Real-Time Computed Torque Control of Flexible-Joint Robots

by

Jayavel Sankaran

A thesis submitted in conformity with the requirements for the degree of

Master of Applied Science

Department of Mechanical and Industrial Engineering
University of Toronto

©Copyright by Jayavel Sankaran 1997
The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author’s permission.

L’auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L’auteur conserve la propriété du droit d’auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.
Title: Real-Time Computed Torque Control of Flexible-Joint Robots

Degree: Master of Applied Science, 1997

Author: Jayavel Sankaran

Dept.: Department of Mechanical and Industrial Engineering

Univ.: University of Toronto

Abstract

Evergrowing demand for speed, accuracy and higher payload capacity of industrial robots has led to redefining the robot control problem. As speed and accuracy are always at loggerheads, achieving one without compromising on the other, calls for a closer examination of the physical system of the robot, to devise sophisticated control techniques that would meet the rigorous specifications. This thesis aims at modelling the dynamics of a typical industrial robot, incorporating the flexibility effects due to the mechanical compliance of the joints. Using a reduced order model of the flexible-joint system, the computed torque control method is implemented on the robot, in order to evaluate the real-time trajectory tracking performance of the computed torque controller, compared to the PID-based commercial controller of the robot. Issues related to real-time multi-axis servo control, dynamics modelling and analysis, parallel processing, electro-mechanical systems design and integration, are addressed in this thesis.
To the memory of my father

Let us realize that what happens around us
is largely outside our control, but that
the way we choose to react to it
is inside our control

J. Petty. "Apples of Gold"
Acknowledgement

This thesis would not have been possible but for the timely opportunity given by Prof. J.K. Mills. I sincerely thank him for his relentless support, able guidance and encouragement throughout the project. I would also like to thank Prof. R.M.H. Cheng and Prof. R. Ramesh of Concordia University for their support during the initial stages of my graduate study. I wish to extend my thanks to my parents for their moral support which is vital to all my efforts. Thanks are also due to my colleagues, past and present, for their suggestions and help which saved a lot of time.
# Contents

Abstract ................................................................. i

Acknowledgement ......................................................... iii

List of Figures ............................................................. vii

List of Tables ............................................................... xiii

Nomenclature ............................................................... xv

## 1 Introduction

1.1 Motivation .............................................................. 1

1.2 The Challenge ......................................................... 1

1.3 Literature Review .................................................... 3

1.4 Scope of This Thesis ................................................ 4

## 2 Flexible Transmission Effects in Robots

2.1 Compliance Distribution in Robot Mechanisms ....................... 6

2.2 Harmonic Drive ........................................................ 9

2.3 Singular Perturbation Theory in Robots with Transmission Compliance 14

  2.3.1 The Concept of Singular Perturbation .......................... 14

  2.3.2 Modeling Transmission Compliance With The Singular Perturbation Method ............................................................... 16

  2.3.3 Stability Analysis of Mechanical Systems With Transmission Compliance ............................................................... 18
3 Dynamic Modeling of Flexible-Joint Robots 20
  3.1 The Dynamic Model ........................................ 21
    3.1.1 Rigid Manipulator Dynamics ............................. 21
    3.1.2 Flexible-Joint Robot Dynamics .......................... 21
  3.2 Singular Perturbation Analysis of the Flexible-Joint Robot .... 23
    3.2.1 Stability of the Flexible-Joint Robot System .......... 25

4 Computed Torque Control 26
  4.1 Why Computed Torque Control? ................................. 26
    4.1.1 Shortcomings of The PID Control Method .................. 27
    4.1.2 Merits of The Computed Torque Control Method ......... 28
  4.2 What is Computed Torque Control? ............................. 30
    4.2.1 The Computed Torque Control Law ........................ 32

5 Experimental Set-up 34
  5.1 Hardware .................................................. 34
  5.2 Software .................................................. 34
  5.3 Computed Torque Control with The Inner Torque-Loop .......... 36
  5.4 Computed Torque Control without The Inner Torque-Loop ...... 38
    5.4.1 The Current-Based Torque Control Method .............. 39

6 Experimental Results 43
  6.1 Design of Experiments ....................................... 44
    6.1.1 Parameters of Control .................................. 44
    6.1.2 Parameters of Observation ............................... 45
  6.2 Results and Discussion ...................................... 46
    6.2.1 Discussion .............................................. 56

7 Concluding Remarks 59
  7.1 Summary .................................................... 59
  7.2 Conclusion .................................................. 61
  7.3 Contributions ............................................... 63
7.4 Recommendations For Future Work .................................. 64
  7.4.1 Issues Related to Modelling ..................................... 64
  7.4.2 Issues Related to the Experimental Set-up ................... 64

A Optimal Loop-Gain Values ........................................... 70

B Control Software Listing ............................................ 71
  B.1 Network Information File: cto rq.nif ............................ 71
  B.2 Network Configuration File: cto rq.wir ....................... 71
  B.3 Network Load File: cto rq.bat ................................ 72
  B.4 Make File: cto rq.mak .......................................... 72
  B.5 Link File: cto rq.lnk ........................................... 73
  B.6 'C' Source Code ................................................ 74
    B.6.1 General Description ......................................... 74
    B.6.2 Control program description ............................... 75
    B.6.3 cto rq.c ................................................... 77
    B.6.4 contreal.c ................................................ 81
    B.6.5 control.c ................................................. 93
    B.6.6 contini.c ................................................ 99

C Graphical Results of Real-time Experiments on
  the CRS A460 Robot ................................................ 101
List of Figures

2.1 The Harmonic Drive unit .......................................... 10
2.2 Schematic Representation of the Harmonic Drive Driven Joint .... 11
2.3 Block Diagram of the Flexible-Joint Model .......................... 11
2.4 Free-body Diagram of the Flexible-Joint System .................... 12

4.1 A schematic representation of the Computed Torque Control System 30

5.1 The CRS A460 Robot .............................................. 35
5.2 Topology of the transputer network ................................. 37
5.3 Computed Torque Control with the Inner Torque-Loop ............... 38
5.4 Computed Torque Control without the Inner Torque-Loop ........... 39
5.5 The Armature Current Control Scheme ............................. 41

6.1 Position history of joint-1 under computed torque control ........ 47
6.2 Position history of joint-1 under independent-joint PD control .... 48
6.3 Position history of joint-1 under PD control with gravity compensation 49
6.4 Position history of joint-2 under computed torque control .......... 50
6.5 Position history of joint-2 under independent-joint PD control .... 51
6.6 Position history of joint-2 under PD control with gravity compensation 52
6.7 Position history of joint-3 under computed torque control .......... 53
6.8 Position history of joint-3 under independent-joint PD control .... 54
6.9 Position history of joint-3 under PD control with gravity compensation 55

C.1 Position history of joint-1 under computed torque control, for motion aided by gravity .............................................. 102
C.2 Position history of joint-1 under PD control, for motion aided by gravity 103
C.3 Position history of joint-1 under PD control with gravity compensation, for motion aided by gravity 104
C.4 Position history of joint-1 under computed torque control, for motion against gravity 105
C.5 Position history of joint-1 under PD control, for motion against gravity 106
C.6 Position history of joint-1 under PD control with gravity compensation, for motion against gravity 107
C.7 Position history of joint-1 under computed torque control, at a peak joint-speed of 0.1 rad/s and a payload of 0.5Lb 108
C.8 Position history of joint-1 under computed torque control, at a peak joint-speed of 1 rad/s and a payload of 0.5Lb 109
C.9 Position history of joint-1 under computed torque control, at a peak joint-speed of 3 rad/s and a payload of 0.5Lb 110
C.10 Position history of joint-1 under computed torque control, at a peak joint-speed of 0.1 rad/s and a payload of 1.5Lb 111
C.11 Position history of joint-1 under computed torque control, at a peak joint-speed of 1 rad/s and a payload of 1.5Lb 112
C.12 Position history of joint-1 under computed torque control, at a peak joint-speed of 3 rad/s and a payload of 1.5Lb 113
C.13 Position history of joint-1 under computed torque control, at a peak joint-speed of 0.1 rad/s and a payload of 2.5Lb 114
C.14 Position history of joint-1 under computed torque control, at a peak joint-speed of 1 rad/s and a payload of 2.5Lb 115
C.15 Position history of joint-1 under computed torque control, at a peak joint-speed of 3 rad/s and a payload of 2.5Lb 116
C.16 Position history of joint-1 under computed torque control, at a peak joint-speed of 0.1 rad/s and a payload of 3.5Lb 117
C.17 Position history of joint-1 under computed torque control, at a peak joint-speed of 1 rad/s and a payload of 3.5Lb 118
C.18 Position history of joint-1 under computed torque control. at a peak joint-speed of 3 rad/s and a payload of 3.5Lb

C.19 Position history of joint-1 under computed torque control. at a peak joint-speed of 0.1 rad/s and a payload of 4.5Lb

C.20 Position history of joint-1 under computed torque control. at a peak joint-speed of 1 rad/s and a payload of 4.5Lb

C.21 Position history of joint-1 under computed torque control. at a peak joint-speed of 3 rad/s and a payload of 4.5Lb

C.22 Position history of joint-1 under computed torque control. at a peak joint-speed of 0.1 rad/s and a payload of 5.5Lb

C.23 Position history of joint-1 under computed torque control, at a peak joint-speed of 1 rad/s and a payload of 5.5Lb

C.24 Position history of joint-1 under computed torque control, at a peak joint-speed of 3 rad/s and a payload of 5.5Lb

C.25 Position history of joint-2 under computed torque control for motion against gravity

C.26 Position history of joint-2 under independent-joint PD control for motion against gravity

C.27 Position history of joint-2 under PD control with gravity compensation for motion against gravity

C.28 Position history of joint-2 under computed torque control for motion aided by gravity

C.29 Position history of joint-2 under independent-joint PD control for motion aided by gravity

C.30 Position history of joint-2 under PD control with gravity compensation for motion aided by gravity

C.31 Position history of joint-2 under computed torque control, at a peak joint-speed of 0.1 rad/s and a payload of 0.5Lb

C.32 Position history of joint-2 under computed torque control, at a peak joint-speed of 1 rad/s and a payload of 0.5Lb
C.33 Position history of joint-2 under computed torque control, at a peak joint-speed of 3 rad/s and a payload of 0.5Lb.

C.34 Position history of joint-2 under computed torque control, at a peak joint-speed of 0.1 rad/s and a payload of 1.5Lb.

C.35 Position history of joint-2 under computed torque control, at a peak joint-speed of 1 rad/s and a payload of 1.5Lb.

C.36 Position history of joint-2 under computed torque control, at a peak joint-speed of 3 rad/s and a payload of 1.5Lb.

C.37 Position history of joint-2 under computed torque control, at a peak joint-speed of 0.1 rad/s and a payload of 2.5Lb.

C.38 Position history of joint-2 under computed torque control, at a peak joint-speed of 1 rad/s and a payload of 2.5Lb.

C.39 Position history of joint-2 under computed torque control, at a peak joint-speed of 3 rad/s and a payload of 2.5Lb.

C.40 Position history of joint-2 under computed torque control, at a peak joint-speed of 0.1 rad/s and a payload of 3.5Lb.

C.41 Position history of joint-2 under computed torque control, at a peak joint-speed of 1 rad/s and a payload of 3.5Lb.

C.42 Position history of joint-2 under computed torque control, at a peak joint-speed of 3 rad/s and a payload of 3.5Lb.

C.43 Position history of joint-2 under computed torque control, at a peak joint-speed of 0.1 rad/s and a payload of 4.5Lb.

C.44 Position history of joint-2 under computed torque control, at a peak joint-speed of 1 rad/s and a payload of 4.5Lb.

C.45 Position history of joint-2 under computed torque control, at a peak joint-speed of 3 rad/s and a payload of 4.5Lb.

C.46 Position history of joint-2 under computed torque control, at a peak joint-speed of 1 rad/s and a payload of 5.5Lb.

C.47 Position history of joint-2 under computed torque control, at a peak joint-speed of 3 rad/s and a payload of 5.5Lb.
C.48 Position history of joint-3 under computed torque control for motion aided by gravity ........................................ 149
C.49 Position history of joint-3 under independent-joint PD control for motion aided by gravity ........................................ 150
C.50 Position history of joint-3 under PD control with gravity compensation for motion aided by gravity ........................................ 151
C.51 Position history of joint-3 under computed torque control for motion against gravity ........................................ 152
C.52 Position history of joint-3 under independent-joint control for motion against gravity ........................................ 153
C.53 Position history of joint-3 under PD control with gravity compensation for motion against gravity ........................................ 154
C.54 Position history of joint-3 under computed torque control, at a peak joint-speed of 0.1 rad/s and a payload of 0.5Lb ........................................ 155
C.55 Position history of joint-3 under computed torque control, at a peak joint-speed of 1 rad/s and a payload of 0.5Lb ........................................ 156
C.56 Position history of joint-3 under computed torque control, at a peak joint-speed of 3 rad/s and a payload of 0.5Lb ........................................ 157
C.57 Position history of joint-3 under computed torque control, at a peak joint-speed of 0.1 rad/s and a payload of 1.5Lb ........................................ 158
C.58 Position history of joint-3 under computed torque control, at a peak joint-speed of 1 rad/s and a payload of 1.5Lb ........................................ 159
C.59 Position history of joint-3 under computed torque control, at a peak joint-speed of 3 rad/s and a payload of 1.5Lb ........................................ 160
C.60 Position history of joint-3 under computed torque control, at a peak joint-speed of 0.1 rad/s and a payload of 2.5Lb ........................................ 161
C.61 Position history of joint-3 under computed torque control, at a peak joint-speed of 1 rad/s and a payload of 2.5Lb ........................................ 162
C.62 Position history of joint-3 under computed torque control, at a peak joint-speed of 3 rad/s and a payload of 2.5Lb ........................................ 163
C.63 Position history of joint-3 under computed torque control at a peak joint-speed of 0.1 rad/s and a payload of 3.5Lb

C.64 Position history of joint-3 under computed torque control at a peak joint-speed of 1 rad/s and a payload of 3.5Lb

C.65 Position history of joint-3 under computed torque control at a peak joint-speed of 3 rad/s and a payload of 3.5Lb

C.66 Position history of joint-3 under computed torque control at a peak joint-speed of 0.1 rad/s and a payload of 4.5Lb

C.67 Position history of joint-3 under computed torque control at a peak joint-speed of 1 rad/s and a payload of 4.5Lb

C.68 Position history of joint-3 under computed torque control at a peak joint-speed of 3 rad/s and a payload of 4.5Lb

