinear rigid robotic systems appears in the work by Abdallah et al [1]. They discuss current approaches to robust control of rigid robots and summarise the major design methods: the feedback-linearisation approach, the passivity approach, the Lyapunov-based nonlinear approach,
Chapter 2. Robust Tracking Control Methods

the variable structure approach, the saturation approach and the robust-adaptive approach. A survey of the control of flexible joint robots has been discussed by Spong [88]. Different methods such as feedback linearisation, adaptive control, singular perturbation methods and robust control have been used to design effective controllers for flexible joint robots. Many excellent papers are based on robust tracking control of robots using classical linear controls by Takegaki, Arimoto, Gilbert, Wang, Samson, Qu et al [94, 4, 38, 105, 68, 67, 64], Lyapunov-based controls by Corless, Leitmann, Gutman et al [39, 18, 16, 17, 20, 2], sliding mode controls by Slotine, Young, Morgan, Bartolini, Nicosia, Chen et al [83, 107, 56, 8, 58, 15, 84, 76], $H_2$ and $H_\infty$ controls by Postlethwaite, Feng, Schaft et al [103, 33, 60], adaptive controls by Craig, Slotine, Spong et al [24, 80, 82, 89], and I-O feedback by Lu, Spurgeon, Sire-Raunirez, Behtash, Qu et al [54, 73, 72, 74, 9, 61].

For linear computed torque control, local robustness analysis was reported by imposing zero initial tracking errors and an upper bound for the difference between the inertia matrix and its estimate [23]. In [105] it is shown that a computed torque law with very high gain makes tracking errors uniformly bounded for any initial condition. The existing results in [94, 4, 38, 105, 68, 67, 64] represent the progress made in the area of robustness analysis of PD and PID controls. For a general trajectory tracking problem, uniform boundedness has been shown by using sufficiently large, nonlinear PD gains; the error system becomes asymptotically stable in the limit [68, 67]. It has been shown in [43] that a linear PD high gain feedback law can make the tracking error locally uniformly bounded with respect to a bound that becomes proportional to time. In [64], a PD control scheme was shown to be robust and was used to make the tracking error small.

For robotic systems whose dynamics are inherently nonlinear, it is natural to design nonlinear control rather than a linear control in order to achieve better performance. The linear control laws achieve robustness by employing large linear control gains [63]. Nonlinear control can guarantee global stability and can achieve better performance while using less control energy.

Typical Lyapunov-based, nonlinear robust controls are minmax controller [39], saturation-type controller [18, 25, 38] and polynomial-type controller [68, 67]. If Lyapunov theory is adopted for control design of robotic systems, the most difficult thing is to find a proper Lyapunov function. The Lyapunov function is the basis on which the nonlinear analysis
can be performed and the nonlinear control can be designed. A method closely related to the method of the Lyapunov function is the so-called sliding control or variable structure control (VSC). The concept of a sliding surface has been widely used to generate robust controls [83]. The first application of this theory to robot control seems to be in Young's work [107]. Discontinuous sliding control was generated for robot manipulators in [107, 56, 8, 108, 15]. Unfortunately, for most of the schemes, the control effort is discontinuous along the sliding surface and creates chattering which may excite unmodelled high-frequency dynamics. To improve the robustness of the sliding control, one often uses the boundary layer approach which forces the trajectory towards the sliding surface and is smoothed inside a possibly time-varying thin boundary layer [76, 84]. The discontinuous control law is suitably smoothed to achieve an optimal trade-off between control bandwidth and tracking precision. The control first forces the sliding surface to accounts for parameter variations and disturbance inputs, then is smoothed out by forcing only the state asymptotically into a small neighbourhood of the switching surface to achieve robustness of high-frequency unmodelled dynamics [99, 75, 63, 83, 54]. Slotine [76] modified the discontinuous sliding control into the continuous sliding control and applied to robotic manipulators. Adaptive robust controls were first introduced in [19] and applied to the tracking control of robots in [14, 69]. A survey on adaptive control of manipulators can be found in [59]. The adaptive control can be developed using a straightforward Lyapunov argument and is closely related to robust controls [24]. To achieve robustness for adaptive control schemes, robust controls were used in [80, 89, 62]. The learning control was first introduced by Uchiyama [95]. Learning control is studied from an artificial intelligence point of view. More specifically neural network approaches, fuzzy logic and memory-based modelling have been used for the learning control of robots [3, 5, 29]. Compared with learning control based on artificial intelligence, model-based learning controls have two major advantages the theoretical guarantee of stability and performance and applicability to all robotic systems without training. Compared with adaptive control, learning control has the advantage that the parameterisation of system dynamics is not required; a feature that is shared with robust control.

The robust control schemes previously studied are based on full state feedback. Under many circumstances, only partial information about the state of the system is available. The output feedback signal of a robotic system is the vector of joint positions, and its
measurements are always available, reliable, and economic. This prompts us to study the output feedback robust control problem to derive information about the system from its output and then to devise a good control. In an input-output feedback control design for robotic manipulators, we must overcome two obstacles: nonlinear dynamics and uncertainties. Both of these features make output feedback robust control of robots a very challenging problem.

The control of uncertain systems is usually accomplished using either an adaptive (learning) control or a robust control philosophy. In the adaptive approach, one designs a controller that attempts to learn the uncertain parameters of the system and, if properly designed, will eventually be a best controller for system in question [81, 59]. In the robust approach, the controller has a fixed structure that yields acceptable performance for a class of plants which include the plant in question [86, 76]. In general, the adaptive approach is applicable to a wider range of uncertainties, but robust controllers are simpler to implement and no time is required to tune the controller to the particular plant.

In this thesis, the robust tracking control developed should be applicable to a real robot - the Tetrabot. The Tetrabot was built in the late 1980's. The Tetrabot low-level controller (including computer hardware and machine code software) is a 68000 SBC (Single Board Computer) which cannot be easily deal with on-line calculation at high speed. For this reason, a robust tracking control is the best option for the Tetrabot.

Previous studies have considered most of robust tracking control from a common viewpoint: the Lyapunov method. This treatment allows us to fully explore the power of the Lyapunov method and to bring out the similarities of various control schemes. Although all controls have certain robustness, they have different levels of compensation for nonlinearities, uncertainties, and disturbances. Two methods of robust tracking control will be studied in here. Lyapunov theory-based approaches to the problem of tracking control are presented in Section 2.5. For a class of desired state motions, a control strategy is expounded which ensures that the system asymptotically tracks the desired motion to any desired degree of accuracy. Another new scheme is presented for the robust tracking control of robot manipulators in the presence of model uncertainty and disturbances. Based on the robust sliding mode control methodology, the control scheme addresses the following problem: given the extent of parametric uncertainty and the external uncertainties,
design a nonlinear sliding mode controller to achieve robust tracking precision in Section 2.4. The explicit robustness guarantees provided by the methodology are demonstrated using elastic joint and rigid manipulator models.

### 2.3 Robust Tracking Control Problems

Design of a robust control system for robotic systems should proceed through several phases: motion analysis and dynamic modelling for a specific configuration of the manipulator under study; formulation of control problems such as free-space tracking, constrained motion, and force control; trajectory planning; robust control design; and finally simulation and implementation. It is assumed in the rest of the thesis that trajectory tracking is the control problem and that a desired trajectory is given. This simplification allows us to focus our investigation on robust control design.

In general the tracking control problem of a manipulator can be conveniently divided into two coherent subproblems: the motion (trajectory) planning subproblem and motion control (path tracker) subproblem. The trajectory planning interpolates and/or approximates the desired path by a class of polynomial functions and generates a sequence of time-based "control set points" for the control of the manipulator from the initial location to the destination location. Chapter 4 discusses trajectory planning schemes for tracking motion. In general, the motion control problem consists of (a) obtaining dynamic models of the manipulator, and (b) using these models to determine control laws or strategies to achieve the desired system response and performance. The path tracker attempts to make the robot's actual position and velocity match some desired values of position and velocity: the desired values are provided to the controller by the trajectory planner. The trajectory planner receives as input some sort of geometric path descriptor from which it calculates a time history of the desired positions and velocity. The path tracker then tries to minimise the deviation of the actual position and velocity from the desired values. The control scheme is divided in this way because the dynamics of all but the simplest robots are highly nonlinear and coupled.

The robot controller design is defined as follows [20, 16, 53]: given the desired trajectories of the joint position \( q_R(t) \), velocity \( \dot{q}_R(t) \) and acceleration \( \ddot{q}_R(t) \), measurements of the
Chapter 2. Robust Tracking Control Methods

joint position $q(t)$ and velocity $\dot{q}(t)$, and with some or all the manipulator parameters being unknown, with model uncertainty and external uncertainties, derive a control law for the actuator torque and a tracking law for uncertain systems such that the actual trajectories $q(t), \dot{q}(t), \ddot{q}(t)$, closely track the desired trajectories $q_R(t), \dot{q}_R(t), \ddot{q}_R(t)$.

A trajectory control problem may arise when a robot is performing tasks such as laser cutting or arc welding where the tracking accuracy set by the allowable tolerance of workplaces is stringent, or even when the task is merely to move a load from its initial position to a final desired position. This may be due to requirements on the maximum time to complete the task or may be necessary to avoid overshoot or otherwise bumping into obstacles during the task [77].

Dynamics of robotic systems are usually not totally known. Common uncertainties include unknown parameters, friction, load variation, unmodelled dynamics, and disturbances. Identifying bounding function is one of the key steps in designing robust control. These bounding functions are guaranteed by the inherent properties of robotic systems. A robust controller that estimates the uncertainties and bounding function can benefit robot tracking control as follows:

1. To account for the possibly large unknown loads the manipulator may carry simply by considering the securely grasped load.

2. To account for the possibly large lumped uncertainty and the disturbances.

3. To solve voltage controlled servo problems with improved robustness at a lower level in order to release the control algorithm embedded in a microprocessor-based computer control (the Tetrabot system).

4. To modify the robust control to achieve improved performance after the parameters have successfully converged to more accurate values.

2.4 Robust Tracking via Dynamic Sliding Mode Control

In this section, the robust output tracking problem is treated from the perspective of feedback linearisation and robust sliding mode control. Robust sliding mode control schemes
for uncertain nonlinear systems [53] are employed. The uncertain system, with appropriately chosen sliding mode control, is shown to be ultimately bounded if the zero dynamics of the nominal system are uniformly asymptotically stable. In the derivation of robust controllers, the Fliess' generalised controller canonical form (GCCF) is used [36, 37]. Robust tracking controller design approaches for nonlinear control systems usually require the evaluation of upper bounds on nonlinear quantities of the process. The usefulness of robustness criteria is determined by the accuracy of the robustness bounds. The eventual aim is to design and assess a robust tracking controller using the sliding mode technique to deal with uncertainties for robots with \(n\) controlled degrees of freedom.

2.4.1 Sliding Mode

Variable structure control (VSC) with a sliding mode was first described by Soviet authors including Emel'yanov et al., [30, 31], Utkin [97, 98, 100] and Itkis [42]. A survey paper by Utkin [99] introduces many of the early contributions available in translation. Recent survey and tutorial papers with numerous references have been written by Utkin [101, 102] and DeCarlo et al. [26]. Draženović established early results on the invariance of VSC systems to a class of disturbances and parameter variations. More recently the subject has attracted great interest because of these excellent invariance properties. Consequently, VSC is particularly suited to the deterministic control of uncertain and nonlinear control systems. There have been major studies in the use of VSC and allied techniques in, for instance, model following and model reference adaptive control, the development of variable structure tracking control, the geometrical interpretation of the sliding mode in VSC and the control of uncertain dynamical systems using Lyapunov control.

The central feature of VSC is sliding motion. This occurs when the system state repeatedly crosses and immediately re-crosses a switching manifold, because all motion in the neighbourhood of the manifold is directed inwards. Depending on the form of the control law selected, sliding motion may occur on individual switching surfaces in the state space, on a selection of surfaces, or on all the switching manifolds together. When the last of these cases occurs, the system is said to be in the sliding mode.

The design of the sliding control system may be divided into two independent proce-
Chapter 2. Robust Tracking Control Methods

16

dues: the proper choice of sliding surface according to the performance task. This may be considered as a geometric manifold which may be determined by a set of geometric equations [96] or may be determined by a set of control independent differential equation [78]. The second procedure involves the proper choice of sliding mode control strategy which guarantees satisfaction of the sliding reachability condition. The major practical disadvantage of this approach is the fundamental requirement of a discontinuous control structure that chatters about the switching surface at high frequency. This has resulted in the development of continuous approximations to the discontinuous elements [10] and also the use of boundary layer techniques [75, 76, 84]. Sliding control has been successfully applied to robot manipulators, underwater vehicles, automotive transmissions and engines, high-performance electric stability

Sliding mode controllers differ from simpler relay controllers in that they rely on extremely high speed switching among the control values. The major advantage of a sliding controller is effectively to account for structured uncertainty (e.g., unknown plant parameters and payload) and unstructured uncertainties (e.g., modelling frictions, disturbances and unmodelled dynamics in a reasonably rigid mechanical system) once in the sliding mode. Sliding modes exist if all phase space trajectories are directed toward the switching line which defines the new trajectory. Once the system reaches the sliding mode, the fast switching control keeps the trajectory near the switching line.

For the class of system to which it applies, sliding controller design provides a systematic approach to the problem of maintaining stability and consistent performance. By allowing the trade-offs between modelling and performance to be quantified in a simple way, sliding mode controller is the most appropriate for the application.

2.4.2 Output Tracking for the Uncertain System

Suppose uncertain differential I-O system models exist [103].

\[ y_i^{(n_i)} = \varphi_i(\dot{y}, \dot{u}, t) + \Delta_i(\dot{y}, \omega, t) \]

\[ \ldots \]

\[ y_p^{(n_p)} = \varphi_p(\dot{y}, \dot{u}, t) + \Delta_p(\dot{y}, \omega, t) \]  

(2.1)
where \( u \in \mathbb{R}^m \) is the control, \( y \in \mathbb{R}^p \) is the output. \( \hat{u} = (u_1, \ldots, u_1^{(\beta_1)}, \ldots, u_m, \ldots, u_m^{(\beta_m)}) \) and \( \hat{y} = (y_1, \ldots, y_1^{(n_1-1)}, \ldots, y_p, \ldots, y_p^{(n_p-1)}) \) with \( n_1 + \ldots + n_p = n \), \( n_i \geq 1 \), where \( n \) is the dimension of the system state-space. The uncertainties \( \Delta(\hat{y}, \omega, t) = (\Delta_1(\hat{y}, \omega, t), \ldots, \Delta_p(\hat{y}, \omega, t)) \) are Lebesgue measurable and satisfy

\[
\|\Delta_i(\hat{y}, \omega, t)\| \leq \rho_i \|\hat{y}\| + l_i \tag{2.2}
\]

\( \rho_i \geq 0, \ l_i \geq 0, \ i = 1, \ldots, p. \)

where \( \omega \in \Omega \) represents the uncertain time-varying parameter and \( \Delta \) is an unknown vector field. Here \( \|\cdot\| \) denotes the Euclidean norm of a vector [92].

The nominal system of equation (2.1) given by

\[
y^{(n)}_i = \varphi_i(\hat{y}, \hat{u}, t) \tag{2.3}
\]

is a differential I-O system. It is assumed that \( p = m \), all \( \varphi_i(\cdot, \cdot, \cdot) \) are \( C^1 \)-functions and the regularity condition holds \( \frac{\partial \varphi_i}{\partial (\varphi_j)} \) with \( i, j = 1, \ldots, m \), i.e. there partial derivatives exist and satisfy

\[
\det \left[ \frac{\partial (\varphi_1, \ldots, \varphi_m)}{\partial (u_1^{(\beta_1)}, \ldots, u_m^{(\beta_m)})} \right] \neq 0 \tag{2.4}
\]

The system (2.3) may be expressed in the following Fliess Generalised Controller Canonical Form (GCCF) [35]

\[
\dot{\eta}_1^{(1)} = \eta_2^{(1)} \\
\vdots \\
\dot{\eta}_{n_1-1}^{(1)} = \eta_{n_1}^{(1)} \\
\dot{\eta}_{n_1}^{(1)} = \varphi_1(\eta, \hat{u}, t) + \Delta_1(\eta, \omega, t)
\]
Chapter 2. Robust Tracking Control Methods

\begin{align}
\eta^{(m)}_1 &= \eta^{(m)}_2 \\
\vdots \\
\eta^{(m)}_{m-1} &= \eta^{(m)}_m \\
\eta^{(m)}_m &= \varphi_m(\eta, \hat{u}, t) + \Delta_m(\eta, \omega, t)
\end{align}

where \( \eta^{(i)} = (\eta^{(i)}_1, \ldots, \eta^{(i)}_m) = (y_i, \ldots, y_i^{(m-1)}) \), \( \eta = (\eta^{(1)}, \ldots, \eta^{(m)})^T \) and \( i = 1, \ldots, m \).

Let \( \hat{y}_R(t) \) be a prescribed reference output function, assumed to be sufficiently smooth. The \textit{asymptotic output tracking problem} consists of specifying a dynamical controller, possibly described by an implicit time-varying scalar ordinary differential equation, using the \textit{output reference signal} \( \hat{y}_R(t) \) with a finite number of its time derivatives and the \textit{uncertain GCCF system} (2.5) to construct a control input function \( \hat{u} \), which locally forces the system actual output \( \hat{y} \) to asymptotically track the desired output reference signal \( \hat{y}_R(t) \).

Define a tracking error function \( \hat{e}(t) \) as the difference between the actual system output \( \hat{y} \) in (2.1) and the output reference signal \( \hat{y}_R(t) \):

\[
\hat{e}(t) = \hat{y} - \hat{y}_R(t)
\]

Here \( \hat{y}_R \) is used to denote \( (y_{R1}, \hat{y}_{R1}, \ldots, y_{R1}^{(n_1-1)}, \ldots, y_{Rm}, \hat{y}_{Rm}, \ldots, y_{Rm}^{(n_m-1)}) \) and \( \hat{y} \) is used to denote \( (y_1, \hat{y}_1, \ldots, y_1^{(n_1-1)}, \ldots, y_m, \hat{y}_m, \ldots, y_m^{(n_m-1)}) \). Define \( \hat{e} = (e_1, e_1^{(1)}, \ldots, e_1^{(n_1-1)}, \ldots, e_m, e_m^{(1)}, \ldots, e_m^{(n_m-1)}) \), as components of an error vector \( \hat{e} \). Based on the uncertain system (2.1) and (2.5), the tracking error system may be expressed in \textit{GCCF} as:

\begin{align}
\hat{e}^{(1)}_1 &= \hat{e}^{(1)}_2 \\
\vdots \\
\hat{e}^{(1)}_{n_1-1} &= \hat{e}^{(1)}_{n_1} \\
\hat{e}^{(1)}_{n_1} &= \varphi_1(\hat{y}_R + \hat{e}, \hat{u}, t) + \Delta_1(\hat{y}_R + \hat{e}, \omega, t) - y_{R1}^{(n_1)}(t) \\
\vdots 
\end{align}
Chapter 2. Robust Tracking Control Methods

\[
\begin{align*}
\dot{e}_1^{(m)} &= e_2^{(m)} \\
\vdots \\
\dot{e}_{n_m-1}^{(m)} &= e_{n_m}^{(m)} \\
\dot{e}_{n_m}^{(m)} &= \varphi_m(\hat{y}_R + \hat{e}, \hat{u}, t) + \Delta_m(\hat{y}_R + \hat{e}, \omega, t) - y_{R_m}^{(n_m)}(t)
\end{align*}
\]

Based on the uncertain system (2.1), the tracking error system (2.7) may be expressed in vector form as:

\[
e^{(n)} = \varphi(\hat{y}_R + \hat{e}, \hat{u}, t) + \Delta(\hat{y}_R + \hat{e}, \omega, t) - y_R^{(n)}(t)
\]

with

\[
e^{(n)} = [e_1^{(n_1)}, \ldots, e_m^{(n_m)}]^T, \quad y_R^{(n)} = [y_{R_1}^{(n_1)}, \ldots, y_{R_m}^{(n_m)}]^T,
\]

\[
\varphi = [\varphi_1, \ldots, \varphi_m]^T, \quad \Delta = [\Delta_1, \ldots, \Delta_m]^T.
\]

\[
n_1 + \ldots + n_m = n
\]

The nominal system is represented by the uncertain error system (2.7) with \( \Delta_i \equiv 0 \). Suppose that the asymptotic equilibrium point of the controlled tracking nominal error system, for a suitable control input strategy, is given by \( \hat{e} = 0 \). The remaining dynamics of the nominal system (2.3) can be defined by

\[
\varphi(\hat{y}_R, \hat{u}, t) = \hat{y}_R(t)
\]

For the particular case in which the tracking signal \( \hat{y}_R(t) \) is identically zero, or a given constant, i.e., for the case of an output nulling, or stabilisation task the expression (2.9) constitutes the so called \textit{zero dynamics} \( \varphi(0, \hat{u}, t) = 0 \) which is the same zero dynamics defined in \([50],[51]\) and \([52]\).

The stability features of (2.9) for reference signals \( \hat{y}_R(t) \) which are bounded with bounded derivatives, also determine, to a large extent, the physical reliability of any tracking control strategy which asymptotically achieves the perfect tracking condition \( \hat{e} = 0 \). It is
assumed that the solution of (2.9) is bounded for all bounded input functions \( y_R(t) \) which also exhibit bounded derivatives.

**Definition 2.1:** The error system (2.7) with \( \Delta_i \equiv 0 \) is defined to be a minimum-phase nonlinear system if its corresponding zero dynamics are exponentially stable.

### 2.4.3 Robust Output Tracking Control Scheme

Static sliding mode controller design is recognised as robust with respect to system uncertainties in both theoretical research and application by Ryan and Corless [65], DeCarlo et al [26], Utkin [96]. However, static sliding mode controller design requires that the system has a controllable linearisation or assumes an appropriate regular form. In addition, high frequency chattering appears in the ideal sliding mode which is practically unacceptable. Dynamic sliding mode control remedies the defects of high frequency chattering to some extent by passing the control through a filter [78, 72, 71]. The designed controllers can be continuous or discontinuous.

A robust sliding mode controller is designed by selecting sliding surfaces to prescribe desirable system performance and choosing a sliding reachability condition to ensure the sliding mode is attained. Consider the direct sliding surface \( s \), defined in terms of the output tracking error coordinates \( \hat{e} \) (2.6) as

\[
\mathbf{s} = e^{(n-1)} + \mathbf{A}\hat{e}
\]

where \( \mathbf{\hat{e}} = (e_1, e_1^{(1)}, \ldots, e_1^{(n-2)}, \ldots, e_m, e_m^{(1)}, \ldots, e_m^{(n_m-2)}) \) and \( \mathbf{A} = \text{diag}[A_1, \ldots, A_m] \) with \( A_i \) the companion matrix of the Hurwitz polynomial \( \sum_{j=1}^{n_i-1} a_j^{(i)} \lambda^{j-1} \), where \( i = 1, \ldots, m \).

Different types of sliding reachability are defined in [50, 51, 79]. For robust controller design, a strong sliding reachability condition which can be written in vector form as defined in [53] is required:

\[
\dot{s} = -\gamma(k, s) = -\mathbf{K}s - \mathbf{K}_0\text{sign}(s)
\]
where $\text{sign}(s) = [\text{sign}(s_1), \ldots, \text{sign}(s_m)]^T$ is the signum function, $s = [s_1, \ldots, s_m]$, $K_0 = [k_{01}, \ldots, k_{0m}]$ and $K = \text{diag}[K_j] \in \mathbb{R}^{n \times m}$ is a positive definite matrix that form a set of design parameters.

Ideally a discontinuous sliding reachability condition (2.11) is used to eliminate deviations from the sliding surface in the presence of uncertainty. However, in practice, due to finite switching time, the frequency is not infinitely high. The control is discontinuous across the switching surface and chattering takes place. A common approach to reduce chattering is to introduce a boundary layer around the sliding surface as in [53] and [84] and use a continuous sliding reachability condition within the boundary layer. Using a saturation function $\text{sat}_\epsilon(s)$ instead of $\text{sign}(s)$ in controller design will reduce chattering. The term $\text{sat}_\epsilon(x_i)$, with a saturation limit vector $\epsilon$ for $\epsilon > 0$, defined on $\mathbb{R}$ as follows. Let $\epsilon_i$ and $\text{sat}_\epsilon(x_i)$ denote the $i$th element in vectors $\epsilon$ and $\text{sat}_\epsilon(x)$, respectively. Then

$$
\text{sat}_\epsilon(x_i) \triangleq \text{sat} \left( \frac{x_i}{\epsilon_i} \right) = \begin{cases} 
1 & \text{if } x_i > \epsilon_i \\
\frac{x_i}{\epsilon_i} & \text{if } |x_i| \leq \epsilon_i \\
-1 & \text{if } x_i < -\epsilon_i
\end{cases}
$$

(2.12)

The continuous sliding reachability condition becomes

$$
\dot{s} = -\gamma(k, s) = -Ks - K_0\text{sat}_\epsilon(s)
$$

(2.13)

where $\text{sat}_\epsilon(s) = [\text{sat}_\epsilon(s_1), \ldots, \text{sat}_\epsilon(s_m)]^T$. $K$ and $K_0$ are positive parameters to be tuned during controller design.

**Definition 2.2:**

A strong sliding reachability condition is defined as

$$
\dot{s} = -\gamma(k, s)
$$

where $\gamma$ and $k$ are a set of constant design parameters that $\gamma(k, s) = [\gamma_1(k, s), \ldots, \gamma_m(k, s)]$ satisfies [53]:
1. $\gamma(k, s)$ is continuous if $s \neq 0$;
2. $\gamma(k, 0) = 0$.
3. $s^T \gamma(k, s) > s^T K s$ when $s \neq 0$.

This condition implies that $s$ converges to 0 globally and at least exponentially.

Differentiating the sliding surface (2.10) along the trajectories of the uncertain GCCF (2.7), it follows that

$$\dot{s} = e^{(n)} + A \bar{e}$$

(2.14)

where $\bar{e} = (e_1^{(1)}, e_2^{(2)}, \ldots, e_1^{(m-1)}, \ldots, e_1^{(1)}, e_2^{(2)}, \ldots, e_m^{(m-1)}), e^{(n)} = [e_1^{(n)}, e_2^{(n)}, \ldots, e_m^{(n)}]^T$

and we have

$$\dot{s} = \varphi(\dot{y}_R + \dot{e}, \ddot{u}, t) + \Delta(\dot{y}_R + \dot{e}, \omega, t) + A \bar{e} - y_R^{(n)}(t)$$

(2.15)

with

$$\|\Delta\| = \|\Delta_i\| \
\leq \rho_i \|\dot{y}_R + \dot{e}\| + l_i$$

(2.16)

To estimate the uncertainty bound as in (2.16), let $\rho_i$ and $l_i$ denote the upper bound on the lumped uncertainty, choose parameters $0 < \theta < 1$, $\theta_0 + \theta = 1$, $\rho^{(0)} = \sqrt{\rho_1^2 + \rho_2^2 + \cdots + \rho_m^2}$ and calculate the parameter $\rho$ as follows

$$\rho^{(1)} = \rho^{(0)} \left(1 + \max_i \{|A_i| \sqrt{n-1}\}\right)$$

$$\rho = \rho^{(0)} + \frac{(\rho^{(1)})^2}{4\theta}.$$  

(2.17)

Use the strong continuous sliding reachability condition (2.13) and the differentiation of the sliding surface (2.15) to set
\[ \varphi(\hat{y}_R + \hat{e}, \hat{u}, t) - y_R^{(n)}(t) + \Delta e(t) = -Ks - K_0\text{sate}(s) \]  
\[ (2.18) \]

Equation (2.15) becomes

\[ \dot{s} = -Ks - K_0\text{sate}(s) + \Delta(\hat{y}_R + \hat{e}, \omega, t) \]
\[ (2.19) \]

The robust tracking controller corresponding to the tracking system (2.8) can be determined by

\[ \varphi(\hat{y}_R + \hat{e}, \hat{u}, t) = y_R^{(n)}(t) - Ks - K_0\text{sate}(s) - A\hat{e}(t) \]
\[ (2.20) \]

Suppose \( s^T\gamma_0(k, s) \geq s^TKs \) and \( K \) satisfies

\[ \lambda_{\text{min}}(K) - \left[ \frac{1}{\theta_0} [BD]^T [BD] + \rho I_m \right] > 0 \]
\[ (2.21) \]

where \( A = \text{diag}[A_1, \ldots, A_{m_1}] \) is the companion matrix of Hurwitz polynomial, \( D = \text{diag}[D_1, \ldots, D_{m_1}]^T \) with \( D_i = \text{diag}[0, \ldots, 0, 1]^T \) for \( i = 1, \ldots, m_1 \). \( A \) and \( B \) satisfy the Lyapunov equation

\[ A^T B + B A = -D \]
\[ (2.22) \]

and \( K_0 \) satisfies

\[ K_0 > l^{(0)}I_m \]
\[ (2.23) \]

where \( l^{(0)} = \sqrt{l_1^2 + l_2^2 + \cdots + l_m^2} \). Then the stability of the uncertain system is ensured as proved in [53].

The choice of robust tracking control law is apparent from the normal GCCF form (2.7) with \( \Delta_i \equiv 0 \). From the robust control law (2.20) the highest order derivatives of control \( u^{(\beta)} = (u_1^{(\beta_1)}, \ldots, u_m^{(\beta_m)}) \) can be solved by the Implicit function Theorem as
Chapter 2. Robust Tracking Control Methods

\( \mathbf{u}(\beta) = \nu(\dot{y}_R + \dot{e}, \dot{u}, t) \) \hspace{1cm} (2.24)

The dynamic feedback (2.24) together with (2.8) yields a closed loop system of dimension \( n + \beta \) with \( n = \sum_{i=1}^{m} n_i \) and \( \beta = \sum_{i=1}^{m} \beta_i \).

**Theorem 2.1** Assume the nominal GCCF system (2.5) is minimum phase and the control law is chosen according to (2.24) with \( \nu = [\nu_1, \ldots, \nu_m] = \varphi \) as in (2.20). For arbitrarily given but bounded \( \epsilon > 0 \), there exists a robust tracking control law

\[ \nu = y_R^{(n)} - Ks - K_0 \text{sat}_s(s) - A\tilde{e}(t) \] \hspace{1cm} (2.25)

with

\[ \lambda_{\text{min}}(K) I_m > \left[ \frac{1}{2\theta} [BD]^T [BD] + \rho^{(0)} I_m \right. \]
\[ + \left[ \rho^{(0)} (1 + \max_i \{|A_i|\} \sqrt{n - 1}) \right] \frac{1}{4\theta} I_m \]
\[ K_0 > l^{(0)} I_m \] \hspace{1cm} (2.26)

which yields a tracking error system which is ultimately bounded by \( \epsilon \).

This is in contrast to the usual sliding mode control design. Addition of the linear in \( s \), is necessary owing to the presence of uncertainty in the control vector field which introduces a term in the dynamics of \( s \) depending on \( \tilde{e} \).

**Proof:** Utilising sliding surface \( s \) defined in (2.10), lets \( e^{(n-1)} = \dot{\tilde{e}} \) and utilising (2.19) we can, through a coordinate change, write the system as

\[ \dot{\tilde{e}} = -A\tilde{e} + Ds \] \hspace{1cm} (2.27)
\[ \dot{s} = -Ks - K_0 \text{sat}_s(s) + \Delta(\tilde{e}, \omega, t) \]

with the uncertainty bounds
Chapter 2. Robust Tracking Control Methods

\[ \|\Delta(\hat{e}, \omega, t)\| \leq \rho(0)\|\hat{e}\| + t(0) \]

An estimate of the uncertainty bounds in the \((e, s, t)\) coordinates is required

\[ \|\Delta(e, s, t)\| \]
\[ \leq [\rho(0)\left(\|\hat{e}\| + \|e(n-1)\|\right) + t(0)] \]
\[ \leq [\rho(0)\left(\|\hat{e}\| + \|s\| + \max_i\{|A_i|\}\|\hat{e}\|\sqrt{n - 1}\right) + t(0)] \]
\[ \leq (\rho(1)\|\hat{e}\| + \rho(0)\|s\| + t(0)) \quad (2.28) \]

where \(\rho(1) = \rho(0)(1 + \max\{|A_i|\}\sqrt{n - 1})\). A candidate Lyapunov function can be proposed as

\[ V = \hat{e}^T B \hat{e} + \frac{1}{2} s^T s \quad (2.29) \]

where \(B\) satisfies (2.22). Differentiating \(V\) along (2.27) with (2.28), we have

\[ \dot{V} \leq -\|\hat{e}\|^2 + 2\hat{e}^T B D s - s^T K s - s^T K_0 \text{sat}_e(s) \]
\[ + \|s\|(\rho(1)\|\hat{e}\| + \rho(0)\|s\| + t(0)) \quad (2.30) \]

Using the inequalities

\[ \rho(1)\|\hat{e}\|\|s\| \leq \theta\|\hat{e}\|^2 + \frac{(\rho(1))^2}{4\theta} \|s\|^2 \]
\[ \|s\| \leq \sum_{i=1}^{m} |s_i| = s^T \text{sign}(s) \]

Utilising the saturation function \(\text{sat}_e(s)\) defined in (2.12) and all the \(s_i\) satisfy \(|s_i|\), it now becomes
Chapter 2. Robust Tracking Control Methods

\[
\mathbf{K}_0 \|\mathbf{s}\| \leq \mathbf{K}_0 \sum_{i=1}^{m} |s_i| = \mathbf{s}^T \mathbf{K}_0 \text{sat}_e(\mathbf{s}) = \begin{cases} 
\mathbf{K}_0 & \text{if } |s_i| > \epsilon; \\
\frac{|s_i|}{\epsilon} \mathbf{K}_0 & \text{otherwise.}
\end{cases}
\]

\[
\dot{V} \leq -\|\mathbf{e}\|^2 + 2\mathbf{e}^T \mathbf{B} \mathbf{D} \mathbf{s} - \lambda_{\text{min}}(\mathbf{K})\|\mathbf{s}\|^2 - \mathbf{s}^T \mathbf{K}_0 \text{sat}_e(\mathbf{s}) + \theta\|\mathbf{e}\|^2 + \left(\frac{\rho^{(1)}}{4\theta}\right)^2 \|\mathbf{s}\|^2 + \rho^{(0)}\|\mathbf{s}\|^2 + l^{(0)}\|\mathbf{s}\|
\]
\[
\leq - (\theta - 1)\|\mathbf{e}\|^2 + 2\mathbf{e}^T \mathbf{B} \mathbf{D} \mathbf{s} - \lambda_{\text{min}}(\mathbf{K})\|\mathbf{s}\|^2 + \left(\rho^{(0)} + \frac{\rho^{(1)}}{4\theta}\right)^2 \|\mathbf{s}\|^2 - \mathbf{s}^T (\mathbf{K}_0 - l^{(0)}) \text{sat}_e(\mathbf{s})
\]
\[
\leq -\theta_0\|\mathbf{e}\|^2 + 2\mathbf{e}^T \mathbf{B} \mathbf{D} \mathbf{s} - (\lambda_{\text{min}}(\mathbf{K}) - \rho)\|\mathbf{s}\|^2 - \sum (\mathbf{K}_0 - l^{(0)})|s|
\]

where

\[
\theta_0 = \theta - 1
\]
\[
\rho = \left(\rho^{(0)} + \left[\rho^{(0)}(1 + \max_i\{|A_i|\})\sqrt{n - 1}\right]^2\right)\frac{4\theta}{4\theta}
\]

Letting

\[
|2\mathbf{e}^T \mathbf{B} \mathbf{D} | \leq |\mathbf{B}|^T |\mathbf{B}| \|\mathbf{e}\|\|\mathbf{s}\|
\]

\[
\dot{V} \leq -\theta_0\|\mathbf{e}\|^2 + |\mathbf{B}|^T |\mathbf{B}| \|\mathbf{e}\|\|\mathbf{s}\| - (\lambda_{\text{min}}(\mathbf{K}) - \rho \mathbf{I}_m)\|\mathbf{s}\|^2 - \sum (\mathbf{K}_0 - l^{(0)})|s|
\]

\[
(2.31)
\]

When \(|s_i| > \epsilon\), we assume \(\mathbf{K} > \rho \mathbf{I}_m\) then \(\dot{V} < 0\) if and only if \(\mathbf{K}\) and \(\mathbf{K}_0\) satisfy
\[ \lambda_{\text{min}}(K) I_m > \left[ \frac{1}{\theta_0} [BD]^T [BD] + \rho I_m \right] \quad (2.32) \]

\[ K_0 > l^{(0)} I_m \]

Otherwise, \( \dot{V} < 0 \) if and only if \( K \) satisfies (2.32) and \( K_0 \) satisfies \( K_0 > ml^{(0)} \epsilon I_m \). Thus choosing \( 0 < \epsilon \leq \min \{ \lambda_{\text{min}}(K), \lambda_{\text{min}}(K_0) \} / ml^{(0)} \), the system (2.27) is ultimately bounded by \( \epsilon \).

Here \( \lambda_{\text{max(min)}}(\cdot) \) denotes the maximum (minimum) eigenvalue of \( \cdot \).

In the following example, the continuous round off sliding reachability condition is used to reduce chattering [63] and [83].

### 2.5 Lyapunov Theory-based Robust Tracking Control

Into the framework of the above we shall now add a Lyapunov theory-based nonlinear tracking control which assures that the trajectory of the controlled system can track a desired trajectory to within any desired degree of accuracy.

#### 2.5.1 Lyapunov Theory

Design of nonlinear tracking controllers for robotic systems is based on Lyapunov theory [83]. Basic Lyapunov theory comprises two methods introduced by Lyapunov, the indirect method and the direct method. The indirect method, or linearisation method, states that the stability properties of a nonlinear system in the close vicinity of an equilibrium point are essentially the same as those of its linearised approximation. The direct method is a powerful tool for nonlinear system analysis, and therefore the so-called Lyapunov analysis often actually refers to the direct method. The direct method is a generalisation of the energy concepts associated with a mechanical system: the motion of a mechanical system is stable if its total mechanical energy decreases all the time. In using the direct method to analyse the stability of a nonlinear system, the idea is to construct a scalar energy-like function (a Lyapunov function) for the system, and to see whether it decreases.
Although Lyapunov's direct method was originally a method of stability analysis, it can be used for other problems in nonlinear control. One important application is the design of nonlinear controllers. The idea is to somehow formulate a positive function of the system states, and then choose a control law to make this function decrease along trajectories of the system. A nonlinear control system thus designed will be guaranteed to be stable. Such a design approach has been used to solve many complex design problems, including these in robot control. The direct method can also be used to estimate the performance of a control system and its robustness.

The approach of early research workers such as Leitmann, Corless [18], Gutman [39] and Ryan [65] follows a Lyapunov approach. Using a Lyapunov function and specified magnitude bounds on the uncertainties, a nonlinear control law is developed to ensure uniform ultimate boundedness of the closed-loop feedback trajectory to achieve sufficient accuracy. The resulting controller is a discontinuous control function with generally continuous control in a boundary layer which prevents the exciting of high frequency unmodelled parasitic dynamics. Controllers have been devised for numerous types of system with many different approaches and allow for a range of expected system variation.

In particular, a quantitative description of the behaviour of a system subject to any continuous approximation of a discontinuous control law is given in [47] and in [18]. Through the use of Lyapunov arguments, they evaluate the final precision that can be guaranteed under the worst realisation of the uncertainties affecting the system. The objective achievable by the controller is the practical stability [20] of the closed loop rather than its asymptotic stability.

2.5.2 Uncertain System

When modelling a real system, the model usually contains uncertain elements. The model of an uncertain system is of the form

$$\dot{x}(t) = \bar{F}(t, x(t), u(t), \omega) \quad (2.33)$$
\[ \omega \in \Omega \]  \hfill (2.34)

where \( t \in \mathbb{R} \) is the time variable, \( x \in \mathbb{R}^n \) is the system state, and \( u \in \mathbb{R}^m \) is the control input to the system. \( \Omega \) is some known, non-empty set and \( \bar{F} : T \times X \times U \times \Omega \rightarrow \mathbb{R}^n \) is known. All the uncertainty in the system is represented by the lumped uncertain element \( \omega \). For each \( \omega \in \Omega \), there corresponds a system function \( \bar{F}(\cdot, \omega) : \mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}^n \).

The control input will be generated by a state-feedback controller \( p : \mathbb{R} \times \mathbb{R}^n \rightarrow \mathbb{R}^m \) so that,

\[ u(t) = p(t, x(t)) \]  \hfill (2.35)

Substituting (2.35) into (2.33) results in the feedback-controlled system

\[ \dot{x}(t) = f(t, x, \omega) \]  \hfill (2.36)

where

\[ f \triangleq \bar{F}(t, x, p(t, x), \omega) \]  \hfill (2.37)

To solve the tracking problem for the system (2.33) and (2.34), define a desired state motion, \( \bar{x}(\cdot) : \mathbb{R} \rightarrow \mathbb{R}^n \). Ideally a feedback control function \( p \) is required which will ensure the feedback controlled system is globally uniformly asymptotically stable (GUAS) about \( \bar{x} \) for all possible \( \omega \).

**Definition 2.3:** Consider the desired model for the systems to tracking described by

\[ \dot{\bar{x}}(t) = \bar{A}\bar{x}(t) + \bar{B}\bar{u}(t) \]

where \( \bar{x} \in \mathbb{R}^n \) is the desired model state, \( \bar{u} \) is the desired model input. We assume that the desired model pair \( \bar{A} \in \mathbb{R}^{n \times n} \) and \( \bar{B} \in \mathbb{R}^{n \times m} \) are stabilisable [6, 91].
The state tracking error is defined by

\[ z(t) \triangleq x(t) - \bar{x}(t) \quad (2.38) \]

and the error system is described by

\[ \dot{z}(t) = f_c(t, z, \omega) \quad (2.39) \]

where

\[ f_c(t) \triangleq \bar{F}(t, \bar{x}(t) + z, p(t, \bar{x}(t) + z), \omega) - \dot{\bar{x}}(t) \quad (2.40) \]

The control objective is thus equivalent to ensuring *globally uniformly asymptotically stable* (*GUAS*) of (2.39) about zero. The requirement of *GUAS* about zero for an uncertain system is extremely stringent, and may require the use of discontinuous control signals, for example. For this reason the control objective is relaxed and controllers which ensure that (2.39) asymptotically tracks zero to within a *small* tolerance \( d > 0 \) are sought.

**Definition 2.4:**

System (2.39) is called to **asymptotically tracks 0 to within** \( d \) if the following properties hold:

- For each \( t_0 \in \mathbb{R}, z_0 \in \mathbb{R}^n \), there exists a solution of \( z(\cdot) \) of (2.39) with \( z(t_0) = z_0 \). Every solution \( z(\cdot) : [t_0, t_1) \to \mathbb{R}^n \) of (2.39) has an extension over \([t_0, \infty)\);

- For each initial condition bound \( r \in \mathbb{R}_+ \), there is a solution bound \( d(r) \in \mathbb{R}_+ \) such that for any \( t_0 \in \mathbb{R} \) and any solution \( z(\cdot) \) of (2.39) with \( z(t_0) \in r \), then \( ||z(t_0)|| \leq d(r) \) for all \( t \in [t_0, t_1) \).

- For each \( \epsilon > 0 \), there exists \( \delta > 0 \) such that for any \( t_0 \in \mathbb{R} \) and any solution of \( z(\cdot) \) of (2.39) with \( z(t_0) \in \delta \), then \( ||z(t_0)|| < d(\epsilon) + \epsilon \) for all \( t \in [t_0, t_1) \).
Chapter 2. Robust Tracking Control Methods

- For each initial condition bound \( r \in \mathbb{R}_+ \), and each \( \epsilon > 0 \), there exists \( T(r, \epsilon) \in \mathbb{R}_+ \) such that for any \( t_0 \in \mathbb{R} \) and any solution \( z(\cdot) \) of (2.39):
  \[
  \|z(t_0)\| \leq r \Rightarrow \|z(t)\| \leq \delta(r) + \epsilon; \quad \forall \ t \geq t_0 + T(r, \epsilon);
  \]

If system (2.40) satisfies Definition 2.4 with \( \delta \geq 0 \), then the tolerance \( \delta \) can be regarded as a measure of closeness of the system behavior to \textit{GUAS}.

\textit{Definition 2.5:} System (2.36) and (2.37) asymptotically tracks \( \bar{x} \) to within \( \delta \) if the corresponding error system (2.39) - (2.40) asymptotically tracks 0 to within \( \delta \).

It is shown in [16] that appropriate tracking controllers exist if there exist matrices \( \bar{A} \) and \( \bar{B} \) such that the following conditions hold

\textit{Assumption 2.1:} There exist matrices \( \bar{A} \) and \( \bar{B} \) as defined by \textit{Definition 2.3}, satisfying

1. For each \( \omega \in \Omega \) there are continuous functions \( G(\cdot, \omega): \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}^{m \times m} \) and \( h(\cdot, \omega): \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}^m \) such that

   \[
   \bar{F}(t, x, u, \omega) = \bar{A}x(t, x, \omega) + \bar{B}[h(t, x, \omega) + G(t, x, \omega)u]
   \]

   for all \( t \in \mathbb{R} \), \( x \in \mathbb{R}^n \) and \( u \in \mathbb{R}^m \).

2. There are continuous bounding functions \( \hat{\beta}_0 \), \( \hat{\beta}_1 \) and \( \hat{\beta}_2 \) so that for all \( \omega \in \Omega \), \( t \in \mathbb{R} \) and \( x \in \mathbb{R}^n \):

   \[
   \|h(t, x, \omega)\| \leq \hat{\beta}_0(t, x) \]
   \[
   \|G(t, x, \omega)\| \leq \hat{\beta}_1(t, x) \]
   \[
   \lambda_{\text{min}}[G(t, x, \omega) + G(t, x, \omega)^T] \geq \hat{\beta}_2(t, x) > 0
   \]

   where \( \lambda_{\text{max(min)}}(\cdot) \) denotes the maximum (minimum) eigenvalue of (\cdot).