C.69 Position history of joint-3 under computed torque control at a peak joint-speed of 1 rad/s and a payload of 5.5 Lb

C.70 Position history of joint-3 under computed torque control at a peak joint-speed of 3 rad/s and a payload of 5.5 Lb
List of Tables

C.1 Joint-1 peak position error in degrees ......................... 172
C.2 Joint-2 peak position error in degrees ......................... 172
C.3 Joint-3 peak position error in degrees ......................... 172
C.4 Joint-1 peak position error in degrees, under computed torque control 173
C.5 Joint-2 peak position error in degrees, under computed torque control 173
C.6 Joint-3 peak position error in degrees, under computed torque control 173
Nomenclature

$C(\theta, \dot{\theta})$ vector of Coriolis and centrifugal forces
$D(\theta)$ inertia tensor of the rigid manipulator
$G(\theta)$ vector of gravitational forces
$J_m$ inertia of the motor, referred to the load-side
$J$ overall inertia on the load-side
$K$ joint stiffness matrix
$K_p$ proportional gain matrix
$K_v$ derivative gain matrix
$K_{lp}$ torque loop gain matrix
$n$ transmission matrix
$R$ joint damping matrix
$T$ vector of joint-torque input
$T_m$ vector of motor-torque input
$w$ transformation variable defined as $\delta K$
$\mu$ singular perturbation parameter
$\tau$ stretched time variable
$\theta$ vector of joint position coordinates
$\dot{\theta}$ vector of joint velocity
$\ddot{\theta}$ vector of joint acceleration
$\theta_d$ vector of actual joint-position coordinates
$\dot{\theta}_d$ vector of actual joint-velocity
$\delta$ joint compliance vector
$\dot{\delta}$ time-derivative of the joint compliance vector
$^+$ variables associated with the fast subsystem
$^-$ variables associated with the slow subsystem
Chapter 1

Introduction

1.1 Motivation

The brain of a robot is the controller which controls the body of the robot, the mechanism consisting of physical links, actuators and transmission elements. Therefore the successful performance of a robot largely hinges upon the efficacy of the controller, among other factors. Most industrial robots are primarily, position-controlled devices. Though this type of control is adequate for most industrial applications, tasks that require high precision as in electronic assembly, safety critical applications like robot-assisted surgery, high speed tasks like visual servoing, applications involving high payload as in space craft manoeuvres, call for more sophisticated control strategies. This was the motivation for investigating model-based control techniques like the computed torque control method that promise to cater to the needs of applications that demand higher accuracy and higher speeds while retaining flexibility. The advantages of using the computed torque control method over the independent-joint PID control, will become clear in the ensuing chapters.

1.2 The Challenge

Any model-based control method requires a reasonably accurate mathematical model of the physical system under control. Therefore, modelling a physical system as
realistically as possible is the first step in the application of model-based control. A model derived under the assumptions of rigid body dynamics may not represent the real attributes of the physical system as every mechanical system exhibits structural flexibility to some extent. The flexibility effects of the physical system have to be modeled as well, in order to mathematically capture the actual behaviour of the system. Hence, the dynamics of the system can no longer be governed by the laws of rigid body dynamics alone. Flexible body dynamics of the system can be analysed by advanced techniques like the singular perturbation method which will be discussed in detail, in the following chapter.

After formulating the model, the next challenge is to implement the control strategy using the model. There are two major problems associated with using the complete flexible-joint model for real-time control: Firstly, acquiring joint-compliance data and its derivatives in real-time, is difficult, if not impossible, due to hardware constraints. Secondly, in the computed torque control method, the complete inverse dynamics computation has to be carried out in real-time. As the complexity of the model grows to represent the actual physical system, the computational burden increases as well. Though the present-day computers have overcome many bottlenecks in highspeed computation, it is cumbersome to use the complete higher-order flexible-joint model for real-time calculations. Through parallel processing, online inverse dynamics calculations for a system as complex as a multi-axis robot is not impossible. However, simplifying the higher-order model to recover a reduced-order model, which would represent the actual system with reasonable accuracy, is quite acceptable. In this thesis, the flexible-joint model is simplified to generate the reduced-order model used for real-time control.

From the implementation point of view, the torque control method involves servoing the joint motors to the commanded torque computed by the controller. The method of torque servo-loop using a strain-gauge based torque feedback loop is problematic due to the tendency of the strain-gauge bridge to pick up noisy signals as a result of weak surface bonding and temperature variations [20]. Therefore, alternate means of implementing the torque control loop, like the current-based torque control
method, have to be investigated.

1.3 Literature Review


Robot dynamics models have been developed for several industrial robot manipulators using rigid body assumptions. Armstrong [2] derived the dynamics model for a Puma560 robot, incorporating friction models, in 1988. Tuttle [30] has dealt with
modelling flexibility of the harmonic drive, in great detail. Due to the complexity of flexible-joint robot models, real-time control implementation is difficult, if not impossible. Readman [27] has treated flexible-joint robot modelling and control quite extensively. Flexible-joint robot simulations have been reported by Tahboub [29], Lin and Yu [18]. De Luca [10] has reported feedback linearization of robots with mixed rigid and elastic joints. Tosunoglu [31] has dealt with modelling flexible elements in robotic systems.

1.4 Scope of This Thesis

This thesis addresses issues related to real-time multi-axis servo control, parallel processing, dynamics modelling, and electro-mechanical systems design and integration. The thesis attempts to model the flexibility effects of a typical industrial robot, with a view to implement computed torque control on the flexible system, aiming at achieving high-speed and high-precision real-time trajectory tracking.

While structural flexibility can arise from a variety of sources, as most industrial robots have sufficiently rigid links, only joint-flexibility is dealt with. Among the popular high-torque transmission devices used in industrial robots, the harmonic-drive transmission is known to exhibit a built-in compliance, contributing to the flexibility of robot joints, substantially. A detailed discussion of the modelling and analysis of the flexibility effects of harmonic drives is presented in Chapter 2.

Based on the insight gained into flexible-joint system modelling, a comparative study of the dynamics model, with and without joint-flexibility, of a CRS A460 type robot of CRS Robotics Corporation, is presented in Chapter 3. However, the dynamics modelling and analysis is restricted to the first three joints of the six degrees of freedom CRS robot. The first three joints of the robot, namely, the waist, the shoulder and the elbow, are used for attaining the three degrees of freedom for positioning the end-effector, while the last three joints of the wrist, namely, pitch, roll and yaw, are used for the other three degrees of freedom, required for orienting the end-effector in space. The last three wrist-joints have lower inertia and smaller dynamics effects.
like centrifugal, Coriolis, frictional and interaction forces, compared to the first three joints, due to their lower torque-handling necessity and smaller size. Hence, if the control system can be proved to be successful for controlling the first three joints, the technique can be extended to the last three joints, subsequently. Moreover, due to the practical difficulty of implementing methods to measure compliance and its higher derivatives in real-time, in commercially available industrial robots like the CRS A460 robot, using the complete flexible-joint model of the robot for real-time control, is difficult, is not impossible. Therefore, only a reduced order model has been adopted for the real-time implementation of computed torque control of the CRS A460 robot, in this thesis.

Chapter 4 deals with the computed torque control method, presenting a thorough understanding of the technique and its merits over the independent-joint PID control method. From the point of view of controlling the joint-motor torque, computed by the model-based controller for a particular manoeuvre, the computed torque control method can be categorized as the method with the inner-torque loop \cite{11}, \cite{5} and that without the inner-torque loop. While the former method employs strain gauges to actually measure the joint torque so as to implement a closed-loop torque control at the lowest or the "innermost" level of the control system, the latter method senses the motor torque via the current drawn by the armature circuit. Due to the elimination of the afore-said strain gauge based inner-torque loop, the current-based torque control method has been advantageously adopted in this thesis.

Chapter 5 presents the detailed description of the experimental design, explaining the method of implementing the computed torque controller without the inner-torque loop. The experimental results are presented, compared and analyzed in Chapter 6 and finally, Chapter 7 summarizes the thesis and presents a critical analysis of the conclusions derived.
Chapter 2

Flexible Transmission Effects in Robots

The theoretical concept of a rigid body that does not deform under external forces, is an engineering simplification of the actual physical characteristics of the material which forms the body. In practice, however, all materials have finite stiffness and hence deform under external forces and moments. The mechanical structure of a robot is composed of links, joints and drive components, each of which is flexible. Sources of compliance include gear teeth, timing belts, cables, transducers used to measure forces and moments, built-in flexibility as exhibited by harmonic drives, and the compressibility of the transmission medium as in pneumatic drives. While some of the compliance may be purposely introduced by mechanical design, as in the flexspline of a harmonic drive, a majority of them arise due to many inevitable factors associated with the properties of materials used and manufacturing methods employed. A discussion of the sources of compliance in the mechanism of a robot is in order.

2.1 Compliance Distribution in Robot Mechanisms

The mechanism of the robot consists of linkages, supporting elements like bearings, guideways, and power transmission elements like gear drives, timing belts, cables,
hydraulic and pneumatic cylinders. Every element in the mechanism contributes to the overall compliance of the mechanical structure of the robot, although in varying proportions. Compliance is desirable in certain cases as in a RCC (Remote Center Compliance) device. However in many other cases, structural compliance is undesirable but inevitable in mechanical design. Increasing the cross sections or using high-modulus conventional materials, as a means to reduce compliance, are in many cases either counterproductive or not cost effective. Over-design increases the weight and results in larger deflections due to larger inertia forces. In many instances, both the external and internal dimensions of the links are limited by design and application constraints. Though stiffness enhancement by means of additional supports is very effective, it limits the universality of the robot as the supports have to be custom designed for specific applications.

In order to minimize, if not eliminate, the compliance of the overall structure, thus to improve the performance and accuracy characteristics of the robot without losing generality, one has to understand the sources of structural compliance, compute a breakdown of compliance among the various structural constituents like linkages and transmission elements and direct design efforts toward their improvement. Since the most important goal is reduction of deflections in structural chains caused by a force at a certain point, the use of a compliance parameter instead of a stiffness parameter is natural in many cases as compliance breakdown is equivalent to deflection breakdown.

Effective compliance of a structure is measured by the response of the structure to certain performance-induced forces like inertial forces, and forces caused by interactions with the environment. Structural compliance is a result of the following four basic factors:

- Structural deformations of load-transmitting components which are idealized for computational purposes as beams, rods, plates, shells, etc., also with idealized loading and support conditions.

- Contact deformations between parts contacting along nominally small contact surfaces like balls, roller, etc., or nominally large but actually small contact
surfaces like imperfectly machined flat surfaces.

- Deformations in the transmission devices caused by mechanical deflection of gear tooth, compressibility of a working medium as in hydraulic or pneumatic systems, deformation of electromagnetic field in electric motors, etc.

- Modifications of numerical stiffness values caused by kinematic transformations between the area in which the deformations originate and the point for which the effective stiffness is analyzed.

There are many ways of measuring and analyzing compliance distribution in a manipulator structure. They include static loading of the linkage and measuring ensuing static deflections and measuring natural frequencies under shock or sinusoidal excitation, compiling mathematical models, calculating masses and moments of inertia, and finally, deducting compliance values. To construct the distribution, compliances of structural elements as well as transmission devices, have to be computed.

The significance of each of the contributors to the overall effective compliance depends not only on the magnitudes of their compliance but also on their operating speeds. To evaluate the effect of individual contributors, all the compliance magnitudes have to be reflected to one selected structural element. If such a reduction is properly done, neither natural frequencies nor modes of vibration are affected and the overall compliance referred to a certain component of the system would be the same as the compliance value measured by the application of force or torque to this component and recording the resulting deflection. The condition to be satisfied for the reduced algorithm to be correct is that both potential and kinetic energy are the same for mathematical models of the original and the reduced systems.

A direct consequence of structural compliance is that it would result in loss of overall steady state positional accuracy of the robot. If the positional parameters are measured only at the driver-end of the transmission, assuming total rigidity of the transmission components. This kinematic error could be corrected by suitable calibration procedures. A more serious effect, however, is that flexibility introduces complexities in the dynamics of the otherwise simpler rigid system due to the addi-
tional degrees of freedom. Therefore, in practice, a robot's mechanical structure is nonrigid and dynamically more complex than predicted by the rigid body dynamic model. If the flexible transmission effects are neglected, then the closed-loop bandwidth of the joint angle and task space control loops will be limited, and dynamic positioning errors can occur if the fast dynamics are excited.

This thesis deals with compliance introduced only by joint-flexibility. Robot joints are actuated by electric motors or hydraulic or pneumatic actuators and the mechanical motion is transmitted to the links via transmission components like gear box, timing belts, cables, etc., except direct-drive robots in which case the actuators are mounted directly onto the links, thereby minimizing any flexible transmission effects, at the cost of large-sized and expensive actuators, comparatively. Due to the compliance of these individual transmission elements, the overall joint of the robot becomes flexible. One of the typical mechanical transmission systems used in industrial robots, which introduce joint-flexibility, are harmonic drives. It is therefore of interest to study the flexibility effects of harmonic drives in detail.

2.2 Harmonic Drive

Harmonic drives are characterized by a high transmission ratio and compact size. They are favoured for electromechanical systems with space and weight constraints. Moreover, they result in the use of relatively light high-speed motors. The harmonic drive is kinematically akin to a planetary drive with one gear that is made with a flexible rim. It consists of a flexspline, a circular spline and a wave generator, as illustrated in figure 2.1. The flexspline lies inbetween the circular spline and the wave generator and it takes the shape of the wave generator which rotates in close contact with the flexspline. The wave generator is elliptical in shape so that only a fraction of the teeth of the flexspline are in contact with the corresponding teeth on the circular spline. Also, the flexspline has a couple of teeth less than the circular spline. A schematic representation of a typical harmonic drive driven joint is depicted in figure 2.2.
The gear ratio and the direction of rotation can be modified by fixing any one of the three components and using the other two as the input link and the output link, respectively. Typically, the wave generator is driven by a high speed motor and the flexspline is connected to the output link, while the circular spline is kept fixed.

Flexspline is the primary flexible element in a harmonic drive, which introduces compliance into the transmission. Torsional stiffness of the flexspline, like many other spring elements, is nonlinear. However, since the variations in torsional stiffness of the flexspline are unpredictable and minimal, for all practical purposes it can be considered to be constant, without loss of generality. Thus the compliance of the harmonic drive can be modelled by means of a spring element of constant stiffness and a damping element to represent the cumulative damping effects of the drive as shown in Figure 2.3.