3. The desired state motion \( \bar{x}(\cdot) \) is \( C^1 \) and there exists a continuous control function \( \bar{u}(\cdot): \mathbb{R} \to \mathbb{R}^m \) such that for all \( t \in \mathbb{R} \)

   \[
   \dot{x}(t) = \bar{A}x(t) + \bar{B}\bar{u}(t) \]
Chapter 2. Robust Tracking Control Methods

2.5.3 Lyapunov Functions for Tracking

We shall consider an uncertain system described by (2.33) and (2.34). Since the only information available on $\omega$ is a set $\Omega$ to which $\omega$ belongs, we attempt to solve the above problem by looking for a feedback control function which stabilises (2.33) about for all $\omega \in \Omega$.

**Definition 2.6:** A function $V : T \times \mathbb{R}^n \rightarrow \mathbb{R}_+$ is a candidate Lyapunov function which is continuously differentiable and there exist functions $\gamma_i, i = 1, 2, 3$ of class-$\mathcal{K}$, and constant $\epsilon, \gamma_3$ satisfy

$$\lim_{r \to \infty} \gamma_3(r) > \epsilon$$

such that, for all $(t, x) \in T \times \mathbb{R}^n$,

$$\gamma_1(\|z\|) \leq V(t, z) \leq \gamma_2(\|z\|)$$

$$\frac{\partial V}{\partial t}(t, z) + \frac{\partial V}{\partial z}(t, z)f(t, z, \omega) \leq -\gamma_3(\|z\|) + \epsilon$$

2.5.4 Design of Tracking Controllers

A class of controllers which ensures that the system (2.36) asymptotically tracks $\bar{x}(\cdot)$ to within $\delta$ for all uncertainty $\omega \in \Omega$ is now described. From (2.33), (2.41) and (2.43) the tracking error system is given by

$$\dot{z} = \bar{A}z + \bar{B}[h(t, x, \omega) - \bar{u} + G(t, x, \omega)u]$$

The proposed controller strategy consists of three parts

$$p = p^1 + p^2 + p^f$$
with

\[
\begin{align*}
\mathbf{p}^1 &= -\gamma(t, x)\mathbf{B}^T\mathbf{P}z \\
\mathbf{p}^2 &= \gamma_2 \ddot{q} \\
\mathbf{p}^\varepsilon &= -\rho(t, x)S_\varepsilon
\end{align*}
\]

(2.49)

for any tolerance \(d \geq 0\), letting \(u = \mathbf{p}\). This yields the error dynamics

\[
\dot{z} = \mathbf{A}z + \mathbf{B}\mathbf{G}\mathbf{p}^1 + \mathbf{B}[e + \mathbf{G}\mathbf{p}^\varepsilon]
\]

(2.50)

where

\[
e = h(t, x, \omega) - \ddot{u}(t) + G(t, x, \omega)\mathbf{p}^2(t, x)
\]

(2.51)

The controllers \(\mathbf{p}^1\), \(\mathbf{p}^2\) and \(\mathbf{p}^\varepsilon\) are constructed as follow

The function \(\mathbf{p}^2\) must be continuous. There are obviously broad classes of design objectives which can be incorporated within this broad constraint.

The function \(\mathbf{p}^1\) is chosen so that

\[
\dot{z} = \mathbf{A}z + \mathbf{B}\mathbf{G}\mathbf{p}^1
\]

(2.52)

is GUAS about zero for all \(\omega \in \Omega\). Choose symmetric, positive-definite matrices \(\mathbf{P} \in \mathbb{R}^{n \times n}\), \(\mathbf{Q} \in \mathbb{R}^{n \times n}\), any positive scalar \(\sigma\) and any continuous function \(\gamma\) which satisfies

\[
\gamma(t, x) \geq 2\tilde{\beta}_2^{-1}\sigma
\]

(2.53)

where \(\gamma\) is guaranteed by the bounding conditions (3) in Assumption 2.1. \(\mathbf{P}\) is the solution of the algebraic Riccati equation
Chapter 2. Robust Tracking Control Methods

\[ PA + A^T P - 2\sigma PBB^T P - 2Q = 0 \]  \hspace{1cm} (2.54)

Then \( p^1 \) is given by

\[ p^1 = -\gamma(t,x)\bar{B}^T P z \]

The function \( p^e \) is chosen to compensate for the, possibly destabilising, term \( e \). It can be likened to the unit vector nonlinearity which is used to provide robustness in sliding mode control configurations. For any \( \epsilon > 0 \), the controller \( p^e \) is selected by any continuous functions \( \rho \) and \( k \) which satisfies

\[ \rho \geq \frac{1}{2} \| \hat{\beta}_1 \| e(t, x, \omega) \| \quad k \geq \| e(t, x, \omega) \| \]  \hspace{1cm} (2.55)

where \( \rho \) and \( k \) are guaranteed by the bounding conditions (3) in Assumption 2.1.

Let \( \alpha = : \bar{B}^T P z, \mu = k\alpha \) and \( S_\epsilon \) be any continuous function which satisfies

\[ \| \mu \| S_\epsilon = \| S_\epsilon \| \mu \] \hspace{1cm} (2.56)

\[ \| k\alpha \| S_\epsilon = \| S_\epsilon \| k\alpha \]

\[ S_\epsilon = \| \alpha \|^{-1} \| S_\epsilon \| \alpha \]

\[ \| S_\epsilon \| \geq 1 - \| k\alpha \|^{-1} \| e(t,x) \|, \quad if \quad \| \mu \| > 0 \]

\( p^e \) becomes

\[ p^e = -\rho(t,x)S_\epsilon \]

Theorem 2.2: Consider a tracking error system with uncertainty described by (2.39) - (2.40) and any desired state motion \( \bar{x}(\cdot) \) which together satisfy Assumption 2.1. Suppose
\textbf{Chapter 2. Robust Tracking Control Methods}

\begin{align*}
p \text{ in equation } (2.48) \text{ is a collection of feedback control functions. If there exists a candidate Lyapunov function } V \text{ and a class-}\mathcal{K}\text{ function } \gamma \text{ such that give any } \epsilon > 0 \text{ there exists } p \text{ which assures that for all } \omega \in \Omega, \\
\hat{z}(t) = f_e(t, z, p(t, z), \omega) \quad (2.57)
\end{align*}

has existence and indefinite extension of solutions

\begin{align*}
\frac{\partial V}{\partial t}(t, z) + \frac{\partial V}{\partial z}(t, z)f_e(t, z, p, \omega) \leq -\|z\|^2_Q + \epsilon \quad (2.58)
\end{align*}

then the control function \( p \) practically stabilises (2.39) - (2.40) about zero.

\textit{Proof:} \quad \text{The proof proceeds by considering the function } V \text{ given by}

\begin{align*}
V(z) = \frac{1}{2}z^TPz \quad (2.59)
\end{align*}

where \( P \) satisfies (2.54). The error system (2.50) can be described by

\begin{align*}
f_e = \bar{A}z + \bar{B}Gp^1 + \bar{B}[e + Gp^\gamma]
\end{align*}

Differentiating \( V \) along (2.60), utilising (2.49), (2.54) - (2.56) and \( \beta_0k\alpha \neq 0 \) we have

\begin{align*}
\frac{\partial V}{\partial t}(t, z) + \frac{\partial V}{\partial z}(t, z)f_e(t, z, p, \omega) &= z^TP\left\{\bar{A}z + \bar{B}Gp^1 + \bar{B}[e + Gp^\gamma]\right\} \\
&\leq \frac{1}{2}z^T[P\bar{A} + \bar{A}^TP]z + z^TP\bar{B}Gp^1 + \alpha^T[e + \alpha^TGp^\gamma] \\
&\leq z^T[\alpha P\bar{B}B^TP + Q]z - \gamma^TP\bar{B}Gp^\gamma \beta_G^T Pz + \|\alpha\| \|e\| - \rho \alpha^T G S_e \\
&\leq -z^TQz + z^T[\alpha P\bar{B}B^TP]z - \frac{1}{2}\gamma^TP\bar{B}(G + G^T) \beta_G^TPz + \|\alpha\| \|e\| \\
&- \frac{1}{2}\rho \|\alpha\|^{-1} \|S_e\| \alpha^T(G + G^T)\alpha
\end{align*}

Using the inequalities
Chapter 2. Robust Tracking Control Methods

\[ z^T Q z \leq \|z\|_Q^2; \]  
\[ z^T [P \hat{B} + \hat{B}^T P] z \leq \|\alpha\|^2; \]  

Utilising (2.53), (2.55), (2.56) and substituting (2.63) into (2.61), differentiating \( V \) becomes

\[
\frac{\partial V}{\partial t}(t, z) + \frac{\partial V}{\partial z}(t, z)f_e(t, z, p, \omega) \\
\leq - \|z\|_Q^2 + \sigma \|\alpha\|^2 - \frac{1}{2} \gamma \beta \|\alpha\| \|e\| - \frac{1}{2} \rho \beta \|\alpha\| \left(1 - \|k\alpha\|^{-1}\epsilon\right) \\
\leq - \|z\|_Q^2 + \sigma \|\alpha\|^2 - \sigma \|\alpha\|^2 + \|e\| \|\alpha\| - \|e\| \|\alpha\| \left(1 - \|k\alpha\|^{-1}\epsilon\right) \\
\leq - \|z\|_Q^2 + \|e\| \|\alpha\| \|k\alpha\|^{-1}\epsilon \\
\leq - \|z\|_Q^2 + \epsilon
\]

If \( k\alpha = 0 \), it follows from (2.56) that \( S_e = 0 \); hence \( p^e = 0 \) and (2.61) becomes

\[
\frac{\partial V}{\partial t}(t, z) + \frac{\partial V}{\partial z}(t, z)f_e(t, z, p, \omega) \leq - \|z\|_Q^2
\]  

(2.63)

2.6 Concluding Remarks

Tracking control methods based on the deterministic control of uncertain systems are considered. These **Sliding Mode** and **Lyapunov Control** design techniques have outstanding feature robustness and invariance properties. The tracking control problem for a manipulator can be divided into two subproblems: the motion planning subproblem and the motion control subproblem. The motion controller design problem is to derive a tracking law for the unknown parameters, such that the actual positions \( q_i(t) \) closely track the desired trajectories \( \tilde{q}_i(t) \). The application of these deterministic control methods to particular robot problems and geometries will be considered in the rest of the thesis. First it is necessary to discuss the dynamics of the robot systems.
Chapter 3

Dynamics of Robotic Systems

3.1 Introduction

In this chapter we present a discussion of the dynamics of robotic systems. This will include the motion of the rigid robotic manipulator, joint flexibility issues and actuator dynamics. The discussion introduces generic expressions that are used to derive useful properties of the dynamics and to develop bounding functions for the unknowns, which will be useful for controller design and analysis.

3.2 Dynamics of Robotic Manipulators

3.2.1 Euler-Lagrange equations

For a robotic system, the dynamic equations can be derived using Lagrange’s Method [61], which is given by the following partial differential equation:

\[ \frac{d}{dt} \left( \frac{\partial L}{\partial \dot{q}} \right) - \frac{\partial L}{\partial q} = \tau - \Delta_d \]  

(3.1)

where \( q \in \mathbb{R}^n \) is a vector of generalised coordinates of robotic systems (the generalised coordinates in this case are the joint positions), \( \dot{q} \) denotes the time derivative of \( q \), \( L \) is the Lagrangian of the system defined by
Chapter 3. Dynamics of Robotic Systems

\[ L(q, \dot{q}) = K_{e}(q, \dot{q}) - P(q), \quad K_{e}(q, \dot{q}) = \frac{1}{2} \dot{q}^T M(q) \dot{q} \quad (3.2) \]

\( K_{e}(q, \dot{q}) \) represents the kinetic energy, \( M(q) \) is the inertia matrix, \( P(q) \) denotes the potential energy, \( \tau \in \mathbb{R}^n \) is the vector of generalised external input functions (torques), and \( \Delta_d \in \mathbb{R}^n \) is a vector of generalised input due to disturbances or unmodelled dynamics.

3.2.2 Rigid Robots

The dynamics of a rigid-body robotic manipulator can be represented by the following second-order ordinary differential equation:

\[ \tau = M(q)\ddot{q} + H(q, \dot{q}) \quad (3.3) \]

where \( H(q, \dot{q}) \) is the lumped sum of all nonlinearities given by

\[ H(q, \dot{q}) = C(q, \dot{q})\dot{q} + G(q) + \Delta_d \]

\( M(q) \in \mathbb{R}^{n \times n} \) denotes the inertia matrix, \( C(q, \dot{q}) \in \mathbb{R}^{n \times n} \) is the matrix containing centripetal and Coriolis terms and \( G(q) \in \mathbb{R}^n \) is the vector containing gravity terms, \( q(t) \in \mathbb{R}^n \) is the vector of joint variables.

Dynamic equation (3.3) can be derived directly from the Lagrange formulation (3.1) by simply substituting (3.2) into (3.1).

3.2.3 Robots with Flexible Joints

Control design for robot manipulators with flexible joints has attracted much attention from control researchers in recent years. The main reason is that joint flexibility must be taken into account in both modelling and control design in order to achieve high tracking performance [93]. The model of flexible joints can be written as
Chapter 3. Dynamics of Robotic Systems

\[ 0 = M(q_1)\ddot{q}_1 + H(q_1, \dot{q}_1) + K_s(q_1 - q_2) \]  
(3.4)

\[ J\ddot{q}_2 = K_s(q_1 - q_2) - D\dot{q}_2 + \tau + \Delta_f(q_1, \dot{q}_1) \]  
(3.5)

\[ H(q_1, \dot{q}_1) = C(q_1, \dot{q}_1)q_1 + G(q_1) + \Delta_d \]

where \(q_1 \in \mathbb{R}^n\) is the vector of joint angles and \(M(q_1) \in \mathbb{R}^{n \times n}, C(q_1, \dot{q}_1) \in \mathbb{R}^{n \times n}, G(q_1) \in \mathbb{R}^n, \) and \(\Delta_d \in \mathbb{R}^n\) have the same definitions as those in equation (3.3). The presence of flexible joints introduces another second-order vector differential equation in which the torque output \(\tau\) of the actuators drives the joint angle \(q_1\) indirectly through the deformation \(q_1 - q_2\). The variable \(q_2 \in \mathbb{R}^n\) represents the vector of motor angles, \(K_s\) is a diagonal matrix representing joint stiffness, \(J\) is a diagonal matrix of actuator inertias, \(D\) is a diagonal matrix of torsional damping coefficients, and \(\Delta_f(q_1, \dot{q}_1)\) denotes the vector of disturbance or additive unmodelled dynamics. Due to the fact that each joint is usually driven by a dedicated actuator, equation (3.5) is decoupled and decentralised.

3.2.4 Actuator Dynamics

In this section we consider the dynamics of the actuators that generate the generalised torques \(\tau\) in equations (3.3) and (3.5). A systematic approach will be to compensate for actuator dynamics in the form of electrical effects and joint flexibilities.

In practical implementation, many factors limit the ultimate speed of a manipulator, including the finite capabilities of the robot's actuators, links and bearings. Actuator saturation may occur when one of the torques specified by the algorithm reaches the physical limit of the corresponding actuator.

Dynamics of a permanent magnet DC machine are described by the following differential equations [45]:

\[ \tau = K_m I_a \]  
(3.6)
\[ L_m \frac{dI_a}{dt} + R_m I_a = V_d - E_b + \Delta_e(q_2, \dot{q}_2) \]
\[ E_b = -K_b \dot{q}_2 \]

where \( q_2 \) is the vector of the motor angles. When joint flexibility does not exist, equation (3.6) depends on link angles, since \( q_2 = q_1 \). The symbols \( R_m \) and \( L_m \) denote the resistance and inductance of the armature of the DC motor; \( K_m \) and \( K_b \), which depend on the field flux, are torque and back-e.m.f parameters of the motor, respectively; \( I_a \) is the current in the armature of the motor, \( V_d \) is the armature voltage, \( E_b \) is the back EMF generated by the DC motor and \( \Delta_e(q_2, \dot{q}_2) \) denotes the vector of disturbances or additive unmodelled dynamics.

### 3.3 Properties of Robot Dynamics

In this section, properties of robot dynamics (3.3) can be observed from the following derivation. Substituting (3.2) into (3.1) yields

\[ \tau - \Delta_d = \dot{M}(q)\dot{q} + M(q)\ddot{q} - \frac{1}{2}q^T \left( \frac{\partial}{\partial q} M(q) \right) \dot{q} + \frac{\partial}{\partial q} P(q) \]  
\[ (3.7) \]

Comparing the preceding equation with robot equation (3.3), we have

\[ C(q, \dot{q})\dot{q} = \dot{M}(q)\dot{q} - \frac{1}{2}q^T \left( \frac{\partial}{\partial q} M(q) \right) \dot{q} \]
\[ (3.8) \]

and

\[ G(q) = \frac{\partial}{\partial q} P(q) \]

From these expressions, there are three types of term in the robot equation (3.3). The first involve the second derivative of the generalised coordinates. The second are quadratic terms in the first derivatives of \( q \), where the coefficients may depend on \( q \). The third type of terms is those involving only \( q \).
Although the equations of robotic motion are complex, nonlinear equations for all but the simplest robots, they have several fundamental properties which can be exploited to facilitate control system design. We can conclude the following properties.

**Property 2.1:**

1. The inertia matrix $M(q)$ is symmetric, positive definite, and both $M(q)$ and $M(q)^{-1}$ are uniformly bounded as a function of $q(t) \in \mathbb{R}^n$.

2. The difference between the time derivative of the inertia matrix $M(q)$ and centripetal/Coriolis matrix $C(q, \dot{q})$, $\dot{M} - 2C(q, \dot{q})$, is skew symmetric. That is

   $$y^T \dot{M} y = 2y^T C(q, \dot{q}) y, \quad \forall y, q, \dot{q} \in \mathbb{R}^n$$

   see Ortega and Spong [59].

3. $C(q, \dot{q})$ is linear in $\dot{q}$, and its dependence on $q$ is similar to that of $M(q)$. That is

   $$C(q, z)y = C(q, y)z, \quad \forall y, z \in \mathbb{R}^n$$

In the context of control design, Property 2.1 (1) is fundamental and therefore always used. Positive definiteness of the inertia matrix reflects the fact that most systems in motion always have nonzero kinetic energy ($K(q, \dot{q}) > 0$, whenever $||\dot{q}|| \neq 0$). Property 2.1 (2) and (3) derive from equation (3.8), and their physical root is the assumption of the rigid body motion to which the Euler-Lagrange method applies. Property 2.1 (3) will be used in developing bounding functions for robot dynamics.

### 3.4 Uncertainty and Bounding Functions

The dynamics of robot systems as given in (3.3) are usually not totally known. Common uncertainties include unknown parameters, frictions, load variation, unmodelled dynamics, and disturbances. All the terms in (3.3) can be expressed without loss of any generality into two parts:
Chapter 3. Dynamics of Robotic Systems

\[ M = \hat{M} + \Delta_m \quad H = \hat{H} + \Delta_h \]

where

\[ \dot{H}(q, \dot{q}) = \dot{\hat{C}}(q, \dot{q})\dot{q} + \hat{\dot{G}}(q) \]
\[ \Delta_h(q, \dot{q}) = \Delta_c(q, \dot{q})\dot{q} + \Delta_g(q) + \Delta_d \]

where \( \hat{M}, \hat{H}, \hat{C} \) and \( \hat{G} \) are the estimates of the corresponding \( M, H, C \) and \( G \); and \( \Delta_m, \Delta_h, \Delta_c \) and \( \Delta_g \) denote the mismatch between the actual and estimated \( M, H, C, G \), and \( \Delta_d \), respectively.

Robot dynamics are nonlinear and may contain large uncertainties. The existence of large uncertainties makes the trajectory tracking problem challenging and reveals the need for robust control design. As will be shown later, known nonlinear dynamics can be easily stabilised. For many cases, uncertainties are largely unknown but their maximum sizes may be predicted. Control design should be done for the worst possible uncertainties within the predicted bounds; the robust control design is such a design. To this end, we describe in the following the common assumptions on the size of the uncertainties and the associated justification. Assumptions are made and stated for \( M(q), C(q, \dot{q}), G(q), \Delta_d \), and so forth. Note that in most cases a control contains two parts: known nonlinear dynamics are stabilised by a feedback control portion, and only the unknown parts are bounded and then compensated for by a robust control portion. That is, the assumptions on uncertainties are needed only for unknown parts such as \( \Delta_m(q), \Delta_c(q, \dot{q}) \) and \( \Delta_g(q) \). The following are the important assumptions that will be used:

**Assumption 3.1:**

1. Inertia matrix \( M(q) \) satisfies

\[ mI_n \leq M(q) \leq \bar{m}(q)I_n, \quad \forall q \in \mathbb{R}^n \]
where \( m \) is a positive constant, \( \bar{m}(q) \) is a positive definite function of \( q \), and \( I_n \in \mathbb{R}^{n \times n} \) is the identity matrix. It is assumed that \( m \) and \( \bar{m}(q) \) are known.

2. The uncertainty in the Coriolis and centripetal term \( C(q, \dot{q}) \) is bounded by

\[
\| \Delta_c(q, \dot{q}) \| \leq \xi_c(q) \| \dot{q} \|, \quad \forall q, \dot{q} \in \mathbb{R}^n
\]

where \( \xi_c(q) \) is a known, positive definite function of \( q \).

3. Gravity uncertainty \( \Delta_g(q_1) \) and the lumped uncertainty \( \Delta_d \) are bounded as

\[
\| \Delta_g(q_1) \| + \| \Delta_d \| \leq \xi_g(q_1) + \xi_2, \quad \forall q_1 \in \mathbb{R}^n
\]

It is assumed that \( \xi_2 \) is a known constant, and \( \xi_g(q_1) \) is a known positive definite function.

The lumped sum of uncertainties \( \Delta_h \) is given by

\[
\| \Delta_h \| \leq \xi_c(q_1) \| \dot{q}_1 \| + \xi_g(q_1) + \xi_2
\]

### 3.4.1 Flexible Joint

The procedure for developing bounding functions for robot dynamics can be applied to determine bounds for joint flexibility, actuator dynamics, and unmodelled dynamics. First, every parameter can be decomposed into two parts, known and unknown. Then, magnitude bounds on unknown parts can be found using minimum prior information. We introduce the following assumptions for these dynamics.

**Assumption 3.2:**

1. Joint Flexibility: The parameters \( K_s, J, \) and \( D \) in equations (3.4) and (3.5) are assumed to be within known intervals as

\[
0 < K_s \leq \bar{K}_s, \quad 0 < J \leq \bar{J}, \quad 0 < D \leq \bar{D}
\]
where \( \mathbf{K}_s, \mathbf{J}, \mathbf{D}, \mathbf{K}_s, \mathbf{J} \) and \( \mathbf{D} \) are positive matrices which are assumed the lower and upper bound of \( \mathbf{K}_s, \mathbf{J} \) and \( \mathbf{D} \). In addition, if the parameters are time varying, magnitudes of their derivatives up to a necessary order are bounded by known constants as well.

2. The external disturbance or uncertain term \( \Delta_f(q_1, \dot{q}_1) \) is bounded as:

\[
\|\Delta_f\| \leq \xi_f(q_1, \dot{q}_1)
\]

where \( \Delta_f(q_1, \dot{q}_1) \) is some known, well defined function.

**Assumption 3.3:**

1. Motor Dynamics: It is assumed without loss of any generality that

\[
0 < K_m \leq K_m \leq \overline{K}_m, \quad 0 < L_m \leq L_m \leq \overline{L}_m,
\]

\[
0 < R_m \leq R_m \leq \overline{R}_m, \quad 0 < K_b \leq K_b \leq \overline{K}_b,
\]

where \( K_m, L_m, R_m, K_b, \overline{K}_m, \overline{L}_m, \overline{R}_m \) and \( \overline{K}_b \) are known lower and upper bounding matrices of \( K_m, L_m, R_m \) and \( K_b \).

2. The disturbance or uncertainty term \( \Delta_e(q_1, \dot{q}_1) \) in the actuator dynamics is bounded as:

\[
\|\Delta_e(q_1, \dot{q}_1)\| \leq \xi_e(q_1, \dot{q}_1)
\]

where \( \Delta_e(q_1, \dot{q}_1) \) is some known, well defined function \( \xi_e(q_1, \dot{q}_1) \).

Developing bounding functions is one of the key steps in designing a robust controller. Existence of these bounding functions is guaranteed by the inherent properties of robotic systems. It follows from the preceding discussions that, for a given robot, explicit expressions of these bounding functions can always be found. However, finding the coefficients in these bounding functions requires some knowledge of the uncertainties such as ranges.
of parameter variations, maximum variation of load, and size bound on disturbances. In most applications, this information is available. In the case where these coefficients are not known due to lack of information, a robust control can still be designed, as will be shown in Chapter 5 and Chapter 8.

3.5 Concluding Remarks

In this chapter the dynamics of rigid robots, joint flexibility robots and actuators have been discussed. These properties are important in subsequent stability and performance analysis which will be used to derive successful control laws and develop bounding functions for robot manipulators.
Chapter 4

Analysis of the Tetrabot System

4.1 Introduction

The purpose of this chapter is to analyse the computer-controlled Tetrabot system and to give a description of a dynamic model of the Tetrabot. First, analysis of the Tetrabot includes the kinematics, dynamics, trajectory interpolation and linear control of the Tetrabot system as described in Sections 4.1, 4.2, 4.4 and 4.6. Understanding the computer-controlled Tetrabot is an important step towards the development of any new controllers for the real system. The hardware and software architecture of the Tetrabot will be discussed in Chapter 9. The dynamic model of the Tetrabot for the purpose of simulation is described in Section 4.5.

4.2 Tetrabot Mechanism

The Tetrabot mechanism in Figure 4.1 is partitioned into two structures, a three degree of freedom positioning mechanism to which is attached a three degree of freedom wrist. The axes of all three rods intersect at their remote ends and pivot about a single point P at the apex of the tetrahedron. The point P can be placed with three degrees of freedom of position, relative to the upper supporting structure, by changing the lengths of the actuator rods (2) in Figure 4.1.

As it is impossible in practice to have the three rods physically intersecting and rotating
at a single point, a toolplate (4) is introduced which has three symmetrically arranged, radial pivot axes (IV). Each rod end is attached, with a single axis pivot joint (5), to one of these axes perpendicular to the rod axis. A cranked offset in each rod end equal to the toolplate radius causes the rods' axes to intersect at point $P$.  

The simplest six degree of freedom manipulator using this mechanism would have a three degree of freedom wrist attached to the toolplate. In order to simplify the solution of the kinematics equations, a passive radial arm (6) is introduced which is constrained to move in spherical coordinates about a point $G$ by a two axis gimbal (V) and transitional joint (VI), located in the supporting structure. A three axis gimbal joint or spherical joint (8) centred on the point $P$ and connecting the former toolplate to the radial arm accommodates relative rotation between the two. The radial arm is keyed against axial rotation to provide a reference for wrist rotations.

Servo motors driving rotation nuts on the lead screws adjust rod lengths. Incremental encoders attached to these servomotors enable the rod lengths to be determined. To avoid measurement error caused by any superimposed axial rod rotation, each actuator rod is restrained against axial rotation by attaching to it a parallel anti-rotation rod (9) and also a pivot joint (10) at the lower end of, and co-axial with, the actuator rod, allowing relative rotation between cranked rod end and actuator rod. Each actuator rod and anti-rotation rod pair rotates and translates, as one rigid body, about the upper attachment point (3). The net result of the changes to the basic mechanism is that a point $W$ on the linear axis of the radial arm is located in spherical coordinates referred to point $G$ by location of point $P$ which is in turn located by the adjustable lengths of the 3 actuator rods (2).

The mounting of a three axis wrist on the lower end of the radial arm provides the required three $DOF$ of rotation. The wrist mechanism is conventional, comprising three further links ($q_1, q_2, q_3$) attached serially to the lower end of the radial arm. The three wrist axes intersect at point $W$.

Point $T$, the end-effector mounting point on (13), is located with three $DOF$ of position and three $DOF$ of orientation with respect to point $G$ by the three linear rod axes and three rotational wrist axes.

**Tetrabot Manipulator Mechanism ( Figure 4.1)**
Chapter 4. Analysis of the Tetrabot System

Figure 4.1: Tetrabot Manipulator Mechanism
where

\begin{align*}
\mathbf{b} & : \text{base coordinates frame with origin at } \mathbf{B}; \\
\mathbf{B} & : \text{origin of base coordinate frame of the manipulator;} \\
\mathbf{E} & : \text{end effector point, i.e. origin of end effector coordinates frame;} \\
\mathbf{G} & : \text{upper pivot point of } \mathit{rod}_4; \\
\mathbf{Q}_i & : \text{position of } \mathit{rod}_i \text{ upper pivot point;} \\
\mathbf{P} & : \text{common intersection point of actuator rods axes } (1,2,3) \text{ and } \mathit{rod}_4 \text{ axis (central tube);} \\
\mathbf{T} & : \text{toolplate mounting point;} \\
\mathbf{W} & : \text{wrist centre point i.e. intersection point of wrist rotation axes.}
\end{align*}

### 4.3 Tetrabot Kinematics

#### 4.3.1 Tetrabot Kinematics Notation

**4×4 Transformation Transform**

\[
H = \begin{bmatrix}
R & d \\
0 & 1
\end{bmatrix}
\]

\[R \in SO(3)\]

Transformation matrices of the form \(H\) are called homogeneous transformations. \(SO(3)\) stands for Special Orthogonal group of order 3, where \(0\) denotes \((0 \ 0 \ 0)\).

**Tetrabot Kinematics and Geometry (Figure 4.2)**

where

\[
\mathbf{B} : \text{base coordinate frame } (x_b, y_b, z_b);
\]
Figure 4.2: Tetrabot Kinematics and Geometry.
Chapter 4. Analysis of the Tetrabot System

E: end effector coordinate frame \((x_e, y_e, z_e)\) attached to point E;

T: toolplate coordinate frame \((x_t, y_t, z_t)\) attached to point T;

\(E_b : \mathbb{R}^{4 \times 4}, E\) in B;

\(E_t : \mathbb{R}^{4 \times 4}, E\) in T, this is dependent on the end effector geometry.

\(T_b : \mathbb{R}^{4 \times 4}, T\) in B.

\(e_b\): position of E in B;

\(q_i\): rod vector i from \(Q_i\) to P;

\(p_b\): rod vector P from \(Q_i\) to P;

\(Q_i\): position of \(Q_i\) in B;

\(Q_4\): position of the pivot point of radial arm;

\(t_b\): position of T in B;

\(w_b\): position of W in B.

\(q_i\): length of actuator rod from \(Q_i\) to P, \((i = 1,2,3)\)

\(\theta_j\): wrist rotation angle, \((j = 1,2,3)\)

4.3.2 Tetrabot Forward Kinematics

The forward kinematics solution involves the solution for the position and orientation of the toolplate point T given the joint coordinates (base frame B). This solution has not been derived but it is expected that there is an explicit analytical solution, as the position of the point P is determined by the intersection point of three spheres, each having a radius equal to the length of a rod. This part of the solution would be performed iteratively followed by concatenation of three transformations (for the serial wrist links) to achieve the complete solution [22, 46].

The forward kinematics solution computes the forward kinematics from the joint space B(base coordinate \((x_b, y_b, z_b)\)) in Figure 4.2 to the intermediate task space, G and W points then to the toolplate mounting point T.

The conversion of the joint coordinate B(base frame) to the toolplate mounting point T is given by the transformation matrix \(T_b \in \mathbb{R}^{4 \times 4}\).
Given \( E_b \in \mathbb{R}^{4 \times 4} \), consisting of a position vector \((e_b)\) of the point \(E\) and \(3 \times 3\) orientation matrix of frame \(E\) in \(B\), the manipulator joint coordinates are required. \(E_t \in \mathbb{R}^{4 \times 4}\) describes a transformation mapping of a vector from the end effector frame to the toolplate frame. Suppose that the position and orientation of \(T_b\) is given as:

\[
T_b = E_b \cdot (E_t)^{-1}
\]

\[
= \begin{bmatrix}
  n_x & o_x & a_x & t_{bx} \\
  n_y & o_y & a_y & t_{by} \\
  n_z & o_z & a_z & t_{bz} \\
  0 & 0 & 0 & 1
\end{bmatrix}
\]

(4.1)

where \(n, o, a\) (normal, orientation and approach unit vectors) are the directions, in \(B\) of the axes \((x_t, y_t, z_t)\) of the coordinate frame \(T\) with components \((a_x, a_y, a_z)^T\), \((o_x, o_y, o_z)^T\) and \((n_x, n_y, n_z)^T\). \(R\) denote a \(3 \times 3\) rotation matrix and \(d\) denote a three-vector.

Now suppose \(A^j_i\) is the homogeneous matrix that transforms the coordinates of a point from frame \(j\) to frame \(i\). This is called, by convention, a transformation matrix, and is usually denoted by \(T_i\). Let \(A^T_b\) consist of \(A^G_b\), \(A^W_b\) and \(A^T_b\). The conversion of the joint coordinate to the toolplate transformation matrix solution has to be

\[
A^T_b = A^G_b \cdot A^W_b \cdot A^T_b
\]

Let \(c \theta_i = \cos(\theta_i), s \theta_i = \sin(\theta_i), i = 1, 2, 3\). \(c \phi_j = \cos(\phi_j), s \phi_j = \sin(\phi_j), j = 1, 2\).

\[
A^T_b = T_b = \begin{bmatrix}
  n_x & o_x & a_x & t_{bx} \\
  n_y & o_y & a_y & t_{by} \\
  n_z & o_z & a_z & t_{bz} \\
  0 & 0 & 0 & 1
\end{bmatrix}
\]

(4.2)
where 

\[ n_x = c_3(c_2 c_1 c_3 + s_2 s_3) - c_2 s_1 s_3 \]
\[ n_y = c_3(c_1 c_2 s_1 s_2 - c_1 c_2 s_1 - c_2 s_1 s_2) + [-c_1 c_2] + s_1 s_2 s_3 \]
\[ n_z = c_3[-(c_1 c_1 c_2 s_2) - c_2 s_1 s_1 + c_1 c_2 s_2] + [-c_2 s_1 + c_1 s_1 s_3] \]
\[ a_x = -(c_2 s_1 s_3) - (c_2 c_1 c_2 + s_2 s_3) s_3 \]
\[ a_y = c_2 s_1 s_1 - (c_1 c_2 s_1 s_2 - c_1 c_2 s_2 - c_2 s_1 s_2) s_3 \]
\[ a_z = c_3[-(c_1 s_2) + s_1 s_2 s_1] - [-c_1 c_1 c_2 s_2] - c_2 s_1 s_1 + c_1 c_2 s_2 s_3 \]
\[ a_x = -(c_2 s_2 s_2) + c_2 c_1 s_2 \]
\[ a_y = c_2 c_2 s_2 s_1 + c_2 s_1 s_2 - c_2 s_1 s_2 \]
\[ a_z = -(c_1 c_2 s_2) - c_1 c_1 s_1 s_2 - s_1 s_2 s_3 \]
\[ t_{bx} = -(q_4 + z_3) c_2 s_2 + z_4[-(c_2 s_2) + c_2 c_1 s_2] \]
\[ t_{by} = (q_4 + z_3) c_2 s_1 + z_4(c_2 c_2 s_1 s_2 + c_1 s_1 s_2 - c_1 s_1 s_2) \]
\[ t_{bz} = -z_2 - (q_4 + z_3) c_1 c_2 + z_4[-(c_1 c_2 c_2) - c_1 c_1 s_2 s_2 - s_1 s_1 s_2] \]

4.3.3 Tetrabot Inverse Kinematics

The inverse kinematics (referring to Figs. 4.1 and 4.2) solution involves deriving the joint coordinates (three rod lengths, \( q_1, q_2, q_3 \), and three wrist rotation angles, \( \theta_1, \theta_2, \theta_3 \)) for any specified position and orientation of point \( T \) referred to a base coordinate frame \((x_b, y_b, z_b)\) with origin located at point \( B \) and fixed with respect to the support structure. It can be obtained explicitly for the Tetrabot provided that all three wrist axes intersect at a single point \( W \), being almost identical to that for a spherical axis serial manipulator with the addition that the vector position of point \( P \) is required in order to determine the actuator rod lengths. The inverse kinematics solution is given in the Tetrabot inverse kinematics solution.

In the Tetrabot forward kinematics solution we determine the toolplate mounting point \( T \) position and orientation in terms of the joint variables, the homogeneous transformations matrix of the toolplate mounting point \( T \) in the base frame \( B \) is give by equation (4.1):
Chapter 4. Analysis of the Tetrabot System

\[
T_b = \begin{bmatrix}
    n_x & o_x & a_x & t_{bx} \\
    n_y & o_y & a_y & t_{by} \\
    n_z & o_z & a_z & t_{bz} \\
    0 & 0 & 0 & 1
\end{bmatrix}
\]

The vector \( \mathbf{w}_b \) donates the position of the wrist centre point \( \mathbf{W} \) in the base frame \( \mathbf{B} \):

\[
\mathbf{w}_b = \mathbf{t}_b - \mathbf{a} \cdot z_{45}
\]  

(4.3)

where \( z_{45} \) is the distance of \( z_4 \) plus the distance of \( z_5 \) in Fig. (4.2). The vector \( \mathbf{p}_b \) donates the position of the lower pivot point \( \mathbf{P} \) in the base frame \( \mathbf{B} \):

\[
\mathbf{p}_b = \mathbf{Q}_4 + \left[1 - \frac{z_3}{|\mathbf{q}_4|}\right] \cdot \mathbf{q}_4
\]

(4.4)

with

\[
\mathbf{q}_4 = \mathbf{w}_b - \mathbf{Q}_4
\]

(4.5)

where \( |\mathbf{q}_4| \) denotes the modules of the vector. The vector \( \mathbf{Q}_4 = (0, 0, -z_2)^T \) is the position of the rod \( 4 \) upper pivot point. The vector \( \mathbf{q}_4 = (q_{4x}, q_{4y}, q_{4z})^T \) is the radial arm from \( \mathbf{Q}_4 \) to the lower pivot point \( \mathbf{P} \).

Actuator rod vectors and lengths \( (i = 1, 2, 3) \) are

\[
\mathbf{q}_i = \mathbf{p}_b - \mathbf{Q}_i
\]

(4.6)

\[
q_i = |\mathbf{q}_i|
\]

The direction of the axis of rod \( 4 \) (\( \mathbf{q}_4 \)) can be forward from the datum configuration by two rotations \( \phi_1, \phi_2 \) about two perpendicular axes passing through the point \( \mathbf{G}(x_g, y_g, z_g) \):
rotation of $\phi_1$ about $x_g$ following by a rotation of $\phi_2$ about $y_g$. The two Euler rotation angles $\phi_1$ and $\phi_2$ associated with rod_4 are determined as follows:

$$\phi_1 = A \tan 2(q_4y, -q_4z)$$

$$\phi_2 = A \tan 2[-q_4x, \sqrt{(q_4^2 + q_4^2)}]$$

where $A \tan 2(x, y)$ computes $\tan^{-1}(\frac{y}{x})$ but uses the signs of both $x$ and $y$ to determine the quadrant in which the resulting angle lies.

The wrist rotation axes will be referred to as wrist axes 1, 2, 3 in positional order and the respective rotation angles $\theta_1$, $\theta_2$ and $\theta_3$ as in Figure 4.2. In order to satisfy the range of orientations required by the manipulator specification, minimum angular limits of $\pm 90^\circ(\theta_1)$, $\pm 135^\circ(\theta_2)$ and $\pm 180^\circ(\theta_3)$ are required.

Let $c\theta_i = \cos(\theta_i)$, $s\theta_i = \sin(\theta_i), i = 1, 2, 3$. $c\phi_j = \cos(\phi_j)$, $s\phi_j = \sin(\phi_j), j = 1, 2$. The wrist rotation angles are as follow:

$$\theta_1 = \theta'_1 \ (\theta_1 = \theta'_1 + \pi)$$

$$\theta_2 = A \tan 2(s\theta_2, c\theta_2)$$

$$\theta_3 = A \tan 2(s\theta_3, c\theta_3)$$

where

$$\theta'_1 = A \tan 2([-a_zc\phi_1s\phi_2 + a_ys\phi_1s\phi_2 + a_zc\phi_2), a_zs\phi_1 + a_yc\phi_1$$

$$c\theta_2 = -a_zc\phi_1c\phi_2 + a_ys\phi_1c\phi_2 - a_zs\phi_2$$

$$s\theta_2 = (a_zs\phi_1 + a_yc\phi_1)c\theta_1 - (a_zc\phi_1s\phi_2 - a_ys\phi_1s\phi_2 - a_zc\phi_2)s\theta_1$$

$$c\theta_3 = (-o_zc\phi_1c\phi_2 + a_ys\phi_1c\phi_2 - a_zs\phi_2)s\theta_2 - [(a_zs\phi_1 + a_yc\phi_1)c\theta_1 - (o_zc\phi_1s\phi_2 - a_ys\phi_1s\phi_2 - a_zc\phi_2)s\theta_1)c\theta_2$$

$$s\theta_3 = (o_zs\phi_1 + a_yc\phi_1)s\theta_1 + (o_zc\phi_1s\phi_2 - a_ys\phi_1s\phi_2 - a_yc\phi_2)c\theta_1$$
Equations (4.1), (4.3), (4.5) and (4.6) can be used to calculate $q_4$ as:

$$q_4 = p_b - Q_i$$

$$= Q_4 + \left[ 1 - \frac{z_3}{|Q_4|} \right] \cdot q_4 - Q_i$$

$$= Q_4 + \left[ 1 - \frac{z_3}{|w_i - Q_4|} \right] \cdot (w_i - Q_4) - Q_i$$

$$= Q_4 + \left[ 1 - \frac{z_3}{|t_b - z_{45} \cdot a - Q_4|} \right] \cdot [(t_b - z_{45} \cdot a) - Q_4] - Q_i$$  \hspace{1cm} (4.12)

Recalling equation (4.6) and utilising (4.12):

$$q_4 = Q_4 + \left[ 1 - \frac{z_3}{|t_b - z_{45} \cdot a - Q_4|} \right] \cdot [(t_b - z_{45} \cdot a) - Q_4] - Q_i$$  \hspace{1cm} (4.13)

### 4.4 Tetrabot Trajectory Interpolation

#### 4.4.1 Trajectory Generation for the Tetrabot

In this section, methods of computing a trajectory in multidimensional space to describe the desired motion of the manipulator are considered. How trajectories are represented in the computer after they have been planned is considered. The trajectory refers to a time history of position, velocity, and if needed, acceleration for each degree of freedom. There is the problem of actually computing the trajectory from the internal representation, or generating the trajectory. Since these trajectories are computed on digital computers, the trajectory points are computed at a certain rate, called the path update rate [87].

#### 4.4.2 Linear Function with Parabolic Blends (LFPB) for Tetrabot Trajectory Implementation

The LFPB trajectory is appropriate when a constant velocity is desired along a portion of the path. It is such that the manipulator should accelerate to its specific position initially and then decelerate near the goal position. To achieve this, the desired trajectory is specified in three parts. At first part from the start time $t_0$ to time $t_m$ the velocity is ramped
up with a quadratic polynomial. This results in a linear acceleration. At time $t_m$ the position trajectory switches to a linear function; this corresponds to a constant velocity. Finally at time $t_e - t_m$, the trajectory switches once again to a quadratic polynomial to produce linear deceleration.

Choose the blend time $t_m$ so that the position curve is symmetric. Supposing at time $t_0 = 0$, $q_i(0) = q_0$, $\dot{q}_i(t_e) = \ddot{q}_i(0) = 0$ and $\dddot{q}_i(t_e) = \dddot{q}_i(0) = 0$. Between time 0 and $t_m$ we have a quadratic trajectory of the form

$$
\begin{align*}
q_i(t) &= a_0 + a_1 t + a_2 t^2 \\
\dot{q}_i(t) &= a_1 + 2a_2 t \\
\ddot{q}_i(t) &= 2a_2 
\end{align*}
$$

Utilising equations (4.14) with the desired constraints $q_0 = 0$, $\dot{q}(0) = 0$ and $\ddot{q}(0) = 0$ to solve these equations for $a_i$ we obtain

$$
a_0 = q_0; \quad a_1 = 0; \quad a_2 = 0.
$$

At specified time $t_m$, we have a constant velocity $v$

$$
\dot{q}_i(t) = 2a_2 t_m = v
$$

then we have

$$
a_2 = \frac{v}{2t_m} = \frac{a}{2}
$$

where $a$ denotes the acceleration.