Since the wave generator and the drive-motor are mounted coaxially, their rotor inertias can be lumped together and reflected onto the load side of the gear box. The transmission ratio, $n$, of the harmonic drive in this configuration can be calculated as follows:

$$n = -\frac{n_{fs}}{n_{cs} - n_{fs}}$$ (2.1)
Figure 2.2: Schematic Representation of the Harmonic Drive Driven Joint

Figure 2.3: Block Diagram of the Flexible-Joint Model [27]
where,

\( n_{fs} \) = is the number of teeth on the flexspline

\( n_{cs} \) = is the number of teeth on the circular spline

The free body diagram of the harmonic drive joint is shown in Figure 2.4. Accordingly, the dynamic model of the flexible-joint is formulated as follows:

\[
\begin{align*}
J_m \ddot{\theta}_m - \frac{n}{n_k} T_m + n C \left( \frac{\dot{\theta}_m}{n} - \dot{\theta}_l \right) + n K \left( \frac{\dot{\theta}_m}{n} - \theta_l \right) & = n T_m \\
J_l \ddot{\theta}_l + C \left( \dot{\theta}_l - \frac{\dot{\theta}_m}{n} \right) + K \left( \theta_l - \frac{\theta_m}{n} \right) & = 0
\end{align*}
\] (2.2) (2.3)

where, all inertial quantities are referred to the load-side of the drive, unless stated otherwise:

\( n \) = transmission ratio

\( J_m \) = effective polar moment of inertia on the motor-side

\( = (J_{motor} + J_{wavegenerator}) \times n^2 \)
\( J_l \) = effective polar moment of inertia on the load-side
\( C \) = effective damping of the joint
\( K \) = torsional stiffness of the flexspline
\( \theta_{\text{m}} \) = input link position
\( \dot{\theta}_{\text{m}} \) = input link velocity
\( \ddot{\theta}_{\text{m}} \) = input link acceleration
\( \theta_i \) = output link position
\( \dot{\theta}_i \) = output link velocity
\( \ddot{\theta}_i \) = output link acceleration

The flexible joint model of the harmonic-drive driven joint can be further simplified by rewriting Equations 2.2 and 2.3 in terms of torsional compliance, \( z \), of the flexspline instead of the actual link coordinates. Therefore, by defining the torsional compliance, \( z \) and its derivatives as follows,

\[
\begin{align*}
    z &= \left( \frac{\theta_{\text{m}}}{n} - \theta_i \right) \\
    \dot{z} &= \left( \frac{\dot{\theta}_{\text{m}}}{n} - \dot{\theta}_i \right) \\
    \ddot{z} &= \left( \frac{\ddot{\theta}_{\text{m}}}{n} - \ddot{\theta}_i \right)
\end{align*}
\]

the fourth order flexible joint system modelled by equations 2.2 and 2.3 can be rewritten in terms of the joint compliance \( z \) as

\[
\begin{align*}
    J_m \frac{\dot{\theta}_{\text{m}}}{n} + C \dot{z} + Kz &= n T_m \\
    J_i \ddot{\theta}_i - C \dot{z} - Kz &= 0
\end{align*}
\]

Further transposition of equations 2.7 and 2.8, yields

\[
\begin{align*}
    \frac{\dot{\theta}_{\text{m}}}{n} &= \frac{n T_m}{J_m} - \frac{C}{J_m} \dot{z} - \frac{K}{J_m} z \\
    \ddot{\theta}_i &= \frac{C}{J_1} \dot{z} + \frac{K}{J_1} z
\end{align*}
\]
By subtracting equation 2.10 from equation 2.9, the fourth order system in $\theta$, shown in equations 2.2 and 2.3, can be simplified to a second order system in terms of the transmission compliance $z$, as follows:

\begin{align*}
\ddot{z} &= n\frac{T_m}{J_m} - \frac{C\dot{z}}{J} - \frac{Kz}{J} \\
J_l\ddot{\theta}_l &= C\dot{z} + Kz
\end{align*}

where,

\[ J = \frac{J_m J_l}{J_m + J_l}. \]

\section*{2.3 Singular Perturbation Theory in Robots with Transmission Compliance}

\subsection*{2.3.1 The Concept of Singular Perturbation}

The singular perturbation method is a tool to simplify complex dynamic models without much loss of generalization. The dynamic model of a physical system can be considered to be made up of two coupled dynamic systems, namely, the usual rigid body or low frequency dynamics and the other, high frequency dynamics due to compliance \cite{27}. This decomposition into slow and fast systems is governed by a separation of time scales. Typically, the reduced model represents the average phenomena while the boundary layer models evolve in faster time scales and represent deviations from the predicted slow behavior.

In principle, the singular perturbation approach involves the identification of a parameter in the model, which can tend to become a negligibly small quantity in the limiting case. This parameter is termed as the perturbation variable. As this perturbation variable tends to zero, the model is said to be singularly perturbed. Upon suitable manipulation of the perturbation parameter and the time scale, the
high frequency dynamics of the system can be analysed [27]. Thus, the singular perturbation method of analysis, in essence, involves the following procedure:

- Singularity perturbed system modeling: The system is represented in the standard form and the perturbation parameter is identified.

- Recovery of the rigid body model: Then, the singular perturbation parameter is set to zero and the rigid body model is regenerated from the flexible model.

- Stretching the time scale: The time scale is stretched by a suitable transformation so as to reveal the high frequency dynamics of the flexible system.

- A Reduced Order flexible model is generated to include the effect of joint flexibility without increasing the order of the rigid body model.

- If the high frequency dynamics of the flexible model can be proven to be asymptotically stable, then any traditional controller for rigid robot control can be used to design the controller.

- Corrective control can be designed to achieve the compensation of the joint flexibility using the reduced order flexible model.

The advantages of modelling a flexible system as a singularly perturbed system is that the model obtained is of the same order as the rigid system but is a more accurate model for flexible joint model. The reduced order system is linearizable by nonlinear feedback and a feedback signal for this system does not require information about the joint acceleration and jerk. Moreover, it is of practical interest to deal separately with the fast mode control and rigid body control, which has a much lower frequency, thereby enabling different sampling rates for each subsystem to be designed. Also, the laws that are designed for rigid body control can be readily adopted to control the flexible system whose model accounts for compliance.
2.3.2 Modeling Transmission Compliance with the Singular Perturbation Method

Consider the case of the harmonic drive transmission which has an in-built flexibility due to the flexspline. The one d.o.f. harmonic-drive driven flexible-joint model has been derived as equations 2.11 and 2.12, in Section 2.2.

It is quite evident that the compliance of the joint, \( \frac{1}{k} \), can readily serve as the singular perturbation variable, \( \mu \). Upon applying the mapping: \( z \rightarrow w \), such that

\[
\begin{align*}
  w &= kz \\
  \mu &= \frac{1}{k} \\
  z &= \mu w
\end{align*}
\]

the afore-said harmonic drive model transforms into

\[
\begin{align*}
  \mu \ddot{w} &= \frac{n T_m}{J_m} - \frac{C \mu}{J} \dot{w} - \frac{w}{J} \\
  J_l \ddot{\theta}_l &= C \mu \dot{w} + w
\end{align*}
\]

As the stiffness of the flexspline is known to be high, of the order of \( 10^4 \) Nm/rad \(^{27} \), i.e., \( k \gg 1 \), it follows that the singular perturbation parameter \( \mu \ll 1 \). Therefore, we can regenerate the rigid body model from the flexible model as the singular perturbation, \( \mu \rightarrow 0 \). Hence, from equations 2.16 and 2.17, the slow subsystem can be regenerated as follows:

\[
\begin{align*}
  n \frac{T_m}{J_m} - \frac{\bar{w}}{\bar{J}} &= 0 \\
  J_l \ddot{\bar{\theta}}_l &= \bar{w}
\end{align*}
\]

where, the overline indicates that the variables belong to the "slow" subsystem, due to the rigid body dynamics.

In order to visualize the high frequency dynamics of the "fast" subsystem, the
time scale must be stretched by applying the following temporal transformations to the flexible model:

\[
\frac{t}{\sqrt{\mu}} = \tau \quad \frac{C}{\sqrt{\mu}} = C
\]

In the stretched time scale, the flexible joint model 2.11 and 2.12, becomes

\[
\begin{align*}
    w'' &= n \frac{T_m}{J_m} - \frac{C}{J} \dot{w} - \frac{w}{J} \\
    J_i \ddot{\theta}_i &= \bar{C} \dot{w} + w
\end{align*}
\] (2.20)  (2.21)

To separate the effects of high frequency dynamics due to the fast subsystem, from the overall flexible-joint model, subtract equation 2.20 from equation 2.18 and equation 2.21 from equation 2.19. Thus,

\[
\begin{align*}
    - \frac{\dot{w}}{J} - \frac{C}{J} \dot{w}' &= \dot{w}'' \\
    \bar{C} \dot{w}' + \dot{w} &= 0
\end{align*}
\] (2.22)  (2.23)

where, the "hat", \(^\hat{\cdot}\) indicates variables associated with high frequency dynamics. Clearly,

\[
\begin{align*}
    \dot{w} &= (w - \bar{w}) \\
    \dot{w}' &= (w' - 0) \\
    \dot{w}'' &= (w'' - 0).
\end{align*}
\] (2.24)  (2.25)  (2.26)

Hence, Equations 2.22 and 2.23 represent the high frequency dynamics of the flexible joint, due to the compliance of the system.
2.3.3 Stability Analysis of Mechanical Systems With Transmission Compliance

Consider the high frequency dynamics of the flexible joint model, represented by equation 2.22. It is apparent that the rigid body dynamics of the afore-said system is stable. Therefore, the overall system will be stable if and only if the high frequency dynamics of the flexible joint model does not destabilize the system [15]. Hence, if we can prove that equation 2.22 is asymptotically stable, then the overall system will be stable.

Transforming equation 2.22 to Laplace transform domain, we have

\[ Js^2 \ddot{w}(s) + Cs \dot{w}(s) + \dot{w}(s) = 0 \Rightarrow s = \frac{-\bar{C} \pm \sqrt{\bar{C}^2 - 4J}}{2} \quad (2.27) \]

We can prove that this system will be stable if the poles of the characteristic equation lie on the negative half of the s-plane. As the parameters \( C \) and \( J \) are physical constants, they are positive definite. Further analyzing the determinant of the characteristic equation 2.27 reveals the following three cases:

1. \((\bar{C}^2 - 4J) = 0\)
   \[ \Rightarrow s < 0 \]
2. \((\bar{C}^2 - 4J) < 0\)
   \[ \Rightarrow s < 0 \]
3. \((\bar{C}^2 - 4J) > 0\)
   \[ \Rightarrow s < 0 \]

In all the three cases, it can be seen that the poles lie on the negative half of the s-plane. This proves that the high frequency dynamics of the flexible joint system are asymptotically stable.

Also, the complete solution of the flexible-joint system dynamics, consists of two components, namely, that due to the slow subsystem and the other due to the fast subsystem. i.e.,
Complete solution = Slow-system solution + $O(\mu)$

where, $O(\mu)$ is the perturbation due to the fast subsystem dynamics. Since the fast subsystem has been proved to be asymptotically stable, Tikonov’s theorem [15] proves that the overall flexible-joint system represented by equation 2.22 is stable.
Chapter 3

Dynamic Modeling of Flexible-Joint Robots

Compliance in the mechanical structure of a robot, complicates the dynamics of the system as it is no longer governed by rigid body dynamics alone. The nonlinear system would consist of a slow subsystem which is contributed by the rigid body dynamics and a fast subsystem which is a consequence of the flexibility in the system. High frequency dynamics thus introduced into the system due to compliance can be modelled by the singular perturbation method, discussed in Section 2.3.2.

Two principal sources of flexibility are link flexibility and joint flexibility. Although both types of compliance can be minimized by mechanical design and proper manufacturing methods, they cannot be completely eliminated. Though direct drive design can be employed to overcome the demerits of transmission compliance, in existing robots, mechanical improvements may not be possible or may prove to be too expensive to be economically feasible. Hence the effects of compliance can be overcome by suitable control strategies. This calls for a knowledge of the high frequency dynamics that are introduced by compliance. If the variables to be controlled can be measured, a controller can be designed to improve the real time response and the control bandwidth of the robot. This chapter focusses on modeling the dynamics of a CRS A-460 robot, incorporating the joint-flexibility introduced by the harmonic drives of the first three joints.

20
3.1 The Dynamic Model

Robot dynamics has been traditionally dealt with respect to rigid manipulators in the literature, by Asada and Slotine (1986), Spong and Vidyasagar (1989). This section discusses about the dynamic model of a jointed-arm, vertically articulated open-chain kinematic configuration type of robot from the rigid body perspective as well as the flexible system approach. However, the flexible manipulator dynamics are curtailed to the treatment of flexibility in the joints of the mechanism.

3.1.1 Rigid Manipulator Dynamics

The dynamic model of a rigid manipulator can be represented as follows:

\[
J(\theta)\ddot{\theta} + C(\theta, \dot{\theta}) + G(\theta) = T
\]  \hspace{1cm} (3.1)

where,

\[J(\theta) \in \mathbb{R}^{n \times n}\] is the inertia matrix of the robot,
\[C(\theta, \dot{\theta}) \in \mathbb{R}^{n}\] is the vector of centrifugal and Coriolis forces expressed in the robot joint space,
\[G \in \mathbb{R}^{n}\] is the vector of gravitational forces,
\[T \in \mathbb{R}^{n}\] is the vector of joint torque input,
\[\theta \in \mathbb{R}^{n}\] is the vector of joint coordinates of the robot.