The trajectory between time 0 and $t_m$ is given by

$$
\bar{q}_i(t) = q_0 + \frac{v}{2t_m} t^2 = q_0 + \frac{a}{2} t^2
$$
Chapter 4. Analysis of the Tetrabot System

\[ \ddot{q}_i(t) = \frac{v}{t_m} t = at \quad (4.17) \]

\[ \ddot{q}_e(t) = \frac{v}{t_m} = a \]

Between time \( t_m \) and \( t_e - t_m \), the trajectory is a linear segment

\[ \tilde{q}_i(t) = a_0' + a_1' t = a'_1 + vt \quad (4.18) \]

By symmetry and equation (4.17)

\[ \tilde{q}_i\left(\frac{t_f}{2}\right) = \frac{q_0 + q_e}{2} = \frac{a}{2} = a'_0 + vt_e \quad (4.19) \]

we have

\[ a'_0 = \frac{q_0 + q_e - vt_e}{2} \quad (4.20) \]

The two segments (4.17) and (4.19) must blend at time \( t_m \)

\[ \tilde{q}_0 + \frac{v}{2} t_m = \frac{q_0 + q_e - vt_e}{2} + vt_m \quad (4.21) \]

The blend time \( t_m \) is

\[ t_m = \frac{q_0 - q_e + vt_e}{v} \quad (4.22) \]

Let \( 0 < t_m \leq \frac{t_e}{2} \). This leads to

\[ \frac{q_e - q_0}{v} < t_e \leq \frac{2(q_e - q_0)}{v} \]

\[ \frac{q_e - q_0}{T_e} < v \leq \frac{2(q_e - q_0)}{t_e} \quad (4.23) \]

The specified velocity must be between these limits.
Chapter 4. Analysis of the Tetrabot System

Between time $t_e - t_m$ and $t_e$, the trajectory is determined by symmetry. The complete \textit{LFPB} trajectory is given as

$$q_i(t) = \begin{cases} 
q_0 + \frac{a}{2}t^2 & 0 \leq t \leq t_m \\
q_e - \frac{a}{2}v t + vt & t_m < t \leq t_e - t_m \\
q_e - \frac{a}{2}t^2_e + at_e t - \frac{a}{2}t^2 & t_e - t_m < t \leq t_e
\end{cases} \quad (4.24)$$

4.4.3 Joint Interpolated Motion of Tetrabot

The Tetrabot motion is point to point motion. The Tetrabot controller only supports joint interpolated motion. It uses a single parameter which is varying over each 'move' that the Tetrabot is asked to make (this assumes that you want the Tetrabot to move in a series of point-to-point motions with a momentary stop between each). This single parameter is varied over its range with a limited first and second derivative. The parameter is used to calculate the intermediate points the robot should visit, one per time slot, in order to achieve a smooth motion. The map from the scalar parameter to the vector position can be given a \textit{joint interpolated motion} (giving a straight line in joint space but a curved path in real space). In each case, the velocity and acceleration limits in the target space have to be converted back to parameter space for each move so that no joint attempts to exceed its maximum torque.

The low level interpolator function has been written in Assembly language and interfaced to a C language at the middle level. The lowest level of programming in the computer system is machine language which defines binary coded words that have specific meaning for a particular CPU.

\textbf{Interpolation Notation}

\textbf{m/c units or mu:} machine units (32-bit binary number)

\textbf{pos}_1: \text{initial position(i.e. motion start point, m/c units)}

\textbf{pos}_2: \text{target position(i.e. motion end point, m/c units)}
maxdp: maximum change in joint position (m/c units)

\[ maxdp = |(pos_2[i] - pos_1[i])| \]

\( P_{\text{max}} \): maximum move parameter (m/c units)

\[ P_{\text{max}} = 65536 \]

\( V_{\text{max}} \): maximum velocity (interpolation velocity scale)

\( V_{\text{sta}} \): joint velocity limits (mm/s, rads/s) is converted to m/c units.

\[ V_{\text{max}} = \frac{P_{\text{max}}}{\max dp} \times V_{\text{sta}} \]

\( A_{\text{max}} \): maximum acceleration (interpolation acceleration scale)

\( A_{\text{sta}} \): joint acceleration limits (mm/s², rads/s²) is converted to m/c units.

\[ A_{\text{max}} = \frac{P_{\text{max}}}{\max dp} \times A_{\text{sta}} \]

\( \text{Time} \): longest joint motion time.

\[ Time = \begin{cases} 
2\sqrt{\frac{dp}{A_{\text{sta}}}}, & \frac{V_{\text{sta}} \times V_{\text{sta}}}{A_{\text{sta}}} > \max dp. \\
\frac{dp}{V_{\text{sta}}} + \frac{V_{\text{sta}}}{A_{\text{sta}}}, & \frac{V_{\text{sta}} \times V_{\text{sta}}}{A_{\text{sta}}} \leq \max dp.
\end{cases} \]  \hspace{1cm} (4.25)

When \( \frac{V_{\text{sta}} \times V_{\text{sta}}}{A_{\text{sta}}} > \max dp \) the joint cannot reach maximum velocity, \( \text{Time} = 2\sqrt{dp/A_{\text{sta}}} \).

When \( \frac{V_{\text{sta}} \times V_{\text{sta}}}{A_{\text{sta}}} \leq \max dp \) the joint can reach maximum velocity, \( \text{Time} = \frac{dp}{V_{\text{sta}}} + \frac{V_{\text{sta}}}{A_{\text{sta}}} \). And \( \text{Time} > \text{Tmax} \)

we can get slowest joint, \( \max dp \) and \( \text{Tmax} = \text{Time} \).
Chapter 4. Analysis of the Tetrabot System

Time Optimum for Interpolation

An important variation of the trajectory is obtained by leaving the final time $t_e$ unspecified and seeking the fastest trajectory between $q_0, q_1, \ldots, q_5$ with a given constant acceleration $a$. This is called a Bang-Bang trajectory since the optimal solution is achieved with the acceleration at its maximum value $+a$ until an appropriate switching time $t_s$ at which time it abruptly switches to its minimum value $-a$ (maximum deceleration) from $t_s$ to $t_e$.

The minimum transition time is:

$$t_{i,\text{min}} = 2\sqrt{\frac{q_f,i - q_0,i}{a_i}};$$  \hspace{1cm} (4.26)

Before Tetrabot interpolation, the minimum displacement demand joint is selected by the point-to-point minimum-time control algorithm. Given multiple joint coordinates solution vectors to the inverse kinematics solution, find the optimum target joint vector for joint interpolated motion from current joint coordinates and return a pointer to the selected target joint vector. The criterion is to minimise the sum of the wrist displacements. This has the desirable effect of eliminating large wrist displacements for small transitional or rotational movement of the toolplate, especially near the degenerate wrist point.

4.5 Tetrabot Dynamics

4.5.1 Tetrabot Dynamic Model

The dynamics of the Tetrabot are properly represented by a set of nonlinear differential equations. The Tetrabot consists of three parallel linear actuator rods and three serial wrist links. The kinematics of the three serial wrist links is largely decoupled from the three upper parallel rods. For a given task, the torques produced by the rod driving systems are generally much greater than those for wrists, and therefore system modelling and control will be concentrated on the three parallel actuator rods and their drive systems.

According to the dynamics representation of a robotic system in Chapter 3, the Tetrabot dynamic model can be expressed as
Chapter 4. Analysis of the Tetrabot System

\[ M(t, q(t), \omega) \ddot{q}(t) = U(t, q(t), \dot{q}(t), \omega) + D(t, q(t), \dot{q}(t), \omega) \dot{u}(t) \]  

(4.27)

\[ \omega \in \Omega \]  

(4.28)

where \( t \in \mathbb{R} \) represents time, \( q(t) \in \mathbb{R}^3 \) is a vector of generalised co-ordinates which describe the configuration of the system, \( \dot{u}(t) \in \mathbb{R}^3 \) is a vector of control inputs and \( \omega \) is the lumped uncertain element. The function \( M : \mathbb{R} \times \mathbb{R}^3 \times \Omega \rightarrow \mathbb{R}^{3 \times 3} \) is the inertia matrix; \( U : \mathbb{R} \times \mathbb{R}^3 \times \mathbb{R}^3 \times \Omega \rightarrow \mathbb{R}^3 \) represents the vector of controlled generalised force, and \( D : \mathbb{R} \times \mathbb{R}^3 \times \mathbb{R}^3 \times \Omega \rightarrow \mathbb{R}^{3 \times 3} \) is a non-singular matrix.

Figure 4.3: The force structure of the Tetrabot

where

\( W' \): centre of mass of total payload (end-effector and payload);
Chapter 4. Analysis of the Tetrabot System

$Z'_3$: axial distance along $rod_4$ from point $P$ to $W'$;

$(x_i, y_i, z_i)$: coordinate frame attached to centre of mass of $rod_i$, $i = 1, 2, 3, 4$;

$a_i$: linear acceleration of $rod_i$ centre of mass; $f_{ri}$: axial force in $rod_i$;

$f_{oi}$: force applied to $rod_i$ by upper pivot at $Q_i$;

$f_{o'i}$: force applied to $rod_i$ by upper pivot at $Q_i$, expressed in $rod_i$ coordinate frame;

$f_{ii'}$: force applied to $rod_i$ at $P$;

$f_{li}$: force applied to $rod_i$ by $P$, expressed in $rod_i$ coordinate frame;

$f_w$: external force applied to point $W'$;

$m_i$: mass of $rod_i$;

$m_p$: total payload mass, end-effector plus payload;

$n_w$: external torque applied to point $W'$;

$g$: gravitational acceleration vector;

$r_i$: distance from $P$ to $rod_i$ centre of mass;

$v_p$: velocity of point $P$;

$v_w$: velocity of point $Q'$;

$w_b$: position of $W'$ in $b$;

$\alpha_i$: angular acceleration of $rod_i$;

$\omega_i$: angular velocity of $rod_i$.

The drive dynamic model in equations (4.27) - (4.28) is used for the purpose of simulation. The inverse dynamic solution has been obtained for the simplified three DOF model of the Tetrabot shown in Figure 4.3. The model consists of five bodies: three similar actuator rods ($rod_1$ to $rod_3$), a radial arm ($rod_4$) and a payload point mass at $W'$. The distance $PW'$ represents an increase over $PW$ to allow for the offset between the payload and the point $P(x_p, y_p, z_p)$ due to the intervening wrist. The three parallel rods are effectively jointed at the lower pivot point $P$ and their lengths are varied via a ball-screw driving mechanism to provide the position of the centre $W$ ($x_w, y_w, z_w$) of the wrist axes in Figure 4.3. The manipulator is modelled as a three DOF device with the wrist mechanism and
radial arm combined together as a single rigid body, rod4. The end-effector and payload are also modelled as a single body, with centre of mass at point W.

In Figure 4.3 the base frame B attached to the fixed equilateral triangle base (1) has its origin at centre of the triangle, with the x-axis pointing at point Q1 and the z-axis along the vertical direction. Frame i is attached to rod_i (i = 1, 2, 3, 4) such that it is aligned with the base frame B when rod_i is not very relevant; only its axis direction is of interest here. Frame i can be obtained by rotating rod_i with an angle of φ_i1 around the x-axis followed by an angle of φ_i2 around the y-axis.

The frame i is transformed into the base frame B as follows:

\[
T^i_b = \begin{bmatrix}
\cos\phi_{i1} & 0 & \sin\phi_{i2} \\
\sin\phi_{i1}\sin\phi_{i2} & \cos\phi_{i1} & -\sin\phi_{i1}\cos\phi_{i2} \\
-\cos\phi_{i1}\sin\phi_{i2} & \sin\phi_{i1} & \cos\phi_{i1}\cos\phi_{i2}
\end{bmatrix}
\]  

(4.29)

It will be shown that φ_i1 and φ_i2 can be determined given the position of the point P. In simulation and control of the Tetrabot, the kinematics from point P represented by (x_p, y_p, z_p) in frame B to the rod lengths q_1, q_2 and q_3, and the inverse kinematics which is from q_i to point P, will be invoked repeatedly.

The position of point P is determined by the intersection point of three spheres, each having a radius equal to the length of a rod, and at Q_i:

\[
(x_p - x_i)^2 + (y_p - y_i)^2 + (z_p - z_i)^2 = q_i^2 
\]

(4.30)

Consider a system status in the frame B as

\[
q_i = \sqrt{(x_p - x_i)^2 + (y_p - y_i)^2 + (z_p - z_i)^2}
\]

\[
\dot{q}_i = \frac{\dot{x}_p(x_p - x_i) + \dot{y}_p(y_p - y_i) + \dot{z}_p(z_p - z_i)}{q_i}
\]

\[
\ddot{q}_i = \frac{\ddot{x}_p(x_p - x_i) + \ddot{y}_p(y_p - y_i) + \ddot{z}_p(z_p - z_i) + \dot{x}_p^2 + \dot{y}_p^2 + \dot{z}_p^2}{q_i}
\]  

(4.31)
Chapter 4. Analysis of the Tetrabot System

The two Euler rotation angles $\phi_1$ and $\phi_2$ associated with $rod_i$ are determined as follows ([22]):

$$\phi_2 = \arcsin \left[ \frac{x_i - x_p}{q_i} \right] \quad \phi_1 = -\arcsin \left[ \frac{y_i - y_p}{q_i \cos \phi_2} \right] \quad (4.32)$$

The dynamic system analysis is desirable to characterise the dynamic variables relating to the Centre of Mass (COM) of each rod. Assuming that the distance from the COM of $rod_i$ to the intersection point $P$ is $r_i$, and and the COM is at $(x_{ri}, y_{ri}, z_{ri})$ in frame $B$, then it is obtained that

$$\frac{x_{ri} - x_p}{x_i - x_p} = \frac{y_{ri} - y_p}{y_i - y_p} = \frac{z_{ri} - z_p}{z_i - z_p} = k \quad (4.33)$$

with

$$k = \frac{r_i}{q_i} \quad (4.34)$$

4.5.2 Inverse Dynamics and Newton-Euler Equations

The inverse dynamic solution has been obtained for the simplified three DOF model of the Tetrabot shown in Figure 4.3. The model consists of five bodies, three similar actuator rods ($rod_1$ to $rod_3$), a radial arm ($rod_4$) and a payload point mass at $W'$. The wrist mechanism is amalgamated with the radial arm to form the rigid body, $rod_4$. The distance $Z_3'$ represents an increase over $Z_3$ to allow for the offset between the payload and point $P$ due to the intervening wrist.

The inverse dynamics of the Tetrabot are amenable to analysis using a form of the Newton-Euler formulation of manipulator dynamics.

If all the system dynamic variables such as linear and angular velocities, linear and angular accelerations are available, then the forces and torques exerted on each rod can be evaluated by Newton-Euler (N-E) equations.

Form the N-E equations of motion for each rod:
Chapter 4. Analysis of the Tetrabot System

\[ F_i = m_i a_i \] (4.35)

\[ N_i = M_i \alpha_i + \omega_i \times (M_i \omega_i) \] (4.36)

where \( M_i \) is the inertia matrix of rod \( i \) in frame \( b \), \( F_i \) is the total vector force exerted on rod \( i \). \( N_i \) is the total vector torque exerted on rod \( i \) about its centre of mass.

In frame \( i \), the Newton-Euler equations for rod \( i \) are given by [57]:

\[ \begin{bmatrix} f_{oi}^i + f_{ti}^i + m_i T_b^i g + T_b^i f_{ext,i} \end{bmatrix} = m_i T_b^i a_i \] (4.37)

\[ f_{oi}^i \times \begin{bmatrix} 0 \\ 0 \\ r_i - q_i \end{bmatrix} + f_{ti}^i \times \begin{bmatrix} 0 \\ 0 \\ r_i \end{bmatrix} + T_b^i n_{ext,i} = M_i \alpha_i^i + \omega_i^i \times (M_i \omega_i^i) \] (4.38)

where \( f_{ext,i} \) is a known external force vector and \( n_{ext,i} \) an external torque vector, both acting at the centre of mass of the rod.

In equation (4.37), \( f_{oi}^i \) and \( f_{ti}^i \) are external forces exerted on the upper pivot point \( Q_i \) and lower pivot point \( P_i \); \( m_i \) is the total rod mass; \( g = [0, 0, -9.8]^T \); \( \alpha_i = [\ddot{x}_{ri}, \ddot{y}_{ri}, \ddot{z}_{ri}]^T \) with \( \ddot{x}_{ri}, \ddot{y}_{ri} \) and \( \ddot{z}_{ri} \) given in [34]. These vectors are zero except for rod \( 4 \) when they have values determined by the payload and by the working force \( f_w \) and torque \( n_w \).

Transform the forces \( f_{oi}^i \) and \( f_{ti}^i \) back into the base coordination frame

\[ f_{oi}^i = T_i^b f_{oi}^i \quad f_{ti}^i = T_i^b f_{ti}^i \] (4.39)

The force sum is

\[ F = f_{t1} + f_{t2} + f_{t3} + f_{t4} \] (4.40)
Chapter 4. Analysis of the Tetrabot System

Referring to Figure 4.3, each actuator $rod_i$ exerts a force of magnitude $f_{ri}$ and directed at the point $P$. These forces sum at $P$ as

$$F = f_{r1}n_1 + f_{r2}n_2 + f_{r3}n_3$$ (4.41)

This can be rewritten as

$$F = Nf = (M^{-1})^Tf$$ (4.42)

The rod forces are given by

$$f = (M^T)F$$ (4.43)

with

$$M_i^i = \begin{bmatrix} I_i & 0 & 0 \\ 0 & I_i & 0 \\ 0 & 0 & 0 \end{bmatrix}$$ (4.44)

which means that $M_i^i$ about the rod direction is neglected.

In equation (4.38) $r_i$ is the distance of the COM of $rod_i$ to the point $P$; $\omega_i^i$ and $\alpha_i^i$ are the angular velocity and acceleration in frame $i$ as given in [34]

The angular velocity $\omega_i = (\omega_{iz}, \omega_{iy}, \omega_{ix})$ of $rod_i$ in the base frame $B$ can be determined by

$$\omega_i = \begin{bmatrix} \dot{\phi}_{i1} \\ \cos \phi_{i1} \phi_{i2} \\ \sin \phi_{i1} \dot{\phi}_{i2} \end{bmatrix}$$ (4.45)

The angular acceleration of the $rod_i$ in the base frame $B$ is
\[ \alpha_i = \dot{\omega}_i = \begin{bmatrix} \ddot{\phi}_{i1} \\ -\sin\phi_{r1}\dot{\phi}_{r1}\dot{\phi}_{r2} + \cos\phi_{i1}\dot{\phi}_{i2} \\ \cos\phi_{r1}\dot{\phi}_{r1}\dot{\phi}_{r2} + \sin\phi_{i1}\dot{\phi}_{i2} \end{bmatrix} \] (4.46)

The angular velocity and acceleration of rod \( i \) in the frame \( i \) are

\[ \omega_i = T^i_b \omega_i = (T^b_i)^T \omega_i \begin{bmatrix} \cos\phi_{i2}\dot{\phi}_{i1} \\ \dot{\phi}_{i2} \\ \sin\phi_{i2}\dot{\phi}_{i1} \end{bmatrix} \] (4.47)

\[ \alpha_i = T^i_b \alpha_i = (T^b_i)^T \alpha_i \begin{bmatrix} -\sin\phi_{r2}\dot{\phi}_{r1}\dot{\phi}_{r2} + \cos\phi_{i2}\dot{\phi}_{i1} \\ \dot{\phi}_{i2} \\ \cos\phi_{r2}\dot{\phi}_{r1}\dot{\phi}_{r2} + \sin\phi_{i2}\dot{\phi}_{i1} \end{bmatrix} \] (4.48)

### 4.5.3 Drive Dynamics

The drive dynamic model equation (4.49) is used for the purpose of simulation. \( q = [q_1, q_2, q_3]^T \) is the status of the Tetrabot system.

The motor torque available to control the \( i \)-th rod of the Tetrabot is given by [34] and [57]

\[ \tau_{mi} = \left( \frac{NmL}{\eta} + \frac{NI_n}{L} + \frac{I_m}{NL} \right) \ddot{q}_i + \frac{LN}{\eta} f_{ri} \] (4.49)

where

\( \ddot{q}_i \): second time derivative of rod length;

\( f_{ri} \): axial force in rod \( i \);

\( L \): lead of ballscrew, meters/revolution (\( m/\text{rads} \));

\( \eta \): efficiency of ballscrew;

\( N \): motor to nut drive ratio = nut rotation/motor rotation;

\( I_n \): driveline moment of inertia of nut assembly (\( \text{kgm}^2 \));
Chapter 4. Analysis of the Tetrabot System

\( I_m \): driveline moment of inertia of motor assembly \((kgm^2)\);

\( m \): total load and screw mass \((kg)\);

\( i \): number of the motor \((i = 1, 2, 3)\).

The force \( f_{ri} \) is the axial component of the upper pivot force at the point \( Q_i \) in Figure 4.3 which is exerted on the rod by the ballscrew nut of the rod drive actuator. The torque generated by the drive motor must provide angular acceleration of the actuator drive, the excess torque being available to generate \( f_{ri} \). The rod forces \( f_{ri} \) can be given by

\[
\begin{bmatrix}
  f_{r1} \\
  f_{r2} \\
  f_{r3}
\end{bmatrix} = T_s^{-1} \sum_{i=1}^{4} T_i^i \begin{bmatrix}
  f_{1a}^i \\
  f_{1v}^i \\
  f_{1s}^i + f_{1ss}^i
\end{bmatrix} + \begin{bmatrix}
  f_{2a}^i \\
  f_{2v}^i \\
  f_{2s}^i + f_{2ss}^i
\end{bmatrix} + \begin{bmatrix}
  f_{3a}^i \\
  f_{3v}^i \\
  f_{3s}^i + f_{3ss}^i
\end{bmatrix} + \begin{bmatrix}
  T_2^i \\
  T_3^i \\
  T_4^i
\end{bmatrix} \begin{bmatrix}
  0 \\
  0 \\
  0
\end{bmatrix} + T_5^i \begin{bmatrix}
  0 \\
  0 \\
  0
\end{bmatrix}
\]

(4.50)

where \( f_{oi} \) and \( f_{hi} \) are external forces exerted on the upper pivot point \( Q_i \) and lower pivot point \( P \); the matrix \( T_s \) is composed of the three columns of \( T_1^1, T_2^2 \) and \( T_3^3 \):

\[
T_s = \begin{bmatrix}
  x_1 - x_P \\
  y_1 - y_P \\
  z_1 - z_P
\end{bmatrix}
\]

(4.51)

4.6 Linear Control of the Tetrabot

In Section 4.5, the Tetrabot dynamic model has been investigated using a set of nonlinear ordinary differential equations. In this section we will discuss a class of linear control systems. The use of linear control techniques is only valid when the system being studied can be mathematically modelled by linear differential equations. However, it is often reasonable to make such approximations, and it also is the case that these linear methods are the ones most often used in current industrial practice. Consideration of the linear approach will serve as a base assessment for the more complex treatment of nonlinear control systems in subsequent chapters.
4.6.1 Actuator Dynamics

Recall the drive dynamic equation (4.49). The motor torque of the $r_{od_i}$ actuator is used for the purpose of simulation.

A block diagram representing each joint of the Tetrabot is shown in Figure 4.4.

![Block diagram for the Tetrabot position control](image)

Figure 4.4: The block diagram for the Tetrabot position control

$$\tau_{mi} = \left( \frac{NmL}{\eta} + \frac{NI_n}{L} + \frac{J_m}{NL} \right) \ddot{q} + \frac{LN}{\eta} f_{ri} \quad (i = 1, 2, 3) \quad (4.52)$$

4.6.2 Set-point Tracking Control

Early control studies for the Tetrabot have assumed the six joints are decoupled, and have resulted in a primary control system for set-point tracking using PID controllers to position each joint in the computer controlled Tetrabot system.

The comprehensive nonlinear model of the Tetrabot dynamics has been developed in Section 4.4. The description of linear models about various operating points can be obtained for controller design studies. Here only the design of $PID$ controllers for the parallel structure is considered. This model includes a rigid mechanical structure, driving systems with linear properties and linear components in each of the amplifier blocks in Figure 4.4.

Let us assume that the Tetrabot is in the horizontal plane ($g(q) \equiv 0$), and that the task
is simply to move it to a given final position, as specified by a constant vector \( \mathbf{q}_R \) of desired joint angles. It is physically clear that a joint proportional-integral-derivative (PID) controller, namely a feedback control law that selects each actuator input independently, based on the local measurements of position errors \( e_i = q_i - q_{ri} \) and joint velocities \( \dot{q}_i \) \((i = 1, \ldots, 3)\), will achieve the desired position control task. Indeed, the control law (4.53) is used, where \( k_{pi}, k_{ii} \) and \( k_{di} \) are strictly positive constants.

\[
\tau_i = -k_{pi}e_i - k_{di}\dot{q}_i - k_{ii}\int e_i dt
\]  

The stability and convergence proof for the PID controller above can then be derived very simply. Let us actually take the control input in a form slightly more general than (4.53):

\[
\tau = -K_P e - K_D \dot{q} - K_I \int e dt
\]  

where \( K_P, K_I \) and \( K_D \) are constant symmetric positive definite matrices.

For controller design studies, the PID control design is evaluated with point to point motion of the end effector. The trajectory involved diagonal movement across the primary working volume at point \( W \) from \( x_{w0} = 0.0, y_{w0} = -0.5 \) (m), \( z_{w0} = -1.475 \) (m) to \( x_{wd} = 0.0, y_{w0} = 0.5 \) (m), \( z_{w0} = -0.74 \) (m), \( \dot{x}_w = 0.0, \dot{y}_w = 0.0 \) and \( \dot{z}_w = 0.0 \), which is the largest distance possible within the primary working volume, using a PID controller. The required driving torques are \( T_{m1} = 0.58992 \text{ (Nm)} \), \( T_{m3} = -0.19392 \text{ (Nm)} \) and \( T_{m3} = 1.16181 \text{ (Nm)} \), and the track errors \( e_1 = -1.1921 \times 10^{-7}, e_2 = -1.5497 \times 10^{-6} \) and \( e_3 = 7.1526 \times 10^{-7} \). Let \( K_{Pi} = 1; K_{ii} = 1; K_{Di} = 1 \). When the payload \( m_p = 0 \text{ kg} \), the PID tracking errors are shown in Figure 4.5. When the payload \( m_p = 6 \text{ kg} \), the PID tracking errors are shown in Figure 4.6. Let \( K_{Pi} = 1; K_{ii} = 4.5; K_{Di} = 1 \). When the payload \( m_p = 0 \text{ kg} \) the PID tracking errors are shown in Figure 4.7. When the payload \( m_p = 6 \text{ kg} \), the PID tracking errors are shown in Figure 4.8.
Chapter 4. Analysis of the Tetrabot System

Figure 4.5: (a) The PID controller for Tetrabot with $K_p=1$, $K_i=1$, $K_d=1$ and $mp=0$ Kg

Figure 4.6: (b) The PID controller for Tetrabot with $K_p=1$, $K_i=4.5$, $K_d=1$ and $mp=0$ Kg

Figure 4.7: (c) The PID controller for Tetrabot with $K_p=1$, $K_i=1$, $K_d=1$ and $mp=6$ Kg
Figure 4.8: (d) The PID controller for Tetrabot with $K_p=1$, $K_i=4.5$, $K_d=1$ and $m_p=6$ Kg

4.7 Concluding Remarks

Analysis of the Tetrabot system is very important to understand and implement a nonlinear controller for the Tetrabot.

The Tetrabot mechanism is partitioned into two structures, a three degree of freedom positioning mechanism to which is attached a three degree of freedom wrist. The forward kinematics solution computes the forward kinematics from the joint space (base coordinate $B(x_b, y_b, z_b)$) to the intermediate task space, $G$ and $W$ points then to the toolplate mounting point. The inverse kinematics solution involves deriving the joint coordinates (three rod lengths, $q_1, q_2, q_3$, and three wrist rotation angles, $\theta_1, \theta_2, \theta_3$) for any specified position and orientation of point $T$ referred to a base coordinate frame with origin located at point $B$ and fixed with respect to the support structure. The trajectory interpolation has been performed using a linear function with parabolic blends ($LFPB$) method. The time optimum control algorithm is used. The Tetrabot dynamics includes inverse dynamics and drive dynamics. The system dynamic model has been modelled as a three DOF device with the wrist mechanism and radial arm combined together as a single rigid body, $rod_4$. The dynamic model using a set of ordinary differential equations will be used for the purpose of simulation to develop and test robust tracking controllers in subsequent Chapters. Finally linear set-point tracking control has been simulated on the Tetrabot model. This will be used as a bench mark for the changes of future controllers which will be developed later in the thesis.
Chapter 5

Robust Tracking via Sliding Mode Control for Flexible Joint Manipulators

5.1 Introduction

In this Chapter, robust tracking via sliding mode theory will be presented and given one case study which relates to flexible joint manipulators. A new scheme is presented in Section 2.4 for robust tracking control via sliding mode technique in the presence of model uncertainty and disturbances. Based on the robust sliding mode control methodology, the control scheme addresses the following problem: given the extent of parametric uncertainty and the external uncertainties, design a nonlinear sliding mode controller to achieve robust tracking precision. In this work, the explicit robustness guarantees provided by the methodology are demonstrated using elastic joint manipulator models. The methodology is compared with the traditional feedback linearisation. A dynamics of flexible joint manipulator is investigated in Section 5.2. A robust tracking control for flexible joint manipulators is developed in Section 5.3. In Section 5.4, the robust tracking controller is applied to a two DOF manipulator with flexible joints. Simulation results show excellent robust performance.
5.2 Elastic Joint Manipulator Dynamic Model

In this section, the specific effects of parameter uncertainty, computational error and model simplification for an elastic joint manipulator are investigated.

The dynamics of an n-link manipulator with elastic joints (3.4) - (3.5) may be described by the equations of motion found from the Euler-Lagrange equations in Section 3.2. Let \( q_1 \in \mathbb{R}^n \) denote the vector of angular positions of the link, \( q_2 \in \mathbb{R}^n \) be the vector of angular positions of the motor and \( y \in \mathbb{R}^n \) denote the vector of system outputs which is the angular positions of the link \( q_1 \). The differential equations governing the controlled motion with uncertainties are given by

\[
\begin{align*}
J_i(q_i)\ddot{q}_i + H(q_i, \dot{q}_i) + k(q_i - q_2) &= 0 \\
J_m\ddot{q}_2 - k(q_i - q_2) &= F \\
y &= q_1
\end{align*}
\]  

(5.1)

where \( J_i(q_i) \in \mathbb{R}^{n \times n} \) denotes the inertia matrix of the rigid robot, \( J_m \in \mathbb{R}^{n \times n} \) denotes the inertia matrix of the motors and \( k \) is a diagonal matrix representing flexible joint stiffness.

Supposing \( H(q_1, \dot{q}_1) \) is the lumped sum of nonlinearities given by

\[
H(q_1, \dot{q}_1) = C(q_1, \dot{q}_1)\dot{q}_1 + G(q_1) + \Delta_d
\]

\[
= \hat{C}(q_1, \dot{q}_1)\dot{q}_1 + \Delta_c(q_1, \dot{q}_1)\dot{q}_1 + \Delta_g(q_1) + \Delta_d
\]

(5.2)

Define the lumped sum of uncertainties \( \Delta_h \) by

\[
\Delta_h = \Delta_c(q_1, \dot{q}_1)\dot{q}_1 + \Delta_g(q_1) + \Delta_d
\]

(5.3)

The \( H(q_1, \dot{q}_1) \in \mathbb{R}^n \) is the vector containing Coriolis, centripetal, and gravitational forces and uncertainties. According to Section 3.4, \( C(q_1, \dot{q}_1) \in \mathbb{R}^{n \times n} \) is the matrix containing
actual centripetal and Coriolis terms, $G(q_1) \in \mathbb{R}^n$ is the vector containing actual gravity terms, $\hat{C}$ and $\hat{G}$ denote estimates of the corresponding $C$ and $G$, $\Delta_c$ and $\Delta_g$ are the mismatch between the actual and estimated centripetal and Coriolis terms, and $\Delta_d \in \mathbb{R}^n$ is the vector of all bounded disturbances and is in general an unknown nonlinear function of $q_1$ and $\dot{q}_1$. 

Supposing the joint forces applied by the actuators have the form

$$F = u - \Delta_f(q_1, \dot{q}_1)$$ (5.4)

The vector $u \in \mathbb{R}^n$ is the additional joint force beyond the compensation forces and will be referred to as the control. $\Delta_f(q_1, \dot{q}_1)$ is a vector of any possible nonlinear external disturbances whose bounding function depends only on $q_1$ and $\dot{q}_1$.

The equation (5.1) for robotic systems is usually not totally known. Common uncertainties include unknown parameters, load variation, unmodelled dynamics and disturbances. The existence of large uncertainties makes the trajectory tracking problem challenging and reveals the need for robust control design. Control design should be done for the worst possible uncertainties within the predicted bounds; we have described the following the common assumptions on the size of uncertainties and their justifications in Section 3.4. Assumptions 3.1 - 3.2 of Section 3.4 are made for $C(q_1, \dot{q}_1)$, $G(q_1)$, $\Delta_d$ and $\Delta_f$. The following assumptions on the uncertainties are needed only for the mismatch parts $\Delta_c(q_1, \dot{q}_1)$, $\Delta_g(q_1)$ and $\Delta_d$.

**Assumption 5.1**

1. The Coriolis and centripetal term $C(q_1, \dot{q}_1)$ is linear in $\dot{q}_1$. It follows that

$$\|\Delta_c(q_1, \dot{q}_1)\| \leq \xi_c(q_1) \|\dot{q}_1\|, \quad \forall q_1 \in \mathbb{R}^n, \dot{q}_1 \in \mathbb{R}^n$$

where $\xi_c(q_1)$ is a known, positive definite function of $q_1$.

2. Gravity $\Delta_g(q_1)$ and lumped uncertainty $\Delta_d$ are bounded as

$$\|\Delta_g(q_1)\| + \|\Delta_d\| \leq \xi_g(q_1) + \xi_d, \quad \forall q_1 \in \mathbb{R}^n$$
It is assumed that $\xi_d$ is a known constant, and $\xi_g(q_1)$ is a known positive definite function.

The lumped sum of uncertainties $\Delta_h$ is bounded by

$$\|\Delta_h\| \leq \xi_c(q_1) \|\dot{q}_1\| + \xi_g(q_1) + \xi_d$$

3. The external disturbance or uncertain term $\Delta_f(q_1, \dot{q}_1)$ is bounded as:

$$\|\Delta_f\| \leq \xi_f(q_1, \dot{q}_1)$$

where $\Delta_f(q_1, \dot{q}_1)$ is some known, well defined function.

**Dynamics of Elastic Joint Manipulators**

The state variables are defined as the angular position of the links $x_1 = q_1 \in \mathbb{R}^n$, the corresponding angular velocities $x_2 = \dot{q}_1 \in \mathbb{R}^n$, the angular positions of the motors $x_3 = q_2 \in \mathbb{R}^n$ and the corresponding angular velocities $x_4 = \dot{q}_2 \in \mathbb{R}^n$. The state variable representation of (5.1) is obtained as

$$\begin{align*}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= -J_l(x_1)^{-1}(H(x_1, x_2) + k(x_1 - x_3)) \\
\dot{x}_3 &= x_4 \\
\dot{x}_4 &= J_m^{-1}k(x_1 - x_3) + J_m^{-1}F \\
y &= x_1
\end{align*}$$

An input-independent state coordinate transformation can be used to obtain a GCCF in the form of (2.5) which can be written in vector form as

$$\begin{align*}
\eta_1 &= x_1 \\
\eta_2 &= x_2 \\
\eta_3 &= -J_l(x_1)^{-1}\{H(x_1, x_2) + kx_1 - kx_3\} \\
\eta_4 &:= A(x_1, x_2, x_3) + J_l(x_1)^{-1}kx_4
\end{align*}$$
Chapter 5. Robust Tracking via Sliding Mode Control for Flexible Joint Robots

with

\[ \Lambda(x_1, x_2, x_3) = -\frac{d}{dt}\{J_1(x_1)^{-1}\}[H(x_1, x_2) + k(x_1 - x_3)] - J_1(x_1)^{-1}\{\frac{\partial H}{\partial x_1}x_2 + \frac{\partial H}{\partial x_2}[-J_1(x_1)^{-1}(H(x_1, x_2) + kx_1 - kx_3)] + kx_2} \]

where \(\eta_1 \in \mathbb{R}^n, \eta_2 \in \mathbb{R}^n, \eta_3 \in \mathbb{R}^n\) and \(\eta_4 \in \mathbb{R}^n\).

The inverse can be found by inspection to be

\[
\begin{align*}
x_1 &= \eta_1 \\
x_2 &= \eta_2 \\
x_3 &= \eta_1 + k^{-1}\{J_1(\eta_1)\eta_3 + H(\eta_1, \eta_2)\} \\
\quad &= \Gamma(\eta_1, \eta_2, \eta_3) \\
x_4 &= k^{-1}J_1(\eta_1)\{\eta_4 - \Lambda(\eta_1, \eta_2, \eta_3)\}
\end{align*}
\]

Substituting \(x_1\) and \(x_3\) of the equation (5.8) into \(\dot{x}_4\) of the equation (5.5),

\[
\dot{x}_4 = -J_m^{-1}\{J_1(\eta_1)\eta_3 + H(\eta_1, \eta_2)\} + J_m^{-1}F
\]

Substituting \(x_1, x_2\) and \(x_3\) of the equation (5.8) into \(\eta_4\) of the equation (5.6), we have

\[
\eta_4 := \Lambda(\eta_1, \eta_2, \Gamma(\eta_1, \eta_2, \eta_3)) + J_1(\eta_1)^{-1}kx_4
\]

Differentiating the equation (5.10),

\[
\dot{\eta}_4 = \left(\frac{\partial \Lambda}{\partial \eta_1} + \frac{\partial \Lambda}{\partial \Gamma} \frac{\partial \Gamma}{\partial \eta_1}\right)\eta_2 + \left(\frac{\partial \Lambda}{\partial \eta_2} + \frac{\partial \Lambda}{\partial \Gamma} \frac{\partial \Gamma}{\partial \eta_2}\right)\eta_3 \\
+ \left(\frac{\partial \Lambda}{\partial \Gamma} \frac{\partial \Gamma}{\partial \eta_3}\right)\eta_4 + J_1(\eta_1)^{-1}k\dot{x}_4
\]
Substituting (5.9) into the equation (5.11)

\[
\dot{\eta}_4 = \left( \frac{\partial \Lambda}{\partial \eta_1} + \frac{\partial \Lambda}{\partial \Gamma} \frac{\partial \Gamma}{\partial \eta_1} \right) \eta_2 + \left( \frac{\partial \Lambda}{\partial \eta_2} + \frac{\partial \Lambda}{\partial \Gamma} \frac{\partial \Gamma}{\partial \eta_2} \right) \eta_3 \\
+ \left( \frac{\partial \Lambda}{\partial \Gamma} \right) \eta_4 \\
- \mathbf{J}_i(\eta_1)^{-1} \mathbf{J}_m^{-1} \mathbf{k} \left[ \mathbf{J}_i(\eta_1) \eta_3 + \mathbf{H}(\eta_1, \eta_2) - \mathbf{F} \right]
\]

(5.12)

Utilising (5.2) - (5.4), the equation (5.12) becomes

\[
\dot{\eta}_4 = \left( \frac{\partial \Lambda}{\partial \eta_1} + \frac{\partial \Lambda}{\partial \Gamma} \frac{\partial \Gamma}{\partial \eta_1} \right) \eta_2 + \left( \frac{\partial \Lambda}{\partial \eta_2} + \frac{\partial \Lambda}{\partial \Gamma} \frac{\partial \Gamma}{\partial \eta_2} \right) \eta_3 \\
+ \left( \frac{\partial \Lambda}{\partial \Gamma} \right) \eta_4 \\
- \mathbf{J}_i(\eta_1)^{-1} \mathbf{J}_m^{-1} \mathbf{k} \left\{ \mathbf{J}_i(\eta_1) \eta_3 + \mathbf{C}(\eta_1, \eta_2) \eta_2 + \mathbf{C}(\eta_1) \right\} \\
- \mathbf{J}_i(\eta_1)^{-1} \mathbf{J}_m^{-1} \mathbf{k} \left[ \mathbf{\Delta}_h + \mathbf{\Delta}_f \right] \\
+ \mathbf{J}_i(\eta_1)^{-1} \mathbf{J}_m^{-1} \mathbf{k} \mathbf{u}
\]

(5.13)

An input-independent state coordinate transformation to obtain a GCCF in the form of (2.5) for the flexible joint manipulator system is thus

\[
\begin{align*}
\dot{\eta}_1 &= \eta_2 \\
\dot{\eta}_2 &= \eta_3 \\
\dot{\eta}_3 &= \eta_4 \\
\dot{\eta}_4 &= \varphi(\eta, \mathbf{\dot{u}}, t) + \mathbf{\Delta}(\eta, \omega, t) \\
&= \mathbf{p}(\eta) + \mathbf{q}(\eta) \mathbf{u} + \mathbf{\Delta}(\eta, \omega, t) \\
\mathbf{y} &= \eta_1
\end{align*}
\]

(5.14)

with

\[
\mathbf{p}(\eta) = \left( \frac{\partial \Lambda}{\partial \eta_1} + \frac{\partial \Lambda}{\partial \Gamma} \frac{\partial \Gamma}{\partial \eta_1} \right) \eta_2 + \left( \frac{\partial \Lambda}{\partial \eta_2} + \frac{\partial \Lambda}{\partial \Gamma} \frac{\partial \Gamma}{\partial \eta_2} \right) \eta_3
\]
Chapter 5. Robust Tracking via Sliding Mode Control for Flexible Joint Robots

\[ \begin{align*}
    + \left( \frac{\partial \Delta}{\partial \Gamma} \right) \eta_4 - J_I(\eta_1)^{-1}J_m^{-1}k \\
    &= \left[ J_I(\eta_1) \eta_3 + \dot{C}(\eta_1, \eta_2) \eta_2 + \dot{G}(\eta_1) \right] \\
    \mathbf{q}(\eta) &= J_I(\eta_1)^{-1}J_m^{-1}k \\
    \dot{\Delta}(\eta, \omega, t) &= -\mathbf{q}(\eta)\Delta(\eta, \omega, t) \\
    \Delta(\eta, \omega, t) &= \Delta_f(\eta, \omega, t) + \Delta_h(\eta, \omega, t)
\end{align*} \]

where \((\eta_1, \eta_2, \eta_3, \eta_4) = (y, y'(1), y'(2), y'(3), y'(4))\).

### 5.3 Robust Output Tracking Control Scheme

Consider the uncertain system (5.14) - (5.15) satisfying the conditions of the uncertainty bounding (2.2). Let \(y_R(t)\) be the desired tracking signal and the output tracking error be

\[ \hat{e}(t) = \hat{y} - y_R(t) \] (5.16)

with

\[ \hat{y} = (y_1, y_1^{(1)}, y_1^{(2)}, y_1^{(3)}, \ldots, y_m, y_m^{(1)}, y_m^{(2)}, y_m^{(3)}) \] (5.17)
\[ \hat{y}_R = (y_R, y_R^{(1)}, y_R^{(2)}, y_R^{(3)}, \ldots, y_R^{(1)}, y_R^{(2)}, y_R^{(3)}) \]
\[ \hat{e}(t) = [e_1, e_1^{(1)}, e_1^{(2)}, e_1^{(3)}, \ldots, e_m, e_m^{(1)}, e_m^{(2)}, e_m^{(3)}]^T \]

The uncertain I-O error dynamics of the flexible joint manipulator system are given by

\[ e^{(4)} = -y_R^{(4)} + \varphi(\hat{y}_R + \hat{e}, \hat{u}, t) + \dot{\Delta}(\hat{y}_R + \hat{e}, \omega, t) \] (5.18)

with

\[ \varphi(\hat{y}_R + \hat{e}, \hat{u}, t) = p(\hat{y}_R + \hat{e}) + q(\hat{y}_R + \hat{e})u \\
\dot{\Delta} = -q(\Delta_f + \Delta_h) \] (5.19)
where

\[ e^{(4)} = [e_1^{(4)}, \ldots, e_m^{(4)}]^T, \quad y_R^{(4)} = [y_{R1}^{(4)}, \ldots, y_{Rm}^{(4)}], \]

\[ \phi = [\phi_1, \ldots, \phi_m], \quad \Delta = [\Delta_1, \ldots, \Delta_m]. \]

The uncertainty can be bounded as

\[
\tilde{\Delta} = -J_i^{-1}J_m^{-1}\kappa(\Delta_f + \Delta_h)
\]

\[
\|\tilde{\Delta}\| = \|J_i^{-1}J_m^{-1}\kappa(\Delta_f + \Delta_h)\|
\leq \|J_i^{-1}\|\|J_m^{-1}\|\|\kappa\|\|\Delta_f + \Delta_h\|
\leq \Xi(\|\Delta_f\| + \|\Delta_h\|)
\]

where \(\|J_i^{-1}\|\|J_m^{-1}\|\|\kappa\| \leq \Xi\) is the overall upper bound for the inertia and joint stiffness matrices. In here, bounding matrices and vectors may increase the conservatism in robust output tracking control design [63]. To alleviate this problem, utilising Assumption 5.1 introduce the

\[
|\Delta_{h1}| = |[\Delta_c y^{(1)}_i]| + |[\Delta_g]_i| + |[\Delta_d]_i| \\
\leq \xi_{c1}(y)\|y^{(1)}\| + \xi_{g1}(y) + \xi_{d1} \quad \forall y \in \mathbb{R}^m \\
|\Delta_{f1}| \leq \xi_{f1}(y, y^{(1)}) \quad i = 1, \ldots, m.
\]

where \([\Delta_c y^{(1)}_i]\) denotes the ith element in the vector \(\Delta_c y^{(1)}\). Letting \(\|M\| \leq \|J_i^{-1}\|\|J_m^{-1}\|\|\kappa\|\) and \(\|My_i\| \leq \Xi_i \|y\|\) with \(y \in \mathbb{R}^m\) represent the ith element in the vector \(My\). The uncertainty (5.20) may be written in output coordinates as

\[
|\tilde{\Delta}_i| \leq \Xi_i \{\xi_{f1}(y, y^{(1)}) + \xi_{c1}(y)\|y^{(1)}\| + \xi_{g1}(y) + \xi_{d1}\}
\leq \rho_i \|\tilde{y}\| + l_i
\]
The constants $\rho_i$ and $l_i$ in (5.20) are independent of $u \in \mathbb{R}^m$. For given bounds, $u$ can be chosen to guarantee the stability of (5.14) as long as $\tilde{\Delta}$ remains bounded by the worst case, where $\xi_{cl} \geq 0$ is a small positive constant, $\xi_{ci}, \xi_{ci}, \xi_{ci}$ are known, positive definite functions.