3.1.2 Flexible-Joint Robot Dynamics

As mentioned earlier, this thesis focusses on the dynamics due to joint-flexibility effects alone. In this section, the dynamic model of a CRS A460 robot is presented with respect to the flexibility effects of the harmonic drives of the first three joints, namely, waist, shoulder and elbow. Through Lagrangian principles, the high-speed dynamics of the flexible-joint robot can be modelled as follows:
\[
\begin{bmatrix}
  n_1 & 0 & 0 \\
  0 & n_2 & 0 \\
  0 & n_2^2 & -n_3
\end{bmatrix}
\begin{bmatrix}
  T_{m1} \\
  T_{m2} \\
  T_{m3}
\end{bmatrix}
= \begin{bmatrix}
  J_{m1} & 0 & 0 \\
  0 & J_{m2} & 0 \\
  0 & 0 & J_{m3}
\end{bmatrix}
\begin{bmatrix}
  \ddot{\theta}_1 \\
  \ddot{\theta}_2 \\
  \ddot{\theta}_3
\end{bmatrix}
\]

\[
\begin{bmatrix}
  J_{m1} & 0 & 0 \\
  0 & J_{m2} & 0 \\
  0 & 0 & J_{m3}
\end{bmatrix}
\begin{bmatrix}
  \ddot{\delta}_1 \\
  \ddot{\delta}_2 \\
  \ddot{\delta}_3
\end{bmatrix}
- \begin{bmatrix}
  R_1 & 0 & 0 \\
  0 & R_2 & 0 \\
  0 & 0 & R_3
\end{bmatrix}
\begin{bmatrix}
  \dot{\delta}_1 \\
  \dot{\delta}_2 \\
  \dot{\delta}_3
\end{bmatrix}
\]

\[
\begin{bmatrix}
  K_1 & 0 & 0 \\
  0 & K_2 & 0 \\
  0 & 0 & K_3
\end{bmatrix}
\begin{bmatrix}
  \delta_1 \\
  \delta_2 \\
  \delta_3
\end{bmatrix}
= (3.2)
\]

where,

- \( n_i \) = transmission matrix
- \( T_{m} \) = motor input torque vector
- \( J_{m} \) = rotor inertia matrix, consisting of the motor and the wave generator inertia, referred to the load-side
- \( \ddot{\theta}_i \) = output link acceleration vector
- \( \delta_i \) = joint-compliance vector
- \( K_i \) = joint stiffness matrix
- \( R_i \) = matrix of joint damping coefficients

Note that the transmission matrix is not diagonal due to the transmission coupling effect between joint 2 and joint 3.

Following the same notation, the slow subsystem due to the rigid body dynamics can be formulated as follows:

\[
\mathbf{rT}_m = \mathbf{J}\ddot{\theta} - \mathbf{J}_m\ddot{\delta} + \mathbf{G}(\theta) + \mathbf{C}(\theta, \dot{\theta})
\]  

(3.3)
\[ C(\theta, \dot{\theta}) = \text{vector of centrifugal and coriolis forces} \]
\[ G(\theta) = \text{vector of gravity forces} \]
\[ J = \text{overall effective inertia matrix, referred to the load-side} \]
\[ \ddot{\delta}_i = \text{second derivative of the joint-compliance} \]
\[ = \text{vector with respect to time} \]

### 3.2 Singular Perturbation Analysis of the Flexible-Joint Robot

The flexible-joint robot model represented by equations 3.2 and 3.3 can be analyzed by the singular perturbation method discussed in Section 2.3.2. By solving equation 3.2 for \( \ddot{\theta} \) and substituting in equation 3.3, we have

\[
\mathbf{nT}_m(1 - J_m^{-1}J)(\ddot{\theta}) + J_m^{-1}JR\dot{\delta} + J_m^{-1}JK\delta + C(\theta, \dot{\theta}) + G(\theta) = 0
\]

(3.4)

In dealing with robot joints, we readily have a singular perturbation parameter, the transmission compliance of the joint itself. Hence, any dynamic model accounting for transmission compliance, can be analyzed by the singular perturbation method discussed in Section 2.3.2, by decomposing the original model into its constituent slow and fast subsystems. Therefore, the singular perturbation parameter,

\[
\mu = \frac{1}{\mathbf{K}}
\]

where, \( \mathbf{K} = \text{diag}\{k_1, k_2, \ldots, k_n\} \). Assuming \( k_i \) to be equal and applying the transformation,

\[
\mathbf{K}\delta = \mathbf{w}
\]

such that,

\[
\mu\mathbf{w} = \delta
\]
In practice it is quite reasonable to assume that the joint stiffness is quite high in industrial robots using a high gear reduction as in harmonic drive driven robot joints. It is known that the flexspline of the harmonic drive exhibits high stiffness of the order of $10^4$ Nm/rad [27]. This assumption of high stiffness would make the perturbation variable tend to zero and thereby simplify the higher order flexible model, without largely compromising on the accuracy of the model.

This fact leads to the singularity condition of the singular perturbation model 3.5 of the flexible-joint system. Since $|K| \gg 1$, it follows that $\mu \ll 1$. Thus, setting $|\mu| = 0$, the slow subsystem due to the rigid body dynamics can be regenerated as follows:

$$nT_m(1 - J_m^{-1}J) = J_m^{-1}Jw + C(\theta, \dot{\theta}) + G(\theta)$$

(3.6)

In order to reveal the fast subsystem dynamics, the time scale can be stretched by defining the following transformations:

$$t = \sqrt{\mu \tau}R = \sqrt{\mu}R$$

where, $\tau$ is the new time variable in the stretched time scale.

Upon stretching the time scale, Equation 3.5 becomes,

$$nT_m(1 - J_m^{-1}J) = (J - J_m)w^{''} + J_m^{-1}JRw' + J_m^{-1}Jw + C(\theta, \frac{1}{\sqrt{\mu}}\theta') + G(\theta)$$

(3.7)

Defining the high speed variables,

$$\dot{w} = w - \overline{w}$$

$$\dot{w}' = w' - 0$$

$$\dot{w}'' = w'' - 0$$

24
and substituting in equation 3.7, the high frequency dynamics of the system can be represented as,

\[(J - J_m)\ddot{w} + J_m^{-1}J\dot{\bar{R}}\dot{w} + J_m^{-1}J\dot{w} = 0\] (3.8)

where, the prime symbol ' indicates differentiation, * indicates variables associated with high frequency dynamics while the overline - refers to the slow system variables.

### 3.2.1 Stability of the Flexible-Joint Robot System

It is clear that the slow subsystem 3.3 of the flexible-joint system modelled by equations 3.2 and 3.3, is stable. As discussed in Section 2.3.2 for a one d.o.f. system, it can also be proved for the multiple d.o.f. robot system by Tikonov's theorem [15] that the stability of the overall system hinges upon the stability of the fast subsystem dynamics represented by equation 3.2. Therefore to prove that the overall system will be stable, it is sufficient to show that the high frequency dynamics of the fast subsystem, represented by equation 3.8, is asymptotically stable [15].

Upon analyzing equation 3.8, it can be seen that all the parameters involved are physical quantities and hence, are positive definite. Furthermore, the term \((J - J_m)\) is always positive, since the quantity \('J'\) is the overall effective inertia of all the components of the flexible joint, referred to the load-side, while the quantity \('J_m'\) is the inertia of the motor armature and the wave generator alone, referred to the load-side and hence, the latter is always smaller than the former.

\[(J - J_m) \geq 0\]

Clearly, all the coefficients \(w\) and its derivatives in equation 3.8 are positive definite. As a result, it is proved that the high frequency dynamics, shown in equation 3.2, of the flexible-joint system represented by equations 3.2 and 3.3, is asymptotically stable. Therefore by Tikonov's theorem [15], it can be concluded that the application of a rigid control law to a system with joint-flexibility, does not destabilize the system.
Chapter 4

Computed Torque Control

4.1 Why Computed Torque Control?

The ever-growing demand for speed and accuracy, at higher payload capacity has led to the burgeoning of model-based control strategies. Speed and accuracy are at loggerheads. A given task can be performed accurately at a slower speed while the same task if done faster, might incur a penalty in the accuracy with which the task is performed. The situation is further aggravated if the task is to be performed with a larger payload. On the other hand, performance can also deteriorate at very slow speeds with large payloads. Because at higher speeds, the inertia of the system, provides enough momentum to follow a given trajectory without much discontinuity, thereby resulting in a "smoother" position trajectory. However, at very slow speeds, due to the lack of any significant inertial effects, the system tends to yield relatively easily, to any inherent disturbances like frictional effects, environmental interaction effects and the dynamics of the payload.

Besides speed, accuracy and payload, the application for which the system is designed also plays a major role in evaluating the importance of the method of control. For instance, in safety-critical applications like robot-assisted surgery, nuclear materials handling in reactors, hazardous waste disposal, space-craft manoeuvres in outer-space, settling for a simpler control method would amount to compromising on the safety standards. In contact applications like robotic deburring, the dynamics of
the closed-loop system, demands a robust controller which can reject external disturbances introduced by the process of application. Hence, for any method which does not even compensate for the dynamics of the system, let alone the dynamics of the payload and the environment, it is difficult, if not impossible, to meet all the specifications of speed, accuracy and payload. With the computed torque control method, if the model of the system is known reasonably accurately, then the system dynamics can be compensated effectively, thereby minimizing deviations from the desired trajectory.

4.1.1 Shortcomings of The PID Control Method

One of the most commonly used commercial robot controllers is the PID controller or more specifically, the independent-joint PID controller. The PID controller is popular because of its simplicity: computationally less intensive than model-based control methods like the computed torque control technique, while being robust enough for most of the less demanding applications. However, the PID control method suffers from the following drawbacks, for controlling a complex electro-mechanical system like a robot:

1. Only kinematic constraints are taken into account while rigid body dynamics are treated as disturbances: The task of the control system is to move the robot from the present position in space to the target position, without regard to the dynamical effects like forces and moments, that may come into play during the motion, affecting the very execution of the task itself. This negligence of dynamics of motion, is suitable for slow speeds and small payloads at low acceleration, in which case dynamic effects caused by rigid-body dynamics, like inertia forces, centrifugal forces, coriolis forces, gravitational forces, viscous friction forces and nonlinear coupling effects are minimal. However, at higher acceleration, higher velocities and larger payload, rigid body dynamics become dominant and hence, if neglected, result in larger performance errors.
2. The underlying assumption made justifying the use of an independent-joint controller is that the joints of the mechanism are dynamically decoupled, with each joint treated as though it is independent of the other joints of the mechanism of the robot: In reality, the interaction between the various links of a robot manipulator, are significant and are cross-coupled through nonlinear dynamics. Hence when the controller attempts to compensate for the performance error of one joint without taking into account the influence of the other joints upon this particular joint, it is difficult to attain the set-point condition in real-time. Any compensatory action taken at one joint may introduce in an offset in the set-point at the rest of the joints. This interaction effect becomes more pronounced at higher speeds and larger payloads.

3. The controller is purely error driven: Error is the cause and the controller action compensating for the error, is the result, although the control signal may or may not be able to completely eliminate the error in real-time. Faster control action can be achieved by using larger gains, however, higher gains call for higher sampling rates, which is limited by the availability of computing hardware. Besides, larger gains may lead to instability. Therefore, for a given hardware, there is an inherent saturation limit for the accuracy that can be attained by the controller.

4.1.2 Merits of The Computed Torque Control Method

Though the Computed Torque Control method is computationally demanding, it overcomes the drawbacks of the Independent-Joint PID control method. Computing power is no longer a bottleneck as present day computers are much faster than those that were available when the PID control method was available. Parallel processing using transputers/ DSPs enables the complete computation of inverse dynamics of the CRS A460 robot in real-time, within one millisecond. The advantages of computed torque control can be summarized as follows:
1. As the controller uses a mathematical model of the system dynamics, in order to generate a control signal to compensate for the rigid-body dynamics of the physical system, performance is superior to that of the PID controller, under similar operating conditions. This difference is more pronounced at very high speeds and high acceleration with a massive payload.

2. Since the exact torque demanded by a particular motion with a particular payload, is computed and is known, it is possible to protect the system from unexpectedly large torques which may arise due to collisions with the environment, by adopting on-line safety measures.

3. The computed torque control method is practically robust to external disturbances like payload variations if the dynamics of the payload is less significant compared to the dynamics of the system under control. However, in situations like handling a large mass of liquid payload, the dynamics of the payload cannot be treated as disturbances and hence, must be incorporated in the dynamics model of the system under control.

4. Compared to the cause-and-effect type of control employed by the PID controller which reacts only after an error occurs, the computed torque controller is an anticipatory type of controller due to the feed-forward component of the control system. Due to inevitable modelling inaccuracies, there is, however, a small error in the prediction made and this is corrected by the feedback component of the controller. This ability of the computed torque controller to actually predict the appropriate control signal to compensate for errors, if any, that may arise during the motion in real-time, is the power of the method.

5. The controller utilizes the model of the system in the feedback loop in order to decouple the dynamics of the system, in real-time. As a result, each joint of the cross-coupled dynamic system can be controlled independently, once the control signals have been computed.
6. Though the computed torque controller uses the dynamics model of the system under control to compute the control signals, model inaccuracies have minimal effect on performance [16]. Robustness of the computed torque control method is a result of using the model in addition to the PID control technique.

4.2 What is Computed Torque Control?

Computed Torque Control [1] is a model-based control method which utilizes the dynamics model of the system to compute the control torque signals that are input to the system, given the present state of the system, in order to bring about the desired motion. Computed Torque Control is regarded as a form of Non-Linearity Cancellation Technique [1], for if the dynamics model of the system is exact, then the nonlinear dynamic perturbations are exactly cancelled. Figure 4.1 illustrates a generalized computed torque control system applied to a robot.
In principle, the computed torque control method is a combination of feedforward and feedback control. The feedforward component is used to compute the control torque signal using the inverse dynamics model while the feedback control component employs a PD (Proportional-Derivative) control law, to estimate a quantity called Corrected Acceleration [1], which is to be attained by the system in order to achieve the desired position at the desired velocity within the desired time, carrying the desired payload and/or imparting the desired force on the environment. The corrected acceleration is used in the place of inertial acceleration, to compute the inertia forces that would come into play during the motion. Other components of the control torque signal, required to compensate for gravitational forces, Coriolis and centrifugal forces, and friction forces, are calculated using the inverse dynamic model of the system. The feedforward computation is done on the basis of the actual trajectory; hence inverse dynamics computations must be done in real-time. Thus the overall control torque signal is computed, every sampling period, using the dynamic model, by knowing the present state of the system, and the desired state, under the temporal and spatial constraints.

Typically, the computed torque control system consists of an inverse dynamics computation module, a trajectory planner, an inverse kinematics module and a torque servo module. The trajectory planner could be part of a task-planning module which generates the sequence of tasks to be carried out by the robot. The task-planning module is at the highest rung in the hierarchy of the control system software. The trajectory planner which is at the next lower level, would eventually generate kinematic target commands, with respect to the task coordinate frame. Through inverse kinematics, the Cartesian target commands - desired position, desired velocity and desired acceleration, are converted to the corresponding joint space commands. Knowing the actual position and the actual velocity in the joint space coordinates, the inverse dynamics module would compute the control torque signals for that particular sampling period, using the inverse dynamic model of the system. The torque servo module forms the lowest level in the control hierarchy; it accepts computed torque or current commands from the inverse dynamics module and executes a low-level servo
loop inorder to bring about the desired motion.

The greatest challenge of using the computed torque control method is that of computing the complete inverse dynamics model of the system, among other calculations, within the sampling period. This is quite challenging, as the sampling period employed in commonly used digital control systems is of the order only a few milliseconds. However, thanks to the modern techniques of parallel computing, with the aid of the present-day commercially available parallel processors like transputers and DSPs (Digital Signal Processors), which employ RISC (Reduced Instruction Set Computing) based design, there is a tremendous amount of computing power available at the disposal of the common user. Furthermore, higher computationally intensive tasks can also be performed using more powerful massively parallel architectures.

4.2.1 The Computed Torque Control Law

Consider the typical robot manipulator system, represented by the following dynamic model:

\[ T = J\ddot{\theta} + G(\theta) + C(\theta, \dot{\theta}) + F(\theta, \dot{\theta}) \]  

(4.1)

where,

\( J(\theta) \in \mathbb{R}^{n \times n} \) is the inertia matrix of the robot,

\( C(\theta, \dot{\theta}) \in \mathbb{R}^{n} \) is the vector of centrifugal and Coriolis forces expressed in the robot joint space,

\( G \in \mathbb{R}^{n} \) is the vector of gravitational forces,

\( T \in \mathbb{R}^{n} \) is the vector of joint torque input,

\( F(\theta, \dot{\theta}) \in \mathbb{R}^{n} \) is the vector of static and viscous frictional forces expressed in the robot joint space,

\( \theta \in \mathbb{R}^{n} \) is the vector of joint coordinates of the robot.

The computed torque control law for the model 4.1 is stated as follows:

\[ T = J\ddot{\theta}^* + G(\theta) + C(\theta, \dot{\theta}) + F(\theta, \dot{\theta}) \]  

(4.2)
where, all the terms are as defined in model 4.1, except the corrected acceleration term, $\ddot{\theta}^*$, which is defined as follows:

$$\ddot{\theta}^* = \ddot{\theta}_d + K_p(\theta_d - \theta) + K_v(\dot{\theta}_d - \dot{\theta})$$  (4.3)

where,

$\ddot{\theta}_d \in \mathbb{R}^n$ is the vector of desired acceleration with respect to the joint coordinates of the robot
$
\dot{\theta}_d \in \mathbb{R}^n$ is the vector of desired velocity with respect to the joint coordinates of the robot
$\ddot{\theta} \in \mathbb{R}^n$ is the vector of actual velocity with respect to the joint coordinates
$\theta_d \in \mathbb{R}^n$ is the vector of desired position with respect to the joint coordinates
$\theta \in \mathbb{R}^n$ is the vector of actual position with respect to the joint coordinates
$K_p$ is the proportional gain matrix
$K_v$ is the derivative gain matrix

The proportional and the derivative gains can be designed by knowing the physical characteristics of the system or by following empirical tuning methods like the Zeigler-Nichols method [33]. Thus, for every sampling period, from the position and velocity feedback information, the corrected acceleration is estimated using Equation 4.3 and thereby, the control torque can be computed using the control law 4.2. The computed torque signal is further communicated to the torque servo module for real-time execution. This process is carried out repeatedly, through the trajectory period, at regular sampling intervals, until the desired trajectory is completed.
Chapter 5

Experimental Set-up

5.1 Hardware

The experimental subject for the computed torque control experiments described in this thesis is a CRS Robotics Corporation A460 type robot with a C500 controller, which is a six-axis vertically articulated jointed arm robot manipulator, having a payload capacity of 2.95Kg (6.5 lb) and capable of attaining a peak joint acceleration of 12.56 rad/s² and a maximum joint velocity of 3.14 rad/s.

In order to conduct the experiment with different payloads, circular steel discs, each weighing 0.23Kg (0.5Lb), were designed such that they can be bolted on to the face plate of the manipulator wrist.

5.2 Software

The real-time control system software runs in a network of six transputers, configured in a customized pipeline architecture, such that the channels of communication between the transputers is serial in nature, as shown in Figure 5.2. The various transputer modules are symbolically named as Runex, Traj, Control, Fti, Servo and Kin. Runex serves as a user interface, Traj does trajectory calculations to generate online setpoint commands for the Control module which is responsible for the inverse dynamics computations and control signal generation. Servo is dedicated to low-level
Figure 5.1: The CRS A460 Robot
servo loop control, while Kin does the inverse kinematics calculations. Fti is an interface for the force/torque sensor mounted on the robot endeffector. The control software is executed within a cycle time of 1ms, that is, at the rate of 1000Hz. The interested reader is referred to [22] for more detail.

5.3 Computed Torque Control with the Inner Torque-Loop

The control torque signal that is computed by the control algorithm has to be executed by the motor control hardware. The torque delivered by the motor or the load torque that can be overcome by the motor must be equal to the commanded torque control signal. This can be accomplished by executing a torque servo loop at the lowest or the innermost level and hence the name, Inner Torque Loop. The method of using the inner torque loop to servo the commanded torque, is illustrated in Figure 5.3.

The inner torque loop can be considered as another PID control loop at the lowest level. The set point to the torque servo loop is the control torque computed by the computed torque controller. The actual load torque on the joint can be measured using a suitably calibrated strain-gauge bridge mounted on the flexspline of the harmonic drive of the joint motor. This measured torque can be used as a feedback on the commanded torque and hence a closed-loop torque control scheme can be developed. This method was adopted by [20], [11] and [14], previously.

The method of using the inner torque loop to servo the computed torque, although straightforward, suffers from the following drawbacks:

- The strain gauges mounted on the flexspline are subject to continuously alternating tension and compression, during the operation of the harmonic drive. As a result of the fatigue loading, the bonding between the strain gauges used for torque-sensing, and the surface of the flexspline, is susceptible to failure. This would result in spurious torque feedback signals, leading to poor performance or even instability.
Figure 5.2: Topology of the transputer network [5]
Frequent preventive maintenance procedures must be adopted to ensure the proper operation of the strain-gauge based torque feedback loop. This requires trained personnel, to follow appropriate calibration procedures, because once the settings are changed, the strain-gauge sensor has to be recalibrated for correct readings. This would result in longer downtime of the machine, during maintenance.

Initial installation as well as operating costs are substantial due to the requirement of additional hardware and software, to implement the torque sensing scheme, using strain-gauges.

5.4 Computed Torque Control without the Inner Torque-Loop

The success of any closed-loop control problem hinges upon the fidelity of the feedback signal, among other parameters. Moreover, closing the servo loop at the lowest control level is detrimental to the control system, if the feedback signal is unreliable. This thesis presents an alternate method to overcome the problems of the afore-said inner torque loop - elimination of the inner torque loop. The proposed scheme is illustrated in Figure 5.4.
5.4.1 The Current-Based Torque Control Method

The torque generated by an electric motor is a function of the current drawn by the motor. Depending on the type of the motor, this functionality could be as simple a linear relationship as in a constantly excited armature controlled DC motor, or as complex as in a AC induction motor. Thus, the required information about the torque delivered by the motor is embedded in the current drawn by it. This is the crux of the current-based torque sensing method. Fortunately, servo motors have a linear current-torque relationship, which is the reason for their ease of control and hence, their popularity. In the case of a permanent magnet armature-controlled DC motor, under a constant exciting voltage, the current flowing through the armature is directly proportional to the torque acting on the motor shaft. Commercially available permanent magnet DC servo motors are designed to have a linearized operating torque-current characteristics.

Mathematically,

\[ T_m = K_i I_a \]  \hspace{1cm} (5.1)

where,

\[ T_m = \text{torque on the motor shaft} \]
\[ K_t = \text{Torque constant} \]
\[ I_a = \text{current drawn by the armature}. \]

Using Equation 5.1, the current that will be drawn by the armature of the motor, for a computed load torque, can be calculated. Thus, it follows that

\[ I_a = \frac{T_l}{nK_t}. \] (5.2)

Therefore, using the armature current information, it is possible to ascertain the torque delivered by the motor or vice versa. The current flowing through the motor armature circuit can be sensed by including a current-sensing resistor in series with the armature coil, and measuring the voltage drop across the resistor [28]. This sensed armature current can be used as a feedback signal to the current command calculated in Equation 5.2. Thus the torque servo control problem discussed in Section 5.3, transforms into a current control problem.

The servo amplifier of the joint motors of the CRS A460 robot, has the feature of switching between current mode and voltage mode. This further simplifies the problem of implementing the armature current control loop. By switching the mode of the amplifier through the specified switch settings provided by the manufacturer, the robot controller can be set to operate in the current control mode as shown in figure 5.5. This is the simplicity of the method and it becomes very cost effective in modifying existing robots to operate under the computed torque control method without the inner torque loop.

It is to be understood that although the inner torque loop is eliminated from the digital control system, the process of controlling the armature current of the motor and hence its shaft torque, is done through the analog control loop implemented in the current-control feature of the servo amplifier. Hence, the current control scheme is in fact a feedback control method, and not a feedforward control as viewed from the digital component of the controller.

The final step in the method, is to calibrate the armature circuit with the amplifier in the current mode, against the DAC (Digital-to-Analog Converter) input data. The
Figure 5.5: The Armature Current Control Scheme

\[ V_o = V_i \frac{R_f}{R_i} \]
\[ V_o = I_a R_o \]
\[ I_a = V_i \frac{R_f}{(R_i + R_o)} \]
DAC is the interface between the amplifier and the transputer. The calibration curve is linear within the operating range.

The method of computed torque control without the inner torque loop, compared to the method using the inner torque loop described in Section 5.3, has the following advantages:

- By knowing the exact current required for a particular motion, with a particular payload, the control system can conduct on-line safety checks to detect the feasibility of a user-specified operation by referring to the design current limits of the system. The significance of this in-built safety feature of current-based computed torque control without inner torque loop, can be better understood by comparison with voltage-based control methods. In the latter case, where only the terminal voltage is controlled, it is still possible for the motor to draw dangerously large amounts of current due to unforeseen loads, resulting in serious damages to the motor or complete system shutdown due to fuse blow-outs.

- Unlike the strain gauge based torque sensing method, the current based torque sensing method is quite rugged, due to the reason that no mechanically moving parts are involved in the sensing process and hence, less wear and tear.

- The method results in practically maintenance-free operation, thereby resulting in shorter downtime.

- It is quite cost-effective, due to the optimum utilization minimal hardware resources, besides eliminating extra software requirements due to the eradication of the inner torque loop. Moreover, it is easier to adapt the existing machines to this method with fewer modifications.
Chapter 6

Experimental Results

The aim of the computed torque control experiments conducted on the set-up described in the previous chapter is three-fold:

- Firstly, to verify the theoretical proof discussed in Section 3.2.1, that the application of a rigid control law on a flexible-joint robotic system such as the CRS A460 robot, does not destabilize the system.

- Secondly, to prove that the computed torque control scheme can provide superior performance to that of the independent-joint PID control scheme, originally adopted by the manufacturer.

- Finally, to eliminate the inner torque loop described in Section 5.3, and test the current-based torque control technique by replacing the unreliable strain-gauge based torque sensing method with the cost-effective and rugged method of armature current control of permanent magnet DC servo motors, discussed in Section 5.4.
Chapter 6

Experimental Results

The aim of the computed torque control experiments conducted on the set-up described in the previous chapter is three-fold:

- Firstly, to verify the theoretical proof discussed in Section 3.2.1. that the application of a rigid control law on a flexible-joint robotic system such as the CRS A460 robot, does not destabilize the system.

- Secondly, to prove that the computed torque control scheme can provide superior performance than the independent-joint PID control scheme, originally adopted by the manufacturer.

- Finally, to eliminate the inner torque loop described in Section 5.3. and test the current-based torque control technique by replacing the unreliable strain-gauge based torque sensing method with the cost-effective and rugged method of armature current control of permanent magnet DC servo motors, discussed in Section 5.4.
6.1 Design of Experiments

The main objective, being comparative performance evaluation of a computed torque controller as against an independent-joint PID controller, the tests to be conducted were categorized on the basis of the controller employed, as follows:

1. Independent-Joint PID Control
2. Independent-Joint PID Control with Gravity Compensation
3. Computed Torque Control without Inner Torque Loop

Since Independent-Joint PID control was the controller commercially provided by the manufacturer, the gain values originally designed by CRS Robotics Corporation, have been adopted for the customized PID controller used for testing purposes. It is of further interest to ascertain whether gravitational effects play a significant role on real-time tracking performance of the robot. Besides, gravitational offsets can be compensated with relatively lesser computation, compared to the complete inverse dynamics model of the robot. Therefore, the PID controller was augmented with a gravity compensator, to study the effects of gravity compensation on tracking accuracy. All the three controllers were tested under identical operating conditions, to facilitate comparative evaluation.

6.1.1 Parameters of Control

The criterion for evaluating the efficiency of a control algorithm, for the experiments described in this thesis, is the positional accuracy of the robot. It has been observed that the tracking performance of the robot is a function of its peak acceleration, maximum operating speed, payload carried by the end-effector and the initial kinematic configuration of the robot, among various other parameters that could possibly affect its operating characteristics. Therefore the experiments call for the investigation of maximum acceleration, maximum velocity, payload carried by the robot and the starting kinematic configuration of the mechanism, on the tracking accuracy of the robot.
In order to minimize the number of control parameters, the peak acceleration of all the three joints was set to the maximum design limit of 12.56 radians/s², in joint space. The experiments were conducted at three different speeds—very slow speed, medium speed and high speed—with frequent reversal of directions. Accordingly, the maximum operating speeds in joint space for the three respective categories of experiments were chosen to be 0.1 radian/s, 1 radian/s and the maximum design limit of 3.14 radian/s, respectively.

The effect of payload variation on performance was observed by varying the payload in steps of 0.23Kg (0.5Lb) to the maximum design limit of 2.72Kg (6Lb). Finally, the kinematic configuration of the starting position of the robot mechanism was varied for different sets of experiments to observe its effect on the performance of the robot. Starting positions, such as arm vertically up, arm horizontal, and various other configurations were adopted, in order to cover the entire work envelope of the robot and to subject the robot to varying frictional effects and different starting gravitational forces/torques.