The choice of robust tracking control law is apparent from the normal GCCF form (5.14) - (5.15) with $\tilde{\Delta} \equiv 0$ and $\nu = \varphi$. Its inverse is defined and the control law is given by

$$u = q^{-1} \{-p + \nu\} \quad (5.21)$$

Consider the direct sliding surface $s$, defined in terms of the output tracking error coordinates $\hat{e}$ (5.16) as

$$s = e^{(3)} + \Lambda \hat{e} \quad (5.22)$$

where $\hat{e} = [e_1^{(1)}, e_1^{(2)}, \ldots, e_m^{(1)}, e_m^{(2)}]^T$ and $\Lambda = \text{diag}[A_1, \ldots, A_m]$ with $A_i$ the companion matrix of the Hurwitz polynomial $\sum_{j=1}^{3} a_j^{(i)} \lambda^{j-1}$, where $i = 1, \ldots, m$.

For robust controller design, a strong continuous sliding reachability condition which was defined in equation (2.13) as:

$$\dot{s} = -\gamma(k, s) = -Ks - K_0 \text{sat}_e(s) \quad (5.23)$$

where $\text{sat}_e(s) = [\text{sat}_e(s_1), \ldots, \text{sat}_e(s_m)]^T$, $s = [s_1, \ldots, s_m]$, $K_0 = [k_{01}, \ldots, k_{0m}]$, $K = \text{diag}[K_{fu}] \in \mathbb{R}^{4 \times m}$ is a positive definite matrix that form a set of design parameters, and $\gamma(k, s) = [\gamma_1(k, s), \ldots, \gamma_m(k, s)]$ satisfies the Definition 2.2.

Differentiating the sliding surface (5.22) along the trajectories of the uncertain GCCF (5.18), it follows that

$$\dot{s} = e^{(4)} + \Lambda \bar{e} \quad (5.24)$$

where $\bar{e} = [e_1^{(1)}, e_1^{(2)}, e_1^{(3)}, \ldots, e_m^{(1)}, e_m^{(2)}, e_m^{(3)}]^T$, $e^{(4)} = [e_1^{(4)}, \ldots, e_m^{(4)}]^T$ and we have
\[ \dot{s} = \varphi(\dot{y}_R + \dot{e}, \dot{u}, t) + \Delta(\dot{y}_R + \dot{e}, \omega, t) \]
\[ + A\dot{e} - y^{(4)}_R(t) \]  
(5.25)

with

\[ \|\Delta\| \leq |\Delta_i| \]
\[ \leq \rho_i\|\dot{y}_R + \dot{e}\| + l_i \]  
(5.26)

To estimate the uncertainty bound as in (5.26) for the flexible joint manipulator system (5.18) - (5.19) define

\[ \|\Delta(\dot{y}_R + \dot{e}, \omega, t)\| \leq \rho^{(0)}\|\dot{y}_R + \dot{e}\| + I^{(0)} \]

Let \( \rho_i \) and \( l_i \) denote the upper bound on the lumped uncertainty that is given by (5.20), choose parameters \( 0 < \theta < 1, \theta_0 + \theta = 1, \rho^{(0)} = \sqrt{\rho_1^2 + \rho_2^2 + \cdots + \rho_m^2} \) and calculate the parameter \( \rho \) as follows

\[ \rho^{(1)} = \rho^{(0)} \left(1 + \max_i \{|A_i|\}\sqrt{3}\right) \]
\[ \rho = \rho^{(0)} + \frac{(\rho^{(1)})^2}{4\theta}. \]  
(5.27)

Use the strong continuous sliding reachability condition (5.23) and the differentiation of the sliding surface (5.25) to set

\[ \varphi(\dot{y}_R + \dot{e}, \dot{u}, t) - y^{(4)}_R(t) + A\dot{e}(t) = -Ks - K_0sat_\epsilon(s) \]  
(5.28)

Equation (5.25) becomes

\[ \dot{s} = -Ks - K_0sat_\epsilon(s) + \Delta(\dot{y}_R + \dot{e}, \omega, t) \]  
(5.29)
The robust tracking controller corresponding to the tracking system (5.18) can be determined by

$$\varphi(y_R + \hat{e}, \hat{u}, t) = y_R^{(4)}(t) - Ks - K_0 \text{sat}_\epsilon(s) - A\hat{e}(t)$$  \hfill (5.30)

Suppose \( s^T\gamma_0(k, s) \geq s^T Ks \) and \( K \) satisfies

$$\lambda_{\text{min}}(K)I_m - \left[ \frac{1}{\theta_0} [BD]^T BD + \rho I_m \right] > 0$$  \hfill (5.31)

where \( D = \text{diag}[D_1, \ldots, D_m]^T \) with \( D_i = \text{diag}[0, \ldots, 0, 1]^T \) for \( i = 1, \ldots, m \). \( A \) and \( B \) satisfy the Lyapunov equation

$$A^TB + BA = -D$$  \hfill (5.32)

and \( K_0 \) satisfies

$$K_0 > l^{(0)}I_m$$  \hfill (5.33)

where \( l^{(0)} = \sqrt{l_1^2 + l_2^2 + \cdots + l_m^2} \). Then the stability of the uncertain system is ensured as proved in Section 2.4.

**Theorem 5.1** Assume the flexible manipulator system (5.1) is minimum phase and the control law is chosen according to (5.21) with \( \nu = [\nu_1, \ldots, \nu_m] = \varphi \). For arbitrarily given but bounded \( \epsilon > 0 \), there exists a robust tracking control law

$$u = q^{-1} \left\{ y_R^{(4)} - p - Ks - K_0 \text{sat}_\epsilon(s) - A\hat{e}(t) \right\}$$  \hfill (5.34)

with

$$\lambda_{\text{min}}(K)I_m > \left[ \frac{1}{\theta_0} [BD]^T BD + \rho^{(0)} I_m \right]$$
which yields a tracking error system which is ultimately bounded by $\epsilon$.

## 5.4 Application to a Two-Link Flexible Joint Manipulator

The dynamical equations governing the behaviour of two-link revolute joints with elastic gears coupling DC-motor actuators and rigid links with inertia $J_i$ about the rotation axis are considered \([61, 85, 90, 70]\). In Section 5.2, we investigate the uncertain GCCF for an $n$-link manipulator with elastic joints in the dynamical equations (5.13) - (5.20). In this section, we use the uncertain dynamical model in Section 5.2 for the two-link flexible joint robot. The equations of motion are

\[
J_i(q_i)\ddot{q}_i + H(q_i, \dot{q}_i) + k(q_1 - q_2) = 0
\]

\[
J_m\ddot{q}_2 - k(q_1 - q_2) = F
\]  

\[
y = q_1
\]

\[
J_i(q_i)\ddot{q}_i + H(q_i, \dot{q}_i) = 0
\]

\[
J_m\ddot{q}_2 - k(q_1 - q_2) = F
\]  

\[
y = q_1
\]

with

\[
H(q_i, \dot{q}_i) = MgL\sin(q_i) + \Delta_h
\]

\[
F = u - \Delta_f(q_i, \dot{q}_i)
\]  

\[
\Delta_h = MgLx_1 + \Delta_d
\]

Let $r_{i_1} = J_{i_1}^{-1}; r_{i_2} = J_{m_1}^{-1}; r_{i_3} = (MgL)_i; \hat{r}_{i_3} = (\hat{MgL})_i$ with $i = 1, 2$. Let $x_1$ denote the angular position of the link and $x_3$ be the motor shaft

\[
x_1 = q_1 = \begin{bmatrix} q_{11} \\ q_{12} \end{bmatrix} \quad x_2 = \dot{q}_1 = \begin{bmatrix} \dot{q}_{11} \\ \dot{q}_{12} \end{bmatrix}
\]
Chapter 5. Robust Tracking via Sliding Mode Control for Flexible Joint Robots

\[ x_3 = q_2 = \begin{bmatrix} q_{21} \\ q_{22} \end{bmatrix} \quad x_4 = \dot{q}_2 = \begin{bmatrix} \dot{q}_{21} \\ \dot{q}_{22} \end{bmatrix} \]  

(5.38)

Utilising (5.37), the GCCF (5.14) - (5.15) for the two-link flexible joint manipulator system is obtained as

\[ \eta_1^{(i)} = \eta_2^{(i)} \]
\[ \eta_2^{(i)} = \eta_3^{(i)} \]
\[ \eta_3^{(i)} = \eta_4^{(i)} \]
\[ \eta_4^{(i)} = \varphi_i(\eta, \dot{\eta}, t) + \tilde{\Delta}_i(\eta, \omega, t) = p_i(\eta) + q_i(\eta)u_i + \Delta_i(\eta, \omega, t) \]
\[ y_1^{(i)} = \eta_i^{(i)} \]

with

\[ p_i(\eta) = r_{i1}r_{i3}\left[\left((\eta_2^{(i)})^2 - r_{i2}k_i\right)\sin\eta_1^{(i)} - \eta_3^{(i)}\cos\eta_1^{(i)}\right] - k_i(r_{i1} + r_{i2})\eta_3^{(i)} \]
\[ q_i(\eta) = r_{i1}r_{i2}k_i \]
\[ \tilde{\Delta}_i(\eta, \omega, t) = -r_{i1}r_{i2}k_i(\Delta_{fi} + \tilde{\Delta}_{fi}) + \frac{\dot{r}_{i3}}{r_{i2}k_i}\eta_3^{(i)} + \Delta_{di} + \tilde{\Delta}_{di} \]
\[ = -q_i(\eta)[\Delta_{fi} + \Delta_{hi}] \quad i = 1, 2. \]

Since \( \Delta_{di} \) is a constant, \( \tilde{\Delta}_{di} \) is zero. The Regularity condition (2.4) for the flexible joint manipulator system (5.39) yields

\[ \det\left[\frac{\partial \varphi_i}{\partial u_i}\right] = r_{i1}r_{i2}k_i \neq 0 \]  

(5.41)

The input-output representation of the manipulator system with \( \Delta_i = 0 \) is

\[ y_1^{(4)} + k_1(r_{11} + r_{12})y_1^{(2)} - r_{11}r_{13}[(y_1^{(1)})^2 - r_{12}k_1]\sin y_1 - y_1^{(2)}\cos y_1] - r_{11}r_{12}k_1u_1 = 0 \]
\[ y_2^{(4)} + k_2(r_{21} + r_{22})y_2^{(2)} - r_{21}r_{23}[(y_2^{(1)})^2 - r_{22}k_2]\sin y_2 - y_2^{(2)}\cos y_2] - r_{21}r_{22}k_3u_2 = 0 \]
5.4.1 Output Tracking of Uncertain System via Feedback Linearisation (FLC)

In this subsection, we consider the use of the uncertain dynamical model (5.39) - (5.40) with a feedback linearisation controller (FLC).

For manipulator trajectory tracking control, define a tracking error function $\hat{e}(t)$ as the difference between the actual system output $\dot{y}$ and the desired reference trajectory $\dot{y}_R(t)$. The system is expressed in terms of error coordinates

$$\hat{e}(t) = \dot{y} - \dot{y}_R(t) \tag{5.42}$$

with

$$\hat{e}(t) = \begin{bmatrix} y - y\text{,}R\text{,}y^{(1)} - y_R^{(1)}, y^{(2)} - y_R^{(2)}, y^{(3)} - y_R^{(3)} \end{bmatrix}^T \tag{5.43}$$

Based on the results of Chapter 2, the system of differential equations describing the tracking error dynamics is

$$\begin{align*}
\dot{e}^{(i)}_1 &= e_2^{(i)} \\
\dot{e}^{(i)}_2 &= e_3^{(i)} \\
\dot{e}^{(i)}_3 &= e_4^{(i)} \\
\dot{e}^{(i)}_4 &= -y_R^{(4)} + \varphi_i(\dot{y}, \dot{u}, t) + \Delta_i(\dot{y}, \omega, t) \\
y^{(1)}_i &= e_1^{(i)} + y_R^{(1)}
\end{align*} \tag{5.44}$$

with

$$\varphi_i(\dot{y}_R + e, \dot{u}, t) = r_{i1}r_{i3}[(e_2^{(i)} + y_R^{(1)})^2 - r_{i2}k_i]sin(e_1^{(i)} + y_R)$$

$$- (e_3^{(i)} + y_R^{(2)})cos(e_1^{(i)} + y_R)$$

$$- k_i(r_{i1} + r_{i2})(e_3^{(i)} + y_R^{(2)}) + r_{i1}r_{i2}k_i u_i \tag{5.45}$$
Chapter 5. Robust Tracking via Sliding Mode Control for Flexible Joint Robots

The tracking error dynamics for the two-link flexible joints manipulator is

\[
\begin{align*}
\varepsilon_4^{(1)} &= -y_{R1}^{(4)} + \varphi_1(\dot{y}, \ddot{u}, t) + \Delta_1(\dot{y}, \omega, t) \\
&= r_{11}r_{13} \left\{ \left( (e_2^{(1)} + y_{R1}^{(1)})^2 - r_{12}k_1 \right) \sin \left( e_1^{(1)} + y_{R1} \right) - \left( e_3^{(1)} + y_{R1}^{(2)} \right) \cos \left( e_1^{(1)} + y_{R1} \right) \right\} \\
&\quad - k_1 (r_{11} + r_{12}) \left( e_3^{(1)} + y_{R1}^{(2)} \right) + r_{11}r_{12}k_1u_1 - r_{11}r_{12}k_1 (\Delta f_1 + \Delta h_1) \\
\varepsilon_4^{(2)} &= -y_{R2}^{(4)} + \varphi_2(\dot{y}, \ddot{u}, t) + \Delta_2(\dot{y}, \omega, t) \\
&= r_{21}r_{23} \left\{ \left( (e_2^{(2)} + y_{R2}^{(1)})^2 - r_{22}k_2 \right) \sin \left( e_1^{(2)} + y_{R2} \right) - \left( e_3^{(2)} + y_{R2}^{(2)} \right) \cos \left( e_1^{(2)} + y_{R2} \right) \right\} \\
&\quad - k_2 (r_{21} + r_{22}) \left( e_3^{(2)} + y_{R2}^{(2)} \right) + r_{21}r_{22}k_2u_2 - r_{21}r_{22}k_2 (\Delta f_2 + \Delta h_2)
\end{align*}
\]

with

\[
\begin{align*}
\Delta h_1 &= \dot{r}_{13}\eta_1^{(1)} + \frac{\dot{r}_{13}}{r_{12}k_1}\eta_3^{(1)} + \Delta d_1 \\
\Delta h_2 &= \dot{r}_{23}\eta_1^{(2)} + \frac{\dot{r}_{23}}{r_{22}k_2}\eta_3^{(2)} + \Delta d_2
\end{align*}
\]

Linearisation of the tracking error dynamics (5.44) is equivalent to having the closed loop error dynamics obey \( e_4^{(i)} = v_i \):

\[
\begin{align*}
\varepsilon_4^{(1)} &= v_1 = -a_{14}e_4^{(1)} - a_{13}e_3^{(1)} - a_{12}e_2^{(1)} - a_{11}e_1^{(1)} \\
\varepsilon_4^{(2)} &= v_2 = -a_{24}e_4^{(2)} - a_{23}e_3^{(2)} - a_{22}e_2^{(2)} - a_{21}e_1^{(2)}
\end{align*}
\]

From (5.44) and (5.46), the feedback linearising control law is simply:

\[
\begin{align*}
u_1 &= \frac{1}{r_{11}r_{12}k_1} \left\{ y_{R1}^{(4)} - r_{11}r_{13} \left\{ \left( (e_2^{(1)} + y_{R1}^{(1)})^2 - r_{12}k_1 \right) \sin \left( e_1^{(1)} + y_{R1} \right) - \left( e_3^{(1)} + y_{R1}^{(2)} \right) \cos \left( e_1^{(1)} + y_{R1} \right) \right\} \right. \\
&\quad + \frac{1}{r_{11}r_{12}k_1} \left( -a_{14}e_4^{(1)} - a_{13}e_3^{(1)} - a_{12}e_2^{(1)} - a_{11}e_1^{(1)} \right) \bigg\}
\end{align*}
\]
Chapter 5. Robust Tracking via Sliding Mode Control for Flexible Joint Robots

\[ u_2 = \frac{1}{r_21r_22k_2} \left\{ y_{r2}^{(4)} - r_21r_23 \left\{ \left( (e_1^{(2)} + y_{r2}^{(2)})^2 - r_22k_2 \right) \sin \left( e_1^{(2)} + y_{r2}^{(2)} \right) \right. \right. \]
\[ - \left( (e_3^{(2)} + y_{r2}^{(2)}) \cos \left( e_1^{(2)} + y_{r2}^{(2)} \right) \right) + k_2 \left( r_21 + r_22 \right) \left( e_3^{(2)} + y_{r2}^{(2)} \right) \]
\[ + \frac{1}{r_21r_22k_2} \left( -a_{24}e_4^{(2)} - a_{23}e_3^{(2)} - a_{22}e_2^{(2)} - a_{21}e_1^{(2)} \right) \} \]

(5.47)

\begin{align*}
r_1 &= 1.0(m^2Kg)^{-1} & r_3 &= 9.8N.m \\
r_2 &= 1.0(m^2Kg)^{-1} & k &= 100N/m rad^{-1}
\end{align*}

(5.48)

Simulation Results for FLC

Simulations for a two-link with the flexible manipulator with the following parameters are presented:

\[
\begin{align*}
r_1 &= 1.0(m^2Kg)^{-1} & r_3 &= 9.8N.m \\
r_2 &= 1.0(m^2Kg)^{-1} & k &= 100N/m rad^{-1}
\end{align*}
\]

(5.48)

The parameters corresponding to the feedback controller were chosen as

\[
a_1 = 512; \quad a_2 = 448; \quad a_3 = 144; \quad a_4 = 20;
\]

(5.49)

with closed loop poles assigned to \{-4 -4 -4 -8\}. The system is required to track a desired angular link position of the form

\[
y_R = \left[ \begin{array}{c} A\cos((2\pi/5)t) \\ A\cos((\pi - \pi/2)t) \end{array} \right]
\]

(5.50)

Having obtained \(a_1, a_2, a_3, a_4\) and the controller parameters, the closed-loop control system is simulated with the feedback controller (5.47). Increasing the magnitude of \(A\) will increase the difficulty in control. All initial conditions for the system and controller were chosen to be zero.

**FLC for the nominal system:** When \(A = 0.0873\), the actual link angular positions for trajectory tracking with feedback linearising controller are shown in Figure 5.1-a. The errors between the desired and actual link angular positions are shown in Figure 5.1-b. The tracking errors is remarkably close to the desired trajectories within two seconds.
The desired link angular trajectories and the feedback linearising input torques are shown in Figure 5.1-c and 5.1-d. When \( A = 1.0 \) in the desired trajectories (5.50), simulation results are shown in Figure 5.2. Respectively, increasing the magnitude of \( A \) shows that the errors have been increased under FLC in Figure 5.3. The nominal performance of the FLC is good.

**FLC for the uncertain system:** The system is simulated with the controller (5.47). The random disturbance is shown in Figure 5.4. When \( A = 0.0873 \), the actual link angular positions are shown in Figure 5.5-a. The errors are shown in Figure 5.5-b. The actual positions are close to the desired trajectories within two seconds but the errors are bounded between \(+0.009 \sim -0.008\). The system uncertainties and the feedback linearising input torques are shown in Figure 5.5-c and 5.5-d. When \( A = 1.0 \), simulation results are shown in Figure 5.6. Increasing the magnitude of \( A \) increases the errors under FLC in Figure 5.7.

The simulation results (Figures 5.5 - 5.7) for the FLC with large uncertain forces and disturbances do not show the good results.

### 5.4.2 Output Tracking via Robust Sliding Mode Control (RSMC)

A robust sliding mode controller has been synthesised in Section 5.3). In this section we apply the RSMC to the two-link flexible joint robot that in modelled by (5.36) - (5.37). The GCCF of the flexible joint robot with uncertainties is given by (5.39) - (5.40). Defining the tracking error function as (5.42) - (5.43) and describing the tracking error dynamics as

\[
\begin{align*}
\mathbf{e}_4^{(1)} & = -\mathbf{y}_{R1} + \varphi_1(\hat{y}, \dot{u}, t) + \tilde{A}_2(\hat{y}, \omega, t) \\
& = r_{11} r_{13} \left\{ \left( (e_2^{(1)} + y_{R1})^2 - r_{12} k_1 \right) \sin (e_1^{(1)} + y_{R1}) - \left( e_3^{(1)} + y_{R1} \right) \cos (e_1^{(1)} + y_{R1}) \right\} \\
& - k_1 \left( r_{11} + r_{12} \right) \left( e_3^{(1)} + y_{R1} \right) + r_{11} r_{12} k_1 u_{1} - r_{11} r_{12} k_1 (\Delta \gamma_1 + \Delta h_1) \\
& \vdots \\
\mathbf{e}_4^{(2)} & = -\mathbf{y}_{R2} + \varphi_2(\hat{y}, \dot{u}, t) + \tilde{A}_2(\hat{y}, \omega, t) \\
& = r_{21} r_{23} \left\{ \left( (e_2^{(2)} + y_{R2})^2 - r_{22} k_2 \right) \sin (e_1^{(2)} + y_{R2}) - \left( e_3^{(2)} + y_{R2} \right) \cos (e_1^{(2)} + y_{R2}) \right\}
\end{align*}
\]
\[-k_2 (r_{21} + r_{22}) \left( e_3^{(2)} + y_{R2}^{(2)} \right) + r_{21} r_{22} k_2 u_2 - r_{21} r_{22} k_2 (\Delta f_2 + \Delta h_2)\]

with

\[
\begin{align*}
\tilde{\Delta}_1(y, \omega, t) &= -r_{11} r_{12} k_1 (\Delta f_1 + \Delta h_1) = - \left( \Delta h_1 + \dot{r}_{13} \eta_1^{(1)} + \frac{\dot{r}_{13}}{r_{12} k_1} \eta_3^{(1)} + \Delta d_1 \right) \\
\tilde{\Delta}_2(y, \omega, t) &= -r_{21} r_{22} k_2 (\Delta f_2 + \Delta h_2) = - \left( \Delta h_2 + \dot{r}_{23} \eta_1^{(2)} + \frac{\dot{r}_{23}}{r_{22} k_2} \eta_3^{(2)} + \Delta d_2 \right)
\end{align*}
\]

According to Assumption 3.1 and equation (5.20), considering the region \(-\frac{1}{2} \leq \frac{\Delta y^2}{2} \leq \frac{1}{2}\), the estimate of a worst case bound on the function \(\tilde{\Delta}_i\), is given by

\[
\begin{align*}
\|\tilde{\Delta}_1\| &\leq k_1 \dot{J}_{11}^{-1} \dot{J}_{m1}^{-1} \{\xi_{f1} + \xi_{c11} \|y_1\| + \xi_{c21} + \xi_{d1}\} \\
&\leq k_1 \dot{J}_{11}^{-1} \dot{J}_{m1}^{-1} \xi_{11} \|y_1\| + k_1 \dot{J}_{11}^{-1} \dot{J}_{m1}^{-1} \xi_{12} \\
\|\tilde{\Delta}_2\| &\leq k_2 \dot{J}_{12}^{-1} \dot{J}_{m2}^{-1} \{\xi_{f2} + \xi_{c12} \|y_2\| + \xi_{c22} + \xi_{d2}\} \\
&\leq k_2 \dot{J}_{12}^{-1} \dot{J}_{m2}^{-1} \xi_{21} \|y_2\| + k_2 \dot{J}_{12}^{-1} \dot{J}_{m2}^{-1} \xi_{22}
\end{align*}
\]

where

\[
\begin{align*}
\rho_1 &= k_1 \dot{J}_{11}^{-1} \dot{J}_{m1}^{-1} \xi_{11} \\
\rho_2 &= k_2 \dot{J}_{12}^{-1} \dot{J}_{m2}^{-1} \xi_{21} \\
l_1 &= k_1 \dot{J}_{11}^{-1} \dot{J}_{m1}^{-1} \xi_{12} \\
l_2 &= k_2 \dot{J}_{12}^{-1} \dot{J}_{m2}^{-1} \xi_{22}
\end{align*}
\]

According to Theorem 5.1 in Section 5.3, the robust tracking controller is given by

\[
u_1 = \frac{1}{r_{11} r_{12} k_1} \left\{ y_{R1}^{(2)} - r_{11} r_{13} \left\{ \left( e_2^{(1)} + y_{R1}^{(1)} \right)^2 - r_{12} k_1 \right\} \sin (e_1^{(1)} + y_{R1}) \\
- \left( e_3^{(1)} + y_{R1}^{(2)} \right) \cos (e_1^{(1)} + y_{R1}) \right\} + k_1 (r_{11} + r_{12}) (e_3^{(1)} + y_{R1}) \\
+ \frac{1}{r_{11} r_{12} k_1} \left( -K_{11}s_1 - K_{12}s_2 - K_{01} sat_e(s_1) - m_{13} e_3^{(1)} - m_{12} e_2^{(1)} - m_{11} e_1^{(1)} \right) \right\}
\]

\[(5.51)\]
\[
\begin{align*}
    u_2 &= \frac{1}{r_2^2 k_2} \left\{ y_{R2}^{(4)} - r_2 r_23 \left\{ \left( e_2^{(2)} + y_{R2}^{(2)} \right)^2 - r_2 k_2 \right\} \sin (e_1^{(2)} + y_{R2}^{(2)}) \\
    &\quad - \left( e_3^{(2)} + y_{R2}^{(2)} \right) \cos (e_1^{(2)} + y_{R2}^{(2)}) \right\} + k_2 \left( r_21 + r_22 \right) \left( e_3^{(2)} + y_{R2}^{(2)} \right) \\
    &\quad + \frac{1}{r_2^2 k_2} \left( -K_{21} \delta_1 - K_{22} \delta_2 - K_{02} \delta_3(s_2) - m_{23} \dot{e}_3^{(2)} - m_{22} \dot{e}_2^{(2)} - m_{21} \dot{e}_1^{(2)} \right) \right\} \\
\end{align*}
\]

where \( K \) satisfies

\[
\lambda_{\text{min}}(K) I_2 > \left[ \frac{1}{\theta_0} [BD]^T [BD] + \rho I_2 \right] 
\]

where \( D = \text{diag} [D_1, D_2] \) with \( D_i = [0, 0, 1]^T \); \( A = \text{diag} [A_1, A_2] \) is the companion matrix of the \( \sum_{j=1}^3 \lambda_j^{i-1} \) with \( i = 1, 2 \), \( B \) satisfies the Lyapunov equation (5.32) and \( K_0 \) satisfies

\[
K_0 > l(0) I_2
\]

where \( 0 < \theta < 1, \theta_0 + \theta = 1, \rho(0) = k \dot{J}_i^{-1} \dot{J}_m^{-1} \xi_1 \) and \( l(0) = k \dot{J}_i^{-1} \dot{J}_m^{-1} \xi_2 \);

\[
\rho^{(1)} = \rho(0) \left\{ 1 + \max_i \left[ \max (|A_i|) \right] \sqrt{3} \right\} \\
\rho = \rho(0) + \frac{\rho^{(1)} \tilde{z}}{4\theta} 
\]

**Simulation Results for RSMC**

Simulations were performed for a flexible joint manipulator with the same parameters previously described in Section 5.4.1. In this case the robust sliding mode controller, given in equation (5.51), was used. Suppose that the coefficients \( \dot{J}_i, \dot{J}_mi \) and \( MgL = \dot{r}_{3i} \) are unknown but bounded as

\[
0.8 \leq \dot{J}_i \leq 1.5; \quad 0.8 \leq \dot{J}_mi \leq 1.5; \quad 5 \leq MgL \leq 10; \quad i = 1, 2.
\]

Choose controller parameters as \( \rho_i = k_i \dot{J}_i^{-1} \dot{J}_mi^{-1} \xi_{1i} \) and \( l_i = k_i \dot{J}_i^{-1} \dot{J}_mi^{-1} \xi_{2i} \). The parameters in the model are set as in equation (5.48). Other dynamics can be bounded similarly as \( \xi_{1i} = 0.00012, \xi_f = \text{rand}(1) \) and \( \xi_{2i} = 0.0035 \). We have
\[ \rho_1 = k_1 \hat{J}_{m1}^{-1} \hat{J}_{m1}^{-1} \xi_{11} = 100 \times \frac{1}{0.8} \times \frac{1}{0.8} \times 0.0012 = 0.0187 \]
\[ \rho_2 = k_2 \hat{J}_{m2}^{-1} \hat{J}_{m2}^{-1} \xi_{21} = 100 \times \frac{1}{0.8} \times \frac{1}{0.8} \times 0.0012 = 0.0187 \]  \hspace{1cm} (5.55)
\[ l_1 = k_1 \hat{J}_{m1}^{-1} \hat{J}_{m1}^{-1} \xi_{12} = 100 \times \frac{1}{0.8} \times \frac{1}{0.8} \times 0.0035 = 0.54688 \]
\[ l_2 = k_2 \hat{J}_{m2}^{-1} \hat{J}_{m2}^{-1} \xi_{22} = 100 \times \frac{1}{0.8} \times \frac{1}{0.8} \times 0.0035 = 0.54688 \]

then

\[ \rho^{(0)} = \sqrt{\rho_1^2 + \rho_2^2} = \sqrt{0.0187^2 + 0.0187^2} = 0.0265 \]  \hspace{1cm} (5.56)
\[ l^{(0)} = \sqrt{l_1^2 + l_2^2} = \sqrt{0.54688^2 + 0.54688^2} = 0.77341 \]

\[ \| \Delta \| < 0.0265 \| y \| + 0.77341 \]  \hspace{1cm} (5.57)

The parameters corresponding to the continuous controller were chosen as

\[ m_1^{(i)} = 210; \quad m_2^{(i)} = 107; \quad m_3^{(i)} = 18; \quad i = 1, 2. \]

Corresponding to ideal sliding mode poles assigned to \{-7, -6, -5\}. Use \( \theta = \theta_0 = 0.5 \) and

\[
A = \begin{bmatrix}
A_1 & 0 \\
0 & A_2
\end{bmatrix}
= \begin{bmatrix}
0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 \\
-210 & -107 & -18 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & -210 & -107 & -18
\end{bmatrix}
\]

Use equation (5.54)

\[ \rho^{(1)} = \rho^{(0)} \left( 1 + \max_i \{ \max_j (|A_i|) \sqrt{3} \} \right) = 9.6714 \]
\[ \rho = \rho^{(0)} + \frac{\rho^{(1)}^2}{4 \theta} = 0.0265 + \frac{(9.6714)^2}{4 \times 0.5} = 46.7943 \]
Chapter 5. Robust Tracking via Sliding Mode Control for Flexible Joint Robots

\[ B = \text{diag}[B_1, B_2] \] is obtained from the Lyapunov equation with \( A = \text{diag}[A_1, A_2] \) and \( C = I \)

\[
B = \begin{bmatrix} B_1 & 0 \\ 0 & B_2 \end{bmatrix} = \begin{bmatrix}
0.3576 & -0.5000 & -1.2273 & 0 & 0 & 0 \\
-0.5000 & 1.2273 & -0.5000 & 0 & 0 & 0 \\
-1.2273 & -0.5000 & 17.3182 & 0 & 0 & 0 \\
0 & 0 & 0 & 0.3576 & -0.5000 & -1.2273 \\
0 & 0 & 0 & -0.5000 & 1.2273 & -0.5000 \\
0 & 0 & 0 & -1.2273 & -0.5000 & 17.3182
\end{bmatrix}
\]

Let

\[
D_1 = [0, 0, 1]^T \quad D_2 = [0, 0, 1]^T \quad D = \text{diag}[D_1, D_2]
\]

\[
\frac{1}{\theta_0} [BD]^T [BD] + \rho I_2 = 2 \times \begin{bmatrix} 301.6694 & 0 \\ 0 & 301.6694 \end{bmatrix} + 46.7943 I_2
\]

\[
= \begin{bmatrix} 348.46 & 0 \\ 0 & 348.46 \end{bmatrix}
\]

The control parameters of the \( RSMC \), \( K \) and \( K_0 \), must satisfy equations (5.52) and (5.53).

Choose

\[
K = \begin{bmatrix} 360 & 10 \\ 10 & 360 \end{bmatrix} \quad K_0 = \begin{bmatrix} 1.5 & 0 \\ 0 & 1.5 \end{bmatrix} \quad \epsilon = 0.1
\]

Substituting \( K \) and \( K_0 \) into equations (5.52) and (5.53) satisfy

\[
\lambda_{\min}(K) I_2 - \left( \frac{1}{\theta_0} [BD]^T [BD] + \rho I_2 \right) = \begin{bmatrix} 350 & 0 \\ 0 & 350 \end{bmatrix} - \begin{bmatrix} 348.46 & 0 \\ 0 & 348.46 \end{bmatrix} = \begin{bmatrix} 1.54 & 0 \\ 0 & 1.54 \end{bmatrix} > 0
\]

\[
K_0 - l(0) I_2 = \begin{bmatrix} 1.5 & 0 \\ 0 & 1.5 \end{bmatrix} - \begin{bmatrix} 0.77341 & 0 \\ 0 & 0.77341 \end{bmatrix} = \begin{bmatrix} .72659 & 0 \\ 0 & .72659 \end{bmatrix} > 0
\]
To reduce the effects of the uncertain function $\hat{A}$ on the tracking error $e$, a large gain is chosen in the robust tracking controller (5.51).

**RSMC for the nominal system:** When $A = 0.0873$, the actual angular positions are shown in Figure 5.8-a. The errors between the desired and actual angular positions are shown in Figure 5.8-b. The tracking errors are remarkably close to the desired trajectories within two seconds. The sliding surfaces and the dynamically generated input torques are shown in Figure 5.8-c and 5.8-d. When $A = 1.0$, simulation results are shown in Figure 5.9. Errors under $RSMC$ with increasing magnitude of $A$ are shown in Figure 5.10. The performance of the $RSMC$ without uncertainties shows excellent results.

**RSMC for the uncertain system:** When $A = 0.0873$, the actual angular positions are shown in Figure 5.11-a. The errors between the desired and actual angular positions are shown in Figure 5.11-b. The tracking errors are very close to the desired trajectories within two seconds and the error is bounded between $+0.002 \sim -0.002$. The sliding surfaces and the dynamically generated input torques are shown in Figure 5.11-c and 5.11-d. When $A = 1.0$, simulation results are shown in Figure 5.12. The errors under $RSMC$ with increasing magnitude of $A$ are shown in Figure 5.13.

The simulation results (Figures 5.11 - 5.13) for the uncertain system with a large uncertain forces and disturbances shows the excellent results.

Comparing Figures 5.7 with Figures 5.13 we see that sliding mode component reduces the tracking error caused by the parameter uncertainties, modelling errors and disturbance when compared with the feedback linearisation controller.

### 5.5 Concluding Remarks

In this chapter an n-link uncertain manipulator with elastic joints has been investigated. Based on robust tracking control via sliding modes, we give a robust output tracking control scheme for the elastic joint manipulators. The function $||\hat{A}||$ is a nonlinear function of the state, and bounds the maximum size of the overall uncertainties. The design of the robust tracking controller involves selection of $u$ and the bounding function $||\hat{A}||$ to achieve stability and good tracking performance. The relative performance is studied by
Figure 5.1: a. The actual trajectories with the FLC; b. The error between the desired and actual trajectories; c. The desired trajectories; d. The torques. $A = 0.0876$. No uncertainty.

Figure 5.2: a. The actual trajectories with the FLC; b. The error between the desired and actual trajectories; c. The desired trajectories; d. The torques. $A = 1.0$. No uncertainty.
Figure 5.3: a. The errors between the desired and actual trajectories with different $A$ using a FLC with the nominal system for the link one. b. The errors between the desired and actual trajectories with different $A$ using a FLC with the nominal system for the link two.

Figure 5.4: The uniformly distributed random disturbance.
Figure 5.5: a. The actual trajectories with the FLC; b. The error between the desired and actual trajectories; c. The desired trajectories; d. The torques. $A = 0.0876$. Uncertain system.

Figure 5.6: a. The actual trajectories with the FLC; b. The error between the desired and actual trajectories; c. The desired trajectories; d. The torques. $A = 1.0$. Uncertain system.
Figure 5.7: a. The errors between the desired and actual trajectories with different $A$ using the FLC with uncertain system for the link one. b. The errors between the desired and actual trajectories with different $A$ using the FLC with uncertain system for the link two.
Figure 5.8: a. The actual trajectories with the $RSMC$; b. The error between the desired and actual trajectories; c. The sliding surfaces; d. The torques. $A = 0.0873$. Nominal system.

Figure 5.9: a. The actual trajectories with the $RSMC$; b. The error between the desired and actual trajectories; c. The sliding surfaces; d. The torques. $A = 1.0$. Nominal system.
Figure 5.10: a. The errors between the desired and actual trajectories with different \( A \) using the SMC with nominal system for the link one. b. The errors between the desired and actual trajectories with different \( A \) using the SMC with nominal system for the link two.
Figure 5.11: a. The actual trajectories with the \textit{RSMC}; b. The error between the desired and actual trajectories; c. The sliding surfaces; d. The torques. $A = 0.0873$. Uncertain system.

Figure 5.12: a. The actual trajectories with the \textit{RSMC}; b. The error between the desired and actual trajectories; c. The sliding surfaces; d. The torques. $A = 1.0$. Uncertain system.
Figure 5.13: a. The errors between the desired and actual trajectories with different $A$ using the RSMC with uncertain system for the link one. b. The errors between the desired and actual trajectories with different $A$ using the RSMC with uncertain system for the link two.
simulation of a two-link flexible joint robot. In Section 5.4, FLC has been applied to the nominal and uncertain systems of elastic joints manipulator. The tracking errors have been increased by the uncertain system or the increasing the magnitude of $A$. RSMC also has been applied to the nominal and uncertain systems. The simulation results show excellent robust performance.
Chapter 6

Robust Tracking via Dynamical Sliding Mode Control Schemes for Modified Flexible Joint Manipulators

6.1 Introduction

Dynamical sliding mode control was introduced by Fliess and Messager [36, 37] in the context of linear dynamical systems. Static sliding mode control design is recognised as robust with respect to system uncertainties in both theoretical research and application [26, 65, 96]. Dynamic sliding mode control remedies the defects of high frequencies chattering [78, 72, 71, 50, 51]. Dynamic sliding mode control combines the advantage of dynamic feedback control and sliding mode techniques in the controller design phase while simultaneously asymptotically linearising the nonlinear system.

In Chapter 5 robust sliding mode schemes have been proposed that address the joint flexibility issue. Robust dynamical sliding mode control schemes for nonlinear uncertain systems have been investigated in Chapter 2. The uncertain system with appropriately chosen dynamical sliding mode control is shown to be ultimately bounded if the zero dynamics of the nominal system are uniformly asymptotically stable.

In this chapter, the robust output tracking problem of modified elastic joint manipulators is treated from the perspective of dynamical feedback linearisation and robust dynamical sliding mode control.
6.2 Modified Elastic Joint Manipulator Dynamic Model

The dynamical equations governing the behaviour of an n-link flexible joint manipulator with elastic gears coupling a DC-motor actuator and a rigid link with inertia $J_l$ about the rotation axis will be considered (Sira-Ramirez [70], Spurgeon [90] and Spong [85]). Let $q_1 \in \mathbb{R}^n$ denote the vector of angular positions of the link of half length and mass $m$ and $q_2 \in \mathbb{R}^n$ be the vector of angular positions of the motor. In Chapter 5, taking the link position as the output of the system $y = q_1$, the system is relative degree four and it is exactly linearisable by nonlinear static state feedback using link position, velocity, acceleration and jerk. These variables are usually very hard to measure in practice. In this chapter, by using motor position $q_2$ as the vector of system outputs, the system becomes relative degree two and the GCCF will involve second order time derivatives of the control input torque. It will be shown that a nonlinear second order dynamical system acts as the controller. The modified differential equations governing the controlled motion with uncertainties are given by

$$
\begin{align*}
J_l(q_1)\ddot{q}_1 + H(q_1, \dot{q}_1) + k(q_1 - q_2) &= 0 \\
J_m\ddot{q}_2 - k(q_1 - q_2) &= F \\
y &= q_2
\end{align*}
$$

(6.1)

where $J_l(q_1) \in \mathbb{R}^{n \times n}$ denotes the inertia matrix of the rigid robot, $J_m \in \mathbb{R}^{n \times n}$ denotes the inertia matrix of the motor and $k$ is a diagonal matrix representing flexible joint stiffness.

According to Section 5.2, the lumped sum of nonlinearities $H(q_1, \dot{q}_1)$ (5.2), the lumped sum of uncertainties $\Delta_h$ (5.3) and the joint forces $F$ (5.4) are given by

$$
\begin{align*}
H(q_1, \dot{q}_1) &= \dot{\mathcal{C}}(q_1, \dot{q}_1)\dot{q}_1 + \dot{\mathcal{G}}(q_1) + \Delta_h \\
\Delta_h &= \Delta_c(q_1, \dot{q}_1)\dot{q}_1 + \Delta_g(q_1) + \Delta_d \\
F &= u - \Delta_f(q_1, \dot{q}_1)
\end{align*}
$$

(6.2)

The assumption on the uncertainties is as given in Assumption 5.1 in Section 5.2.
Chapter 6. Robust Tracking via Dynamic Sliding Mode Control for Flexible Joint Robot

Dynamics of Modified Elastic Joint Manipulators

The state variables are defined as the angular position of the motors \( x_1 = q_2 \in \mathbb{R}^n \), the corresponding angular velocities \( x_2 = \dot{q}_2 \in \mathbb{R}^n \), the angular positions of the links \( x_3 = \theta_1 \in \mathbb{R}^n \) and the corresponding angular velocities \( x_4 = \dot{\theta}_1 \in \mathbb{R}^n \). The state variable representation of \((6.1)\) is obtained as

\[
\begin{align*}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= -J_m^{-1}k(x_1 - x_3) + J_m^{-1}F \\
\dot{x}_3 &= x_4 \\
\dot{x}_4 &= -J_1(x_3)^{-1}\{H(x_3, x_4) - k(x_1 - x_3)\} \\
y &= x_1
\end{align*}
\]

An input-independent state coordinate transformation can be used to obtain a GCCF in the form of \((2.5)\) which can be written in vector form as

\[
\eta_1 = x_1 \\
\eta_2 = x_2 \\
\eta_3 = -J_m^{-1}k(x_1 - x_3) + J_m^{-1}F \\
\eta_4 = -J_m^{-1}k(x_2 - x_4) + J_m^{-1}F
\]

where \( \eta_1 \in \mathbb{R}^n, \eta_2 \in \mathbb{R}^n, \eta_3 \in \mathbb{R}^n \) and \( \eta_4 \in \mathbb{R}^n \).

The inverse can be found by inspection to be

\[
\begin{align*}
x_1 &= \eta_1 \\
x_2 &= \eta_2 \\
x_3 &= J_m k^{-1}\{J_m^{-1}k\eta_1 + \eta_3 - J_m^{-1}F\} \\
x_4 &= J_m k^{-1}\{J_m^{-1}k\eta_2 + \eta_4 - J_m^{-1}F\}
\end{align*}
\]

The Jacobean matrix of the state coordinate transformation \((6.4)\) is given by
Differentiating the equation (6.4),

\[ \dot{\eta}_4 = -J_m^{-1}k(x_3 - \dot{x}_4) + J_m^{-1}\ddot{\tilde{p}} \]

\[ = -J_m^{-1}k\left\{x_3 + J_l^{-1}[H - k(x_1 - x_3)] \right\} + J_m^{-1}\ddot{\tilde{p}} \]

\[ = -J_m^{-1}k\left\{x_3 + J_l^{-1}H - J_l^{-1}kx_1 + J_l^{-1}kx_3 \right\} + J_m^{-1}\ddot{\tilde{p}} \]

\[ = -J_m^{-1}k\left\{J_l^{-1}H - J_l^{-1}kx_1 + (1 + J_l^{-1}k)x_3 \right\} + J_m^{-1}\ddot{\tilde{p}} \]

Substituting \( x_3 \) and \( x_3 \) of the equation (6.5) into \( \dot{\eta}_4 \) of the equation (6.6), we have

\[ \dot{\eta}_4 = -J_l^{-1}J_m^{-1}kH + J_l^{-1}J_m^{-1}k^2\dot{\eta}_1 \]

\[ - \left( J_l^{-1}k + 1 \right) \left( J_m^{-1}k\eta_1 + \eta_3 - J_m^{-1}\tilde{F} \right) + J_m^{-1}\ddot{\tilde{p}} \]

\[ = -J_l^{-1}J_m^{-1}kH - J_m^{-1}k\dot{\eta}_1 \]

\[ - \left( J_l^{-1}k + 1 \right) \eta_3 + J_m^{-1}\left( J_l^{-1}k + 1 \right)F + J_m^{-1}\ddot{\tilde{p}} \]

Substituting \( x_3 \) into (6.2)

\[ H = \dot{C}J_mk^{-1}\left( J_m^{-1}k\eta_2 + \eta_4 - J_m^{-1}\ddot{\tilde{p}} \right) + \dot{\tilde{G}} + \Delta_h \]

\[ \Delta_h = \Delta e\dot{q}_1 + \Delta g + \Delta_d \]

\[ F = u - \Delta_f \]
Chapter 6. Robust Tracking via Dynamic Sliding Mode Control for Flexible Joint Robot

Utilising (6.8), the equation (6.7) becomes

\[
\dot{\eta}_4 = - J_t^{-1} \left[ \hat{C} \left( J_m^{-1} k \eta_2 + \eta_4 \right) + \hat{G} \right] - J_m^{-1} k \eta_1 - \left( J_t^{-1} k + 1 \right) \eta_3 \\
+ J_m^{-1} \left( J_t^{-1} k + 1 \right) \ddot{u} + J_t^{-1} J_m^{-1} \hat{C} \ddot{u} + J_m^{-1} \dddot{u} \\
- J_t^{-1} J_m^{-1} k \Delta_h - J_m^{-1} \left( J_t^{-1} k + 1 \right) \Delta_f + J_t^{-1} J_m^{-1} \hat{C} \Delta_f - J_m^{-1} \dddot{f} \\
= - J_m^{-1} k \eta_1 - J_t^{-1} J_m^{-1} k \hat{C} \eta_2 - \left( J_t^{-1} k + 1 \right) \eta_3 - J_t^{-1} \hat{C} \eta_4 - J_t^{-1} \hat{G} \\
+ J_m^{-1} \left( J_t^{-1} k + 1 \right) \ddot{u} + J_t^{-1} J_m^{-1} \hat{C} \ddot{u} + J_m^{-1} \dddot{u} \\
- J_t^{-1} J_m^{-1} k \Delta_h - J_m^{-1} \left( J_t^{-1} k + 1 \right) \Delta_f + J_t^{-1} J_m^{-1} \hat{C} \Delta_f - J_m^{-1} \dddot{f} \\
(6.9)
\]

An input-independent state coordinate transformation to obtain a GCCF in the form of (2.5) for the flexible joint manipulator system is thus

\[
\begin{align*}
\dot{\eta}_1 &= \eta_2 \\
\dot{\eta}_2 &= \eta_3 \\
\dot{\eta}_3 &= \eta_4 \\
\dot{\eta}_4 &= \varphi(\eta, \ddot{u}, t) + \Delta(\eta, \omega, t) \\
y &= \eta_1 \\
\end{align*}
(6.10)
\]

with

\[
\begin{align*}
\varphi(\eta, \ddot{u}, t) &= - J_m^{-1} k \eta_1 - J_t^{-1} J_m^{-1} k \hat{C} \eta_2 - \left( J_t^{-1} k + 1 \right) \eta_3 - J_t^{-1} \hat{C} \eta_4 - J_t^{-1} \hat{G} \\
+ J_m^{-1} \left( J_t^{-1} k + 1 \right) \ddot{u} + J_t^{-1} J_m^{-1} \hat{C} \ddot{u} + J_m^{-1} \dddot{u} \\
\Delta(\eta, \omega, t) &= - J_t^{-1} J_m^{-1} k \Delta_h - J_m^{-1} \left( J_t^{-1} k + 1 \right) \Delta_f + J_t^{-1} J_m^{-1} \hat{C} \Delta_f - J_m^{-1} \dddot{f}
\end{align*}
(6.11)
\]

where \((\eta_1, \dot{\eta}_1, \ddot{\eta}_2, \dot{\eta}_3, \dddot{\eta}_4) = (y, y^{(1)}, y^{(2)}, y^{(3)}, y^{(4)})\).