6.1.2 Parameters of Observation

It is of interest to measure the real-time positional tracking accuracy of the robot. Hence, the joint encoders are sampled at 100Hz for recording, although the actual control sampling frequency is 1000Hz. From the shaft encoder readings, the joint-wise positional errors are calculated. Besides position errors, the computed torque signal and hence the current command signals are evaluated. The current command signal gives an idea of the current waveform generated in the armature in real-time. Moreover, it is useful in predicting the operating life of the motor and other power electronics, by knowing the maximum current drawn.
6.2 Results and Discussion

Graphical results of the various experiments mentioned previously are illustrated joint-wise, in this section. This provides a comprehension of the relative performance of the afore-said controllers on the real-time trajectory tracking performance of the robot. The experiments involve a combination of different manoeuvres covering the entire work envelope, so as to evaluate the performance of the respective controllers without spatial biases, if any. As mentioned earlier, in this thesis, real-time position error is used as the major criterion for evaluating the performance of a controller. The maximum current drawn by the motor is also considered a critical design condition for evaluating the motion control system, as larger current surges would result in shorter brush life of the motor.
Figure 6.1: Position history of joint-1 under computed torque control

Payload = 5.5 Lbf
Max. joint velocity = 3 rad/s

Joint #1: Desired & Actual Position Trajectory

Joint #1: Position Error

Joint #1: Computed Torque Command

Joint #1: Computed Current Command
Payload = 5.5 Lbf
Max. joint velocity = 3 rad/s

Figure 6.2: Position history of joint-1 under independent-joint PD control
Payload = 5.5 Lbf  
Max. joint velocity = 3 rad/s

Figure 6.3: Position history of joint-1 under PD control with gravity compensation
Payload = 5.5 Lbf
Max. joint velocity = 3 rad/s

Figure 6.4: Position history of joint-2 under computed torque control
Figure 6.5: Position history of joint-2 under independent-joint PD control

Payload = 5.5 Lbf
Max. joint velocity = 3 rad/s
Payload = 5.5 Lbf  
Max. joint velocity = 3 rad/s

Figure 6.6: Position history of joint-2 under PD control with gravity compensation
Payload = 5.5 Lbf
Max. joint velocity = 3 rad/s

Figure 6.7: Position history of joint-3 under computed torque control
Payload = 5.5 Lbf
Max. joint velocity = 3 rad/s

Figure 6.8: Position history of joint-3 under independent-joint PD control
Payload = 5.5 Lbf
Max. joint velocity = 3 rad/s

Figure 6.9: Position history of joint-3 under PD control with gravity compensation
6.2.1 Discussion

Each of the graphs typically shows the desired position trajectory in joint space co-ordinates, the actual trajectory of the joint, the joint-wise position error, the computed torque command and the current signal. Consider the computed torque control experimental results shown in Figures 6.1, 6.4 and 6.7. The elbow joint-3 resulted in the highest position error compared to the base joint-1 and the shoulder joint-2. This is due to the additional unmodelled flexibility effects of the timing belt used in the elbow joint transmission.

It is observed that the computed torque controller without the inner torque loop performs much better than the PID-based independent-joint controller. The position error in the case of a PID controller is quite high compared that of a computed torque controller. For instance in the case of the elbow joint-3, as illustrated by Figures 6.9 and 6.7, position error attained by the PID controller was about 20 times the error that occurred due to the computed torque controller. This can be attributed to the fact that while the PID controller treats the dynamics of the robot as disturbances, compensating only for kinematic aberrations, the computed torque controller takes into account the complete dynamics of the system. Hence, better compensation results in better performance, especially at high operating speeds and higher payloads.

The results show that the effect of gravity compensation on the performance of a PID controller was not significant as evident from Figures 6.5 and 6.6. For instance, showing similar position-error profiles. This is due to the reason that at higher acceleration, speed and payload, inertia forces and other interaction forces are predominant over gravitational effects. Therefore, mere compensation of gravitational effects without regard to other dominating factors like inertial effects, as treated by an independent-joint PID controller, does not improve performance.

Irrespective of the type of controller used, the position error profile tends to be oscillatory. The computed torque command and hence the current signal, exhibit high frequency components. The oscillations are higher at lower speeds because the inertial
effects of the system which are predominant at higher speeds are less pronounced at lower operating speeds. Hence at higher speeds, inertia of the system tends to smoothen the oscillations, resulting in a smoother profile of actual position, and hence, a less oscillatory computed torque and current signals.

Generally, the oscillatory behavior of the system can be explained on the basis of the flexibility effects exhibited by the flexspline of the harmonic drive, which were not compensated for, due to the difficulty of measuring joint-compliance and its derivatives, in real-time. The optical shaft-encoder is the only position-sensing device which gives a feedback of the controlled system, to the controller without an inner torque loop. As the elliptical wave generator of the harmonic drive rotates with the motor, the flexspline deforms continuously. When the system attains higher acceleration and higher speed under high payload, the high frequency component of the flexible-joint system is excited. The sensitive optical encoder mounted on the motor shaft, picks up these high frequency oscillations and hence relays the actual oscillatory position feedback to the controller. Since the dynamics model of the rigid-body controller does not account for the flexibility effects of the harmonic drive, the closed loop controller attempts to compensate for the oscillatory behaviour of the system, treating the oscillations as disturbances. Since the loop gain of the system is set high, small disturbances excite the structural oscillations of the flexspline. However, the magnitude of the high frequency oscillations is small and hence the damping effect of the derivative component of the controller protects the system against instability.

The efficacy of the derivative action of the controller is dependent on the fidelity of the velocity feedback. This information, however, is obtained through numerical differentiation of the position feedback signal with respect to time. For the experimental set-up described in the previous chapter, the velocity feedback is derived at every sampling interval, by taking the temporal difference of the present and the previous position samples. This is a first order numerical approximation of the velocity feedback signal. Due to this approximation, the velocity feedback is not the actual velocity of the system, but it tends to approach the actual velocity as the sampling rate is increased. Higher the sampling rate, higher the accuracy of the velocity feedback.
Hence, the overall damping characteristics of the controller and thereby, its ability to compensate for the oscillatory behaviour of the system, is affected by the simple first-order numerical differentiation of actual velocity of the system. This could be overcome by using a higher order differentiation of the position feedback, taking more samples of actual position before deriving the velocity information. Alternate means for measuring velocity through hardware like stroboscopes, tachometers, etc., can be used in the place of a software based velocity estimator.

The current demand is higher when the position error achieved is lower. This is because of the necessity to move the motor abruptly in either direction, to compensate for faster changes in the commanded trajectory without losing fidelity of the demanded torque signal.

It is found that payload variations has little effect on the tracking accuracy of the computed torque controller. This is ample proof for the robustness of the controller and hence, its disturbance rejecting capacity. On the other hand, performance of the PID controller is affected at higher payloads: higher the payload, higher the position error. This is not unexpected because the dynamics of the system are augmented by higher payload and hence cannot be treated as mere disturbances.

It is interesting to note that the elimination of the inner torque loop results in increased robustness of the computed torque controller as compared to the computed torque method with the strain gauge based inner torque loop, used by [5]. Therefore, it can be concluded that the computed torque controller is far superior in trajectory tracking performance compared to the independent-joint PID controller with or without gravity compensation. Moreover the computed torque controller employing current-based torque sensing scheme, as against that with an inner torque loop, is rugged, cost-effective and reliable.
Chapter 7

Concluding Remarks

7.1 Summary

Flexibility in mechanical systems is inevitable. Structural compliance imparts additional degrees of freedom to the otherwise simpler rigid body dynamics, which if neglected, would result in the loss of accuracy or even instability. A flexible system dynamics model can be partitioned into a slow subsystem and a fast subsystem. The slow subsystem is due to the purely rigid body dynamics while the fast subsystem is a result of compliance. The stability of a system with structural compliance can be analyzed by the singular perturbation method.

Flexible transmission effects in robots is primarily due to joint compliance, timing belts, cables, links and other transducers. Joint compliance is a result of the flexspline of harmonic drives used in transmission. The flexible dynamics model of harmonic drives was derived from first principles, in this thesis. The concept of singular perturbation applied to flexible system dynamics was introduced and then applied to the typical case of harmonic drives. Further, a dynamics model of the CRS A460 robot, incorporating the flexibility of the harmonic drives of the first three joints, was presented as derived using the Lagrangian formulation. The technique of singular perturbation was used to analyze the flexible dynamics model of the robot. It was further proved that the application of a rigid control law to the flexible-joint system does not destabilize the system.
The need for computed torque control was emphasized and its merit over the independent-joint PID control method discussed. Then the method of computed torque control was described in detail as applied to the CRS robot. From the practical point of view, the implementation of computed torque control without the inner torque loop was discussed. The advantages of eliminating the inner torque loop was also illustrated.

Finally, the experimental set-up available in the Laboratory for Nonlinear Systems Control, University of Toronto, consisting of the CRS A460 robot and the customised transputer-based computed torque controller, was described. The following experiments were conducted:

- Computed Torque Control Without the Inner Torque Loop
- Independent-Joint PID Control as implemented by CRS Robotics Inc.
- Independent-Joint PID control with Gravity Compensation

The experiments were conducted at different speeds and with different payloads, with manoeuvres spanning the entire work envelope of the robot, keeping the maximum acceleration as per the design limit.
1.2 Conclusion

It was proved that the application of a rigid control law to a flexible-joint system does not destabilize the system. Application of the rigid computed torque control law on the reduced order dynamics model of the flexible-joint CRS A460 robot verifies the theoretical explanation. Under the following conditions,

- maximum joint-acceleration = 100% design limit (12.56 rad/s²)
- maximum joint-velocity = 100% design limit (3.14 rad/s)
- maximum payload = 100% design limit (6 Lbf. 2.72 Kgf)

Tests performed using a computed torque controller and an independent-joint PID controller, successively under identical operating conditions, reveals that, typically, the ratio of peak position errors in joint space, due to the PID controller to that due to the computed torque controller, as follows:

- Joint 1 = 500%
- Joint 2 = 2000%
- Joint 3 = 2000%

Thus it can be concluded that the computed torque controller clearly outperforms the independent-joint PID controller at high as well as slow speeds.

Elimination of the inner torque loop, besides simplifying the controller hardware, helps in the following ways:

- A significant reduction in the initial installation costs in new systems and a reduction in maintenance and operating costs in existing systems, due to the elimination of strain gauges and the associated hardware and software required to sense the feedback torque.
- An increase in the reliability of the system due to the elimination of the problem of bonding strain gauges to the flexspline of the harmonic drive.
- An overall improvement in the performance and the robustness of the control system due to the elimination of a possible noisy feedback in the inner torque loop.

Model inaccuracies have minimal effect on the performance of the computed torque control scheme. The ability of the controller to servo a commanded trajectory with accuracy, and hence its robustness to external disturbances, can be increased by incorporating higher loop gains. However, higher gains would call for higher sampling rates.
7.3 Contributions

There are two major contributions made by this thesis:

- **Derivation of a flexible-joint dynamics model for the CRS A460 robot:** The complete derivation of the flexible-joint dynamics of the A460 robot incorporating transmission flexibility due to the harmonic drives of the first three joints, using Lagrangian mechanics, gives a good understanding of the actual flexible dynamics of the robot.

- **Elimination of the inner torque loop:** Replacing the strain gauge based torque feedback loop with a rugged and cost-effective current-based torque control scheme, has not only resulted in improved performance but has also reduced maintenance costs and downtime. Moreover, this method can be easily adopted to new as well as existing robots, with little investment towards additional hardware and software requirements.

- **Demonstration of the superiority of the computed torque control method:** It has been proved beyond doubt that the computed torque control method outperforms the independent-joint PID control method, traditionally used in most of the industrial robots. The results reveal that the computed torque method is robust to payload variations, even at very high speeds or very slow speeds of operation, compared to the PID controller whose performance deteriorates drastically at high speeds and high payload. It is noteworthy to mention that in some instances, the PID controller resulted in twenty times the error due to the computed torque controller.
7.4 Recommendations For Future Work

7.4.1 Issues Related to Modelling

- The afore-said flexible-joint model of the CRS robot can be extended to include the last three joints as well, taking into account other flexibility effects due to links and timing belts. It has been further assumed that the mass distribution of the links is uniform and that the stiffness of the flexspline is linear. A more accurate flexible dynamics model can be derived by reconsidering the assumptions made.

- The method used in this thesis, for deriving the dynamics model of the CRS A460 robot, is based on Lagrangian mechanics. It is reported in literature that Kane's method [13] of deriving dynamics models, results in compact and much simpler models, compared to the models derived using the Lagrangian method. A simpler model can be quite useful for real-time implementation. Therefore, it is of interest to investigate the use of Kane's method to derive a dynamics model for the CRS A460 robot.

- It is often difficult, if not impossible, to model the friction and backlash effects quite accurately, due to its unpredictable nature over the entire operating range. This difficulty can be overcome by developing numerical models based on neural networks or fuzzy logic, to model friction and backlash, which are difficult to model analytically. This numerical model can be used in conjunction with the already derived analytical model for a more accurate computed torque control scheme.

7.4.2 Issues Related to the Experimental Set-up

- From the implementation point of view, the velocity feedback can be made more accurate by increasing the order of the numerical differentiation. Taking more than two position samples to estimate velocity, would result in better reading and hence, better performance.
Elimination of the inner-torque loop has made the associated analogue-to-digital conversion hardware and software, irrelevant. Therefore, by eliminating the redundant or unused inter-transputer communications, software overheads can be minimized, resulting in a faster control system. This could pave way for attaining higher sampling rates, resulting in a better control and superior performance.

The transputer file server which runs on the host PC is a polling-type of server. This "locks-up" the host PC and hence, cannot be utilized except to serve the transputer requests, in a polling mode. The PC could be used to display real-time data or other supervisory control tasks, if the transputer file server can be made to operate in the interrupt-driven mode as outlined in [8].
Bibliography


Appendix A

Optimal Loop-Gain Values

Proportional gain matrix for the computed torque control experiment for the CRS A460 (alpha) robot.

\[
K_p = \begin{bmatrix}
6000 & 0 & 0 \\
0 & 6500 & 0 \\
0 & 0 & 8000 \\
\end{bmatrix}
\begin{pmatrix}
\frac{1}{s^2}
\end{pmatrix}
\]

Derivative gain matrix for the computed torque control experiment for the CRS A460 (alpha) robot.

\[
K_v = \begin{bmatrix}
80 & 0 & 0 \\
0 & 80 & 0 \\
0 & 0 & 60 \\
\end{bmatrix}
\begin{pmatrix}
\frac{1}{s} \\
\frac{1}{s^2}
\end{pmatrix}
\]

Torque-to-current conversion matrix, which is equivalent to the inner torque-loop gain matrix, for the CRS A460 (alpha) robot.

\[
K_{tp} = \begin{bmatrix}
600 & 0 & 0 \\
0 & 400 & 0 \\
0 & 0 & 400 \\
\end{bmatrix}
\]
Appendix B

Control Software Listing

B.1 Network Information File: ctorq.nif

This file defines the network topology and the corresponding transputer executable software modules to be loaded:

1,runexnn,R0,0,,2;
2,trajnn,R0,,3,1,4;
3,ctorq,R0,5,4,2,;
4,fti,R0,6,,3,2;
5,servoex,R0,,6,,3;
6,kinfinal,R0,,,5,4;

B.2 Network Configuration File: ctorq.wir

This code configures the Inmos C004 programmable link switch to connect the output of the transputer mounted on the PC. to the RS232 port. It is executed through the network configuration software:

s0 3 t e0 .
B.3  Network Load File: ctorq.bat

This is the windows-based user interface for loading the transputer network. It is executed through a "pif" file:

```
ld-net e:\exprment\curntorq\ctorq.nif cio
```

B.4  Make File: ctorq.mak

This is a generic makefile for use with Logical Systems Transputer Toolset. Change only the flags and specific construction rules and dependencies as required:

```
.SUFFIXES:
.SUFFIXES:.c .pp .tal .trl .tld

INC = -ie:\exprment\curntorq\include
PP_FLAGS = $(INC) -v -dT800 -dDOUBLE32 -dNEURAL
TCX_FLAGS = -f2 -v -c -p85 -ps -w2
TASM_FLAGS = c: -v -c -t -q2

.c.trl:
pp $*.c $(PP_FLAGS)
tcx $* $(TCX_FLAGS)
tasm $* $(TASM_FLAGS)
   @del $*.pp
   @del $*.tal

PATH = E:\exprment\curntorq\include
HEADER1 = $(PATH)\xputer.h $(PATH)\lawvars.h $(PATH)\crsc500.h
all: ctorq
ctorq: ctorq.tld
ctorq.tld: control.trl contreal.trl contnn.trl ctorq.trl
```
atdb.trl contini.trl
	lnk ctorq.lnk
contini.trl :contini.c $(HEADER1) ctorq.mak
atdb.trl :atdb.c $(HEADER1) ctorq.mak
control.trl :control.c $(HEADER1) ctorq.mak
contreal.trl :contreal.c $(HEADER1) ctorq.mak
contnn.trl :contnn.c $(HEADER1) ctorq.mak
ctorq.trl :ctorq.c $(HEADER1) ctorq.mak

B.5  Link File: ctorq.lnk

The link file contains the directive commands for linking the .trl files with the transputer library files to form the transputer executable .tld files. The ctorq.lnk file, used for generating the ctorq.tld executable file, is listed below:

FLAG c
INPUT atdb contini contreal control contnn ctorq
ENTRY _ns_main
LIBRARY t832lib.tll
OUTPUT e:\exprment\curntorq\ctorq.tld
LOAD 0x80004000
STACK 0x80004000
B.6 'C' Source Code

B.6.1 General Description

The C500 Test Bed uses five transputers to perform an experiment. The programs running in each transputer are listed below in the same order that they are booted up. Therefore, communication can take place in the same sequence 1-2-3-4-5 and not from 1 to 5 directly.

1. RUNEX
2. TRAJ
3. CONTROL
4. FTI
5. SERVOEX

Each of these programs will have a minimum of two or three modules (or files):

1. an initialisation module
2. a parameters passing module
3. a real-time expr module

In the initialisation module, communication channels are setup (both hard and logical). The main() program entry point will be executed here. If the program contain more than one processes, then all the processes are setup and executed here. The parameters-passing module will have three common routines to pass various parameters off-line. Parameters files are open in RUNEX and pass to TRAJ, CONTROL and FTI as a string of known length to all transputers. This arrangement allows each transputer program to access information from User Interface Windows without a direct link. If additional parameters are required, this scheme could be follow by either using one of the existing routines or creating a new routine.
i) Run_para() - passes run-time parameters, such as trajectory period, data collection rate, etc.

ii) Joint_path() - decides which joint will be used for the trajectory path selected. It generates the joint angles set point from the Kinematics functions for the trajectory path.

iii) Control_law() - passes control law parameters

In the real_time_exp module, all computations are done in real time under one millisecond. This is where the transputer network can hanged up, if one of the programs is not fast enough to keep up with the rest. CONTROL transputer must complete torque, force and trajectory set points data collections and report the computed desired torque to the SERVO transputer under one milisecond. If this is not the case, then the WatchDog Timer in the SERVO will be violated, causing a hang up.

B.6.2 Control program description

It calculates motor torque for the first three joints from the strain gauge sensor signals read by A/D TRAM. In addition, it reads position data from Servo transputer and force data from Force transputer. Based on the set points, Control program can computes the desired torque for the motors. The computation of desired motor torque depends on the user control strategy and therefore could be changed.

Control program is made from these following modules:

1) contini.c
2) control.c
3) contreal.c
4) user.c
5) atdb.c

along with a common standard header file xputer.h., crsc500.h, lawvar.h, mathcrs.h.

contini.c - an initialisation module Functions Called: Run_para() Functions Called: atdb() Joint_path() Functions Called: initorq() Control_law()
Functions Called: \texttt{initorq()} \texttt{Real\_time\_exp()}

Control.c - parameter reading module

Chan In:

RunParaBuf
TrajBuf
DynamicBuf

Contreal.c - In Contreal.c, all system variables are read in and motor output calculations are performed. It reads position data from Servoex and motor torque signals calculated by A/D TRAM every millisecond. At every two millisecond (500Hz), it reads the end-effector force signal from the Fti program, as discussed above. Once all system variables are read, it calculates the torque required to move the motors based on the control law supplied by the user. These two operations of reading system variables and computation of motor torque must be completed in 1 millisecond to satisfy RAPL timing requirement for each motor.

Called By: \texttt{Contini()} Functions Called: \texttt{user\_cont()} Chan In: Position Torque Force Joint Set Points i.e. \texttt{q, qd, qdd} Chan Out: Motor Inputs

User.c - The user module is named by user to execute its own control (e.g. Inertia.c)

Atdb.c - Collections of functions from Sunnyside A/D manual for sampling A/D signals.
B.6.3  ctnorq.c

#include <stdio.h>
#include <stdlib.h>
#include <conc.h>
#include <math.h>
#include "e:\exprment\curntorq\include\xputer.h"
#include "e:\exprment\curntorq\include\crsc500.h"
#include "e:\exprment\curntorq\include\lawvar.h"
#include "e:\exprment\curntorq\include\neural.h"
#define SC(a,s,c)  ( s=sinf(a), c=cosf(a) )
#define SIGN(a)    ( (a<0) ? -1 : 1 )

extern float  p[6], v[6], claw_op[3];
extern int    qSet[6], position[6];
extern float  q[6], qd[6], qdd[6];
float          pdiff[6], vdiff[6];
float  tvisfric[3], tcolfric[3], torqdes[3];
extern float  acc[3], error[3];
extern float  u[3], sout_final[3];
float        u_new[3];
int cycle_counter = 1;
int contlaw_cal = 1;
int contlaw_execetime = 0;
float shid1[NDIM], shid2[NDIM];
float t_interval=0.0001;
float e_sum[3];
extern float Kp[3], Ki[3], Kd[3];

void user_control(int traj)
{ 
    int i, j;
    float s1, s2, s23, s3;
    float c1, c2, c23, c3;
    float c2c23, s2s23, c2s23, c23s23, c2s2;
    float v1v2, v1v3, v2v3, v1sq, v2sq, v3sq;
    float D1, D2, D3, D12, D23, D32, D13, D23;
    float t1, t2, t3, t12, t23, t33, t32,

    /**************************************************************************
    /* Start Computed Torque algorithm */
    for (j = 0; j < 3; j++)
    {
        pdiff[j] = q[j] - p[j];  /* work in rad. */
        vdiff[j] = qd[j] - v[j];
        if (fabsf(v[j]) < EPSILON)
            tcolfric[j] = 0.0;
        else
            tcolfric[j] = fc[j] * SIGN(v[j]);
        /* viscous friction */
        tvisfric[j] = fv[j] * v[j];
    }
    SC(p[J1], s1, c1);
    SC(p[J2], s2, c2);
    SC(p[J3], s3, c3);
    SC(p[J2] + p[J3], s23, c23);
    c2c23 = c2 * c23;
    s2s23 = s2 * s23;
    c2s23 = c2 * s23;
    c23s23 = c23 * s23;

    78
\[ c_{2s2} = c_2 \times s_2; \]
\[ v_{1v2} = (v_{[J1]}) \times (v_{[J2]}); \]
\[ v_{1v3} = (v_{[J1]}) \times (v_{[J3]}); \]
\[ v_{2v3} = (v_{[J2]}) \times (v_{[J3]}); \]
\[ v_{1sq} = (v_{[J1]}) \times (v_{[J1]}); \]
\[ v_{2sq} = (v_{[J2]}) \times (v_{[J2]}); \]
\[ v_{3sq} = (v_{[J3]}) \times (v_{[J3]}); \]
\[ D_{11} = k_{12} + c_2 \times c_2 \times k_{13} - 2 \times c_2 \times s_2 \times k_{1} + c_2 \times c_2 \times k_{3}; \]
\[ D_{22} = k_{14} - 2 \times s_3 \times k_{1}; \]
\[ D_{23} = D_{32} = k_4 - s_3 \times k_{1}; \]
\[ D_{33} = k_2; \]
\[ D_{112} = -c_2 \times s_2 \times k_{13} - (c_2 \times c_2 - s_2 \times s_2) \times k_{1} - c_2 \times s_2 \times k_{3}; \]
\[ D_{113} = -c_2 \times s_2 \times k_{3} - c_2 \times c_2 \times k_{1}; \]
\[ D_{211} = -D_{112}; \]
\[ D_{223} = D_{233} = -c_3 \times k_{1}; \]
\[ D_{311} = -D_{113}; \]
\[ D_{322} = -D_{223}; \]
\[ G_2 = -c_2 \times g_{r21}; \]
\[ G_3 = s_2 \times g_{r31}; \]
\[ t_{11} = \text{qdd}_{[J1]} + (\text{vdiff}_{[J1]}) \times k_{v11}; \]
\[ t_{21} = (\text{pdiff}_{[J1]}) \times k_{p11}; \]
\[ t_{13} = t_{12} = \text{qdd}_{[J2]} + (\text{vdiff}_{[J2]}) \times k_{v22}; \]
\[ t_{23} = t_{22} = (\text{pdiff}_{[J2]}) \times k_{p22}; \]
\[ t_{33} = t_{32} = \text{qdd}_{[J3]} + (\text{vdiff}_{[J3]}) \times k_{v33}; \]
\[ t_{43} = t_{42} = (\text{pdiff}_{[J3]}) \times k_{p33}; \]
\[ \text{claw\_op}_{[J1]} = D_{11} \times (t_{11} + t_{21}) + 2 \times D_{112} \times v_{1v2} + 2 \times D_{113} \times v_{1v3} + \text{tvisfric}_{[J1]} + \text{tcolfric}_{[J1]}; \]
\[ \text{claw\_op}_{[J2]} = D_{22} \times (t_{12} + t_{22}) + D_{23} \times (t_{32} + t_{42}) + D_{211} \times v_{1sq} + D_{233} \times v_{3sq} + 2 \times D_{223} \times v_{2v3} + G_2 + G_3 + \text{tvisfric}_{[J2]} + \text{tcolfric}_{[J2]}; \]
claw_op[J3] = D32*(t13 + t23) + D33*(t33 + t43) +
    D311*vlst + D322*v2st + G3 + tvisfric[J3] +
tcolfric[J3]; /* end user_control() */
B.6.4 contreal.c

/* Filename: contreal.c (Alpha Robot Regular version)
 * Created: Oct 15/93
 * Rev.1: JAN 05/95
 * Rev.2: Feb 09/95
 * Rev.3: Feb 24/95
 * 
 * Purpose: Calculate desired torque depending on the control law
****************************************************************************
INNER TORQUE LOOP REMOVED FOR COMPUTED TORQUE CONTROL
BASED ON CURRENT SERVO
****************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <math.h>
#include <string.h>
#include "e:\exprment\curntorq\include\atd.h"
#include "e:\exprment\curntorq\include\xputer.h"
#include "e:\exprment\curntorq\include\crsc500.h"
#include "e:\exprment\curntorq\include\lawvar.h"
#include "e:\exprment\curntorq\include\alpha.h"
float acc[3], error[3], velkal[3];
float u[3], sout_final[3];
/* Initialization of Kalman filter variables done in contnn.c */
float phi[7][7];
float kal[7];
float xkal[7];
float xkal_dt[7];
float gamma1, gamma2, gamma3;
int start_time, mid_time, end_time;
extern int contlaw_execetime; /* use of testing time temporarily */
int ucssubloop; /* user_control subloop flag */
int contactflag, firstcontact, offcontact;
int qSet[6], q456[3], qdSet[6], qddSet[6], position[6], oldposition[6];
int ft[6], test;
float ydotbuff[POINTS], qbuff[POINTS][3], posbuff[POINTS][6], vbuff[POINTS][3];
float tmeasbuff[POINTS][3], claw_opbuff[POINTS][3];
float xtra1buff[POINTS][3], xtra2buff[POINTS][3], xtra3buff[POINTS][3], xtra4buff[POINTS][3];
float del_t; /* used for calculating velocities */
float p[6], q[6], v[6], qd[6], qdd[6];
float yydydd[3], zzdzdd[3], yydydd_0_last;
float Txlast[2][3], Txprelast[2][3]; /* used in wfctrl12.c only */
float xdActual[3];
float load[6], fd[3];
float torqmeas[3], oldtorqmeas[3];
int torqadjust[3], torque[3];
Byte freq_copy, temp[9], torqbuff[9];
float motinp[3], claw_op[3], claw_op_last[3];
float sgcc[3] = {SGCC1, SGCC2, SGCC3};
extern int uctime;
extern char userpid[3]; /* flag to run user’s pid or torque control */
extern float TR_N[8], XR[8], link_length[8], Tool_Transform[8];
extern float ktp[3], ktv[3];
extern Channel *ToFti, *FromFti;
extern Channel *ToGrip, *FromGrip;
extern Channel *ToServo, *FromServo;
extern Channel *ADT_CHAN_IN, *ADT_CHAN_OUT;
void real_time_exp()
{
    int   i,j;
    int   traj, trajold, tim, xtime, bdex;
    contactflag = firstcontact = offcontact = 0;
    ucsubloop = 0;
    traj = trajold = 0;
    xtime = 0;
    bdex = 0;

    for(i=0; i<2; i++)
    {
        for(j=0; j<3; j++)
        {
            Txlast[i][j]=0.0,
            Txprelast[i][j]=0.0;
        }
    }

    for(i=0; i<3; i++)
    {
        motinp[i] = 0;
        oldtorqmeas[i] = 0;
        torqmeas[i] = 0;
        fd[i]=0.0;
    }