6.3 Robust Output Tracking Control Scheme

For manipulator trajectory tracking control, define a tracking error function \(e(t)\) as the difference between the actual system output \(y \in \mathbb{R}^m\) and the desired reference trajectory...
\[ \mathbf{y}_R(t) \in \mathbb{R}^m. \] Let \( \mathbf{y}_l(t) \) be a desired reference for the link angular position and \( \mathbf{y}_R(t) \) be a desired reference for the motor position. From the dynamical equation (6.1), \( \mathbf{y}_R(t) \) corresponding to a given \( \mathbf{y}_l(t) \) can be seen to be

\[
\mathbf{y}_R(t) = k^{-1} \left[ \mathbf{J}_l \mathbf{y}_l^{(2)}(t) + \mathbf{C} \mathbf{y}_l + \mathbf{G} \right] + \mathbf{y}_l(t) \quad (6.12)
\]

For manipulator trajectory tracking control, define a tracking error function \( \hat{\mathbf{e}}(t) \) as the difference between the actual system output \( \hat{\mathbf{y}} \) and the desired reference trajectory \( \hat{\mathbf{y}}_R(t) \)

\[
\hat{\mathbf{e}}(t) = \hat{\mathbf{y}} - \hat{\mathbf{y}}_R(t) \quad (6.13)
\]

with

\[
\hat{\mathbf{e}}(t) = \left[ \mathbf{y} - \mathbf{y}_R, \mathbf{y}_l^{(1)} - \mathbf{y}_R^{(1)}, \mathbf{y}_l^{(2)} - \mathbf{y}_R^{(2)}, \mathbf{y}_l^{(3)} - \mathbf{y}_R^{(3)} \right]^T
\]

The uncertain I-O error dynamics of the flexible joint manipulator system are given by

\[
\mathbf{e}^{(4)} = -\mathbf{y}_R^{(4)} + \varphi(\hat{\mathbf{e}} + \hat{\mathbf{y}}_R, \hat{\mathbf{u}}, t) + \mathbf{\bar{\Delta}}(\hat{\mathbf{y}}_R + \hat{\mathbf{e}}, \omega, t) \quad (6.14)
\]

The uncertainty is given by

\[
\mathbf{\bar{\Delta}} = -\mathbf{J}_l^{-1} \mathbf{J}_m^{-1} \mathbf{k} (\Delta_f + \Delta_h) + \mathbf{J}_m^{-1} \Delta_f - \mathbf{J}_l^{-1} \mathbf{J}_m^{-1} \mathbf{\hat{C}} \mathbf{\hat{A}}_{f} + \mathbf{J}_m^{-1} \mathbf{\hat{A}}_{f}
\]

Since \( \Delta_f \) is a constant, \( \hat{\Delta}_f \) and \( \hat{\Delta}_f \) are zeros. Therefore the \( \mathbf{\bar{\Delta}} \) can be bounded as

\[
\| \mathbf{\bar{\Delta}} \| \leq \| \mathbf{J}_l^{-1} \| \| \mathbf{J}_m^{-1} \| \| \mathbf{k} \| (\| \Delta_f \| + \| \Delta_h \|) + \| \mathbf{J}_m^{-1} \| \| \Delta_f \|
\]

(6.15)

Utilising the uncertainty function (5.20) in Chapter 5, Assumption 5.1 and letting \([J^{-1}_m y]_i\) represent the \( i \)th element in the vector \( J^{-1}_m y \) and assuming \( ||[J^{-1}_m y]_i|| \leq \Xi_{mi} \| y \| \), the uncertainty (6.15) may be written as
\[
|\dot{\Delta}_i| \leq \sum \xi_i(y_i, y_i^{(1)}) + \xi_p(y) + \xi_s(y) + \xi_{di}
\]
\[
\leq \rho_i \|y\| + l_i
\]

where \(\xi_{di} \geq 0\) is a small positive constant, and \(\xi_s, \xi_p, \xi_{fi}\) are known, positive definite functions. Let \(\rho_i\) and \(l_i\) denote the upper bound on the lumped uncertainty, choose parameters \(0 < \theta < 1, \theta_0 + \theta = 1, \rho^{(0)} = \sqrt{\rho_1^2 + \rho_2^2 + \cdots + \rho_m^2}\) and calculate the

\[
\rho^{(1)} = \rho^{(0)} \left(1 + \max_i \{|A_i|\} \sqrt{3}\right)
\]
\[
\rho = \rho^{(0)} + \frac{(\rho^{(1)})^2}{4\theta}.
\]

According to Theorem 2.1 in Chapter 2, there exists a robust tracking control law

\[
\varphi(\dot{e} + \dot{y}_R, \ddot{u}, t) = y_R^{(4)} - Ks - K_0\text{sat}(s) - A\ddot{e}(t)
\]

Substituting (6.11) into (6.18), the robust tracking control law may be written by

\[
\ddot{u} + J_i^{-1} \dddot{u} + (J_i^{-1} + 1) u = J_m \left\{y_R^{(4)} - p - Ks - K_0\text{sat}(s) - A\ddot{e}(t)\right\}
\]

with

\[
p = -J_m^{-1}k(e + y_R) - J_i^{-1}J_m^{-1}kC(\dot{e}_1 + y_R^{(1)})
\]
\[-(J_i^{-1}k + 1)(\dot{e}_2 + y_R^{(2)}) - J_i^{-1}\ddot{C}(\dot{e}_3 + y_R^{(3)}) - J_i^{-1}\dddot{G}
\]

Suppose \(K\) and \(K_0\) satisfy

\[
\lambda_{\text{min}}(K)I_m - \left[\frac{1}{\theta_0} [BD]^T[BD] + \rho I_m\right] > 0
\]
\[
K_0 > l^{(0)}I_m
\]
Chapter 6. Robust Tracking via Dynamic Sliding Mode Control for Flexible Joint Robot

The choice of robust tracking control law is apparent from the normal GCCF form (6.10) with $\tilde{\Delta} \equiv 0$. From the robust control law (6.18) the highest order derivatives of control $u^{(\beta)} = (u_1^{(\beta_1)}, \ldots, u_m^{(\beta_m)})$ was given $u^{(\beta)} = v(\dot{\gamma}_R + \dot{e}, \ddot{e}, t)$ as in (2.24).

Theorem 6.1 \begin{align*}
\text{Assume the nominal GCCF system of the flexible joint manipulator (6.10) is minimum phase and the control law is chosen according to (2.24) in Chapter 2 with } \nu = [\nu_1, \ldots, \nu_m] = \varphi \text{ as in (6.19). For arbitrarily given but bounded } \epsilon > 0, \text{ there exists a robust tracking control law }

\nu = y_R^{(4)} - K_s - K_0sat_\epsilon(s) - A\dot{e}(t) \tag{6.20}
\end{align*}

which yields a tracking error system which is ultimately bounded by $\epsilon$.

6.4 Application to a Two-link Flexible Joint Robot

In this section, the dynamical equations of the two-link flexible joint robot are considered by the uncertain dynamical model (6.1) - (6.2). Let $J_l = J_l I_2; J_m = J_m I_2; k = k I_2; MgL = MgLI_2; \dot{C} = \dot{C}I_2; \dot{G} = MgLsin(q_1)I_2; \Delta_g = MgLq_1 I_2$. Let $r_1 = J_l^{-1}; r_2 = J_m^{-1}; r_3 = MgL; \dot{r}_3 = \dot{M}gL; r_4 = \dot{C}; x_1$ denote the motor shaft and $x_3$ be the angular position of the link

\begin{align*}
x_1 &= q_2 = \begin{bmatrix} q_{21} \\ q_{22} \end{bmatrix} & x_2 &= \dot{q}_2 = \begin{bmatrix} \dot{q}_{21} \\ \dot{q}_{22} \end{bmatrix} \\
x_3 &= q_1 = \begin{bmatrix} q_{11} \\ q_{12} \end{bmatrix} & x_4 &= \dot{q}_1 = \begin{bmatrix} \dot{q}_{11} \\ \dot{q}_{12} \end{bmatrix} \tag{6.21}
\end{align*}
Utilising (6.1), the GCCF (6.10) - (6.11) for the two-link flexible joint manipulator system is obtained as

\begin{align*}
\dot{\eta}_1 &= \eta_2 \\
\dot{\eta}_2 &= \eta_3 \\
\dot{\eta}_3 &= \eta_4 \\
\dot{\eta}_4 &= -r_2 k \eta_1 - r_1 r_2 r_4 k \eta_2 - (r_1 k + 1) \eta_3 - r_1 r_4 \eta_4 \\
&\quad - r_1 r_3 \sin \left[ \frac{1}{1} \left( r_2 k \eta_1 + \eta_3 - r_2 u \right) \right] + r_2 (r_1 k + 1) u + r_1 r_2 r_4 \dot{u} + r_2 \ddot{u} \\
&\quad - r_1 r_2 k \Delta_h - r_2 (r_1 k + 1) \Delta_f + r_1 r_2 r_4 \Delta_f - r_2 \ddot{\Delta}_f \\
\gamma_1 &= \eta_1
\end{align*} 

(6.22)

Since $\Delta_f$ is a constant, $\dot{\Delta}_f$ and $\ddot{\Delta}_f$ are zero. The Regularity condition (2.4) of the flexible joint manipulator system (6.22) yields

\[ \det \left[ \frac{\partial \varphi}{\partial u} \right] = r_1 r_2 k \neq 0 \] 

(6.23)

The input-output representation of the manipulator system with $\Delta_i = 0$ is

\begin{align*}
\gamma^{(4)} + r_1 r_4 \gamma^{(3)} + (r_1 k + 1) \gamma^{(2)} + r_1 r_2 r_4 k \gamma^{(1)} + r_2 k \gamma \\
+ r_1 r_3 \sin \left[ \frac{1}{1} \left( r_2 k \gamma + \gamma^{(2)} - r_2 u \right) \right] - r_2 (r_1 k + 1) u - r_1 r_2 r_4 \dot{u} - r_2 \ddot{u} &= 0
\end{align*} 

(6.24)

### 6.4.1 Output Tracking of Uncertain System via Dynamic Feedback Linearisation (DFLC)

In section 6.2, we investigate the uncertain GCCF for the $n$-link flexible joint manipulator (6.10) - (6.11). In section 6.4, we investigate the uncertain GCCF for the two-link flexible joint robot (6.22). In this subsection, we consider the use of the uncertain dynamical model (6.22) with a feedback linearisation controller (FLC).

For manipulator trajectory tracking control, define a tracking error function $\hat{e}(t)$ as the
difference between the actual system output $\hat{y}$ and the desired reference trajectory $\hat{y}_R(t)$ (6.13) and (6.12).

Based on the results of Section 6.3, the system of differential equations describing the tracking error dynamics is

\[
\dot{e}_1 = e_2 \\
\dot{e}_2 = e_3 \\
\dot{e}_3 = e_4 \\
\dot{e}_4 = \begin{align*}
-y_R^{(4)} - r_2 k (e_1 + y_R) - r_1 r_4 k (e_1 + y_R^{(1)}) \\
- (r_1 k + 1)(e_2 + y_R^{(2)}) - r_1 r_4 (e_3 + y_R^{(3)}) \\
- r_1 r_3 \sin \left[ r_2^{-1} k^{-1} \left( r_2 k (e + y_R) + (e_2 + y_R^{(2)}) - y_R^{(1)} \right) \right] \\
+ r_2 (r_1 k + 1) u + r_1 r_4 u + r_2 \dot{u} + \Delta
\end{align*}
\]

\[
y_1 = e_1 + y_R
\]

where

\[
\Delta = -r_1 r_2 k \Delta_h - (r_1 k + 1) \Delta_f
\]

with

\[
\Delta_h = \Delta_c \dot{q}_1 + \Delta_g + \Delta_d
\]

Linearisation of the tracking error dynamics (6.25) is equivalent to having the closed loop error dynamics obey $e_4 = v$:

\[
\dot{e}_4 = v = -a_4 e_4 - a_3 e_3 - a_2 e_2 - a_1 e_1
\]

From (6.25) and (6.27), the feedback linearising control law is simply:

\[
\ddot{u} + r_1 r_4 \dot{u} + (r_1 k + 1) u = r_2^{-1} \left( y_R^{(4)} + r_2 k (e_1 + y_R) + r_1 r_4 k (e_2 + y_R^{(1)}) \right)
\]
Chapter 6. Robust Tracking via Dynamic Sliding Mode Control for Flexible Joint Robot

\[ + (r_1 k + 1) (e_3 + y_R^{(2)}) + r_1 r_4 (e_4 + y_R^{(3)}) \]  \hfill (6.28)  
\[ + r_1 r_3 \sin \left[ r_2^{-1} k^{-1} \left( r_2 k (e_1 + y_R) + (e_3 + y_R^{(2)}) - r_2 u \right) \right] \]  
\[ + (\text{4th term}) \]  
\[ + (\text{5th term}) \]  

Using the state coordinate transformation (6.3) and (6.4) the nonlinear dynamical controller is

\[ \dot{u} = -r_1 k^2 x_1 + r_2 k (r_1 k + 1) x_3 \]  
\[ + r_1 r_2 r_4 k x_4 + r_2^{-1} r_1 r_3 \sin (x_3) \]  
\[ + r_2^{-1} \left\{ y_R^{(4)} - a_4 e_4 - a_3 e_3 - a_2 e_2 - a_1 e_1 \right\} \]

Substituting the error function (6.13) into here, DFLC is given by

\[ \ddot{u} + a_4 \dot{u} + a_3 u = \left( a_3 k - r_2^{-1} a_1 - r_1 k^2 \right) x_1 \]  
\[ + \left( a_4 k + r_2^{-1} a_2 \right) x_2 + k (r_1 k - a_3 + 1) x_3 \]  
\[ + k (r_1 r_4 - a_4) x_4 + r_2^{-1} r_1 r_3 \sin (x_3) \]  
\[ + r_2^{-1} \left\{ y_R^{(4)} a_4 y_R^{(3)} + a_3 y_R^{(2)} + a_2 y_R^{(1)} + a_1 y_R \right\} \]

Simulation Results for DFLC

Simulations for the two-link flexible joint manipulator, the parameters in the model is chosen as in equations (5.48) and (5.49) in Chapter 5.

The parameters of the controller are chosen as

\[ a_1 = 120; \quad a_2 = 154; \quad a_3 = 71; \quad a_4 = 14; \]  \hfill (6.29)

with closed loop poles assigned to \{-2 -3 -4 -5\}. Comparing with Chapter 5, the dynamic sliding mode controller can achieve better performance while using less control energy.

The system is required to track a desired angular link position \( y_1 \) of the form
Chapter 6. Robust Tracking via Dynamic Sliding Mode Control for Flexible Joint Robot

\[ y_1 = \begin{bmatrix} A\cos\left(\frac{2\pi}{5t}\right) \\ A\cos\left[(\pi - \pi/2)t\right] \end{bmatrix} \] (6.30)

The system is required to track the desired reference trajectory \( y_R(t) \) which can be computed from (6.12). All initial conditions for the system and controller were chosen to be zero.

Having obtained \( a_1, a_2, a_3, a_4 \) and the controller parameters, the closed-loop control system is simulated with the feedback controller (6.28). Increasing the magnitude of \( A \) will increase the difficulty in control. All initial conditions for the system and controller were chosen to be zero.

**DFLC for the uncertain system:** The system is simulated with the controller (6.28). The random disturbance is shown in Figure 6.1. When \( A = 0.0873 \), the actual link angular positions for trajectory tracking with feedback linearising controller are shown in Figure 6.2-a. The errors are shown in Figure 6.2-b. The actual positions are close to the desired trajectories within two seconds, the errors are approached to +0.0032. The desired link angular trajectories and the feedback linearising input torques are shown in Figure 6.2-c and 6.2-d. When \( A = 1.0 \), simulation results are shown in Figure 6.3. Respectively, increasing the magnitude of \( A \) shows that the errors have been increased under DFLC in Figure 6.4 and Table 6.1.

The simulation results (Figures 6.2 - 6.4) for DFLC with large uncertainty forces and disturbances are not good.

6.4.2 Robust Tracking via Dynamic Sliding Mode Control (RDSMC)

According to Section 6.3, a robust sliding mode controller is synthesised by the sliding surface and the strong sliding reachability condition. In this section we apply RDSMC to the two-link flexible joint manipulator that is modelled by (6.10) - (6.11). The GCCF of the flexible joint robot with uncertainties is given by (6.22). Defining the tracking error function as (6.13) and (6.12) and describing the tracking error dynamics as (6.25) and the uncertainties as (6.26):
\( \tilde{\Delta} = -r_1 r_2 k (\Delta_h + \Delta_f) - r_2 \Delta_f \)

with

\[
\Delta_h = \Delta_c q_1 + \Delta_g + \Delta_d
\]
\[
= \Delta_c \left[ (e_2 + y_R^{(1)}) + r_2^{-1} k^{-1} (e_4 + y_R^{(2)}) - k^{-1} \hat{u} \right] + \hat{r}_{i3} (e_1 + y_R) + \text{rand}(1)
\]

According to Assumption 5.1 and equation (6.16), considering the region \(-\frac{\Delta \pi^2}{2} I_2 \leq y^{(1)}, y^{(2)} \leq \frac{\Delta \pi^2}{2} I_2\), the estimate of a worst case bound on the function \(\tilde{\Delta}\), is given by

\[
\left\| \tilde{\Delta} \right\| \leq k \hat{r}_i I_2 \sum_{i=1}^{2} (\xi_{fi} + \xi_{ci1} \|y\| + \xi_{c2i} + \xi_{di}) + \hat{r}_{i3} (e_1 + y_R)
\]
\[
\leq k \hat{r}_i I_2 \sum_{i=1}^{2} (\xi_{ci} \|y\| + \xi_{2i})
\]

with

\[
\xi_{ci} = \hat{r}_{i3}; \quad \xi_{c2i} = \frac{1}{2} r_{i3}^{-1} k_i^{-1} \left| \Delta \pi^2 \right| \hat{r}_{i3}; \quad i = 1, 2;
\]
\[
\xi_{fi} = \delta_{i1} \xi_{ci1}; \quad \xi_{i2} = \delta_{i2} (\xi_{fi} + \xi_{c2i} + \xi_{di}) + r_{i2} \xi_{fi};
\]

where

\[
\rho = k \hat{r}_i I_2 \xi_{i2} I_2
\]
\[
l = k \hat{r}_i I_2 \xi_{i2} I_2
\]

According to equation (6.19) in Section 6.2, the robust tracking controller is given by
\[ \ddot{u} + r_1 r_4 \dot{u} + (r_1 k + 1) u = r_2^{-1} \begin{bmatrix} y_R^{(4)} + r_2 k (e_1 + y_R) + r_1 r_2 r_4 k (e_2 + y_R^{(1)}) \\ + (r_1 k + 1) (e_3 + y_R) + r_1 r_4 (e_3 + y_R^{(3)}) \\ + r_1 r_3 \sin \left( r_2^{-1} k^{-1} \left( r_2 k (e_1 + y_R) + (e_3 + y_R^{(2)}) - r_2 u \right) \right) \\ + (-K_0 \text{sat}_e(s) - m_3 \dot{e}_3 - m_2 \dot{e}_2 - m_1 \dot{e}_1) \end{bmatrix} \] (6.31)

where \( K \) satisfies

\[ \lambda_{\min}(K) I_2 > \left[ \frac{1}{\theta_0} [BD]^T [BD] + \rho I_2 \right] \] (6.32)

where \( D = \text{diag}[D_1, D_2] \) with \( D_i = [0, 0, 1]^T \); \( A = \text{diag}[A_1, A_2] \) is the companion matrix of the switching surface coefficients (\( \Sigma_{j=1}^3 m_j^{(i)} \lambda^j \) with \( i = 1, 2 \)); \( B \) satisfies the Lyapunov equation (5.32) and \( K_0 \) satisfies

\[ K_0 > l^{(0)} I_2 \] (6.33)

where \( 0 < \theta < 1, \theta_0 + \theta = 1, \rho^{(0)} = k J_t^{-1} J_m^{-1} \xi_1 \) and \( l^{(0)} = k J_t^{-1} J_m^{-1} \xi_2 \);

\[ \rho^{(1)} = \rho^{(0)} \left( 1 + \max_i \left[ \max(|A_i|) \right] \sqrt{3} \right) \]
\[ \rho = \rho^{(0)} + \frac{(\rho^{(1)})^2}{4 \theta} \] (6.34)

Simulation Results for RDSMC

Simulations were performed for the same flexible joint manipulator described in Section 6.4.1. Suppose that the coefficients \( \hat{J}_t, \hat{J}_m \) and \( \hat{M}_g L \) are unknown but bounded as

\[ 0.8 I_2 \leq \hat{J}_t \leq 1.5 I_2; \quad 0.8 I_2 \leq \hat{J}_m \leq 1.5 I_2; \quad 5 I_2 \leq \hat{M}_g L \leq 10 I_2; \]

Choose the controller parameters as \( \rho = k \hat{J}_t^{-1} \hat{J}_m^{-1} \xi_1 \) and \( l = k \hat{J}_t^{-1} \hat{J}_m^{-1} \xi_2 \). The parameters in the model are set as in equation (5.48). Let \( \xi_1 = 0.00012, \xi_f = \text{rand}(1) \) and \( \xi_2 = 0.0035 \). We have
\[ \rho = k \dot{J}_i^{-1} \dot{J}_m^{-1} \xi_1 = 0.0187I_2 \quad l = k \dot{J}_i^{-1} \dot{J}_m^{-1} \xi_2 = 0.54688I_2 \] (6.35)

\[ \rho^{(0)} = \left( \sqrt{\rho_1^2 + \rho_2^2} \right) I_2 = 0.0265I_2 \quad l^{(0)} = \left( \sqrt{l_1^2 + l_2^2} \right) I_2 = 0.77341I_2 \]

then

\[ \| \Delta \| < 0.0265 \| y \| + 0.77341 \] (6.36)

The parameters corresponding to the continuous controller were chosen as

\[ m_1 = 36; \quad m_2 = 33; \quad m_3 = 10; \]

Corresponding to ideal sliding mode poles assigned to \{-3, -3, -4\}. Let \( \theta = \theta_0 = 0.5 \) and

\[ A = \begin{bmatrix} A_1 & 0 \\ 0 & A_2 \end{bmatrix} \]

Use equation (6.34)

\[ \rho^{(1)} = \rho^{(0)} \left( 1 + \max \left\{ \max_{i} \left( |A_i| \right) \right\} \sqrt{3} \right) = 1.6799 \]

\[ \rho = \rho^{(0)} + \left( \frac{\rho^{(1)}}{4\theta} \right)^2 = 1.4376 \]

Let \( B = \text{diag}[B_1, B_2] \) is obtained from the Lyapunov equation with \( A = \text{diag}[A_1, A_2] \) and \( C = I \)
\( \mathbf{B} = \begin{bmatrix} B_1 & 0 \\ 0 & B_2 \end{bmatrix} = \begin{bmatrix} 0.6778 & -0.5000 & -0.8401 & 0 & 0 & 0 \\ -0.5000 & 0.8401 & -0.5000 & 0 & 0 & 0 \\ -0.8401 & -0.5000 & 4.7245 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.6778 & -0.5000 & -0.8401 \\ 0 & 0 & 0 & -0.5000 & 0.8401 & -0.5000 \\ 0 & 0 & 0 & -0.8401 & -0.5000 & 4.7245 \end{bmatrix} \)

Let

\( \mathbf{D}_1 = [0, 0, 1]^T \quad \mathbf{D}_2 = [0, 0, 1]^T \quad \mathbf{D} = \text{diag}[\mathbf{D}_1, \mathbf{D}_2] \)

\[
\frac{1}{\theta_0} [\mathbf{BD}]^T [\mathbf{BD}] + \rho \mathbf{I}_2 = 2 \times \begin{bmatrix} 20.2786 & 0 \\ 0 & 20.2786 \end{bmatrix} + 1.4376 \mathbf{I}_2 \\
= \begin{bmatrix} 21.7162 & 0 \\ 0 & 21.7162 \end{bmatrix}
\]

The control parameters of the \textit{RDSMC}, \( K \) and \( K_0 \), must satisfy equations (6.19). Choose

\[
K = \begin{bmatrix} 25 & 2 \\ 2 & 25 \end{bmatrix} \quad K_0 = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix} \quad \epsilon = 0.1
\]

\[
\lambda_{\text{min}} (K) \mathbf{I}_2 - \left( \frac{1}{\theta_0} [\mathbf{BD}]^T [\mathbf{BD}] + \rho \mathbf{I}_2 \right) = \begin{bmatrix} 23 & 0 \\ 0 & 23 \end{bmatrix} - \begin{bmatrix} 21.7162 & 0 \\ 0 & 21.7162 \end{bmatrix} = \begin{bmatrix} 1.2838 & 0 \\ 0 & 1.2838 \end{bmatrix} > 0
\]

\[
K_0 - \mathbf{I}^{(0)} = \begin{bmatrix} 1.0 & 0 \\ 0 & 1.0 \end{bmatrix} - \begin{bmatrix} 0.77341 & 0 \\ 0 & 0.77341 \end{bmatrix} = \begin{bmatrix} 0.22659 & 0 \\ 0 & 0.22659 \end{bmatrix} > 0
\]

To reduce the effects of the uncertain function \( \Delta \) on the tracking error \( \mathbf{e} \), a large gain was chosen in the robust tracking controller (5.51) in Chapter 5. For the dynamical feedback
robot systems, a small gain is chosen in the dynamical robust sliding mode controller (6.31) that only use small system energy but achieve good tracking performance.

**RDSMC for the uncertain system:** When $A = 0.0873$, the actual and desired angular positions are shown in Figures 6.5-a and 6.5-b. The errors between the desired and actual angular positions are shown in Figure 6.5-c and 6.5-d. The tracking errors are very close to the desired trajectories within two seconds, the errors are approached to 0.0003. The sliding surfaces and the dynamically generated input torques are shown in Figure 6.5-e and 6.5-f. When $A = 1.0$, simulation results are shown in Figure 6.6. The errors under RDSMC with increasing magnitude of $A$ are shown in Figure 6.7 and 6.1.

The simulation results (Figures 6.5 - 6.7) for the uncertain system with a large uncertain forces and disturbances shows the excellent results.

Comparing Figures 6.4 with Figures 6.7 we see that sliding mode component reduces the tracking error caused by the parameter uncertainties, modelling errors and disturbance in the system.

### 6.5 Concluding Remarks

In this chapter the modified n-link uncertain elastic manipulator has been investigated for the case when the system becomes relative degree two and the GCCF involves second order time derivatives of the control input torque. A dynamical feedback control scheme results. A robust tracking controller has been readily obtained. The approach has led to a dynamical linearising controller and also allows for the design of dynamical sliding mode control schemes based on the output function. The function $||\Delta||$ is a nonlinear function of the state, and bounds the points of the maximum size of the overall uncertainties. The design of the robust tracking controller involves determining a robust tracking control $u$ with bounding function $||\Delta||$ to achieve stability and good tracking performance. To reduce the effects of the uncertain function $\Delta$ on the tracking error $e$, a large gain was chosen in robust tracking controller (5.51) in Chapter 5 for FLC and RSMC. For the robot systems with dynamic feedback, the same closed loop poles are chosen as in the feedback linearising controller but a better performance is achieved than with the feedback
linearising controller in Chapter 5. Small gains are chosen in the dynamic sliding mode controller (6.31) so that low energy is used but excellent tracking performance is achieved. The relative performance is studied using the simulation of a two-link flexible joint robot. The simulation results obtained show excellent robust performance.

<table>
<thead>
<tr>
<th>Control A</th>
<th>$DFLC$ (1) ($\times 10^{-3}$)</th>
<th>$DRMSC$ (1) ($\times 10^{-3}$)</th>
<th>$DFLC$ (2) ($\times 10^{-3}$)</th>
<th>$DRMSC$ (2) ($\times 10^{-3}$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0873</td>
<td>3.2</td>
<td>0.3</td>
<td>3.1</td>
<td>0.3</td>
</tr>
<tr>
<td>0.1750</td>
<td>3.4</td>
<td>0.4</td>
<td>3.1</td>
<td>0.4</td>
</tr>
<tr>
<td>0.3490</td>
<td>$+2.0 \sim +4.0$</td>
<td>$-0.4 \sim +0.9$</td>
<td>$+1.0 \sim +5.0$</td>
<td>$-0.9 \sim +1.4$</td>
</tr>
<tr>
<td>1.0000</td>
<td>$-17.0 \sim +20.0$</td>
<td>$-16.0 \sim +9.0$</td>
<td>$-37.0 \sim +40.0$</td>
<td>$-26.0 \sim +26.0$</td>
</tr>
</tbody>
</table>

Table 6.1: DFLC and RDSMC controllers maximum tracking errors for two links

![Distributed random disturbance](image)

Figure 6.1: The uniformly distributed random disturbance.
Figure 6.2: a. The actual trajectories with the *DFLC*; b. The error between the desired and actual trajectories; c. The desired trajectories; d. The torques. $A = 0.0876$. Uncertain system.

Figure 6.3: a. The actual trajectories with the *DFLC*; b. The error between the desired and actual trajectories; c. The desired trajectories; d. The torques. $A = 1.0$. Uncertain system.
Figure 6.4: a. The errors between the desired and actual trajectories with different $A$ using the DFLC with uncertain system for the link one. b. The errors between the desired and actual trajectories with different $A$ using the FLC with uncertain system for the link two.
Chapter 6. Robust Tracking via Dynamic Sliding Mode Control for Flexible Joint Robot

Figure 6.5: a. The actual trajectories with the RDSMC; b. The error between the desired and actual trajectories; c. The sliding surfaces; d. The torques. $A = 0.0873$. Uncertain system.

Figure 6.6: a. The actual trajectories with the RDSMC; b. The error between the desired and actual trajectories; c. The sliding surfaces; d. The torques. $A = 1.0$. Uncertain system.
Figure 6.7: a. The errors between the desired and actual trajectories with different $A$ using the $RDSMC$ with uncertain system for the link one. b. The errors between the desired and actual trajectories with different $A$ using the $DRSMC$ with uncertain system for the link two.
Chapter 7

Robust Tracking via Sliding Mode Control for Rigid Robotic Manipulators

7.1 Introduction

In this chapter we apply robust tracking control scheme in Chapter 5 to rigid manipulators in the presence of model uncertainty and disturbances to achieve robust tracking. The explicit robustness guarantees provided by the methodology are demonstrated using rigid manipulator models. The methodology is compared with a traditional feedback linearisation and a discontinuous robust sliding mode control. Section 7.2 investigates the dynamics of an n-link rigid robot with uncertainties. Section 7.3 develops a robust sliding mode controller for a rigid robot. In Section 7.4 the controller is applied to the three DOF Tetrabot. Simulation results show excellent robust performance.

7.2 Rigid Manipulator Dynamical Model

In this section, the dynamics of an n-link rigid robot manipulator are described and the tracking control problem for such a manipulator is formalised. The dynamics of an n-link rigid manipulator were described in Section 3.2. They may be expressed as the second-order nonlinear vector differential equation
\[ M(q)\ddot{q} + H(q, \dot{q}) = F \]  
\[ y = q \]  

(7.1)

with

\[ M(q) = \dot{M}(q) + \Delta_m \]
\[ H(q, \dot{q}) = \dot{C}(q, \dot{q})\dot{q} + \dot{G}(q) + \Delta_h \]  
\[ F = u - \Delta_f(q, \dot{q}) \]
\[ \Delta_h = \Delta_c(q, \dot{q})\dot{q} + \Delta_g(q) + \Delta_d \]  

(7.2)

where \( q \in \mathbb{R}^n \) is vector of joint positions, \( \dot{q} \in \mathbb{R}^n \) is vector of joint velocities, \( M(q) \in \mathbb{R}^{n \times n} \) is the inertia matrix and positive definite and \( y \in \mathbb{R}^n \) denotes the vector of system outputs. The \( H(q, \dot{q}) \in \mathbb{R}^{n \times n} \) is the matrix containing Coriolis, centripetal, and gravitational forces and uncertainties. \( C(q, \dot{q}) \in \mathbb{R}^{n \times n} \) is the matrix containing actual centripetal and Coriolis terms and \( G(q) \in \mathbb{R}^n \) is the vector containing actual gravity terms. The vector \( u \in \mathbb{R}^n \) is the additional joint force beyond the compensation forces and will be referred to as the control.

The equation (7.1) for robotic systems is usually not totally known. Assumptions are made and stated for \( M(q), C(q, \dot{q}), G(q), \Delta_d \) and \( \Delta_f \). The assumptions on the uncertainties are needed only for the unknown parts \( \Delta_c(q, \dot{q}), \Delta_g(q) \) and \( \Delta_d \) that have been made in Assumptions 5.1. The following assumption is for the unknown part \( \Delta_m(q) \) which was based on the uncertainty and bounding function in Section 3.4:

**Assumption 7.1**

1. Inertia matrix \( M(q) \) satisfies
   \[ mI_n \leq M(q) \leq \bar{m}I_n, \quad \forall q \in \mathbb{R}^n \]

   where \( m \) and \( \bar{m} \) are positive constants, \( I_n \) is the identity matrix. It is assumed that \( m \) and \( \bar{m} \) are known.
Chapter 7. Robust Tracking via Sliding Mode Control for Rigid Robot

2. The desired trajectory

\[ \sup_{t \geq 0} \| \dot{y}_R \| \leq Q \]

where \( Q \) is a positive constant.

Dynamics of Rigid Robot Manipulators

The state variables are defined as the angular position of the links \( x_1 = q \in \mathbb{R}^n \), the corresponding angular velocities \( x_2 = \dot{q} \in \mathbb{R}^n \). The state variable representation of (7.1) is obtained as

\[
\begin{align*}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= -M(x_1)^{-1} [H(x_1, x_2) - F] \\
y &= x_1
\end{align*}
\] (7.3)

Utilising (7.2), an input-independent state can be used to obtain a GCCF in the form of (2.5) which can be written in vector form as

\[
\begin{align*}
\eta_1 &= x_1 \\
\eta_2 &= x_2 \\
\eta_3 &= -\ddot{M}(\eta_1)^{-1} [\dot{C}(\eta_1, \eta_2)\eta_2 + \dot{G}(\eta_1) - u + \Delta_m \eta_{R3} + \Delta_h + \Delta_f] \\
&= -\ddot{M}(\eta_1)^{-1} [\dot{C}(\eta_1, \eta_2)\eta_2 + \dot{G}(\eta_1) + u] - \ddot{M}(\eta_1)^{-1} [\Delta_m \eta_{R3} + \Delta_h + \Delta_f] \\
&= -\ddot{M}(\eta_1)^{-1} [\dot{C}(\eta_1, \eta_2)\eta_2 + \dot{G}(\eta_1)] + \ddot{M}(\eta_1)^{-1} - \ddot{M}(\eta_1)^{-1} [\Delta_m \eta_{R3} + \Delta_h + \Delta_f]
\end{align*}
\] (7.4)

where \( \eta_1 \in \mathbb{R}^n \), \( \eta_2 \in \mathbb{R}^n \) and \( \eta_3 \in \mathbb{R}^n \).

An input-independent state coordinate transformation can be used to obtain the following GCCF in the form of (2.5) for the rigid robot manipulator system.
## Chapter 7. Robust Tracking via Sliding Mode Control for Rigid Robot

\begin{equation}
\dot{\eta}_1 = \eta_2
\end{equation}

\begin{equation}
\dot{\eta}_2 = \varphi(\eta_1, \dot{u}, t) + \tilde{\Delta}(\eta_1, \omega, t)
\end{equation}

\begin{equation}
y = \eta_1
\end{equation}

with

\begin{equation}
\tilde{\Delta}(\eta, \omega, t) = -\tilde{M}(\eta_1)^{-1}\Delta(\eta, \omega, t)
\end{equation}

\begin{equation}
\Delta(\eta, \omega, t) = \Delta_m \eta R^3(\eta, \omega, t) + \Delta_f(\eta, \omega, t) + \Delta_h(\eta, \omega, t)
\end{equation}

where \( \eta = (\eta_1, \eta_2, \dot{\eta}_2 = (y, \dot{y}, \ddot{y}) \).

### 7.3 Robust Sliding Mode Control Schemes

Consider the uncertain system (7.5) - (7.6) satisfying the conditions (2.2). Let \( y_R(t) \) be the desired tracking signal and the output tracking error be

\begin{equation}
\hat{e}(t) = \hat{y} - \hat{y}_R(t)
\end{equation}

where \( \hat{y} = (y_1, \dot{y}_1, \ldots, y_m, \dot{y}_m, \hat{y}_R = \hat{y}_{R1}, \hat{y}_{R1}, \ldots, \hat{y}_{Rm}, \hat{y}_{Rm} \) and \( \hat{e}(t) = [e_1, \dot{e}_1, \ldots, e_m, \dot{e}_m]^T \).

The uncertain I-O error dynamics of the rigid robotic manipulator system are given by

\begin{equation}
\ddot{e} = \ddot{y}_R + \varphi(\hat{y}_R + \hat{e}, \dot{u}, t) + \tilde{\Delta}(\hat{y}_R + \hat{e}, \omega, t)
\end{equation}

with

\begin{equation}
\tilde{\Delta} = -\tilde{M}^{-1}(\Delta_m \hat{y}_R + \Delta_h + \Delta_f)
\end{equation}

where \( \hat{e} = [\hat{e}_1, \ldots, \hat{e}_m]^T \), \( \hat{y}_R = [\hat{y}_{R1}, \ldots, \hat{y}_{Rm}] \), \( \varphi = [\varphi_1, \ldots, \varphi_m] \) and \( \tilde{\Delta} = [\tilde{\Delta}_1, \ldots, \tilde{\Delta}_m] \).
We can find a continuous function $\rho \|\dot{y}\| + l$, satisfying the inequalities

$$\|\Delta\| \leq \rho \|\dot{y}\| + l \quad \rho \geq 0; \quad l \geq 0. \quad (7.10)$$

The uncertainty can be bounded as

$$\Delta = -\hat{M}^{-1} [\Delta_m \ddot{y}_R + \Delta_h + \Delta_f]$$

$$\|\Delta\| = \|\hat{M}^{-1}(\Delta_m \ddot{y}_R + \Delta_f + \Delta_h)\|$$

$$\leq \Xi(\|\Delta_m\| \|\ddot{y}_R\| + \|\Delta_f\| + \|\Delta_h\|) \quad (7.11)$$

where $\|\hat{M}^{-1}\| \leq \Xi$ is the overall upper bounding function for the inertia matrix. According to the uncertainty function (2.2) and Assumptions 5.1 and 7.1 for rigid manipulators, the uncertainty can be bounded as

$$|\Delta_m| |\ddot{y}_R| \leq m_i Q_i$$

$$|\Delta_h| \leq \xi_{ci}(y) |\ddot{y}| + \xi_{gi}(y) + \xi_{di} \quad \forall y \in \mathbb{R}^m$$

$$|\Delta_f| \leq \xi_{fi}(y, \dot{y}) \quad i = 1, \ldots, m.$$  

The uncertainty (7.11) may be written in the output coordinates as

$$|\hat{\Delta}_i| \leq \Xi_i \{m_i Q_i + \xi_{fi}(y, \dot{y}) + \xi_{ci}(y) |\ddot{y}| + \xi_{gi}(y) + \xi_{di}\} \quad (7.12)$$

$$\leq \rho_i |\dot{y}| + l_i$$

For given bounds, $u$ can be chosen to guarantee the stability of (7.5), where $\xi_{di} \geq 0$ is a small positive constant and $\xi_{gi}, \xi_{ci}, \xi_{fi}$ are known, positive definite functions.

The choice of a robust tracking control law is apparent from the normal GCCF form (7.8) - (7.9) with $\hat{\Delta} \equiv 0$
A robust sliding mode controller $\nu$ is designed by selecting sliding surfaces to prescribe desirable system performance and choosing a sliding reachability condition to ensure the sliding mode is attained. Consider the sliding surface $s$, defined in terms of the output tracking error coordinates $\hat{e}$ (7.7) as

$$s = \hat{e} + Ae$$

(7.14)

where $e = [e_1, \ldots, e_m]^T$ and $A = diag[A_1, \ldots, A_m]$.

The strong continuous sliding reachability condition which can be written in vector form as defined in Chapter 2 is required:

$$\dot{s} = -\gamma(k, s) = -Ks - K_0 sat_e(s)$$

(7.15)

Differentiating the sliding surface (7.14) along the trajectories of the uncertain GCCF (7.14), it follows that

$$\dot{s} = \ddot{e} + A\dot{e}$$

(7.16)

where $\dot{e} = [\dot{e}_1, \ldots, \dot{e}_m]^T$, $\ddot{e} = [\ddot{e}_1, \ldots, \ddot{e}_m]^T$ and we have

$$\dot{s} = \varphi(\dot{y}_R + \hat{e}, \dot{u}, t) + \tilde{\Delta}(\dot{y}_R + \hat{e}, \omega, t) + A\dot{e} - \ddot{y}_R(t)$$

(7.17)

with

$$\|\tilde{\Delta}(\dot{y}_R + \hat{e}, \omega, t)\| \leq \rho \|\dot{y}_R + \hat{e}\| + l$$

(7.18)

Using (7.15) and (7.16), the robust control law (7.13) becomes

$$\nu = \varphi(\dot{y}_R + \hat{e}, \dot{u}, t) = \ddot{y} - \ddot{e}$$

(7.13)
\[ \nu = \ddot{y}_R - K_s - K_0 \sigma_t(s) - A \dot{e} \]  

(7.19)

To estimate the uncertainty bound as in (7.18) for the rigid manipulator system (7.8) - (7.9), let \( \rho \) and \( l \) denote the upper bound on the lumped uncertainty that is given by (7.11), choose parameters \( 0 < \theta < 1, \theta_0 + \theta = 1, \rho^{(0)} = \| \rho \| \) and where \( l^{(0)} = \| l_i \| \) with \( i = 1, \ldots, m \). Then the stability of the uncertain system is ensured as proved in Chapter 2.

**Theorem 7.1** Assume the rigid manipulator system (7.1) is minimum phase and the control law is chosen according to (7.19) with \( \nu = [\nu_1, \ldots, \nu_m] = \varphi \). For arbitrarily given but bounded \( \epsilon > 0 \), there exists a robust tracking control law

\[ u = \hat{M} \left\{ \ddot{y}_R - \hat{M}^{-1} \left[ \hat{C} (\dot{y} + \dot{e}) + \hat{G} \right] - K_s - K_0 \sigma_t(s) - A \dot{e} \right\} \]  

(7.20)

with

\[ \lambda_{\text{min}}(K) > \left\{ \frac{1}{\theta_0} [BD]^T [BD] + \rho^{(0)} + \frac{\rho^{(0)} (1 + \max_i \{|A_i|\})^2}{4 \theta} I_m \right\} \]

\[ K_0 > l^{(0)} I_m \]

\[ \rho^{(1)} = \rho^{(0)} \left( 1 + \max_i \{|A_i|\} \right) \quad \rho = \rho^{(0)} + \frac{(\rho^{(1)})^2}{4 \theta} \]  

(7.21)

which yields a tracking error system which is ultimately bounded by \( \epsilon \).

### 7.4 Application to Three DOF Tetrabot Robot

#### 7.4.1 The Tetrabot Model

The GEC Tetrabot in Figure 4.1 is an experimental assembly robot with a novel three DOF parallel and three DOF serial geometric configuration as described in Chapter 4. This leads to a six DOF robot capable of very precise and rapid movements over a working
Chapter 7. Robust Tracking via Sliding Mode Control for Rigid Robot

volume of $1 \, m^3$. The Tetrabot is a nonlinear multivariable system, with inevitable model uncertainties, which is required to operate with variable payloads over a given workspace at high speeds. The dynamical equations governing the behavior of the Tetrabot are obtained from Section 4.5. The differential equations governing the controlled motions are given by (7.1)

$$M(q)\ddot{q} + H(q, \dot{q}) = \tau \tag{7.22}$$

with

$$M(q) = \hat{M}(q) + \Delta_m$$
$$H = \hat{C}(q, \dot{q})\dot{q} + \hat{G}(q) + \Delta_h$$
$$\tau = u - \Delta_f(q, \dot{q}) \tag{7.23}$$
$$\Delta_h = \Delta_c(q, \dot{q})\dot{q} + \Delta_g(q) + \Delta_d$$

The expression of the dynamics described by (7.22) and (7.23) is simplified by defining the disturbance vector

$$\Delta(q, \dot{q}) = \Delta_m\ddot{q}_R + \Delta_h + \Delta_f(q, \dot{q}) \tag{7.24}$$

The drive dynamic model in equations (7.22) - (7.24) is used for the purpose of simulation. We also include modelling uncertainty on the value of the inertia $M$ to account for the range of possible load values handled by the manipulator. The inverse dynamic solution has been obtained for the simplified three DOF model of the Tetrabot shown in Figure 4.1. The six DOF motion at the end-effector mounting point thus comprises three cartesian positions and three orientation angles. The minimum range of orientation of the end effector can be described as follows. Referred to a coordinate frame whose $'z'$ axis points vertically down, this is satisfied by Euler rotation angles ($\pm 90^\circ$, $\pm 90^\circ$, $\pm 180^\circ$) about axes ($Z, X', Z$). The primary working volume of the Tetrabot is a cylinder with a diameter of $1.0m$ and a height of $0.74m$ which maximum extreme reach to $2.0m$
diameter. The central axis of the cylinder is defined in the base frame as \((0, 0, z)\) with \(-1.475m \leq z \leq -0.740m\) and the radius of the cylinder base is 0.5m. The primary working volume can be described by the co-ordinates of \(W\) as \(\sqrt{x_w(0)^2 + y_w(0)^2} \leq 0.5m\) and \(-1.475m \leq z_w(0) \leq -0.740m\). The maximum velocity of the end-effector mounting point is 1 m/s and the maximum acceleration of the end-effector mounting point is 10 \(m/s^2\). The maximum payload including the end-effector (gripper + payload) is 6 kg. The desired accuracy during on-line motion is ±1.00mm, ±1.50\(^\circ\) and during initial calibration should be ±0.05mm, ±0.01\(^\circ\).

The three rods are actuated by servomotors. The peak torque is 8.4 \(Nm\). The drive amplifier characteristics ensure that the rated continuous torque (3.0 \(Nm\)) is available, with negligible reduction. The three wrists are actuated by servomotors where the peak torque for wrist 1 is 1.536 \(Nm\), wrist 2 is 1.61 \(Nm\) and wrist 3 is 0.567 \(Nm\) where the rated continuous torque for the wrists 1 and 2 are 0.647 \(Nm\), and for the wrist 3 is 0.329 \(Nm\). All six incremental optical encoders are directly attached to the motor shaft. All of the optical encoders have 1000 lines per revolution with complementary output to enable all joint positions to be read in real-time. The motor torque available to control the \(i\)th rod of the Tetrabot is given by equation (4.49) in Chapter 4

\[
\tau_i = \left( \frac{Nm_i L}{\zeta} + \frac{NI_n}{L} + \frac{I_m}{NL} \right) \dot{q}_i + \frac{LN}{\zeta} f_{ri} \tag{7.25}
\]

with

\[
\begin{align*}
\tau_i &= \tau_{mi} - \Delta f_i \quad i = 1, 2, 3. \\
\left( \frac{Nm_i L}{\zeta} + \frac{NI_n}{L} + \frac{I_m}{NL} \right) &= \left( \frac{\dot{N} m_i L}{\zeta} + \frac{NI_n}{L} + \frac{I_m}{NL} \right) + \Delta_{mi} \tag{7.26} \\
\frac{LN}{\zeta} f_{ri} &= \frac{LN}{\zeta} \dot{f}_{ri} + \Delta_{ci}(q, \dot{q}) \ddot{q} + \Delta_{gi}(q) + \Delta_{di} = \frac{LN}{\zeta} \dot{f}_{ri} + \Delta_{hi}(q, \dot{q})
\end{align*}
\]

where

\(\ddot{q}_R\): desired second time derivative of rod length.

\(\dot{f}_{ri}\): estimating axial force \(f_{ri}\) items.
\( \hat{m}_i \): estimating load and screw mass \( m_i \) (kg).

\( i \): number of the motor \((i = 1, 2, 3)\).

\( \Delta_{mi} \): load and mass of uncertainties.

\( \Delta_{hi} \): lumped force of uncertainties.

\( \Delta_{fi} \): external disturbance or uncertain term.

\( \Delta_i \): total uncertainties.

The Tetrabot dynamics with uncertainties become

\[
\ddot{q}_i = \frac{\zeta_N L}{N^2 \hat{m}_i L^2 + \zeta N^2 I_n + \zeta I_m} \tau_{mi} - \frac{(LN)^2}{N^2 \hat{m}_i L^2 + \zeta N^2 I_n + \zeta I_m} \dot{f}_{ri} + \Delta_i
\]  

with

\[
\Delta_i = \Delta_{mi} \ddot{q}_{Ri} + \Delta_{hi}(q, \dot{q}) + \Delta_{fi}(q, \dot{q}) \quad i = 1, 2, 3.
\]

Let \( \tau_{mi} = u_i, f_{ri} \) is given by equation (4.50) in Chapter 4, the uncertain GCCF system (5.14) for the Tetrabot is

\[
\begin{align*}
\dot{\eta}_1 &= \eta_2 \\
\dot{\eta}_2 &= \frac{\zeta_N L}{N^2 m_1 L^2 + \zeta N^2 I_n + \zeta I_m} u_1 - \frac{(LN)^2}{N^2 m_1 L^2 + \zeta N^2 I_n + \zeta I_m} \dot{f}_{r1} + \Delta_1 = \varphi_1 + \tilde{\Delta}_1 \\
y_1 &= \eta_1 \\
\dot{\eta}_3 &= \eta_4 \\
\dot{\eta}_4 &= \frac{\zeta_N L}{N^2 m_2 L^2 + \zeta N^2 I_n + \zeta I_m} u_2 - \frac{(LN)^2}{N^2 m_2 L^2 + \zeta N^2 I_n + \zeta I_m} \dot{f}_{r2} + \Delta_2 = \varphi_2 + \tilde{\Delta}_2 \\
y_2 &= \eta_3 \\
\dot{\eta}_5 &= \eta_6 \\
\dot{\eta}_6 &= \frac{\zeta_N L}{N^2 m_3 L^2 + \zeta N^2 I_n + \zeta I_m} u_3 - \frac{(LN)^2}{N^2 m_3 L^2 + \zeta N^2 I_n + \zeta I_m} \dot{f}_{r3} + \Delta_3 = \varphi_3 + \tilde{\Delta}_3 \\
y_3 &= \eta_5
\end{align*}
\]
with

\[ \varphi_i = \frac{\zeta NL}{N^2 \hat{m}_i L^2 + \zeta N^2 I_n + \zeta I_m} u_i - \frac{(LN)^2}{N^2 \hat{m}_i L^2 + \zeta N^2 I_n + \zeta I_m} \dot{\hat{f}}_{ri} \]  

(7.30)

where \( \eta = [\eta_1, \dot{\eta}_1, \eta_2, \dot{\eta}_2, \eta_3, \dot{\eta}_3, \eta_4, \dot{\eta}_4, \eta_5, \dot{\eta}_5, \eta_6] = [y_1, \dot{y}_1, y_2, \dot{y}_2, y_3, \dot{y}_3, y_4, \dot{y}_4, y_5, \dot{y}_5, y_6] \).

### 7.4.2 Output Tracking of Uncertain System via Feedback Linearisation (FLC)

In Section 7.2, we investigate the uncertain GCCF for an \( n \)-link rigid robotic manipulator (see equations (7.4) - (7.6)). In Section 7.4.1, we investigate the uncertain GCCF for the Tetrabot. In this subsection, we use the uncertain dynamical model (7.29) - (7.30) with the feedback linearisation controller (FLC). According to the uncertain error dynamics of the manipulator system (7.8) - (7.9), the system of differential equations describing the uncertain error dynamics is

\[ \begin{align*}
\dot{e}_i &= e_i \\
\ddot{e}_i &= \frac{\zeta NL}{N^2 \hat{m}_i L^2 + \zeta N^2 I_n + \zeta I_m} u_i - \frac{(LN)^2}{N^2 \hat{m}_i L^2 + \zeta N^2 I_n + \zeta I_m} \dot{\hat{f}}_{ri} \\
+ & \hat{\Delta}_i (\hat{y}_R + \hat{e}, \omega, t) - \hat{y}_{Ri}
\end{align*} \]  

(7.31)

with

\[ \hat{\Delta}_i = - \frac{\zeta NL}{N^2 \hat{m}_i L^2 + \zeta N^2 I_n + \zeta I_m} [\Delta_m \hat{q}_{Ri} + \Delta_h + \Delta_f] \]  

(7.32)

\[ \Delta_{hi} = \Delta_{ct} \hat{y}_i^{(1)} + \Delta_{gi} + \Delta_{di} \]

This linearisation of the tracking error dynamics (7.31) is equivalent to having the closed loop error dynamics obey \( \ddot{e}_i = v_i \):

\[ v_i = -a_{2i} \dot{\hat{e}}_i - a_{1i} e_i \]  

(7.33)
From (7.31) and (7.33), the feedback linearising control law is simply:

\[
u_i = \frac{N^2 \dot{m}_i L^2 + \zeta N^2 I_n + \zeta I_m}{\zeta NL} [\ddot{y}_{Ri} - a_2 \dot{e}_i - a_1 e_i] + \frac{LN}{\zeta} \ddot{f}_{ri}
\]  

(7.34)

7.4.3 Simulation Results for FLC

For numerical simulation the system parameter values are as given in [34, 57, 106] and the Appendix A; in particular, the lower bound on the load \( m = 3 \) kg and the upper bound \( \bar{m} = 6 \) kg. The trajectory involved diagonal movement across the primary working volume to move the wrist centre point \( W \) from the initial point \( x_{wo} = 0.0m, y_{wo} = -0.5m, z_{wo} = -1.475m \) to the final point \( x_{wd} = 0.0m, y_{wd} = 0.5m, z_{wd} = -0.74m \), with the velocities \( \dot{x}_w = 0.0, \dot{y}_w = 0.0, \) and \( \dot{z}_w = 0.0 \). The desired trajectories are shown in Figure 7.1.

The parameters corresponding to the feedback controller were chosen as

\[a_{1i} = 14I_3; \quad a_{2i} = 40I_3.\]

with closed loop poles assigned to \{-4, -10\}. The systems are required to track three desired fifth-order polynomial trajectories as shown in Figure 7.1

\[y_{Ri} = b_{i0} + b_{i1} t + b_{i2} t^2 + b_{i3} t^3 + b_{i4} t^4\]

Having obtained \( a_{1i}, a_{2i} \) and the controller parameters, the closed-loop control system is simulated with the feedback controller (7.34). The initial and final desired trajectory velocities and accelerations are chosen to be zero.

When the payload \( m = 3 \) kg the trajectory tracking errors, \( error_i \), and the driving torques \( T_{mi} \) are shown in Figure 7.2. After 2.0 seconds, the tracking errors of rod 1, rod 2 and rod 3 are \(-1.0 \times 10^{-4}m, -5.0 \times 10^{-4}m\) and \(-2.0 \times 10^{-3} \) respectively. The actual torques are \(0.62627Nm, -0.15497Nm\) and \(1.1777ATm\). When the payload \( \bar{m} = 6 \) kg the trajectories tracking errors and the driving torques \( T_{mi} \) are shown in Figure 7.2. After 2.0 seconds, the tracking errors of rod 1, rod 2 and rod 3 are \(-3.0 \times 10^{-4}m, -1.0 \times 10^{-4}m\) and
Chapter 7. Robust Tracking via Sliding Mode Control for Rigid Robot

\(-3.0 \times 10^{-4}\). The actual torques are 0.7758Nm, -0.1799Nm and 1.2124Nm. In both cases the tracking errors do not satisfy the accuracy requirement \(5.0 \times 10^{-5}\) and the driving torques satisfy the rated torque limit 8.8Nm and the continuous torque limit 3.0Nm given in Section 7.4.1. Increasing the closed loop poles could satisfy the accuracy requirement. The simulation results for the FLC with large uncertain forces and disturbances do not show good results.

7.4.4 Output Tracking via Robust Sliding Mode Control (RSMC)

According to Section 7.3 a robust sliding mode controller is synthesised by the sliding surface and the strong sliding reachability condition. In this section we apply RSMC to the Tetrabot with three DOF that is modelled by (7.29) - (7.30). The GCCF of the Tetrabot with uncertainties is given by (7.28). Define the tracking error function as (7.31) - (7.32) and describe the tracking error dynamics as (7.31) and the uncertainties as (7.32):

\[
\tilde{\Delta}_i = -\Xi_i [\Delta_{mi}\dot{y}_i + \Delta_{ci}\dot{y}_i + \Delta_{gi} + \Delta_{di} + \Delta_{fi}]
\]

where

\[
\Xi_i = \frac{\zeta NL}{N^2 \hat{m}_i L^2 + \zeta N^2 I_n + \zeta I_m}
\]

According to Assumptions 5.1 and 7.1 and equation (7.12), define the region \(-\frac{|a^2|}{2}I_3 \leq y^{(1)} \leq \frac{|a^2|}{2}I_3\), and the estimation of a worst case bound on the function \(\tilde{\Delta}\) is given by

\[
|\tilde{\Delta}_i| \leq \Xi_i \{\bar{m}_i Q_i + \bar{x}_{fi}(y, \dot{y}) + \bar{x}_{ci}(y) \|y\| + \bar{x}_{gi}(y) + \bar{x}_{di}\}
\]

\[
\leq \Xi_i \{\xi_{ci} \|y\| + \xi_{di}\}
\]  

with

\[
\xi_{c1} = \frac{LN}{\zeta} \hat{f}_{ri-1}; \quad \xi_{c2} = \frac{1}{2} \frac{LN}{\zeta} \hat{f}_{ri-2} |a^2|; \\
\xi_1 = \delta_1 \xi_{c1}; \quad \xi_2 = \delta_2 (\xi_f + \xi_{c2} + \xi_d);
\]
where

\[ \rho = \Xi \xi_1 I_3 \quad l = \Xi \xi_2 I_3 \]

According to Theorem 7.1, the robust tracking controllers (7.20) for the Tetrabot become

\[
\begin{align*}
u_1 &= \frac{N^2 \hat{m}_1 L^2 + \zeta N^2 I_n + \zeta I_m}{\zeta N L} \left[ y_{R1} - k_{11} \xi_1 - k_{12} \xi_2 - k_{13} \xi_3 - k_{01} \text{sat}_\epsilon (\xi_1) - A_1 \xi_1 \right] + \frac{LN}{\zeta} \dot{f}_{r1} \\
u_2 &= \frac{N^2 \hat{m}_2 L^2 + \zeta N^2 I_n + \zeta I_m}{\zeta N L} \left[ y_{R2} - k_{21} \xi_1 - k_{22} \xi_2 - k_{23} \xi_3 - k_{02} \text{sat}_\epsilon (\xi_2) - A_2 \xi_2 \right] + \frac{LN}{\zeta} \dot{f}_{r2} \\
u_3 &= \frac{N^2 \hat{m}_3 L^2 + \zeta N^2 I_n + \zeta I_m}{\zeta N L} \left[ y_{R3} - k_{31} \xi_1 - k_{32} \xi_2 - k_{33} \xi_3 - k_{03} \text{sat}_\epsilon (\xi_3) - A_3 \xi_3 \right] + \frac{LN}{\zeta} \dot{f}_{r3}
\end{align*}
\]

(7.36)

Let \( 0 < \theta < 1 \), \( \theta_0 + \theta = 1 \), \( \rho^{(0)} = ||\Xi \xi_1|| \) and \( l^{(0)} = ||\Xi \xi_2|| \), \( K = (k_{ij}) \), \( K_0 \) satisfies

\[
\lambda_{\min}(K) > \left\{ \frac{1}{\theta_0} [BD]^T [BD] + \rho^{(0)} I_3 + \left[ \frac{\rho^{(0)} (1 + \max_i \{|A_i|\})}{4 \theta} \right]^2 I_3 \right\}
\]

(7.37)

with \( K_0 = diag[k_{01}, k_{02}, k_{03}] \), \( \text{sat}_\epsilon (s) = [\text{sat}_\epsilon (s_1), \text{sat}_\epsilon (s_2), \text{sat}_\epsilon (s_3)]^T \) and \( A = diag[A_1, A_2, A_3] \) with \( A_i = \begin{bmatrix} 0 & 1 \\ -1 & -a \end{bmatrix} \).

7.4.5 Simulation Results DRSMC and CRSMC

Four different types of simulation will be discussed in this subsection. Discontinuous robust sliding mode controller (DRSMC) with/without uncertainties will show good tracking performance but chattering is present in the controls. Continuous robust sliding mode controller (CRSMC) with/without uncertainties will show excellent robust tracking performance and the discontinuity of the control is smoothed out by forcing the state asymptotically into a small neighborhood of the switching surface.

The discontinuous robust sliding mode controller (DRSMC) for the Tetrabot is given by
$$u = \Xi [\dot{y}_R - Ks - K_0 \text{sign}(s) - A\dot{e}] + \Xi_0 \dot{f}_r$$  \hspace{0.5cm} (7.38)

with \(\text{sign}(s) = [\text{sign}(s_1), \text{sign}(s_2), \text{sign}(s_3)]^T\) and \(\Xi_0 = \frac{LN}{\zeta} I_3\).

The continuous robust sliding mode controller (CRSMC) for the Tetrabot is given by

$$u = \Xi [\dot{y}_R - Ks - K_0 \text{sat}_e(s) - A\dot{e}] + \Xi_0 \dot{f}_r$$  \hspace{0.5cm} (7.39)

The desired trajectories are shown in Figure 7.1. The robust tracking controllers, given in equations (7.38) and (7.36), were used. Suppose that the coefficients \(\Xi_{oi} = 0.0034; \hat{M}_i, \hat{\xi}_i\) and \(\hat{f}_r i\) are unknown but bounded as

$$3 \leq \hat{M}_i \leq 6; \quad 2 \leq \hat{\xi}_i \leq 10; \quad 0.2 \leq \hat{f}_r i \leq 0.8; \quad i = 1, 2, 3.$$

Let \(\xi_{c1} = \Xi_0 \hat{f}_{r - 1} = 0.0017, \xi_{c2} = 0.5 \Xi_0 \hat{f}_{r - 1} |\pi|^2 = 0.0084, \delta_1 = 8, \delta_2 = 4\). Choose controller parameters as \(\rho = \delta_1 \Xi_c I_3 = 0.1247 I_3\) and \(l = \delta_2 \Xi_c I_3 = 0.3110 I_3\), then \(\rho(0) = 0.2160\) and \(l(0) = 0.5387\).

The parameters corresponding to the controller were chosen as

$$A_i = \begin{bmatrix} 0 & 1 \\ -1 & -a \end{bmatrix} \quad a = 20; \hspace{0.5cm} (7.40)$$

$$K = \begin{bmatrix} 65.3914 & 2.5 & 2.5 \\ 2.5 & 65.3914 & 2.5 \\ 2.5 & 2.5 & 65.3914 \end{bmatrix} \quad K_0 = \begin{bmatrix} 0.866 & 0 & 0 \\ 0 & 0.866 & 0 \\ 0 & 0 & 0.866 \end{bmatrix}$$

Substituting \(K\) and \(K_0\) into equation (7.37)

$$\lambda_{\text{min}}(K) I_3 - \left( \frac{1}{\theta_0} [BD]^T [BD] + \rho I_3 \right) = \begin{bmatrix} 63.3952 & 0 & 0 \\ 0 & 63.3952 & 0 \\ 0 & 0 & 63.3952 \end{bmatrix}$$
Chapter 7. Robust Tracking via Sliding Mode Control for Rigid Robot

\[
\begin{bmatrix}
60.8952 & 0 & 0 \\
0 & 60.8952 & 0 \\
0 & 0 & 60.8952
\end{bmatrix} > 0
\]

\[
K_0 - l^{(0)} I_3 = \begin{bmatrix}
0.866 & 0 & 0 \\
0 & 0.866 & 0 \\
0 & 0 & 0.866
\end{bmatrix} - \begin{bmatrix}
0.54 & 0 & 0 \\
0 & 0.54 & 0 \\
0 & 0 & 0.54
\end{bmatrix} > 0
\]

For numerical simulation the system parameter values and desired trajectories are as given in Section 7.4.2.

**DRSMC for the nominal system:** The discontinuous robust tracking controller is given by equation (7.38). When the payload \( m = 3kg \) the trajectory tracking errors, \( \text{error}_i \), and the driving torques \( T_{mi} \) are shown in Figure 7.3 and Figure 7.4. When the payload \( \bar{m} = 6Kg \) the trajectories tracking errors and the driving torques \( T_{mi} \) are shown in Figure 7.3 and Figure 7.4. Figure 7.5 represents the sliding surface with DRSMC. The tracking errors of rod 1, rod 2, rod 3 and the actual torques are given in Table 7.1. In both cases the tracking errors satisfy the accuracy requirement \( 5.0 \times 10^{-5} \) and the driving torques satisfy the rated torque limit \( 8.8Nm \) and the continuous torque limit \( 3.0Nm \) given in Section 7.4.

**DRSMC for the uncertain system:** The uncertainties of the Tetrabot system are shown in Figure 7.6. Simulations were performed with the same parameters previously described above. When the payload \( m = 3kg \) the trajectory tracking errors, \( \text{error}_i \), and the driving torques \( T_{mi} \) are shown in Figure 7.7 and Figure 7.8. After 2.0 seconds, the tracking errors of rod 1, rod 2 and rod 3 are \(-2.55 \times 10^{-5}m\), \(-2.26 \times 10^{-5}m\) and \(-2.15 \times 10^{-5} \) respectively. The actual torques are \(0.4402Nm, 0.2393Nm\) and \(0.7157Nm\). When the payload \( \bar{m} = 6Kg \) the trajectories tracking errors and the driving torques \( T_{mi} \) are shown in Figure 7.7 and Figure 7.8. After 2.0 seconds, the tracking errors of rod 1, rod 2 and rod 3 are \(-1.17 \times 10^{-5}m\), \(-1.84 \times 10^{-5}m\) and \(1.581 \times 10^{-5} \). The actual torques are \(0.5896Nm, -0.1962Nm\) and \(1.1619Nm\). Figure 7.9 gives the sliding surfaces with DRSMC.

The response of the system under DRSMC control is shown in Figures 7.7 - 7.9, while
Figure 7.10 shows that the DRSMC has a large amount of chattering present in the motor torques. Since chattering is not really practicable in the Tetrabot system, the chattering is controlled by approximately the DRSMC control by a CRSMC control as follows.

**CRSMC for the nominal system:** The continuous robust sliding mode controller (7.39) described in Section 7.4.4 is substituted, using a value of $\epsilon = 0.01$. Simulations were performed with the same parameters previously described above. When the payload $m = 3\, kg$ the trajectory tracking errors, $error_i$, and the driving torques $T_{mi}$ are shown in Figure 7.11 and Figure 7.12. When the payload $\bar{m} = 6\, Kg$ the trajectories tracking errors and the driving torques $T_{mi}$ are shown in Figure 7.11 and Figure 7.12. Figure 7.13 gives the sliding surfaces with CRSMC. The tracking errors of rod 1, rod 2, rod 3 and the actual torques are given in Table 7.1.

**CRSMC for the uncertain system:** The uncertainties of the Tetrabot system are shown in Figure 7.6. When the payload $m = 3\, kg$ the trajectory tracking errors, $error_i$, and the driving torques $T_{mi}$ are shown in Figure 7.14 and Figure 7.15. After 2.0 seconds, the tracking errors of rod 1, rod 2 and rod 3 are $0.68 \times 10^{-5}m$, $1.209 \times 10^{-5}m$ and $0.89 \times 10^{-5}$ respectively. The actual torques are $0.4775Nm$, $-0.1588Nm$ and $0.7434Nm$. When the payload $\bar{m} = 6\, Kg$ the trajectories tracking errors and the driving torques $T_{mi}$ are shown in Figure 7.14 and Figure 7.15. After 2.0 seconds, the tracking errors of rod 1, rod 2 and rod 3 are $0.63 \times 10^{-5}m$, $0.362 \times 10^{-5}m$ and $0.63 \times 10^{-5}$. The actual torques are $0.4863Nm$, $-0.0172Nm$ and $0.8957Nm$. Figure 7.16 gives the sliding surfaces with CRSMC.

The resulting responses and controls under CRSMC control are shown in Figures 7.14 - 7.16 and Figure 7.10. From these figures it is clear that the state and the control have a smooth shape once the sliding mode is approached. Figures 7.14 and 7.16 show some high frequency chatter. This is due to the sampled nature of the simulation process. It could be removed by decreasing the simulation sample period or increasing the value of $\epsilon$ in (7.39). The numerical simulation results show the tracking errors and the actual torques satisfy the required performance. When the mass is changed from the lower bound $3\, kg$ to the upper bound $6\, kg$, the performance only varies slightly (see Table 7.1). The continuous robust sliding mode controller has shown excellent robustness.
### Table 7.1: The FLC, DRSMC and CRSMC controllers maximum tracking errors and torques for three rods ([1]:No uncertainty; [2]:Uncertain system).

<table>
<thead>
<tr>
<th>Control Mode</th>
<th>Mass Kg</th>
<th>$Err_1$</th>
<th>$Err_2$</th>
<th>$Err_3$</th>
<th>$Tm_1$</th>
<th>$Tm_2$</th>
<th>$Tm_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>FLC</strong></td>
<td>3</td>
<td>-10</td>
<td>-50</td>
<td>-20</td>
<td>0.6262</td>
<td>-0.1549</td>
<td>1.1777</td>
</tr>
<tr>
<td>6</td>
<td>-30</td>
<td>-10</td>
<td>-30</td>
<td>0.7758</td>
<td>-0.1799</td>
<td>1.2124</td>
<td></td>
</tr>
<tr>
<td><strong>DRSMC</strong></td>
<td>3</td>
<td>-1.07</td>
<td>-1.8</td>
<td>-1.67</td>
<td>0.5318</td>
<td>-0.1502</td>
<td>1.0357</td>
</tr>
<tr>
<td>6</td>
<td>-1.17</td>
<td>-1.84</td>
<td>-1.581</td>
<td>0.5896</td>
<td>-0.1962</td>
<td>1.1619</td>
<td></td>
</tr>
<tr>
<td><strong>DRSMC</strong></td>
<td>3</td>
<td>2.55</td>
<td>-2.26</td>
<td>-2.15</td>
<td>0.4402</td>
<td>0.2393</td>
<td>0.7157</td>
</tr>
<tr>
<td>6</td>
<td>-2.68</td>
<td>-2.28</td>
<td>-2.51</td>
<td>0.6755</td>
<td>0.3181</td>
<td>1.1619</td>
<td></td>
</tr>
<tr>
<td><strong>CRSMC</strong></td>
<td>3</td>
<td>0.08</td>
<td>0.078</td>
<td>0.01</td>
<td>0.5342</td>
<td>-0.1366</td>
<td>1.0456</td>
</tr>
<tr>
<td>6</td>
<td>-0.1</td>
<td>0.094</td>
<td>0.02</td>
<td>0.5913</td>
<td>-0.1911</td>
<td>1.1632</td>
<td></td>
</tr>
<tr>
<td>$\epsilon = 0.01$</td>
<td>3</td>
<td>-0.18</td>
<td>-0.68</td>
<td>-0.127</td>
<td>0.4785</td>
<td>-0.0667</td>
<td>0.7500</td>
</tr>
<tr>
<td>6</td>
<td>-0.51</td>
<td>0.55</td>
<td>0.96</td>
<td>0.5010</td>
<td>-0.0817</td>
<td>1.0046</td>
<td></td>
</tr>
</tbody>
</table>

7.5 Concluding Remarks

In this chapter an n-link uncertain rigid manipulator has been investigated. Based on robust tracking control via sliding modes, we give a robust output tracking control scheme for any rigid manipulators. The function $||\dot{\Delta}||$ is a nonlinear function of the state, and bounds the maximum size of the overall uncertainties. The design of the robust tracking controller involves the selection of $u$ and the bounding function $||\dot{\Delta}||$ to achieve stability and good tracking performance. In practice, application of a robust sliding mode is more important. The relative performance is studied by simulation of a three DOF rigid robot - a Tetrabot. In this chapter, three typical controllers are applied to the Tetrabot mathematical model. The feedback linearising control does not satisfy the accuracy requirement. Increasing the closed loop poles could satisfy the accuracy requirement. The discontinuous and continuous robust sliding mode controllers have been applied to the nominal and uncertain systems. Both $DRSMC$ and $CRSMC$ satisfy the accuracy requirement. The response of the system under $DRSMC$ control has a large amount of chattering present, particularly in the motor torques. Since chattering is not really practicable in the Tetrabot system, the chattering is controlled using a $CRSMC$ control. $CRSMC$ has used a boundary layer approach by forcing the state asymptotically into a small neighborhood of the switching surface $\epsilon$. When the mass is changed from the lower bound $3kg$ to the upper
bound $6kg$, the performance only varies slightly and reduces chattering. The continuous robust sliding mode controller shows excellent robustness.

Figure 7.1: The desired trajectories for Robust Sliding Mode Control.

Figure 7.2: (1). The rod tracking errors and the motor torques with FLC at the lower bound $m=3Kg$ and (2). The upper bound $m=6Kg$. Uncertain system.
Figure 7.3: (1). The rod tracking errors with DRSMC at the lower bound $m=3\text{Kg}$ and (2). The upper bound $m=6\text{Kg}$. No uncertainty.

Figure 7.4: (1). The motor torques with the DRSMC at the lower bound $m=3\text{Kg}$ and (2). The upper bound $m=6\text{Kg}$. No uncertainty.
Figure 7.5: (1). The sliding surfaces with the DRSMC at the lower bound $m=3$Kg and (2). The upper bound $m=6$Kg. No uncertainty.

Figure 7.6: The uncertainty in the Tetrabot system.
Figure 7.7: (1). The rod tracking errors with DRSMC at the lower bound $m=3$Kg and (2). The upper bound $m=6$Kg. Uncertain system.

Figure 7.8: (1). The motor torques with the DRSMC at the lower bound $m=3$Kg and (2). The upper bound $m=6$Kg. Uncertain system.
Figure 7.9: (1). The sliding surfaces with the DRSMC at the lower bound $m=3\text{Kg}$ and (2). The upper bound $m=6\text{Kg}$. Uncertain system.

Figure 7.10: (1). The sliding reachability functions with DRSMC and (2). The sliding reachability functions with CRSMC. Uncertain system.
Figure 7.11: (1). The rod tracking errors with CRSMC at the lower bound $m=3\text{Kg}$ and (2). The upper bound $m=6\text{Kg}$. No uncertainty.

Figure 7.12: (1). The motor torques with the CRSMC at the lower bound $m=3\text{Kg}$ and (2). The upper bound $m=6\text{Kg}$. No uncertainty.
Figure 7.13: (1). The sliding surfaces with the CRSMC at the lower bound $m=3\text{Kg}$ and 
(2). The upper bound $m=6\text{Kg}$. No uncertainty.

Figure 7.14: (1). The rod tracking errors with CRSMC at the lower bound $m=3\text{Kg}$ and 
$\epsilon = 0.01$ (2). The upper bound $m=6\text{Kg}$. Uncertain system.
Figure 7.15: (1). The motor torques with the CRSMC at the lower bound $m=3$Kg and $\epsilon = 0.01$ (2). The upper bound $m=6$Kg. Uncertain system.

Figure 7.16: (1). The sliding surfaces with the CRSMC at the lower bound $m=3$Kg and $\epsilon = 0.01$ (2). The upper bound $m=6$Kg. Uncertain system.
Chapter 8

Lyapunov-based Robust Nonlinear Tracking Control Schemes for Rigid Manipulators

8.1 Introduction

This chapter considers the development of Lyapunov-based robust tracking controllers for the parallel and serial topology Tetrabot, although the theoretical control strategy presented is applicable to any manipulator tracking problem. The Tetrabot model was previously developed and discussed in Chapter 4. This accurate reduced-order model, which is highly nonlinear and complex, is used to design a nonlinear tracking controller. The uncertain mechanical system is described in Section 8.2. The nonlinear tracking control for the Tetrabot is developed and simulated in Section 8.3. Most importantly, the results of implementation on the three DOF parallel geometry are described in Chapter 9.

8.2 Uncertain Mechanical System

Consider the application of more complex nonlinear control techniques for trajectory tracking of rigid manipulators. The systems treated have a finite number of degrees of freedom (DOFs) and their main characteristic is that the number of independent scalar control inputs is the same as the number of DOFs of system.

The uncertain mechanical system with $n$ DOFs is described by
\[ M(t, q(t), \omega) \dot{q}(t) = U(t, q(t), \dot{q}(t), \omega) + D(t, q(t), \dot{q}(t)) \tilde{u}(t) \]  

(8.1)

\[ \omega \in \Omega \]  

(8.2)

where \( t \in \mathbb{R} \) represents time; \( q(t) \in \mathbb{R}^n \) is a vector of generalised co-ordinates which describe the configuration of the system; \( \tilde{u}(t) \in \mathbb{R}^n \) is a vector of control inputs and \( \omega \) is the lumped uncertain element.

The set \( \Omega \) and the functions \( M : \mathbb{R} \times \mathbb{R}^n \times \Omega \to \mathbb{R}^{n \times n}, U : \mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^n \times \Omega \to \mathbb{R}^n, \) and \( D : \mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^n \times \Omega \to \mathbb{R}^{n \times n} \), are known.

### 8.3 Development of Nonlinear Tracking Controller for the Tetrabot

The control strategy described in Section 2.5 is now applied to the three degree of freedom model of the Tetrabot in Fig (4.1). Using the Tetrabot dynamic model described in Section 4.5, only the design of a nonlinear tracking controller for the parallel structure is considered. A complete description of the 3-DOF model of the Tetrabot is given in Section 4.5.

Let \( q = [q_1 \ q_2 \ q_3]^T \) and \( \tilde{u} = [\tilde{u}_1 \ \tilde{u}_2 \ \tilde{u}_3]^T \), where \( \tilde{u}(t) \in \mathbb{R}^{3 \times 1} \) is the control torque applied to motor \( i \). The Tetrabot can be modelled by equations (8.1) - (8.2) where

\[ \omega \in \Omega = [3, 6] \]

In this model, \( \omega \) is the mass of an uncertain payload modelled as a point mass located at the toolplate. The only information available on \( \omega \) are its lower and upper bounds [34].

The mass matrix \( M(t, q, \omega) \) is symmetric and \( D \) in (8.1) is constant, diagonal and non-singular.

The generalised force term \( U(t, q, \dot{q}, \omega) \) is of the form

\[ U(t, q, \dot{q}, \omega) = U^g(q, \omega) + U^d(t, q, \dot{q}) \dot{q} \]  

(8.3)

Substituting from (8.3) into (8.1) yields
\[ M(t, q, \omega)\ddot{q} = U^g(q, \omega) + U^d(t, q, \dot{q})\dot{q} + D(t, q(t), \dot{q}(t))\ddot{u}(t) \] (8.4)

The expressions for \(M, U^g, U^d\) are used as in [34], [57] and the parameters are as given in Chapter 5. The control problem considered involves robustly tracking a desired reference trajectory, \(q_R\), despite the presence of the uncertainty \(\omega\). It is initially necessary to impose some preliminary assumptions which effectively ensure that the uncertainty is bounded. Known continuous bounding functions \(\beta_0, \beta_1\) and \(\beta_2\) are defined where for all \(q, \dot{q}\) and \(\omega\) the following inequalities hold

\[
\|U^g(q, \omega)\| + \|U^d(q, \dot{q}, \omega)\| \leq \beta_0(t, q, \dot{q})
\]
\[
\lambda_{\text{max}}[M(t, q, \omega)] \leq \beta_1(t, q)
\]
\[
\lambda_{\text{min}}[M(t, q, \omega)] \geq \beta_2(t, q) > 0
\] (8.5)

Define a new control input

\[
\ddot{u} = D(t, q(t), \dot{q}(t))\ddot{u}(t)
\] (8.6)

The evolution of the state equation of (8.4) can be described by:

\[
\dot{x} = Ax + B[h(t, x, \omega) + G(t, x, \omega)\ddot{u}]
\] (8.7)

with \(x = [q_1, q_2, q_3, \dot{q}_1, \dot{q}_2, \dot{q}_3]^T\),

\[
A = \begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix}; \quad B = \begin{bmatrix} 0 \\ I \end{bmatrix}
\] (8.8)

\[
h = M(t, q, \omega)^{-1}U(t, q, \dot{q}, \omega)
\] (8.9)

\[
G = M^{-1}
\] (8.10)
Chapter 8. Lyapunov-based Robust Nonlinear Tracking Control Schemes

To solve the manipulator trajectory tracking control problem, assume the desired state motion is given by \( q_R(t), \dot{q}_R(t) \) and \( \ddot{q}_R(t) \), the desired trajectory position, velocity and acceleration respectively. The model giving the desired behaviour is given by

\[
\dot{x} = Ax + Bu \tag{8.11}
\]

with

\[
\ddot{u} = \dot{q}_R \tag{8.12}
\]

where \( \ddot{x} = [q_{R1}, q_{R2}, q_{R3}, \dot{q}_{R1}, \dot{q}_{R2}, \dot{q}_{R3}]^T \) and \( \dddot{u} = [\dot{u}_1, \dot{u}_2, \dot{u}_3]^T \). The tracking control is intended to produce the same motion for the real system as that of the desired model. Defining \( z = x - \ddot{x} \), the error system is given by

\[
\dot{z} = Az + B[f(t, x, \omega) + G(t, x, \omega)\ddot{u}] \tag{8.13}
\]

where:

\[
f = M(t, q, \omega)^{-1}U(t, q, \dot{q}, \omega) - \ddot{q}_R \tag{8.14}
\]

The problem can be solved by defining a control law (2.48) to define \( \dddot{u} \). Let

\[
\dddot{u} = p^1(t, z) + p^2(t) + p^3(t, z) \tag{8.15}
\]

Thus by substituting (8.15) into (8.13), the error system with control is

\[
\dot{z} = f_x(t, x, \dddot{u}, \omega) + B[g(t, x, \dddot{u}, \omega) + G(t, x, \omega)p^3(t, z)] \tag{8.16}
\]

where

\[
f_x(t, x, \dddot{u}, \omega) = Az + BG(t, x, \omega)p^1(t, z) \tag{8.17}
\]

\[
g(t, x, \dddot{u}, \omega) = f(t, x, \omega) + G(t, x, \omega)p^2(t) \tag{8.18}
\]
Chapter 8. Lyapunov-based Robust Nonlinear Tracking Control Schemes

The function $p_2(t)$ is chosen to reduce the magnitude of the uncertain term $g(t, x, \tilde{u}, \omega)$. Stability and rate of decay of the error contributions are determined by $p_1(t, z)$. Robustness is largely prescribed via $p^\varepsilon(t, z)$. It should be noted that the model configuration in equation (8.13) is not dependent upon any particular operating point and so the controller is global.

For the Tetrabot the parameters were selected as follows. These selections largely follow those proposed in the papers of Corless [16].

\begin{equation}
    p_2(t) = \gamma_2 \dot{q}_R
\end{equation}

where

\begin{equation}
    \gamma_2 = 2\beta_1\beta_2[\beta_1 + \beta_2]^{-1}
\end{equation}

The function $p_1$ is chosen to stabilise the error system (8.17). Choosing a positive definite, symmetric matrix $Q$ and any positive scalar $\sigma > 0$, $p_1$ can be constructed as

\begin{equation}
    p_1(t, z) = -\gamma B^T P z
\end{equation}

where the continuous function $\gamma$ satisfies

\begin{equation}
    \gamma \geq \frac{1}{2} \sigma \beta_1
\end{equation}

and the matrix $P$ is the solution of the algebraic Riccati equation

\begin{equation}
    PA + A^T P - \sigma PBB^T P + 2Q = 0
\end{equation}

For any $\epsilon > 0$, the function $p^\varepsilon$ is selected to provide robustness to uncertainty. Choose functions $\kappa$ and $\rho$ which satisfy

\begin{align}
    \rho(t, x) &\geq \frac{1}{2} \beta_1 \|g(t, x, \tilde{u}, \omega)\| \\
    \kappa(t, x) &\geq \|g(t, x, \tilde{u}, \omega)\|
\end{align}

\begin{equation}
(8.24)
\end{equation}
and define

\[ \alpha = B^T P z \]
\[ \mu = \kappa \alpha \]

The function \( p^f \) for practical stabilisability is

\[ p^f(t, z) = -\rho(t, x)[\epsilon + \mu(t, z)]^{-1} \mu(t, z) \]  \hspace{1cm} (8.25)

where \( \epsilon \) is a small positive constant. As \( \epsilon \to 0 \) this structure approaches the discontinuous control associated with variable structure systems and will be effective against any uncertainty occurring in the range space of the input distribution matrix \( B \). The parameters of the tracking controller depend on the inertia matrix \( M \) and the generalised force term \( U \). The lower and upper bounds of \( ||g|| \) are estimated via open-loop simulation. There are clearly many appropriate selections for \( \rho \) and \( \kappa \) via \( M \) and \( U \).

Substituting (8.3), (8.5), (8.10), (8.14) and (8.19) into (8.18), we have

\[ g(t, x, \bar{u}, \omega) = M^{-1}(U^g + U^d) + (\gamma_2 M^{-1} - I)\ddot{q}_R \]  \hspace{1cm} (8.26)

Utilising (8.26), the particular geometry of the Tetrabot suggests bounding functions of the form (8.24) becomes

\[ \kappa \leq ||M^{-1}U^g|| + ||M^{-1}U^d|| + ||\gamma_2 M^{-1} - I||||\ddot{q}_R|| \]  \hspace{1cm} (8.27)
\[ \rho \leq ||M|| \{ ||M^{-1}U^g|| + ||M^{-1}U^d|| + ||\gamma_2 M^{-1} - I||||\ddot{q}_R|| \}

then \( \rho \) and \( \kappa \) can be chosen as

\[ \kappa = \kappa_1 + \kappa_2 ||\ddot{q}||^2 + \kappa_3 ||\ddot{q}_R|| \]
\[ \rho = \rho_1 + \rho_2 ||\ddot{q}||^2 + \rho_3 ||\ddot{q}_R|| \]
Chapter 8. Lyapunov-based Robust Nonlinear Tracking Control Schemes

with

\[
\begin{align*}
\kappa_1 &= \|M^{-1}U^q\|; \\
\kappa_2 &= \|\dot{q}\|^2 = \|M^{-1}U^d\|; \quad \kappa_3 = \|\gamma_2 M^{-1} - I\| \\
\rho_1 &= \|M\|\|M^{-1}U^q\|; \quad \rho_2 = \|\dot{q}\|^2 = \|M\|\|M^{-1}U^d\|; \quad \rho_3 = \|M\|\|\gamma_2 M^{-1} - I\|
\end{align*}
\]

Note that it is the appropriate choice of the structure of \(\rho\) and \(\kappa\) relative to the particular robot geometry which is needed for general applicability of the method.

MATLAB programs have been used to generate \(P\) and determine the uncertainty bounds. The parameters \(\beta_1, \beta_2, \gamma, \gamma_2, \kappa_1, \kappa_2, \kappa_3, \rho_1, \rho_2\) and \(\rho_3\) are then determined (see Table 8.1). The \(Q\) in (8.23) was chosen so that

\[
B^T Pz = \begin{bmatrix} p_1 & 0 & 0 & p_2 & 0 & 0 \\ 0 & p_1 & 0 & 0 & p_2 & 0 \\ 0 & 0 & p_1 & 0 & 0 & p_2 \end{bmatrix} \begin{bmatrix} q_i - q_{ri} \\ \dot{q}_i - \dot{q}_{ri} \end{bmatrix};
\]

with

\[
p_1 = K_1 \times K_e \times p_{21}; \quad p_2 = K_1 \times p_{22};
\]

where the design parameters \(K_1, K_e, p_{21}\) and \(p_{22}\) are positive and constant.

Tracking controller parameters are presented in Table 8.1.

8.4 Application to a Three DOF Model of Tetrabot

The drive dynamic model in equations (8.1) - (8.2) is used for the purpose of simulation. The primary working volume, the maximum velocity and acceleration of the end-effector mounting point, the desired accuracy during on-line motion, the peak torques and the rated continuous torques have been discussed in Chapters 4 and 5. These limits and constraints are considered during the design procedure.
\[ \lambda_{\text{max}}[M] \geq \beta_1 > 0 \]
\[ \lambda_{\text{min}}[M] \leq \beta_2 \]
\[ \gamma = \frac{1}{2}\sigma\beta_1 \]
\[ \gamma_2 = 2\beta_1\beta_2[\beta_1 + \beta_2]^{-1} \]
\[ \|M^{-1}U^g\| \leq k_1 \]
\[ \|M^{-1}U^g\| \leq k_2 \|\dot{q}\|^2 \]
\[ \|\gamma_2M^{-1} - I\| \leq k_3 \]
\[ \|M\| \|M^{-1}U^g\| \leq \rho_1 \]
\[ \|M\| \|M^{-1}U^g\| \leq \rho_2 \|\dot{q}\|^2 \]
\[ \|M\| \|\gamma_2M^{-1} - I\| \leq \rho_3 \]

Table 8.1: Tracking controller parameters

The motor torque available to control the \( i \)th rod of the Tetrabot is given by (4.49)

\[ \tau_{mi} = \left( \frac{NmL}{\eta} + \frac{NI_n}{L} + \frac{I_m}{NL} \right) \ddot{q}_i + \frac{LN}{\eta}f_{ri} \]

8.4.1 Open-loop Simulation of the Tetrabot

In Section 8.3 the parameters of the tracking controller depend on the inertia matrix \( M \) and the generalised forces \( U^g \) and \( U^d \). The lower bound and the upper bound of \( \| g \| \) are estimated via an open-loop simulation. There are clearly many appropriate selections for the bounding functions \( \kappa \) and \( \rho \) via \( M, U^g \) and \( U^d \). The reduced-order model of the Tetrabot is employed.

Assume that the Tetrabot is released with all initial values being zero except the vertical displacement. Then in the falling process, the central tube is along the z-axis, all the three actuator rod lengths are the same and there is no horizontal motion as far as the point \( W \) is concerned. The open-loop simulation of the Tetrabot was performed with ACSL. The inertia matrix \( M \), the forces \( U^g \) and \( U^d \) were captured. Three sets of \( M \) were
selected at the lowest, highest and midpoint of the trajectories as given in Table 8.2, and the corresponding $U^g$, $U^d$ and $D$ computed.

<table>
<thead>
<tr>
<th>No.</th>
<th>$i = 1, 2, 3$</th>
<th>$M_{i1}$</th>
<th>$M_{i2}$</th>
<th>$M_{i3}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>M.1</td>
<td>$M_{1i}$</td>
<td>2.347933</td>
<td>-0.556833</td>
<td>-0.556833</td>
</tr>
<tr>
<td></td>
<td>$M_{2i}$</td>
<td>-0.556833</td>
<td>2.347990</td>
<td>-0.556889</td>
</tr>
<tr>
<td></td>
<td>$M_{3i}$</td>
<td>-0.556833</td>
<td>-0.556889</td>
<td>2.347990</td>
</tr>
<tr>
<td>M.4</td>
<td>$M_{1i}$</td>
<td>2.391190</td>
<td>-0.580262</td>
<td>-0.580265</td>
</tr>
<tr>
<td></td>
<td>$M_{2i}$</td>
<td>-0.580265</td>
<td>2.391250</td>
<td>-0.580232</td>
</tr>
<tr>
<td></td>
<td>$M_{3i}$</td>
<td>-0.580265</td>
<td>-0.580323</td>
<td>2.391250</td>
</tr>
<tr>
<td>M.8</td>
<td>$M_{1i}$</td>
<td>2.935030</td>
<td>-0.860369</td>
<td>-0.860369</td>
</tr>
<tr>
<td></td>
<td>$M_{2i}$</td>
<td>-0.860369</td>
<td>2.935120</td>
<td>-0.860451</td>
</tr>
<tr>
<td></td>
<td>$M_{3i}$</td>
<td>-0.860369</td>
<td>-0.860451</td>
<td>2.935120</td>
</tr>
</tbody>
</table>

Table 8.2: Inertia Matrix of Tetrabot (Open-Loop)

The controller parameters were computed using MATLAB:

*Input*: $\sigma = 0.25$; $M_i$, $U^g$ and $U^d$ from the open-loop data in Table 8.2.

*Output*: The controller parameters ($\beta_1$, $\beta_2$, $\gamma$, $\gamma_2$, $\kappa$, $\rho_i$) were chosen as in Table 8.3:

The matrix $P \in \mathbb{R}^{6 \times 6}$ is the solution of the algebraic Riccati equation (8.23)

$$PA + A^TP - \sigma PBB^TP + 2Q = 0$$

where $A \in \mathbb{R}^{6 \times 6}$ and $B \in \mathbb{R}^{6 \times 3}$ are as in equation (8.8). $Q \in \mathbb{R}^{6 \times 6}$ is symmetric and positive semi-definite. $A, B$ were given, and $Q$ was chosen as such that

$$B^TP = \begin{bmatrix} 0.7071 & 0 & 0 & 0.9239 & 0 & 0 \\ 0 & 0.7071 & 0 & 0 & 0.9239 & 0 \\ 0 & 0 & 0.7071 & 0 & 0 & 0.9239 \end{bmatrix};$$
### Table 8.3: Tracking Controllers Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>M.1</th>
<th>M.4</th>
<th>M.8</th>
<th>M.11</th>
<th>M.14</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\beta_1$</td>
<td>2.9048</td>
<td>2.9888</td>
<td>3.4450</td>
<td>4.2413</td>
<td>5.4212</td>
</tr>
<tr>
<td>$\beta_2$</td>
<td>1.1996</td>
<td>1.2298</td>
<td>1.2183</td>
<td>1.2113</td>
<td>1.2073</td>
</tr>
<tr>
<td>$\gamma$</td>
<td>0.3631</td>
<td>0.3736</td>
<td>0.4306</td>
<td>0.5302</td>
<td>0.6777</td>
</tr>
<tr>
<td>$\gamma_2$</td>
<td>1.6980</td>
<td>1.7427</td>
<td>1.7999</td>
<td>1.8844</td>
<td>1.9747</td>
</tr>
<tr>
<td>$\kappa_1$</td>
<td>2.8045</td>
<td>2.7562</td>
<td>2.8186</td>
<td>2.8388</td>
<td>2.8405</td>
</tr>
<tr>
<td>$\kappa_2$</td>
<td>4.1681</td>
<td>4.0655</td>
<td>4.1042</td>
<td>4.1279</td>
<td>4.1418</td>
</tr>
<tr>
<td>$\kappa_3$</td>
<td>4.1546</td>
<td>4.1695</td>
<td>4.7745</td>
<td>5.5571</td>
<td>6.3575</td>
</tr>
<tr>
<td>$\rho_1$</td>
<td>8.1465</td>
<td>8.2378</td>
<td>9.7090</td>
<td>12.040</td>
<td>15.399</td>
</tr>
<tr>
<td>$\rho_2$</td>
<td>1.2108</td>
<td>1.2151</td>
<td>1.4137</td>
<td>1.7508</td>
<td>2.4540</td>
</tr>
<tr>
<td>$\rho_3$</td>
<td>1.268</td>
<td>1.2462</td>
<td>1.6446</td>
<td>2.3569</td>
<td>3.4466</td>
</tr>
</tbody>
</table>

#### 8.4.2 Numerical Simulation Results

In numerical simulations we used the system parameter values given in Chapter 7; in particular, we chose

\[ \bar{m} = 6\text{kg}; \quad \sigma = 0.25. \]

We let

\[
P = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix};
\]

\[
K = \begin{bmatrix} P_{21} & P_{22} \end{bmatrix}
\]

\[
B^TP = \begin{bmatrix} P_{21} & P_{22} \end{bmatrix};
\]

For any tolerance $\epsilon > 0$, define a controller $\hat{p} \in \hat{P}$ such that
Chapter 8. Lyapunov-based Robust Nonlinear Tracking Control Schemes

\[ \dot{u} = \dot{\hat{p}}(t, q, \dot{q}) \]

where \( t \in \mathbb{R} \) represents time; \( q(t) \in \mathbb{R}^3 \) is a vector of generalized coordinates which describe the configuration of the system. The mass matrix \( M(t, q, \omega) \in \mathbb{R}^{3 \times 3} \) is symmetric; \( U^g \in \mathbb{R}^{3 \times 1} \) is the rotation matrix and \( U^d \) is the translation constant; \( D \) is constant;

\[
\begin{bmatrix}
  m_{11} & m_{12} & m_{13} \\
  m_{21} & m_{22} & m_{23} \\
  m_{31} & m_{32} & m_{33}
\end{bmatrix}
\cdot
\begin{bmatrix}
  \ddot{q}_{14} \\
  \ddot{q}_{24} \\
  \ddot{q}_{34}
\end{bmatrix}
= 
\begin{bmatrix}
  \dot{x}_1^g \\
  \dot{x}_2^g \\
  \dot{x}_3^g
\end{bmatrix} + U^d \cdot 
\begin{bmatrix}
  \dot{q}_{14} \\
  \dot{q}_{24} \\
  \dot{q}_{34}
\end{bmatrix} + D \cdot 
\begin{bmatrix}
  \dot{u}_1 \\
  \dot{u}_2 \\
  \dot{u}_3
\end{bmatrix}
\]

Letting \( U^d = 0.5 \); \( D = C_1^{-1} \) (\( C_1 \) is the motor parameters); \( \dot{u}_i = \tau_{mi} \ (i = 1, 2, 3) \);

Now we have \( A, B, Q \) the parameters of the controller as in Table 8.3 and the equations of the controller (8.15), (8.21), (8.19) and (8.25). The Tetrabot closed-loop control system is simulated with the nonlinear tracking controller.

The Tetrabot Dynamic Analysis Program (TDASP) has been used as a design tool to investigate the dynamics of the Tetrabot design described in Section 2.5 and develop the nonlinear tracking controllers described in Section 8.3. Essentially this simulation study is performed to test the controller parameterisation prior to implementation. For numerical simulation the system parameter values are as given in [34], [57] and Chapter 5; in particular, the lower bound on the load \( m = 3kg \) and the upper bound \( \bar{m} = 6kg \). The trajectory involved diagonal movement across the primary working volume to move the wrist centre point \( W \) from the initial point \( x_{wo} = 0m, y_{wo} = -0.5m, z_{wo} = -1.475m \) to the final point \( x_{wd} = 0m, y_{wd} = 0.5m, z_{wd} = -0.74m \), with the velocities \( \dot{x}_w = 0.0, \dot{y}_w = 0.0, \text{and} \dot{z}_w = 0.0 \). The desired trajectories are shown in Figure 8.1.

When the payload \( m = 3kg \) the trajectory tracking errors, \( error_i \), and the driving torques \( T_{mi} \) are shown in Figure 8.2. After 2.0 seconds, the tracking errors of rod 1, rod 2 and rod 3 are \( 3.627 \times 10^{-5}m, -0.894 \times 10^{-5}m \) and \( 3.560 \times 10^{-5} \) respectively. The actual torques are \( 0.5358Nm, -0.1344Nm \) and \( 1.0534Nm \). When the payload \( \bar{m} = 6Kg \) the trajectories tracking errors and the driving torques \( T_{mi} \) are shown in Figure 8.3. After 2.0 seconds, the tracking errors of rod 1, rod 2 and rod 3 are \( 4.025 \times 10^{-5}m, -1.289 \times 10^{-5}m \) and
3.954 \times 10^{-5}$. The actual torques are $0.5949 \, Nm$, $-0.1851 \, Nm$ and $1.1664 \, Nm$. Figures 8.2 and 8.3 show some high frequency chatter. This is due to the sampled nature of the simulation process. It could be removed by decreasing the simulation sample period or increasing the value of $\epsilon$ in (8.25). In both cases the tracking errors satisfy the accuracy requirement $5.0 \times 10^{-5}$ and the driving torques satisfy the rated torque limit $8.8 \, Nm$ and the continuous torque limit $3.0 \, Nm$ given in Section 7.4.1.

The numerical simulation results show the tracking errors and the actual torques satisfy the required performance. When the mass is changed from the lower bound $3 kg$ to the upper bound $6 kg$, the performance only varies slightly. The nonlinear tracking controller has shown excellent robustness.

![Figure 8.1: The desired trajectories for the nonlinear tracking controller.](image)

**8.5 Concluding Remarks**

This chapter details Lyapunov-based robust nonlinear tracking control schemes for the Tetrabot. M Corless and G Leitmann have developed a class of uncertain dynamical systems described by ordinary differential equations and characterized by certain structural conditions and known bounding functions. For a feasible class of desired state motions they present a class of controllers which ensure that the controlled system can asymptotically track the desired motion to any desired degree of accuracy. Various classes of controllers are presented. The design of all these controllers is based on Lyapunov theory. This study is based on the Corless-Leitmann nonlinear tracking control schemes. The
Chapter 8. Lyapunov-based Robust Nonlinear Tracking Control Schemes

Figure 8.2: (a) The rod trajectory errors and (b) the motor torques with the payload at the lower bound $m = 3Kg$.

Figure 8.3: (a) The rod trajectory errors and (b) the motor torques with the payload at the upper bound $m = 6Kg$. 
controller has been simulated on the Tetrabot mathematical model in Chapter 4. The simulation results show the tracking errors and the actual torques satisfy the required performance. When the mass is changed from the lower bound 3\( kg \) to the upper bound 6\( kg \), the performance only varies slightly. The nonlinear tracking controller has shown excellent robustness. The theoretical nonlinear tracking controller will be applied to the Tetrabot apparatus in Chapter 9.
Chapter 9

Nonlinear Tracking Controller Applied to the Tetrabot Apparatus

9.1 Introduction

Previous chapters have been devoted to the theoretical aspects of tracking control of the robots, and the results have been confirmed by simulation study in Chapter 4. In order to gain insight into such complex systems, simplifications have been made when modelling to facilitate theoretical derivation and simulation. However, when developing a practical robot controller we face situations that have not been considered, such as the nonlinearity of actuators, sensor noise and configuration dependent friction. Many of the practical considerations can only be verified by extensive experiment. There are a number of experimental studies on tracking control of robot manipulators; examples are [44, 3, 77, 40, 27].

The design of a tracking control experiment includes several topics, such as the choice of reference trajectories, performance criteria, the implementation of the nonlinear tracking controller and the selection of controller design variables to test. It also includes some background aspects, such as what can be predicted from theory, and practical considerations of how to verify the real-time control program.

In this chapter, we will discuss the arrangements that concern experimental design. The available instruments are described in Section 9.2, and the software verification is treated in Section 9.3. Theoretical predictions and practical considerations are dealt with in
Section 9.4, while the proposed control software interface, the parameter estimation and error treatment are described in Section 9.5.

9.2 Experimental Set-up

9.2.1 Computer-controlled Tetrabot System Structure

Tetrabot System Structure

We briefly look at the architecture of the control system of the GEC Tetrabot industrial robot.

The hardware architecture is that of a two-level hierarchy with a Sun Computer serving as the host control computer to a 68000 SBC as the low-level robot controller. An RS232 serial links to terminal (68000 SBC) and remote computer (Sun). The 68000 SBC with six axis parallel I/O interface boards control six individual joints with a PID control law.

The software structure is that of a three-level hierarchy:

1. **Top-level**: MICCON (Methodology for Industrial Command and Control)[21] is the top-level operating system software which provides: a) A real time process scheduler; b) Data and function handling; c) Database handling; d) Command line interpreter.

2. **Mid-level**: General process software is written in C. With such a complex piece of software made up of many hundreds of functions, a system of subdirectories and libraries is used.

3. **Low-level**: Servo software is written in Assembler, mainly dealing with real time control.

9.2.2 Tetrabot Hardware

The Tetrabot controller architecture is depicted in Figure 9.1.
ROBOT INTERFACE

Motor & ambient tent temp. sensors

Over travel limits
Emergency
External contact sensors
cen. tube angle pots.

Main drive brakes
Encoder
channels A & B
Calibration reference

Fault interrupt

Hall effect position sensors
Resolver type sensor
Brushless DC motor
drives (3 phase)

Figure 9.1: The Tetrabot Controller Architecture
Position Sensors (Incremental Optical Encoder) and Converters

Position measurement of the wrist axis is performed by an incremental optical encoder attached to the same shaft as the drive motor. Therefore, the encoder measures motor position directly. Incremental encoders output pulses while rotating and these pulses have to be counted and stored in order to keep track of the wrist position.

As the measurement system adopted uses incremental optical encoders it is necessary to provide a calibration reference, and use travel limit switches to stop the axis from continued rotation past the maximum allowable position. The reference will consist of an optical switch, the beam of which will be broken by an interrupter mounted on the end of the central drive shaft. As this shaft is connected directly to the output of the Harmonic Drive, no errors are introduced between the drive output and the reference. The only error is in the ability to resolve the calibration point. It is possible to combine the limit switches and the calibration reference points by using the limit switch as the reference.

All six incremental encoders are directly attached to the motor shaft. All of the encoders will have 1000 lines per revolution with complimentary output. The encoders will also include a once per revolution marker for use in the calibration procedures. Because the encoders are directly coupled to the motor shafts, their output can also be used as the velocity signal to the servo amplifiers, instead of having a tachometer added to the end of the motor. To do this, the encoder output has to pass through an F to V (frequency to voltage) converter in order to simulate the output of the tachometer.

D/A Converters

Digital-to-Analogue Converters take many forms, but essentially they all rely on the basic concept of using an input digital code to open or close switches in an electronic circuit. The closure of the switches causes a voltage to be generated corresponding to the digital code.

The two 12 bit D to A converters each have their own address within the base address of the axis. To write a value to the servo demand D to A its address is placed on the local address bus and its value is written out on a 12 bit bus which is common to all D to A’s.
9.2.3 Computer Control and Data Collection

A 68000 CPU, a digital I/O interface, an analogue input and an expansion I/O interface make up the built-in part of the Tetrabot controller. Each of these are standard boards which are connected together via a Multibus I backplane. The servo amplifiers were also standard items but with slight modifications for the wrist motors. The buffer board, axis interface boards 1 to 6 and the auxiliary board were designed specifically for the Tetrabot. The motors with the servo amplifier need to be connected to the computer, and effective control performance information needs to be transferred between them and the computer. Each device has its own controller (interface) that permits the communication to take place. The simplest technique is known as unconditioned transfer where the computer carries out the operation whenever the particular section of code is executed. Buffers can become full and data can be lost.

An advanced technique has been used for data collection: using the PC computer RS-232 serial links to the Tetrabot computer system instead of the 68000 SBC so that all the data transference can be collected in real-time; then all the data is transferred to the sun station so that the data can be analysed and plotted by the MATLAB.

9.2.4 Tetrabot Software

MICCON Primary Features

MICCON is based on a message passing, multiprocessor control system MARCOMS, developed in a project studying programmable automatic assembly availability on various microprocessors and minicomputers. C was chosen as the implementation language for MICCON.

Additional ideas for features in MICCON came from various AI programming environments such as LISP and PROLOG. In particular, the use of a symbolic database to hold descriptions of the various language elements, and the concept of object oriented programming form the basis of many features of MICCON.

The target, in producing MICCON, was to develop a versatile programming environment
for real time control system development, whilst at the same time bridging the gap between AI techniques and more conventional computer programming languages.

MICCON has been used successfully on a range of projects involved in control by microprocessors of mechanical systems, such as MKI GADFLY, MKII GADFLY and the Tetrabot. The MKII and the general purpose gripper each use a single 68000 microprocessor implementing both the servo loop software at a time unit of 10 ms and 5 ms respectively and the higher level software that carries out the kinematics calculations and determines the space-time path of the mechanisms during the motions.

The Tetrabot controller is based on a 68000 microprocessor. It is expected that this combination programmed using MICCON, will allow the full integration of sensor feedback the sophisticated controller required for this device.

As well as executing on single board computers, MICCON is also used on a VAX host computer where the flexibility of the MICCON interface allows complex calculations to be carried out on demand [21].

5ms Clock Interrupt

Every 5ms a clock interrupt is generated which causes the CPU to suspend its current operation and service the interrupt. The interrupt performs two primary functions. Firstly, the servo control demands to the motors are updated (position controller). Secondly, the process scheduler suspends the process being revised, before the interrupt, and prepares the next process for servicing on return from the interrupt. Some processes are defined as being time critical (TCP). These are given the highest priority below that of the position controller. If a TCP has not been completed by the following interrupt then an error is generated because the process is no longer synchronised to the clock.

The Process Scheduling System

Figure 9.2 presents a simplified view of the scheduling system employed by MICCON. The technique used for scheduling requires that processes are placed in queues and type of process will dictate which queue it is placed in. The scheduler then moves the processes between queues (Figure 9.2 - a). Processes placed in the TCP queue have priority over
any other (Figure 9.2 - b). Processes may be switched in a number of ways. If the process is a long one, then it will continue until the next clock interrupt. Finally, the process may be completed (Figure 9.2 - c). The execution of processes normally develops into a tree like structure with each main process having a number of subprocesses which may or may not be concurrent. Furthermore, a subprocess may be placed in the TCP queue while its parent is a background process running in the READY queue (Figure 9.2 - d).

![Diagram of process scheduling and execution](image)

**Figure 9.2: The Process Scheduling**

**Tetrabot Controllers Software Structure**

The Tetrabot controller is achieved via a VDU using the MICCON command interpreter. Commands are given to the system in the form of C function calls and macros which are held in the MICCON dictionary. The user also has access to data held in the dictionary. There is a hierarchy of MICCON command levels which can be entered recursively by unyoking certain functions which place additional functions and data in the dictionary and re-enter the MICCON command line interpret. On exit from each level these additional functions and data are deleted from the dictionary. The structure of this hierarchy is as in Figure 9.3. The software of the Tetrabot controller is given in Appendix C and the
Tetrabot controller functions are given in Appendix D.

After the startup procedure the system is inactive with MICCON ready to accept command inputs, as described below:

*enable() Function*

This function activates the manipulator, attempts to calibrate it and moves it to its park position. If successful, the following functions are entered in the MICCON dictionary and the MICCON command line interpreter is re-entered. The enable function is as Figure 9.4.

*demo_mode() Function*

Demonstration mode facility to cause the manipulator to execute a pre-defined sequence of moves. At present there are \( n \) predefined positions, \( demo\_pos \) 0 to \( demo\_pos n \). The positions are available in the MICCON dictionary. The \( demo\_mode \) function is as Figure 9.5.

*run_mode() Function*

Put the system in its normal running mode with the move commands in the dictionary (Figure 9.6).

*manual() Function*

Enter manual move mode in which movements of the manipulator can be controlled by pre-programmed terminal keys.
Chapter 9. Nonlinear Tracking Controller Applied to the Tetrabot Apparatus

Figure 9.4: The Tetrabot enable Function
Chapter 9. Nonlinear Tracking Controller Applied to the Tetrabot Apparatus

demo_mode()

i_demo(): initialize demo.

Call: cds_t_pos(&demo_coords,&demo_pos)
Convert coordinates to pos.

demo()

Call: mv_ec_ii((&demo_pos)
&demo_pos: put demo position.

comms():

Exiting demo_mode().

demo(start, end, n)

No

Yes

call: m v_ee_ji((&demo_pos)

Not calibrated cannot.
execute demonstration

set: start, end points
and how many times n.

No

act=active?

Yes

activate()

No

times<n?

Yes

mv_ec_ii(&demo_pos)
call: mv_tp_ii(tp)

No

act?

Yes

deactivate()

return: succ

Figure 9.5: The Tetrabot demo_mode Function
**run_mode**

- **Yes**
  - calib?
  - **calib = FALSE**

- **No**
  - lost?
  - **lost = FALSE**
  - **Yes**
    - **mv_tp_ji()**

- **No**
  - succ?
  - **succ = FALSE**
  - **Yes**
    - **mv_tp_ji()**

**mv_tp_ji**

- **Yes**
  - op_mode
  - **No**
    - **set_op_mode (NORM)**

- **No**
  - active?
  - **Yes**
    - activate()
    - **comms()**
    - deactivate()
    - **clear and exiting run_mode()**
    - **return TRUE**

**mv_ej_ji**

- **set:** $T_B^B$ toolplace wrt base.
- **$T_B^E$:** transform chain from $T_B$ base to toolplate.
- **$T_B^E: T_E^b$:** transform chain from $E^b$ base to end effector.
- **return:** $mv_{tp}_ji(tp)$

**mv_ej_ji**

- checks: calib, active, succ, lost?
- **Yes**
  - **error information**
- **No**
  - copy (tp, rc_target->tplp) pos.c
  - copy to target configuration.
  - tp_t_jts(rc_target->tplp, rc_target->jts_sols)
  - convert target configuration to joints. th8pose.
  - rc_target->jts=op_jisol(rc_curr->jts, rc_target->jts_sols)
  - Select optimum target joint vector for joint interpolation.
  - **OK**
    - **jts_check**
    - **error information**
  - **Not**
    - jts_j_t_vis(): convert joints to vis.
    - **OK**
      - **mv_t_vis()**
      - **Not**
        - **mv_t_vis()** failed!
        - rc_curr = rc_target: update configuration
        - **return TRUE**

---

* Move end_effect to position P in reference coord frame with joint interpolation.

** Move toolplate to transformation matrix, TP in reference coord frame with joint interpolation.

Figure 9.6: The Tetrabot run_mode Function
test() Function

The functions available in test() were essential for development of the controller hardware and software. Many were written in order to solve a particular problem which required access to data, otherwise unavailable. The con.test() functions allow individual axis position controllers to be designed and implemented. The resulting changes in position error can then be monitored with mon.on().

This facility is used for system development and testing. The detail will be discussed in the following section.

9.3 Software Test Tools

There are five kinds of software test tools to test the controller, controller calibration, input/output ports, analogue I/O board and the motor servo amplifiers.

The software test tools have been controlled by the test() function. All pointers to new objects are stored in the automatic array obj[] in test(). Objects are used by the other functions to enter objects. The test() function deletes all such entries before returning.

System configuration commands:

System configuration commands assign new integer values to MMAX and clock time; read the encoder integer value as signed 20-bit numbers; reset all encoders to h/w per-set value; output the encoder values every millisecond; report trip line faults and display joint positions.

Interpolation test commands

The interpolation test commands have been designed as MICCON commands. The user can give absolute interpolation or the interpolation relative to current position.

Software test tools enter five main tests: control test, calibration test, dm531 analogue I/O board test and Inland BHT motor servo amplifiers test.
9.3.1 Controller Test Tools

There are two components in the controller test tools: the controller test commands and the controller configuration commands.

*The controller test commands:*

The controller test commands enter the controller configuration mode, test the interpolation mode, the robot is initialised, the servo amplifiers are enabled/disabled, and the control is monitored.

*The controller configuration commands:*

The controller configuration commands are a very important controller test tool. The controller is selected, assigned, compressed, displayed, reset, activated and deactivated by the controller configuration commands.

9.3.2 Controller Calibration Test Tools

There are a sequential calibration mode and a parallel calibration mode in the calibration test.

*Calibration test commands:*

The calibration axis gets calibration machine units (mu), resets the servo controller and sets up the linear and the wrist axes calibration speed and acceleration in percentage performance. It ensures the controller is active to move to the calibration point, enters calibration mode which resets the encoder when moving off the calibration position (CSO), then resets the servo controller, exits calibration mode for joint *int*, resets the servo controller, sets the park performance, moves to park position and restores the previous active state.

The sequential calibration mode calibrates each joint of the Tetrabot in turn by moving each joint from the initial park position until the falling edge of the function *is_cso()* is detected and then drives that joint back to the park position. The direction of motion to the calibration point in joint coordinates is given by the function *cal_dir[ ]*; the conversion
Chapter 9. Nonlinear Tracking Controller Applied to the Tetrabot Apparatus

180
to direction in \( m/c \) units is given by the function \( \text{mu.dir}[] \).

The parallel calibration mode gets the calibration machine units \( \text{mu} \), resets the servo controller, and activates it if necessary. Function \( \text{cal.axis}(i) \) \((i = 0, 1 \text{ and } 2)\) calibrates the linear axes in series, the function \( \text{setperf}(\text{calv.r}, \text{cala}) \) sets up the wrist axes calibration speed and acceleration in percentage performance and the function \( \text{run}(\text{mv.to.cso},i) \) \((i = 3, 4 \text{ and } 5)\) moves the wrists concurrently to the calibration point. Waiting for processes to complete, the calibration sets the performance for the wrist and sequentially parks all joints together and deactivates them if necessary.

9.3.3 Input/Output Ports Test Tools

The parallel I/O board, the A/D converter board, the servo amplifiers and interface board are initialised, reset and checked by the I/O ports test tools.

9.3.4 Analogue I/O Board Test Tools

*Analogue I/O board test commands:*

The analogue I/O board test commands: Scans the dm531 measurement, scans the dm531 analogue I/O board, reset the dm531 pointers and registers, initialises the dm531 pointers and registers (adc.c) and scans the dm531 amplifier gain.

9.3.5 Motor Servo Amplifiers Test Tools

*Initialisation of Inland BHT motor servo amplifiers*

The initialisation of the Inland BHT motor servo amplifiers including zero control outputs on all axes, disables/enables the servo amplifiers, turns the torque hold on/off, sets the servo-amp cage on/off, and initialises the torque limits to the maximum.

*Inland BHT motor servo amplifiers test commands*

The inland BHT motor servo amplifiers test commands including initialisation of Inland
BHT motor servo amplifiers, zero motor torque limits and control outputs on all axes, output value $v$ to servo D to A on axis $i$, check servo amps and torque hold status, apply/release axis and status brakes, turn torque hold on/off, set servo-amp cage on/off, reset servo amplifiers, and set motor $i$ torque limit.

### 9.4 Theoretical Design Versus Practical Application

Theoretical analysis and computer simulations of a nonlinear tracking controller are important but not sufficient. The ultimate justification of the value and applicability of a nonlinear tracking controller lies in its actual hardware implementation.

In practical implementation, many factors limit the ultimate speed of a manipulator, including the finite capabilities of the robot’s actuators, links and bearings and the use of suboptimal control systems and trajectory planning. Actuator saturation may occur when one of the torques specified by the algorithm reaches the physical limit of the corresponding actuator. Many practical approaches can be used in order to deal with torque saturation. The speed of the desired trajectories may be reduced, thereby reducing the required magnitude of the actuator torques, since saturation typically occurs when the load is too heavy for the given speed and given torque capacity. In the practical system, the plant inputs will be limited by *maximum* and *minimum* allowable values. The Tetrabot experimental facility contains built-in anti-windup protection which will effectively jacket the nonlinear controller. However, this controller does not have inherent problems with wind-up. For appropriately chosen trajectories, the magnitude of the control signal is effectively bounded.

Throughout Chapter 8 the Tetrabot closed-loop system has been simulated assuming that both the joint positions and velocities are measured exactly. To implement the control system described in Chapter 8, it is necessary to know the desired trajectories’ position, velocity and acceleration ($q_R$, $\dot{q}_R$, $\ddot{q}_R$) and the actual position and velocity ($q$, $\dot{q}$). In practice, it is usual to measure the joint position using optical encoders and then estimate the joint velocities from these position measurements.
9.5 Experimental Method

9.5.1 Voltage Controlled DC Motor

Thus far controllers have been designed at the torque input level. Essentially, any dynamics associated with the actuators have been neglected. In this section a systematic approach will be used to compensate for actuator dynamics (3.6) in the form of electrical effects and joint flexibilities.

Substituting equations (8.19), (8.21) and (8.25) into the tracking controller in equation (8.15)

\[ \dot{u} = -\gamma \alpha + \gamma_2 q_R - \rho(t, \dot{q})[\epsilon + \|\mu(t, x)\|^{-1}\mu(t, x)] \] (9.1)

From (8.1), (8.2) and (8.6), the control torque inputs are

\[ \ddot{u} = [D(t, q, \dot{q})]^{-1}[-\gamma \alpha + \gamma_2 q_R - \rho(t, \dot{q})[\epsilon + \|\mu(t, x)\|^{-1}\mu(t, x)] \] (9.2)

According to Chapter 3, now a control input which is the voltage applied to the motor, where the motor’s inductance is ignored, will be developed. This simplification is made since transient effects due to inductance are much faster than the mechanical actions which are to be controlled. Thus, it is assumed that the physical motor cannot respond to the inductive transients.

Utilising equations (3.6), the current through the motor becomes

\[ I_a = \frac{V_d - E_b}{R_m} \] (9.3)

where \(V_d\) is the voltage applied to the DC motor, \(E_b\) is the back EMF generated by the DC motor and \(R_m\) is the resistance of the armature of the DC motor.

Torque is related to current by
\[
\tau_m = K_m I_a \tag{9.4}
\]

The back EMF of the motor results from its acting as a generator and is proportional to rotational velocity:

\[
E_b = K_b \dot{q} \tag{9.5}
\]

The control torque input is \( \tau_c = \tau_m \), so equation (9.4) becomes

\[
\tau_m = K_m \frac{V_d - E_b}{R_m} \tag{9.6}
\]

so that

\[
V_d = \frac{R_m}{K_m} \tau_m + K_b \dot{q} \tag{9.7}
\]

Substituting (9.2) into (9.7), the control voltage inputs are given by

\[
V_d = \Lambda \left\{ -\gamma \alpha + \gamma_2 \dot{q}_R - \rho(t, \dot{q})[\epsilon + \| \mu(t, x) \|]^{-1} \mu(t, x) \right\} + K_b \dot{q} \tag{9.8}
\]

where

\[
\Lambda = \frac{R_m}{K_m} D^{-1}(t, q, \dot{q}) \tag{9.9}
\]

and the parameters of the Tetrabot are given by \( R_{mi} = 0.85 \, \Omega; \, K_{mi} = 0.34 \, Nm/A; \, K_{bi} = 35.6 \, V/KRPM \) and \( i = 1, \ldots, 6 \).

### 9.5.2 Control Law for Experimentation

Equations (9.8) and (9.9) describe a required voltage. However, the control signal, which is an input to the Tetrabot, is required to take the form of a desired velocity. To develop
this signal, consider the Tetrabot dynamic model described in equations (8.1), (8.2) and (8.6).

Substitute the control law (9.2) into the dynamic model (8.1), (8.2) and (8.6) to obtain

\[
M(t, q(t), \omega)\ddot{q}(t) = U(t, q(t), \dot{q}(t), \omega) + [-\gamma \alpha(t, z) + \gamma_2 \ddot{q}_R - \rho(t, \dot{q})[\epsilon + \mu(t, z)]^{-1}\mu(t, z)]
\]  

The second time derivative of rod length \( \ddot{q} \) is thus

\[
\ddot{q}(t) = M^{-1}(t, q(t), \omega)\{U(t, q(t), \dot{q}(t), \omega) + \gamma \alpha(t, z) + \gamma_2 \ddot{q}_R - \rho(t, \dot{q})[\epsilon + \mu(t, z)]^{-1}\mu(t, z)\}
\]  

Let \( \ddot{q} = \frac{\ddot{q}(k+1) - \ddot{q}(k)}{t_s} \), where \( t_s \) is the sample period of the controller, then

\[
\ddot{q}(k + 1) - \ddot{q}(k) = t_s[M^{-1}(t, q(t), \omega)\{U(t, q(t), \dot{q}(t), \omega) + D(t, q, \dot{q})\dot{u}\}]
\]  

Note that Equation (9.12) is a function of the uncertain robot dynamics and is not implementable. However, in view of the robust nonlinear feedback control strategy designed in Chapter 8, it is sufficient to assume a nominal value of \( M \) and \( U \) for implementation. Essentially, the robust nature of the control law (9.12) will overcome the mismatch between these nonlinear dynamics and the actual dynamics in (9.12). The implemented controller thus has the form

\[
\dot{q}(k + 1) - \dot{q}(k) = t_s[M^{-1}(t, q(t), \dot{\omega})\{U(t, q(t), \dot{q}(t), \dot{\omega}) + D(t, q, \dot{q})\dot{u}\}]
\]  

where \( \dot{\omega} \) denotes a specific value of \( \omega \).

The expression for \( \dot{u} \) in equation (9.2) is complicated to implement. In particular, the desired acceleration \( \ddot{q}_R \) is not available in the Tetrabot control software. The signal could
be constructed, but this would only further increase the complexity of the implemented controller. For this reason, the control component $p^2$ is set to zero and the magnitude of the component $p^e$ is scaled to compensate for this. Essentially, the contribution $-\dot{q}_R$ is now considered as a perturbation to the error dynamics whose effects cannot be directly nullified via the control signal but must be accounted for via increasing the magnitude of the robustifier, $p^e$. The tracking controller is now written as

$$
\dot{u} = [D(t, q, \dot{q})]^{-1}[-\gamma \alpha - K_r \rho(t, \dot{q})[\epsilon + ||\mu(t, z)||]^{-1}\mu(t, z)]
$$

(9.14)

where $K_r$ is a positive design parameter. The simulation and experimental results will be described in the section 9.6. The parameters have been selected as in Chapter 8. The controller configuration for real-time control is given by

$$
\text{control}[i] = \dot{q}(k + 1) - \dot{q}(k)
$$

(9.15)

Substituting (9.15) into (9.13), the control input will be

$$
\text{control}[i] = t_s[M^{-1}(t, q(t), \dot{q})\{U(t, q(t), \dot{q}(t), \dot{\omega}) + D(t, q, \dot{q})\dot{u}\}][i]
$$

(9.16)

Substituting (9.14) into (9.16)

$$
\text{control}[i] = t_s[M^{-1}(t, q(t), \dot{q})\{U(t, q(t), \dot{q}(t), \dot{\omega})

- \gamma \alpha - K_r \rho(t, \dot{q})[\epsilon + ||\mu(t, x)||]^{-1}\mu(t, x)\}] [i]
$$

(9.17)

$$
i = 1, 2, 3
$$

where $t_s = 0.005s$ and $\text{control}[i]$ is the control output to the $D$ to $A$ hardware.
9.5.3 Controller Software Interface

The controller interface consists of three parts (Figure 9.7). It is possible to select a PID controller, a nonlinear tracking controller, or an alternative controller. This facility gives the control designer the chance to implement an alternative controller structure. In Chapter 8 only the design of a nonlinear tracking controller for the parallel structure was considered, the wrists were ignored. Based on this analysis and the Tetrabot software, nonlinear tracking controllers for the three parallel rods and PID controllers for the three serial links have been implemented on the actual robot.

9.6 Experimental Results and Discussions

The effectiveness of the proposed nonlinear tracking control law for the Tetrabot will be demonstrated using real-time implementation results.

9.6.1 Numerical Simulation Results

Having obtained A, B, Q and the controller parameters in Chapter 8.3, the closed-loop control system is simulated with the modified controller (9.17) in Chapter 9.5. The purpose of this simulation exercise is to ensure that no severe degradation in performance has resulted from the changes made to the theoretically developed controller in order to facilitate implementation. Consider moving from the initial point $x_{wo} = 0.0, y_{wo} = -0.5 \text{ m}, z_{wo} = -1.475 \text{ m}$ to the final point $x_{wd} = 0.0, y_{wd} = 0.5 \text{ m}, z_{wd} = -0.74 \text{ m}$, with $\dot{x}_{w} = 0.0, \dot{y}_{w} = 0.0, \text{ and } \dot{z}_{w} = 0.0$ at the $W$ point. The desired joint trajectories are defined by fifth order polynomials interpolated between $q[0.0] = [1.4419, 1.3050, 1.5669]^T$ and $q[0.5] = [0.8573, 1.0342, 0.6328]$ with zero desired velocities and accelerations. The desired trajectories are shown in Figure 8.1. This choice of trajectories effectively ensures that the controller parameters are validated across the full range of possible parameter variations in the nonlinear model. When the payload $m = 3 \text{ kg}$, the errors between the desired and actual trajectories, $error_i$, are shown in Figure 9.8-(a). After 2.0 seconds, the tracking errors of rod 1, rod 2 and rod 3 are $3.627 \times 10^{-5} m, -0.894 \times 10^{-5} m$ and $3.560 \times 10^{-5}$ respectively. The actual torques are $0.5358 N m, -0.1344 N m$ and $1.0534 N m$. When the
Chapter 9. Nonlinear Tracking Controller Applied to the Tetrabot Apparatus

Setpoint Control

i_setpoint():
Initialization of setpoint controllers.

con_off():
Ensure controllers are off.

init_def():
Make default controllers:
MVC, PID or Other.

mak_mvc():
mak_pid() mak_other()

make_con():
Select and make a controller.

mak_mvc():
mak_pid() mak_other()

compress():
Compress long error and control scales to short.

ass_def():
Set default controller for each axis.

setpoint():
Setpoint control to build set point follower.

mak_mvc

rods or
wrist?

rods

select: MVC
for three rods

select: PID
for three wrists

Return

Figure 9.7: The Tetrabot Controller Software Interface.
payload $m = 6 \text{ kg}$, the errors of the desired and actual trajectories, $error_i$, are shown in Figure 9.8-(b). After 2.0 seconds, the tracking errors of rod 1, rod 2 and rod 3 are $2.956 \times 10^{-5} \text{ m}$, $-0.928 \times 10^{-5} \text{ m}$ and $4.84 \times 10^{-5} \text{ m}$. The actual torques are $0.5957 \text{ Nm}$, $-0.1860 \text{ Nm}$ and $1.1688 \text{ Nm}$. In both cases the tracking errors satisfy the accuracy requirement and the driving torques satisfy the rated torque limit and the continuous torque limit which are given in Chapter 8.3.

The modified nonlinear tracking controller thus performs very well in simulation.

9.6.2 Experimental Results and Discussions

The Lyapunov-based Nonlinear Tracking Controller (NTC) has been implemented on the Tetrabot computer control system for real-time control. To demonstrate the performance attainable by this control strategy, the trajectory involved movement across the primary working volume from the end-effect point (0.5, 0.5, -1.5)m to the end-effect point (-0.4, -0.4, -0.8)m which is the largest distance possible within the primary working volume and involved the continuous motion; such a motion will invoke a wide range of the possible nonlinear dynamic representations. The NTC has been developed and tuned by the Tetrabot control software test tools that can access a set of variables to display the information of the system: the parameters, the error scales, the control scales and the actual positions in real time. In the experiments, the design parameters of each controller are tuned to their best values by the control test tools and the tracking accuracy can be improved by increasing the magnitudes of $\kappa_i$ and $\rho_i$. We assumed the upper bound $m = 6 \text{ kg}$, the parameter values used for the NTC method are $\sigma = 0.25$, $\bar{\sigma} = 3.4450$, $\sigma = 1.2183$, $\gamma = 1.378$, $\gamma_2 = 1.7999$, $\kappa_1 = 2.8186$, $\kappa_2 = 4.1042$, $\kappa_3 = 4.7745$, $\rho_1 = 9.7090$, $\rho_2 = 1.4137$, $\rho_3 = 1.6446$, $p_{21} = 0.7071$ and $p_{22} = 0.9239$. The purpose of the experiments is to demonstrate the achievable stability and performance characteristics of the controller and furthermore to compare the performance of the nonlinear tracking controllers with that of the existing PID controllers. The parameter values used for the PID are the same as the original system. The relative desired joint trajectories are shown in Figure 9.9.

Two sets of experimental results are presented below. In the experiments the Tetrabot
Chapter 9. Nonlinear Tracking Controller Applied to the Tetrabot Apparatus

Figure 9.8: (a) The rod tracking errors with the payload at the lower bound $m=3\text{Kg}$ and (b) The upper bound $m=6\text{Kg}$.
was controlled to follow the reference trajectory at 30 percent of its maximum velocity and acceleration. Trajectories are generated on-line by the Tetrabot computer control system and ensure the end point is attained with minimum derivation from the desired speed in the transient stage, it was decided to use this commercially developed tracking, software and associated trajectories rather than academically chosen trajectories. This was performed with the controller as given in equation (9.17). A comparative test was run using first the original PID controllers then the NTC for the same trajectory. To give a measure of the relative performances, the maximum positional tracking errors under calibration for both the NTC and the PID controller are given in Table 9.1. The maximum initial calibration tracking errors for the PID system are $-0.015mm$ and $0.005mm$, and for the NTC system are $-0.005mm$ and $0.005mm$. The NTC and PID systems satisfy the absolute accuracy requirement for initial calibration ($\pm 0.05mm$). In Table 9.1 the maximum on-line tracking errors for the PID system are $-0.715mm$ and $0.315mm$, and for the NTC system are $-0.035mm$ and $0.08mm$. In both cases the tracking errors satisfy the absolute on-line accuracy requirement ($\pm 1.00mm$). The on-line tracking errors of the NTC and PID controllers are shown in Figure 9.10. Using the NTC, the position errors were smaller than those obtained using the original PID controllers. The NTC showed excellent results.

9.7 Concluding Remarks

Careful experiment of design yielding informative results is the basis of a successful validation experiment. In this chapter, we have discussed the Tetrabot computer-controlled system. The software test tools, and how the experiment was arranged based on the theoretical analysis, available hardware, the software design methodology and practical considerations discussed.

It details the software test tools including the controller, the controller calibration, the interpolation, I/O and the motor servo amplifiers of the test tools. A controller is developed and tuned using these software test tools. Theoretical analysis and computer simulations of the nonlinear tracking controller are important but not sufficient, because the value and applicability of a nonlinear tracking controller lies in its actual hardware
Desired Trajectories of Rod for PID, NTC(—)

Figure 9.9: The desired joint trajectories.
Figure 9.10: The rod tracking errors with PID and NTC.
implementation. The Lyapunov-based robust tracking controller, the designing of the controller software interface and the off-line program for the parameter estimation, has set up the experimental methods.

Section 9.6 describes the results of the implementation of a nonlinear tracking controller on a Tetrabot rig facility. The tracking controller parameters have been estimated by off-line simulation based programs and have been tuned during numerical simulation and implementation. The proposed control strategy is robust to variations in robot loading. The experimental results obtained for the closed-loop response indicate that compensation which employs explicit off-line parameter estimation can improve tracking accuracy significantly. The performance of the tracking controller in real-time is good, but not as good as predicted by the numerical simulation studies in Chapter 8. The fact that the simulation model has yet to be validated might account for the differences.
Chapter 10

Conclusions and Further Studies

10.1 Conclusions

This thesis has discussed the development of robust nonlinear tracking controllers for robotic manipulators. The design of robust tracking controllers involves deriving a tracking law for uncertain dynamic systems, such that the actual positions closely track desired trajectories. The foci of this study are the concepts and techniques of robust nonlinear tracking control with a bias toward industrial applications.

Two new schemes, a robust sliding mode control and a Lyapunov-based robust tracking control, have been presented for uncertain dynamical systems in the presence of model uncertainty and disturbances.

Based on the robust sliding mode control methodology, the control scheme addresses the following problem: given the extent of parametric uncertainty and external uncertainties, design a nonlinear sliding mode controller to achieve robust tracking precision. The robust controllers are derived based on the Fliess' generalised controller canonical form (GCCF) ([36, 37]). The GCCF is straightforward to obtain from the system of differential equations describing the manipulator dynamics. For the class of system to which it applies, sliding controller design provides a systematic approach to the problem of maintaining stability and consistent performance. By allowing trade-offs between modelling and performance to be quantified in a simple way, sliding mode control is very appropriate for the application. This approach has been illustrated using two case studies which relate
to elastic joints and rigid manipulators with \( n \) degrees of freedom. The function \( \| \hat{\Delta} \| \) is a nonlinear function of the state, and bounds the maximum size of the overall uncertainties. The design of the robust tracking controller involves selection of \( u \) and the bounding function \( \| \hat{\Delta} \| \) to achieve stability and good tracking performance. The relative performance is studied using simulation models of a two-link flexible joint robot in Chapter 5 and a three DOF rigid robot in Chapter 7.

Dynamic sliding mode control remedies the defects of high frequencies chattering. Dynamic sliding mode control combines the advantage of dynamic feedback control and sliding mode techniques in the controller design phase while simultaneously asymptotically linearising the nonlinear system. The modified \( n \)-link uncertain elastic manipulator has been investigated for the case when the system becomes relative degree two and the GCCF involves second order time derivatives of the control input torque. The approach has led to a dynamical linearising controller and also allows for the design of dynamical sliding mode control schemes based on the output function. For robot systems with dynamical sliding mode control, the same closed loop poles are chosen as in the feedback linearising controller but better performance is achieved. Small gains are chosen in the dynamic sliding mode controller (6.31) to minimise energy but excellent tracking performance is achieved. The relative performance is studied using a two-link flexible joint robot. Simulation results showed excellent robust performance.

Using a Lyapunov function and specified magnitude bounds on the uncertainties, a nonlinear control law is developed to ensure uniform ultimate boundedness of the closed-loop feedback trajectory to achieve sufficient accuracy. The resulting controller is a discontinuous control function with generally continuous control in a boundary layer which prevents the excitation of high frequency unmodelled parasitic dynamics. The development of Lyapunov-based robust tracking controllers for the parallel and serial topology Tetrabot is considered, although the theoretical control strategy presented is applicable to any manipulator tracking problem. For a class of desired state motions a control strategy is expounded which ensures that the system asymptotically tracks the desired motion to any desired degree of accuracy. The Tetrabot model was previously developed and discussed in Chapter 4. Simulation studies are used to parameterise the controller. The nonlinear tracking control for the Tetrabot is developed and simulated in Section 8.3.
Chapter 10. Conclusions and Further Studies

Importantly, the results of implementation on the three DOF parallel geometry are described in Chapter 9.

In implementing the proposed tracking control scheme on a six degree of freedom robot, the Tetrabot, it has been found that

- Robot kinematics deals with the analytical study of the geometry of motion of a robot arm with respect to a fixed reference coordinate system as a function of time without regard to the forces/moments that cause the motion. The forward kinematics solution computes the forward kinematics from the joint space (base coordinate) to the intermediate task space then to the toolplate. The inverse kinematics solution involves deriving the joint coordinates for any specified position and orientation of toolplate referred to a base coordinate frame with origin located at base point and fixed with respect to the support structure. Robot kinematics uses a $4 \times 4$ homogeneous transformation matrix to describe the spatial relationship between two adjacent rigid mechanical links and to reduce the forward kinematics to finding an equivalent $4 \times 4$ homogeneous transformation matrix that relates the spatial displacement of the hand coordinate frame to the reference coordinate frame.

- The kinematics of the three serial wrist links is decoupled from the three upper parallel rods, and for a given task the torques produced by the rod driving systems are generally much greater than those for the wrists. The proposed tracking control scheme design was based on the three parallel structure, ignoring the wrist structure. This means we only have three nonlinear tracking controllers for three parallel rods but do not have same controllers for three serial links. The Tetrabot must be controlled by six independent controllers in the same time.

- The nonlinear tracking controllers for three parallel rods and the PID controllers for three serial links have been implemented on the real task, which are based on Chapter 4: Analysis of the Tetrabot System and Section 8.5: Experimental Methods.

- The controller interface consists of three parts in the Tetrabot control software. The PID controller, the Multi Variable Controller (MVC), and other controller are selectable. The nonlinear tracking controllers for three rods and the PID controller for three wrists are selected by the MVC function.
• The trajectory interpolation used is the Linear Function with Parabolic Blends (LFPB) trajectory. The modified interpolator program has been used by the non-linear tracking controller.

• The tracking controller parameters have been estimated by off-line simulation based programs and have been tuned during numerical simulation and implementation. The proposed control strategy is robust to variations in robot loading. The experimental results obtained for the closed-loop response indicate that compensation which employs explicit off-line parameter estimation can improve tracking accuracy significantly.

• The Tetrabot system software has been further developed for implementation of the nonlinear controller. The nonlinear tracking controller is implemented, developed and tuned by software test tools including the controller, the controller calibration, the interpolation, I/O and the motor servo amplifiers of the test tools. To tune the nonlinear controller much more information is needed from the Tetrabot system. The desired trajectories, actual trajectories and trajectory errors are very important information to tune the nonlinear controller. These information have been extracted, stored and displayed in real time. This information has been captured by the PC and transferred to MATLAB to analyse the control performance.

• Theoretical analysis and computer simulations of the nonlinear tracking controller are important but not sufficient, because the value and applicability of a nonlinear tracking controller lies in its actual hardware implementation. The Lyapunov-based robust tracking controller, the designing of the controller software interface and the off-line program for the parameter estimation, has set up the experimental methods. The performance of the tracking controller in real-time is good, but not as good as predicted by the numerical simulation studies in Chapter 8. The fact that the simulation model has yet to be validated might account for the differences.

• The Nonlinear Tracking Controller (NTC) has been implemented on the Tetrabot computer control system for real-time control. To demonstrate the performance attainable by this control strategy, the trajectory involved movement across the primary working volume from the end-effect point (0.5, 0.5, -1.5)m to the end-effect point (-0.4, -0.4, -0.8)m which is the largest distance possible within the primary
working volume and involved the continuous motion; such a motion will invoke a wide range of the possible nonlinear dynamic representations. The NTC has been developed and tuned by the Tetrabot control software test tools that can access a set of variables to display the information of the system: the parameters, the error scales, the control scales and the actual positions in real time. Using the NTC, the position errors were smaller than those obtained using the original PID controllers. The NTC showed excellent results.

- Robust tracking via dynamic sliding mode techniques has been studied in Chapter 6 this shows excellent robust performance on the simulation but one would be implemented because the available at present Tetrabot system does not have suitable software and hardware.

10.2 Further Studies

Some important issues have been raised following the design of the robust tracking control methodology and the implementation on the computer-control Tetrabot system. Further works will include:

- In this thesis, a Lyapunov theory-based tracking controller has been simulated and applied into the computer-control Tetrabot system. Flexible joints and rigid manipulators have been simulated with robust tracking control via sliding mode. Several individual developmental methods have been used in this thesis: the Tetrabot mathematical model; the off-line program for the parameter estimation and the data transform interface (between the computer-control Tetrabot system and MATLAB) developed in MATLAB. The Tetrabot system software, the software test tools and the Tetrabot control software have been modified on the computer-control Tetrabot system (using C).

Throughout this thesis we can see that the Tetrabot system is a very good example for the study of control of mechanical systems but it is very difficult to develop and tune controllers on the simulation for application on the computer-controlled Tetrabot. One direction of further study is an Integrated control method (ICM)
for the Tetrabot system. It would integrate the simulation and the application of the control. The general processing software of the computer-controlled Tetrabot is written in C and MATLAB with interfaced to C. The ICM software structure considered here would a three-level hierarchy:

1. **Top-level**: The management level including MICCON and MATLAB which provides for the supervision of lower-level functions and for managing the interface to humans. In particular, the management level will interact between the users and the controller and in assessing capabilities of the system. The management level also monitors performance of the lower-level systems, robot activities at the highest level and performs high-level leaning about the user and the lower-lever algorithms.

2. **Mid-level**: The coordination level provides for tuning, scheduling, supervision and redesign of the low-level algorithms, planning capabilities for the coordination of low-level tasks and top-level symbolic decision making and control algorithm management. There are two parts in the mid-level: MATLAB deals with the Tetrabot model, the parameter estimation, the analysis of the data from the low-level and C deals with the general process software to tune the computer-controlled Tetrabot system tests.

3. **Low-level**: The execution level has signal processing and control algorithms, mainly dealing with real time control.

A major advantage of using ICM is that the management level integrates the control method of the Tetrabot system with the friendly user interface, so people can use IMC to develop and tune controllers without requiring knowledge of the complicated Tetrabot model or/and many hundreds of software functions.

- The Tetrabot system software has been further developed to allow the implementation of the nonlinear controller. To tune and analyse the nonlinear controller much more information is needed from the Tetrabot system. The desired trajectories, actual trajectories and trajectory errors are very important information to tune the nonlinear controller. These information have be extracted, stored and displayed in real time but are limited by the memory of the Tetrabot controller.

Suitable hardware is needed to support the Integrated control method (ICM) for the Tetrabot system. The Tetrabot uses a multibus I 68000 SBC (Single Board
Computer) as the built-in part of the Tetrabot controller including a digital I/O interface, an analogue input and an expansion I/O interface. It is very difficult to upgrade 68000 SBC to a PC system as a built-in part, but if we keep the hardware of the Tetrabot controller unchanged, develop the software of the Tetrabot controller and use the PC system as a monitoring station and a host station then a new computer-control Tetrabot system will fully support the ICM.

The 68000 SBC memory map shows a 1024K bytes RAM in total, extending from the start of memory up to 07FFFFFF (512K bytes) which is loaded by the Tetrabot control software at address 00100H, while the address space from 080000H to FFFFFFFH (320K bytes) is unused. This 320K bytes can be used as a buffer area in the host station for the Tetrabot information extracted and stored in real time.

The 68000 SBC has a VERSBUG with a 16K-bytes ROM monitor/debugger program (FE0000H to FE3FFFH), but 48K-bytes ROM (FE4000H to FEFFFFH) is unused which can be programmed to extract and store information (the state, control and demand variables of the system) and the communications to/from the host station. This program will only run in the background and store information in machine units to wait for data processing and send the information to the host station. VERSBUG allows access to the SBC’s memory and CPU registers as well as provide basic communications and debugging facilities.

The host station receives the information from the Tetrabot, then a) changing the machine units into the normal data; b) storing as a MATLAB mat file structure; c) plotting the performance of the control in real-time.

- The Tetrabot nonlinear control has been implemented on the Tetrabot, but the result is not as good as the simulation result in Chapter 8, because the nonlinear controller needs to access a set of variables to effect their control. So far only the desired position, the actual position, the position error and the velocity error can be identified, ignoring desired acceleration and some heavy computational requirements. More variables and computation is needed to achieve perfect performance for the real task.

- For joints velocity estimation from position measurements, it is usual to measure the joint positions using optical encoders, and then estimate the joint velocities from
these position measurements. In Chapter 8 simply computing the joint velocities using the Euler approximation: \( \dot{q}_k = (q_k - q_{k-1})/T \) is virtually doomed to failure, since this high-pass filter amplifies the encoder measurement noise. The velocity estimation filter design can be optimised for the given encoder noise statistics by using an alpha-beta tracking to reconstruct the joint velocity estimation \( v_k \) [48] and [49]. This is a specialized form of Kalman filter. Also other estimation methods via a variable structure control and an adaptive control [83, 12, 11, 13, 109, 32] would solve the implementation problems.
Appendix A

Tetrabot Parameters

A.1 Geometric Measures (Figure 4.2)

\[ z_1 = 0.5m \] distance from the centre of the base of the equilateral triangle to the vertices.

\[ z_2 = 0.0m \] distance from the centre of the base of the equilateral triangle to the upper pivot point of the central tube.

\[ z_3 = 0.205m \] distance from the lower point \( P \) to the centre \( W \) of the wrist axes.

\[ z_4 + z_5 = 0.105m \] distance from the centre \( W \) of the wrist axes to the point \( T \).

\[ z_{6\text{max}} = 1.585m \] Maximum actuator rod \( i \) length.

\[ z_{6\text{min}} = 0.575m \] Minimum actuator rod \( i \) length.

A.2 Masses, Moments of Inertia

\[ m_i = 8.75kg \quad \text{\( i = 1, 2, 3 \). Rod mass.} \]

\[ m_4 = 19.0kg \quad \text{Rod 4 mass.} \]

\[ m_p = 3\ 6kg \quad \text{Total payload, gripper+payload.} \]

\[ I_i = 2.6133Nm \quad \text{\( i = 1, 2, 3 \). Rod moment of inertia.} \]

\[ I_4 = 5.6161Nm \quad \text{Moment of inertia of central tube.} \]

\[ r_i = 0.9779m \quad \text{\( i = 1, 2, 3 \). Distance from the COM of the rod to the datum point \( P \).} \]
Appendix A. Tetrabot Parameters

$r_4 = -0.2670m$ Distance from the COM of the rod to the datum point $P$.

A.3 Driving System Parameters

$L = -\frac{0.025}{2\pi} m$ Lead of ballscrew, metres/revolution.

$\eta = 0.95$ Efficiency of ballscrew.

$N = 0.8$ Motor to nut drive ratio, nut ratio/motor rotation.

$I_m = 0.00028Nm$ Driveline moment of inertia of motor assembly.

$I_n = 0.000775Nm$ Driveline moment of inertia of nut assembly.
Appendix B

Tetrabot Control Software in C

B.1 The Header Files of the Tetrabot Control software (Table B.1)

<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bitops.h</td>
<td>Bit operation macros</td>
</tr>
<tr>
<td>commdefs.h</td>
<td>Common macro definitions</td>
</tr>
<tr>
<td>errorflags.h</td>
<td>Error flag name</td>
</tr>
<tr>
<td>genflags.h</td>
<td>Definitions for flag values</td>
</tr>
<tr>
<td>rbt_config.h</td>
<td>Robot configuration data structure</td>
</tr>
<tr>
<td>tlfthpe.h</td>
<td>Type definitions for trip line fault handling</td>
</tr>
<tr>
<td>vectortype.h</td>
<td>Vector types</td>
</tr>
</tbody>
</table>

Table B.1: The header files of Tetrabot control software

B.2 Tetrabot Controller Software (Tables B.2 and B.3)
<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>activate.c</td>
<td>Activate/deactivate robot once initialized.</td>
</tr>
<tr>
<td>adc.c</td>
<td>Interface routines for dms531 analog I/O board.</td>
</tr>
<tr>
<td>adc.test.c</td>
<td>MICCON i/f for testing of dms531 analogue board.</td>
</tr>
<tr>
<td>calibrate.c</td>
<td>Tetrabot the sequential and parallel calibration.</td>
</tr>
<tr>
<td>cal.test.c</td>
<td>MICCON driven calibration testing.</td>
</tr>
<tr>
<td>common.c</td>
<td>Controller monitoring.</td>
</tr>
<tr>
<td>contest.c</td>
<td>Controller testing.</td>
</tr>
<tr>
<td>control.c</td>
<td>Controller control functions.</td>
</tr>
<tr>
<td>demo.c</td>
<td>Tetrabot demonstration.</td>
</tr>
<tr>
<td>drive.c</td>
<td>Drive robot axes.</td>
</tr>
<tr>
<td>encoder.c</td>
<td>Encoder functions.</td>
</tr>
<tr>
<td>error.c</td>
<td>Centralised error condition handling.</td>
</tr>
<tr>
<td>errservice.c</td>
<td>Error servicing routines.</td>
</tr>
<tr>
<td>initio.c</td>
<td>Initialise/reset all input/output devices.</td>
</tr>
<tr>
<td>interp.c</td>
<td>Generate interpolation function.</td>
</tr>
<tr>
<td>interp_test.c</td>
<td>Interpolation testing.</td>
</tr>
<tr>
<td>mainlin.c</td>
<td>Tetrabot main library.</td>
</tr>
<tr>
<td>maintest2.c</td>
<td>Single axis test version of Tetrabot main program.</td>
</tr>
</tbody>
</table>

Table B.2: The Tetrabot control software – 1
<table>
<thead>
<tr>
<th>Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>manual.c</td>
<td>Tetrabot manual move using VDU keys.</td>
</tr>
<tr>
<td>misc.c</td>
<td>Miscellaneous functions.</td>
</tr>
<tr>
<td>nmask.c</td>
<td>Non maskable interrupt action.</td>
</tr>
<tr>
<td>pos.c</td>
<td>Position (ie. 4x4 hom. transformation) functions.</td>
</tr>
<tr>
<td>servoio.c</td>
<td>Interface functions to Tetrabot servo amplifiers.</td>
</tr>
<tr>
<td>setcontrol.c</td>
<td>Create controller function.</td>
</tr>
<tr>
<td>tbot1io.c</td>
<td>I/O functions for Tetrabot mk1 hardware.</td>
</tr>
<tr>
<td>tbotpos002.c</td>
<td>Kinematic and position calculation functions.</td>
</tr>
<tr>
<td>temmpmon.c</td>
<td>Tetrabot temperature monitoring.</td>
</tr>
<tr>
<td>test.c</td>
<td>Enter test functions running under miccon.</td>
</tr>
<tr>
<td>tlfault.c</td>
<td>Trip line fault (tlf) handling and reporting.</td>
</tr>
<tr>
<td>usr.c</td>
<td>Application specific functions executed outside normal process scheduling.</td>
</tr>
<tr>
<td>util.c</td>
<td>Miscellaneous utility functions.</td>
</tr>
<tr>
<td>vectops.c</td>
<td>Operations on 3-vectors.</td>
</tr>
</tbody>
</table>

Table B.3: The Tetrabot control software – 2
Appendix C

Tetrabot Control Functions

C.1 Tetrabot Controller Functions (Tables C.1 - C.10)

<table>
<thead>
<tr>
<th>Description</th>
<th>File</th>
</tr>
</thead>
<tbody>
<tr>
<td>activate()</td>
<td>activate.c</td>
</tr>
<tr>
<td>adc_test()</td>
<td>adc_test.c</td>
</tr>
<tr>
<td>amp_enable()</td>
<td>tbotio.c</td>
</tr>
<tr>
<td>amp_reset()</td>
<td>tbotio.c</td>
</tr>
<tr>
<td>angles_123()</td>
<td>pos.c</td>
</tr>
<tr>
<td>angles_323()</td>
<td>pos.c</td>
</tr>
<tr>
<td>angles()</td>
<td>pos.c</td>
</tr>
<tr>
<td>ass_con(n, c)</td>
<td>control.c</td>
</tr>
<tr>
<td>ass_def(c)</td>
<td>control.c</td>
</tr>
<tr>
<td>ave_adc1(n)</td>
<td>adc.c</td>
</tr>
<tr>
<td>brakes(status)</td>
<td>tbotio.c</td>
</tr>
<tr>
<td>btst(n, i)</td>
<td>tlfault.c</td>
</tr>
<tr>
<td>cage(status)</td>
<td>tbotio.c</td>
</tr>
<tr>
<td>cal(n, status)</td>
<td>tbotio.c</td>
</tr>
<tr>
<td>calibrate()</td>
<td>calibrate.c</td>
</tr>
</tbody>
</table>

Table C.1: The functions - 1
### Appendix C. Tetrabot Control Functions

<table>
<thead>
<tr>
<th>Description</th>
<th>File</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>cal_axis(i)</code> Calibrate axis <code>i</code>.</td>
<td><code>calibrate.c</code></td>
</tr>
<tr>
<td><code>cal_verify()</code> Verify calibration.</td>
<td><code>calibrate.c</code></td>
</tr>
<tr>
<td><code>cal_test()</code> MICCON driven calibration testing.</td>
<td><code>calibrate.c</code></td>
</tr>
<tr>
<td><code>cds_t_pos()</code> convert coordinates to pos</td>
<td><code>pos.c</code></td>
</tr>
<tr>
<td><code>cdtoa_write</code> Write value v to motor current limit dtoa on axis n.</td>
<td><code>tbotio.c</code></td>
</tr>
<tr>
<td><code>cf_enter()</code> Enter C functions in dictionary.</td>
<td><code>mainlib.c</code></td>
</tr>
<tr>
<td><code>check(i)</code> Check `(i &gt;= 0</td>
<td></td>
</tr>
<tr>
<td><code>che_con(n)</code> Check legality of controller 'n'.</td>
<td><code>control.c</code></td>
</tr>
<tr>
<td><code>che_chan(c)</code> Check legality of channel 'c'.</td>
<td><code>control.c</code></td>
</tr>
<tr>
<td><code>chk_enc()</code> Check that encoder value.</td>
<td><code>calibrate.c</code></td>
</tr>
<tr>
<td><code>che_error(n)</code> Check that 'n' is a legal error no.</td>
<td><code>error.c</code></td>
</tr>
<tr>
<td><code>clip</code> Clip int bv to range <code>1 &lt;= v &lt;= h</code>.</td>
<td><code>misc.c</code></td>
</tr>
<tr>
<td><code>clr_con(n)</code> Zero all parameters of controller <code>n</code>.</td>
<td><code>control.c</code></td>
</tr>
<tr>
<td><code>clr_error()</code> Clear error.</td>
<td><code>error.c</code></td>
</tr>
<tr>
<td><code>clr_tlf()</code> Clear trip line fault latches.</td>
<td><code>tbotio.c</code></td>
</tr>
<tr>
<td><code>(*common())()</code> Real time monitoring and checking of controller action.</td>
<td><code>common.c</code></td>
</tr>
<tr>
<td><code>compress(escale, cscale,es,cs)</code> Convert <code>escale[]</code> to <code>es[]</code> and <code>cscale[]</code> to <code>cs[]</code>.</td>
<td><code>control.c</code></td>
</tr>
<tr>
<td><code>comp_con(n)</code> Compress controller 'n'.</td>
<td><code>control.c</code></td>
</tr>
</tbody>
</table>

Table C.2: The functions – 2
<table>
<thead>
<tr>
<th>Description</th>
<th>File</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>config_con()</code></td>
<td>Enter controller configuration mode.</td>
</tr>
<tr>
<td><code>con_on()</code></td>
<td>Activate all controller channels.</td>
</tr>
<tr>
<td><code>con_off()</code></td>
<td>Deactivate all controller channels.</td>
</tr>
<tr>
<td><code>con_reset()</code></td>
<td>Reset controller.</td>
</tr>
<tr>
<td><code>con_test()</code></td>
<td>Controller test.</td>
</tr>
<tr>
<td><code>copy(a, b)</code></td>
<td>Copy posn a to b.</td>
</tr>
<tr>
<td><code>da_enter()</code></td>
<td>Enter data in dictionary</td>
</tr>
<tr>
<td><code>dclip(v, l, h)</code></td>
<td>Clip double v to range l &lt;= v &lt;= h.</td>
</tr>
<tr>
<td><code>deactivate()</code></td>
<td>Deactivate robot once initialized.</td>
</tr>
<tr>
<td><code>demo(str, end, n)</code></td>
<td>Execute demo moves from start to park posn via end n times.</td>
</tr>
<tr>
<td><code>disp_pos()</code></td>
<td>Display actual positions.</td>
</tr>
<tr>
<td><code>demo_mode()</code></td>
<td>Enter demonstration mode.</td>
</tr>
<tr>
<td><code>drive()</code></td>
<td>Local drive process.</td>
</tr>
<tr>
<td><code>drive_joint()</code></td>
<td>Drive joint n in direction d with v.</td>
</tr>
<tr>
<td><code>dump()</code></td>
<td>Enable robot run.</td>
</tr>
<tr>
<td><code>encr_J(n)</code></td>
<td>Read encoder value n as signed 20 bit.</td>
</tr>
<tr>
<td><code>encoder_test(n)</code></td>
<td>Test encoder 'n'.</td>
</tr>
<tr>
<td><code>encs_reset()</code></td>
<td>Rest all encoders to h/w preset value.</td>
</tr>
<tr>
<td><code>enc.dis(p)</code></td>
<td>Output encoder values every p ms.</td>
</tr>
<tr>
<td><code>(*errvect[])</code></td>
<td>Table of error servicing functions</td>
</tr>
<tr>
<td><code>get_tlf()</code></td>
<td>Get trip line fault status.</td>
</tr>
<tr>
<td><code>help()</code></td>
<td>On line help for single axis moves using vdu keys</td>
</tr>
<tr>
<td><code>hold_con()</code></td>
<td>Hold control at current actual posn.</td>
</tr>
<tr>
<td><code>get_adc1(n)</code></td>
<td>Initiate A/D conversion on channel n.</td>
</tr>
</tbody>
</table>

Table C.3: The functions – 3
### Appendix C. Tetrabot Control Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>File</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>get_temps()</code></td>
<td>Get temperature form acd.</td>
<td><code>temmpmon.c</code></td>
</tr>
<tr>
<td><code>iabs(i)</code></td>
<td>Integer absolute.</td>
<td><code>nmask.c</code></td>
</tr>
<tr>
<td><code>ifbd_select()</code></td>
<td>Select i/f board n with decode mode m.</td>
<td><code>tbotlio.c</code></td>
</tr>
<tr>
<td><code>init_def()</code></td>
<td>Init default controller parameters.</td>
<td><code>control.c</code></td>
</tr>
<tr>
<td><code>init_io()</code></td>
<td>Init parallel I/O board and analogue to digital converter board</td>
<td><code>initio.c</code></td>
</tr>
<tr>
<td><code>init_pio()</code></td>
<td>Initialize parallel i/o board.</td>
<td><code>tbotlio.c</code></td>
</tr>
<tr>
<td><code>init_guard()</code></td>
<td>special version of reset guard.</td>
<td><code>tbotlio.c</code></td>
</tr>
<tr>
<td><code>init_servos()</code></td>
<td>Initialise BHT motor servo amplifiers.</td>
<td><code>servoio.c</code></td>
</tr>
<tr>
<td><code>init_usr()</code></td>
<td>Startup initialization.</td>
<td><code>usr.c</code></td>
</tr>
<tr>
<td><code>inv(a)</code></td>
<td>invert posn</td>
<td><code>pos.c</code></td>
</tr>
<tr>
<td><code>in_error()</code></td>
<td>Check if an error has been flagged.</td>
<td><code>error.c</code></td>
</tr>
<tr>
<td><code>i_error()</code></td>
<td>Initialise error handling.</td>
<td><code>error.c</code></td>
</tr>
<tr>
<td><code>i_demo()</code></td>
<td>Initialize demonstration sequence.</td>
<td><code>demo.c</code></td>
</tr>
<tr>
<td><code>(*i_interp())()</code></td>
<td>Initialise interpolator.</td>
<td><code>interp.c</code></td>
</tr>
<tr>
<td><code>i_mainlib()</code></td>
<td>Initialization mainlib.</td>
<td><code>mainlib.c</code></td>
</tr>
<tr>
<td><code>i_pk_rc()</code></td>
<td>Initialize park configuration.</td>
<td><code>mainlib.c</code></td>
</tr>
<tr>
<td><code>(*i_setpoint())()</code></td>
<td>Initialization of setpoint controllers.</td>
<td><code>control.c</code></td>
</tr>
<tr>
<td><code>i_tbotpos()</code></td>
<td>Initialise tetrabot geometry, kinematics.</td>
<td><code>tbotpos002.c</code></td>
</tr>
<tr>
<td><code>i_tqlim()</code></td>
<td>Initialise torque limits to max</td>
<td><code>servoio.c</code></td>
</tr>
<tr>
<td><code>(*interp())()</code></td>
<td>Build interpolator.</td>
<td><code>interp.c</code></td>
</tr>
<tr>
<td><code>intp_abs(p0, p1,p2,p3,p4,p5)</code></td>
<td>Absolute interpolation, function designed as a miccon command</td>
<td><code>interp.c</code></td>
</tr>
<tr>
<td><code>intp_rel(a0, a1,a2,a3,a4,a5)</code></td>
<td>Interp relative to current position, function designed as a miccon command</td>
<td><code>interp.c</code></td>
</tr>
<tr>
<td><code>intp_test()</code></td>
<td>Interpolation test.</td>
<td><code>interp.c</code></td>
</tr>
</tbody>
</table>

Table C.4: The functions — 4
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>File</th>
</tr>
</thead>
<tbody>
<tr>
<td>io.test()</td>
<td>Test I/O. Functions in tbotlio.c.</td>
<td>test.c</td>
</tr>
<tr>
<td>is.ack()</td>
<td>Check for !ACK on port 9</td>
<td>tbotlio.c</td>
</tr>
<tr>
<td>is.cso(n)</td>
<td>Check for CSO, calib signal on axis 'i'.</td>
<td>tbotlio.c</td>
</tr>
<tr>
<td>is.err(n)</td>
<td>Check for !ERR, interface board n counter overflow.</td>
<td>tbotlio.c</td>
</tr>
<tr>
<td>is.aok()</td>
<td>Check for AOK, servo amplifiers ok.</td>
<td>tbotlio.c</td>
</tr>
<tr>
<td>is.braked()</td>
<td>Status of brakes.</td>
<td>tbotlio.c</td>
</tr>
<tr>
<td>is.tq.hold()</td>
<td>Check torque hold status.</td>
<td>tbotlio.c</td>
</tr>
<tr>
<td>jts.t.mu()</td>
<td>Converts joint coords (mm, rads) to mu.</td>
<td>tbotpos002.c</td>
</tr>
<tr>
<td>jts.t.mu()</td>
<td>Convert coord value p on axis n to mu.</td>
<td>tbotpos002.c</td>
</tr>
<tr>
<td>jts.check(jts)</td>
<td>Check against lower and upper limits.</td>
<td>tbotpos002.c</td>
</tr>
<tr>
<td>jts.check(n,p)</td>
<td>Check joint 'n' (0...5) coord value 'p' against limits.</td>
<td>tbotpos002.c</td>
</tr>
<tr>
<td>jts.t.tp()</td>
<td>Convert joint coords to toolplate transformation matrix</td>
<td>tbotpos002.c</td>
</tr>
<tr>
<td>kill.tmon()</td>
<td>Kill temperature monitoring process.</td>
<td>temmpmon.c</td>
</tr>
<tr>
<td>len(a)</td>
<td>determine length of posn vector of a.</td>
<td>pos.c</td>
</tr>
<tr>
<td>mac.enter()</td>
<td>Enter macros in dictionary</td>
<td>mainlib.c</td>
</tr>
<tr>
<td>main()</td>
<td>Single axis test main prog.</td>
<td>maintest2.c</td>
</tr>
<tr>
<td>mak.pid(n)</td>
<td>Make discrete pid controller 'n'.</td>
<td>control.c</td>
</tr>
<tr>
<td>mak.con(n)</td>
<td>Make controller n.</td>
<td>control.c</td>
</tr>
<tr>
<td>mak.corless(n)</td>
<td>Make discrete nonlinear controller.</td>
<td>control.c</td>
</tr>
<tr>
<td>mak.other(n)</td>
<td>Make discrete other controller 'n'.</td>
<td>control.c</td>
</tr>
<tr>
<td>manual()</td>
<td></td>
<td>manual.c</td>
</tr>
<tr>
<td>mm.t.jts()</td>
<td>Converts mu to joint coords(mm, rads).</td>
<td>tbotpos002.c</td>
</tr>
<tr>
<td>mon.off()</td>
<td>Monitor off.</td>
<td>contest.c</td>
</tr>
<tr>
<td>mon.on()</td>
<td>Monitor on.</td>
<td>contest.c</td>
</tr>
</tbody>
</table>

Table C.5: The functions – 5
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>File</th>
</tr>
</thead>
<tbody>
<tr>
<td>monitor(i)</td>
<td>Monitor servo channel i.</td>
<td>contest.c</td>
</tr>
<tr>
<td>move(a, b)</td>
<td>move a by b ie. a()=b(a()).</td>
<td>pos.c</td>
</tr>
<tr>
<td>mu_check(mu)</td>
<td>Fast limit check of joint coords in mu</td>
<td>tbotpos002.c</td>
</tr>
<tr>
<td>mv_to_cso(i)</td>
<td>Move axis i till is_cso(i) is active.</td>
<td>calibrate.c</td>
</tr>
<tr>
<td>mv_tmu(mu)</td>
<td>Move to m/c units using interpolator.</td>
<td>mainlib.c</td>
</tr>
<tr>
<td>mv_tp_ji(tp)</td>
<td>Move toolplate to 'tp' in reference coord frame with joint interpolation.</td>
<td>mainlib.c</td>
</tr>
<tr>
<td>mv_ej_ji(p)</td>
<td>Move end_effector to posn p in reference coord frame with joint interp.</td>
<td>mainlib.c</td>
</tr>
<tr>
<td>mv_ej_ci(p)</td>
<td>Move end_effector to posn p in ref. coord frame with cartesian interp.</td>
<td>mainlib.c</td>
</tr>
<tr>
<td>mv_ej_pci(p)</td>
<td>Move end_effector to posn p in ref. coord frame with pseudo cartesian interp.</td>
<td>mainlib.c</td>
</tr>
<tr>
<td>mv_to_ncso()</td>
<td>Move axis i till is_cso(i) is inactive.</td>
<td>calibrate.c</td>
</tr>
<tr>
<td>nmask_action()</td>
<td>Non maskable interrupt action</td>
<td>manual.c</td>
</tr>
<tr>
<td>not_impl()</td>
<td>Message: that is not yet implemented.</td>
<td>mainlib.c</td>
</tr>
<tr>
<td>o_error1(i)</td>
<td>Excessive controller error handling process</td>
<td>conmon.c</td>
</tr>
<tr>
<td>o_error(i)</td>
<td>Handling excessive controller error</td>
<td>conmon.c</td>
</tr>
<tr>
<td>(*opt_jisoln (j_0, jts solns)) [NJOINTS]</td>
<td>Find sum of wrist displacements for each possible target, then select minimum displace.</td>
<td>tbotpos002.c</td>
</tr>
<tr>
<td>par_cal()</td>
<td>Parallel calibration of axes.</td>
<td>calibrate.c</td>
</tr>
<tr>
<td>park()</td>
<td>Move manipulator to park position.</td>
<td>mainlib.c</td>
</tr>
<tr>
<td>park_jt(n)</td>
<td>Place joint 'n' in its park position.</td>
<td>mainlib.c</td>
</tr>
<tr>
<td>pause(ms)</td>
<td>Pause process for AT LEAST 'ms'.</td>
<td>ntil.c</td>
</tr>
<tr>
<td>pio_write()</td>
<td>Write v to pio port n</td>
<td>tbot1io.c</td>
</tr>
<tr>
<td>p_read(a)</td>
<td>read a posn matrix (p_re_123(a))</td>
<td>pos.c</td>
</tr>
</tbody>
</table>

Table C.6: The functions – 6
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>File</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>p.re</em>123(a)</td>
<td>read a posn matrix <em>(body.3 123)</em></td>
<td><em>pos.c</em></td>
</tr>
<tr>
<td><em>p.re</em>323(a)</td>
<td>read a posn matrix <em>(body.2 323)</em></td>
<td><em>pos.c</em></td>
</tr>
<tr>
<td><em>p.write</em>(a)</td>
<td>write a position, original MRC func.</td>
<td><em>pos.c</em></td>
</tr>
<tr>
<td><em>p.wr</em>123(a)</td>
<td>write a position, <em>(body.3 123)</em> angles</td>
<td><em>pos.c</em></td>
</tr>
<tr>
<td><em>p.wr</em>323(a)</td>
<td>write a position, <em>(body.2 323)</em> angles</td>
<td><em>pos.c</em></td>
</tr>
<tr>
<td>put.ints()</td>
<td>output string s and n ints start at *d.</td>
<td><em>util.c</em></td>
</tr>
<tr>
<td>put.dbls()</td>
<td>output string s and n doubles start at *d.</td>
<td><em>util.c</em></td>
</tr>
<tr>
<td>ramp(i, v)</td>
<td>Apply ramp input to channel i.</td>
<td><em>util.c</em></td>
</tr>
<tr>
<td>req.error(n)</td>
<td>Request clearance to handle an error of type n.</td>
<td><em>error.c</em></td>
</tr>
<tr>
<td>report()</td>
<td>Report: that is not yet implemented.</td>
<td><em>mainlib.c</em></td>
</tr>
<tr>
<td>reset_adc()</td>
<td>Initialise dms531 pointers and registers. Set adc gain=2.</td>
<td><em>adc.c</em></td>
</tr>
<tr>
<td>reset.guard()</td>
<td>Toggle <em>reset.guard</em> bit low-high.</td>
<td><em>tbotlio.c</em></td>
</tr>
<tr>
<td>reset.io()</td>
<td>Reset parallel I/O and A/D doards</td>
<td><em>initio.c</em></td>
</tr>
<tr>
<td>reset.pio()</td>
<td>Reset PIO</td>
<td><em>tbotlio.c</em></td>
</tr>
<tr>
<td>reset.bd(n)</td>
<td>Reset i/f board n.</td>
<td><em>tbotlio.c</em></td>
</tr>
<tr>
<td>rot(a, d, v)</td>
<td>rotate about axis d by v radians.</td>
<td><em>pos.c</em></td>
</tr>
<tr>
<td>runmon(i)</td>
<td>Run monitor.</td>
<td><em>contest.c</em></td>
</tr>
<tr>
<td>run.mode()</td>
<td>Enter normal running state.</td>
<td><em>mainlib.c</em></td>
</tr>
<tr>
<td>run.interp(pos1,pos2,v,a)</td>
<td>Run interpolator.</td>
<td><em>interp.c</em></td>
</tr>
<tr>
<td>run.tmon(p)</td>
<td>Run temperature monitoring process, checking every p seconds</td>
<td><em>temmpmon.c</em></td>
</tr>
<tr>
<td>scanadc(select,time)</td>
<td>Scan select adc in 'time' seconds then preserve adc display.</td>
<td><em>adc.c</em></td>
</tr>
<tr>
<td>s.adc.gain(g)</td>
<td>Set adc gain (g) for next time.</td>
<td><em>adc.c</em></td>
</tr>
</tbody>
</table>

Table C.7: The functions – 7
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>File</th>
</tr>
</thead>
<tbody>
<tr>
<td>s_adc(m)</td>
<td>Reset adc measure for next time.</td>
<td>adc.c</td>
</tr>
<tr>
<td>seq_cal()</td>
<td>Sequentially calibrate each joint.</td>
<td>calibrate.c</td>
</tr>
<tr>
<td>stopmon()</td>
<td>Stop monitor.</td>
<td>contest.c</td>
</tr>
<tr>
<td>step(i, a)</td>
<td>Apply step change in posn to chan i.</td>
<td>control.c</td>
</tr>
<tr>
<td>square(i, a, p)</td>
<td>Square wave demand.</td>
<td>control.c</td>
</tr>
<tr>
<td>set_conlim()</td>
<td>Set control limit on channel n.</td>
<td>control.c</td>
</tr>
<tr>
<td>sho_typ()</td>
<td>Show current controller type.</td>
<td>control.c</td>
</tr>
<tr>
<td>sel_typ()</td>
<td>Select controller type to use.</td>
<td>control.c</td>
</tr>
<tr>
<td>sho_ass()</td>
<td>Show current assig of controllers.</td>
<td>control.c</td>
</tr>
<tr>
<td>sel_typ()</td>
<td>Select controller type to use.</td>
<td>control.c</td>
</tr>
<tr>
<td>sho_ass()</td>
<td>Show current assig of controllers.</td>
<td>control.c</td>
</tr>
<tr>
<td>sel_con(n)</td>
<td>Select controller n for config.</td>
<td>control.c</td>
</tr>
<tr>
<td>sho_con(n)</td>
<td>Display controller n.</td>
<td>control.c</td>
</tr>
<tr>
<td>(*setpoint())()</td>
<td>Build set point follower.</td>
<td>control.c</td>
</tr>
<tr>
<td>stop_drive()</td>
<td>Stop local drive process.</td>
<td>drive.c</td>
</tr>
<tr>
<td>scr_error(n)</td>
<td>Screen a request for handling of an error of type number of n.</td>
<td>error.c</td>
</tr>
<tr>
<td>ser_FATAL()</td>
<td>Service fatal error.</td>
<td>errservice.c</td>
</tr>
<tr>
<td>ser_NO_ERROR()</td>
<td>Shouldn’t ever get this far.</td>
<td>errservice.c</td>
</tr>
<tr>
<td>ser_TL_FAULT()</td>
<td>A trip line fault is interrupt driven and cannot be cleared easily;</td>
<td>errservice.c</td>
</tr>
<tr>
<td>ser_M_OTEMP()</td>
<td>Service motor over temp.</td>
<td>errservice.c</td>
</tr>
<tr>
<td>ser_A_OTEMP()</td>
<td>Service ambient over temp</td>
<td>errservice.c</td>
</tr>
<tr>
<td>ser_error(n)</td>
<td>Error servicing of error number 'n'</td>
<td>errservice.c</td>
</tr>
<tr>
<td>set_ee(tf)</td>
<td>set end_eff transformat wrt 'tp'.</td>
<td>mainlib.c</td>
</tr>
</tbody>
</table>

Table C.8: The functions — 8
### Function | Description | File
---|---|---
`set_base(tf)` | Set base transformation. | `mainlib.c`
`set_perf()` | Set percentage performance. | `mainlib.c`
`set_op_mode()` | Set manipulator operating mode. | `mainlib.c`
`set_safe()` | Set safe operating mode. | `mainlib.c`
`set_norm()` | Set normal operating mode. | `mainlib.c`
`set_park(p)` | Set park position to p. | `mainlib.c`
`sho_config(n)` | Show config. | `mainlib.c`
`sho_vsta()` | Show vsta. | `mainlib.c`
`sho_asta()` | Show asta. | `mainlib.c`
`set_inhibit()` | Set motion direction inhibiters. | `manual.c`
`sh_inhibit()` | Display motion inhibiters. | `manual.c`
`set_curr(i,c)` | Set servo amp current limit. | `servoio.c`
`set_tqlim()` | Set motor i torque limit. | `servoio.c`
`set_cont(i,v)` | op value 'v' to servo dtoa on axis 'i'. | `servoio.c`
`servos_ok()` | Check servo amps ok. | `servoio.c`
`(*setcontrol())()` | Build controller, Assemble function. | `setcontrol.c`
`sdtoa_write()` | Write value v to servo dtoa n. | `tbotlio.c`
`set_read()` | Enab/disas !READ on i/f bd | `tbotlio.c`
`slo(status)` | Enab/disas !SLO, relevant to bd 6. | `tbotlio.c`
`scan_temps()` | Scan and display temperature. | `tbotlio.c`
`sdtoa_test()` | Test servo dtoa 'i' with value v. | `tbotlio.c`
`set_mmax(n)` | Assign new value 'n' to mmax. | `tbotlio.c`
`ser_test()` | Test servio. Functions in servio.c. | `tbotlio.c`
`strob_usr()` | Application strobing action. | `usr.c`
`test()` | Test serv I/O and 1/O. | `test.c`
`temp_mon(p)` | Temperature monitoring process. | `temmpmon.c`

Table C.9: The functions — 9
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>File</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>term_usr()</code></td>
<td>Clean termination function.</td>
<td><code>usr.c</code></td>
</tr>
<tr>
<td><code>time_adcl(sel,n,count)</code></td>
<td>If <code>sel = 0</code> get <code>adcl(n)&quot;counts&quot; times</code></td>
<td><code>adcl.c</code></td>
</tr>
<tr>
<td><code>tlf_report()</code></td>
<td>Report trip line faults in status.</td>
<td><code>tlfault.c</code></td>
</tr>
<tr>
<td><code>tlf_action()</code></td>
<td>tlf handling function.</td>
<td><code>tlfault.c</code></td>
</tr>
<tr>
<td><code>tq_hold(stat)</code></td>
<td>Turn torque hold on/off.</td>
<td><code>tbotltdo.c</code></td>
</tr>
<tr>
<td><code>tp_t_jts()</code></td>
<td>Converts 'tp' to joint coords.</td>
<td><code>tbotpos002.c</code></td>
</tr>
<tr>
<td><code>tr(a,d,v)</code></td>
<td>translate along axis d (0,1,2) by v</td>
<td><code>pos.c</code></td>
</tr>
<tr>
<td><code>trim_es(n,i)</code></td>
<td>Trim compressed error scales of contr.</td>
<td><code>control.c</code></td>
</tr>
<tr>
<td><code>trim_cs(n,i)</code></td>
<td>Trim compressed control scales of contr.</td>
<td><code>control.c</code></td>
</tr>
<tr>
<td><code>trip_guard()</code></td>
<td>Use with caution!</td>
<td><code>tbotltdo.c</code></td>
</tr>
<tr>
<td><code>tst_cal_axis()</code></td>
<td>Calibration test on joint i.</td>
<td><code>calibrate.c</code></td>
</tr>
<tr>
<td><code>unit(a)</code></td>
<td>initialise posn to unit matrix.</td>
<td><code>pos.c</code></td>
</tr>
<tr>
<td><code>vflush()</code></td>
<td>Flush vdu character buffer ibuf[].</td>
<td><code>manual.c</code></td>
</tr>
<tr>
<td><code>vlen(v)</code></td>
<td>length (modulus) of v, sum=sum+v*v</td>
<td><code>vectops.c</code></td>
</tr>
<tr>
<td><code>vnorm(v,v_nm)</code></td>
<td>normalize a vector, v_nm = v/</td>
<td>v</td>
</tr>
<tr>
<td><code>vscale(sc,v,sc_v)</code></td>
<td>scale a vector, sc_v=sc*v</td>
<td><code>vectops.c</code></td>
</tr>
<tr>
<td><code>vsub(v1,v2,dif)</code></td>
<td>subtract vectors, dif=v1-v2</td>
<td><code>vectops.c</code></td>
</tr>
<tr>
<td><code>vzero(v)</code></td>
<td></td>
<td><code>vectops.c</code></td>
</tr>
<tr>
<td><code>wait_for(cond,time,nchecks)</code></td>
<td>check 'cond' 'nchecks' times and timeout if not true after a minimum of 'time' ms</td>
<td><code>util.c</code></td>
</tr>
<tr>
<td><code>zero(a)</code></td>
<td>initialise posn to zero matrix.</td>
<td><code>pos.c</code></td>
</tr>
<tr>
<td><code>zero_cont()</code></td>
<td>Zero control outputs on all axes.</td>
<td><code>servoio.c</code></td>
</tr>
<tr>
<td><code>zero_temp()</code></td>
<td>Zero temps[]</td>
<td><code>temmpmon.c</code></td>
</tr>
<tr>
<td><code>zero_tqlim()</code></td>
<td>Zero motor torque limits.</td>
<td><code>servoio.c</code></td>
</tr>
</tbody>
</table>

Table C.10: The functions – 10
References


References


References


References


References


References


References