    ProcToHigh();
    tim = Time();
    ChanOutChar(ToServo, choice); /* start communication with SERVO */
ChanOutChar(ToFti, choice);    /* start communication with FTI */
ChanIn(FromServo, position, 24);   /* find current position */
bcopy((char*)(position), (char*)(oldposition), 24);
/* ini. to current pos. */
/* Initialize output to hold against gravity. */

p[J1] = (float)(position[J1])*TR_N[J1];
p[J2] = (float)(position[J2])*TR_N[J2];
p[J2] += PID2;
p[J3] = (float)(position[J3])*TR_N[J3] + p[J2]/100.0;
p[J3] -= PID2;
claw_op_last[J1] = 0.0;
claw_op_last[J3] = sinf((p[J2]+p[J3]))*gr31;
/* del_t is in second */

while (traj < (int)(trajper*1000))
{
    start_time =Time();        /* do it here instead of user_control() */
    if (!strcmp(userpid,"of") )
        /* (return 1 if false) do not sample if running user's pid */
        {

        #pragma asm
        ld1 [tim]
        .ldc 25
        sum
        stl [tim]
        #pragma endasm
        /* Measured Torque sampling */

        freq_copy |= 0x01;

84
ChanOutChar(ADT_CHAN_OUT, RATE);
ChanOutChar(ADT_CHAN_OUT, freq_copy);
    ChanIn(ADT_CHAN_IN, torqbuff, 9);
    freq_copy &= 0xFE;
ChanOutChar(ADT_CHAN_OUT, RATE);
ChanOutChar(ADT_CHAN_OUT, freq_copy);
    ChanInTimeFail(ADT_CHAN_IN, temp, 9, tim);
    ChanOutChar(ADT_CHAN_OUT, RESC);
ChanOutChar(ADT_CHAN_OUT, 0);
}
if(ucsubloop==0)
    /* First sub-loop of the user_control() loop */
{
    yydydd_0_last = yydydd[0];
    if(!strcmp(fbtor2, "on")) /* run position loop */
    ChanIn(FromGrip, qSet, 24);
        /* qSet in abs. motor pulses */
    else /* run torque loop */
    {
    ChanIn(FromGrip, q, 12);
    ChanIn(FromGrip, qd, 12);
    ChanIn(FromGrip, qdd, 12);
    ChanIn(FromGrip, q456, 12);
    ChanIn(FromGrip, yydydd, 12);
    ChanIn(FromGrip, zzdzdd, 12);
    }
/* Get actual joint positions from servo*/
    ChanIn(FromServo, position, 24);
for (j=0; j<6; j++)
    /* convert from motor to joint space */
    {
        if(j==J3)
            p[J3] = (float)(position[J3])*TR_N[J3]+
                    p[J2]/100.0;
        else
            p[j] = (float)(position[j])*TR_N[j];
    }

if(strcmp(fbtor2, "on"))    /* run torque loop */
    {
        p[J2] += PID2;
        p[J3] -= PID2;
    }

    ChanOut(ToGrip, p, 12);
    ChanIn(FromFt, ft, 24);
    for (j=0; j<6; j++)
        load[j] = (float)(ft[j])*FUNIT;

    for (j=0; j<6; j++)     /* In pulses/s */
        {
            if(traj == 0)
                v[j]=0.0;
            else
                v[j] = ((float)(position[j]-
                          oldposition[j])) / del_t;
            oldposition[j] = position[j];
/******** convert to rad/s *********/

for (j=0; j<6; j++)
{
    if(j==J3)
      v[J3] = v[J3]*TR_N[J3] + v[J2]/100.0;
    else
      v[j] = v[j]*TR_N[j];
}

/* Kalman filter for filtering acceleration - used in nnet */
for (i=1; i<=2; i++) {
    xkal_dt[i] = 0.0;
    xkal_dt[i+2] = 0.0;
    xkal_dt[i+4] = 0.0;

    for (j=1; j<=2; j++) {
        xkal_dt[i] += phi[i][j] * xkal[j];
        xkal_dt[i+2] += phi[i+2][j+2] * xkal[j+2];
        xkal_dt[i+4] += phi[i+4][j+4] * xkal[j+4];
    }

    xkal_dt[i] += kal[i] * v[J1];
    xkal_dt[i+2] += kal[i+2] * v[J2];
    xkal_dt[i+4] += kal[i+4] * v[J3];
}
for (i=1; i<=6; i++)
xkal[i] = xkal_dt[i];

acc[J1] = xkal_dt[2];
/* estimated acceleration for joint 1 */
acc[J2] = xkal_dt[4];
/* estimated acceleration for joint 2 */
acc[J3] = xkal_dt[6];
/* estimated acceleration for joint 3 */

/* temp velkal to tune gamma */
velkal[J1] = xkal_dt[1];
velkal[J2] = xkal_dt[3];
velkal[J3] = xkal_dt[5];

/* end of Kalman filtering */
}  } //*************** End of ucsubloop=0 if********** */
if(strcmp(fbtor2, "on"))
/* call user_controller if not running CRS PID*/
user_control(traj);    /* cal. claw_op */
/* Cal. in load torque. Inner torque loop is updated at every ms */
motinp[J1] = ktp[J1]*(claw_op[J1])/100.0;
motinp[J2] = ktp[J2]*(claw_op[J2]/100.0 + claw_op[J3]/10000.0);
motinp[J3] = -ktp[J3]*(claw_op[J3])/100.0;
for (i=0; i<3; i++)
{
  torque[i] = (int)(( (int)(torqbuff[3*i+2]) <<4) |
                  ((int)(torqbuff[3*i+1]) >>4) );
torqmeas[i] = ( (float)(torque[i]-2048-torqadjust[i]) )
              * (VOLT_NM*(sgcc[i]/CVALUE_VOLT));   /* Butterworth filter */
torqmeas[i] = 0.01 * torqmeas[i] + 0.99 * 
    oldtorqmeas[i];
oldtorqmeas[i] = torqmeas[i];
}
for (i=0; i<3; i++)  /* check max. motor torque limit */
{
   if (motinp[i] > maxmt[i])
      motinp[i] = maxmt[i];
   if (motinp[i] < -maxmt[i])
      motinp[i] = -maxmt[i];

   /* motor torque = 0 if position exceed limits 
      if(position[i]>maxjas[i] || position[i]<maxjae[i])
      motinp[i] = 0; */
}

/****** do it here instead of user_control()******/
mid_time = Time();

/* Update servo every ms */
if( !strcmp(fbtor2, "on") ) */ run CRS position loop */
ChanOut(ToServo, qSet, 24);
else  
{
ChanOut(ToServo, motinp, 12);
ChanOut(ToServo, q456, 12);
}

if ( traj<==(int)(recper*1000) )
{

89
if(xtime == (int)(1000/recrate))
{
    ydotbuff[bdex] = xdActual[J1];
    bcopy((char*)(p), (char*)(posbuff[bdex]), 24);
    bcopy((char*)(v), (char*)(vbuff[bdex]), 12);
    bcopy((char*)(torqmeas),(char*)(tmmeasbuff[bdex]), 12);
    bcopy((char*)(claw_op),(char*)(claw_opbuff[bdex]), 12);
    bcopy((char*)(acc), (char*)(xtra1buff[bdex]), 12);
    bcopy((char*)(u), (char*)(xtra2buff[bdex]), 12);
    bcopy((char*)(sout_final),(char*)(xtra3buff[bdex]), 12);
    bcopy((char*)(error), (char*)(xtra4buff[bdex]), 12);
    bdex++;
    xtime = 0;
}
}

if(ucsubloop == (uctime-1))   /* reset every UCTIME ms */
ucsubloop = 0;
else
ucsubloop ++;

xtime++;
traj++;
end_time = Time(); /*do it here instead of user_control()*/
} /* end of 1ms loop */

for(i=0; i<(int)(recper*1000)/(int)(1000/recrate); i++)   /* Recording */
{
    ChanOut(ToGrip, &ydotbuff[i], 4);
    ChanOut(ToGrip, posbuff[i], 24);
ChanOut(ToGrip, vbuff[i], 12);
ChanOut(ToGrip, tmeasbuff[i], 12);
ChanOut(ToGrip, claw_opbuff[i], 12);
ChanOut(ToGrip, xtra1buff[i], 12);
ChanOut(ToGrip, xtra2buff[i], 12);
ChanOut(ToGrip, xtra3buff[i], 12);
ChanOut(ToGrip, xtra4buff[i], 12);
} /* End of recording for */

/* temporary */
ChanOut(ToGrip, &start_time, 4);
ChanOut(ToGrip, &mid_time, 4);
ChanOut(ToGrip, &end_time, 4);

ChanOutChar(ToGrip, '1'); /* end of exp. */

} /* End of real_time_exp loop */

/* Indices for joints: joint 1=index 0; joint 2=index 1; joint 3=index 2
Joint 3 position and motor torque sign are reversed
Also joint 2&3 position need to be converted to D-H convention
joints 1 and 2 do not need correction */

void initq() {
register int tim, i;

ProcToHigh();
tim = Time();
ProcToLow();
for (i=0; i<10; i++)
{
  #pragma asm
  ldl [tim]
  .ldc 30
  sum
  stl [tim]
  #pragma endasm

  freq_copy |= 0x01;
  ChanOutChar(ADT_CHAN_OUT, RATE);
  ChanOutChar(ADT_CHAN_OUT, freq_copy);

  ChanIn(ADT_CHAN_IN, torqbuff, 9);

  freq_copy &= 0xFE;
  ChanOutChar(ADT_CHAN_OUT, RATE);
  ChanOutChar(ADT_CHAN_OUT, freq_copy);

  ProcToHigh();
  ChanInTimeFail(ADT_CHAN_IN, temp, 9, tim);
  ProcToLow();

  ChanOutChar(ADT_CHAN_OUT, RESC);
  ChanOutChar(ADT_CHAN_OUT, 0);

 torqadjust[0] += (int)((int)(torqbuff[2]) <<4) |
    ((int)(torqbuff[1]) >>4);
torqadjust[1] += (int) ((int)(torqbuff[5]) <<4) |
((int)(torqbuff[4]) >>4));
torqadjust[2] += (int) ((int)(torqbuff[8]) <<4) |
((int)(torqbuff[7]) >>4));
}
torqadjust[0] = (torqadjust[0]/10) - 2048;
torqadjust[1] = (torqadjust[1]/10) - 2048;
torqadjust[2] = (torqadjust[2]/10) - 2048;

} /* End of initorq */

B.6.5 control.c

/* Filename: CONTROL.C
* Created: Dec'93
* Robot: ALPHA
* Desc.: Communicate to Root transputer VIA Traj for run parameters
* Run different functions called from contini.c
* Rev.1: MAY 11/94
* General revisions
* Rev.2: May 30/95
* Added User's PID On/Off flag */

#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <conc.h>
#include <string.h>
#include "e:\exprment\curntorq\include\xputer.h"
#include "e:\exprment\curntorq\include\alpha.h"

char choice, force[3];
int uctime, eef, eet, eev, maxmt[3], maxjas[3], maxjae[3];
float trajper, recreate, recper, jtdmax, jtddmax;
int floc, tloc, NumMove, viap[4];
char path[6], joint[3][3], fbtor2[3], userpid[3];
float pAmp[3], p0mega[3], pStAm[3];
float fv[3], fc[3], gr21, gr31, k1, k2, k3, k4, k12, k13, k14;
float Ja[5], Ba[5];
float kv11, kv12, kv13, kv21, kv22, kv23, kv31, kv32, kv33;
float kp11, kp12, kp13, kp21, kp22, kp23, kp31, kp32, kp33;
float kg1, kg2, kg3, alpha22, alpha23, alpha32, alpha33;
float sineamp[3], omega[3], phase[3], stepamp[3], rampamp[3], ktp[3], ktv[3];
float Pwy, threshold, fspec, fstrt, trise, ffini;
float Gv11, Gv22, Gv33, Gp11, Gp22, Gp33, Gtildevl, Gtildefl;
char disconbuf[85], dynamicbuf[164], comptorqbuf[78];
char trajbuf[53], runparabuf[131], torqloopbuf[129];
float TR_N[8];
float XR[8] = {100/PIX2, 100/PIX2, -100/PIX2, -16.07465, -100/PIX2, -100/PIX2, 0, 0};
int RotaryRes[8] = {1000, 1000, 1000, 500, 500, 500, 0, 0};
float link_length[8] = {13.0, A2, D4, 3.0, 0.0, 0.0, 0.0, 1.0};
float Tool_Transform[8] = {TOOLLEN, 0.0, 0.0, 0.0, PI/2.0, 0.0, 0.0, 0.0};

/* Kalman filter eigenvalues */
extern float gamma1, gamma2, gamma3;
extern int position[6];
extern Channel *ToFti, *FromFti;
extern Channel *ToGrip, *FromGrip;
extern Channel *ToServo, *FromServo;
void joint_path()
{
    int i, j;

    ChanIn(FromGrip, trajbuf, 53);

    sscanf(trajbuf, "%s %s %s %s %d %d %d %d %d %d %d %d",
            userid, path, joint[0], joint[1], joint[2],
            &NumMove, &floc, &tlloc, &viap[0], &viap[1], &viap[2], &viap[3]);

    /*comm. with SERVO for pos. only for cpath */
    /* if(!strcmp(path, "Cpath"))
    {
        ChanOutChar(ToServo, choice);
        ChanIn(FromServo, position, 24);
        ChanOut(ToGrip, position, 24);
    } */
}

void run_parameters()
{
    int i;

    ChanIn(FromGrip, runparabuf, 138);
    sscanf(runparabuf, "%f %f %f %d %d %d %d %d %d %d %d %d %d %d %d %d %d %f
            %f %f %f %f %f %f %f %f %f %f %f",
            &recper, &trajper, &recrate, &uctime,
            &eef, &eet, &eev, &maxmt[0], &maxmt[1], &maxmt[2],
            &maxjas[0], &maxjas[1], &maxjas[2],
            &maxjas[2],

&maxjjae[0], &maxjjae[1], &maxjjae[2],
&kt[0], &kt[1], &kt[2], &ktv[0], &ktv[1], &ktv[2],
&jtdmax, &jtdmax, &gamma[1], &gamma[2], &gamma[3]);

ChanOutChar(ToServo, choice); /* START comm. with SERVO after sscanf*/
ChanOut(ToServo, &trajper, 4);
ChanOut(ToServo, &uctime, 4);

ChanOutChar(ToFti, choice); /* START comm. with Fti after sscanf */
ChanOut(ToFti, &trajper, 4);
ChanOut(ToFti, &uctime, 4);

/* INITIALIZATION */
for(i=0; i<8; i++)
TR_N[i] = 1.0/(XR[i]*(float)(RotaryRes[i]));
maxmt[0] = maxmt[0]*AMPSCALE;
/* set 1800dec. of D/A as max. torque, which is 60Nm */
maxmt[1] = maxmt[1]*AMPSCALE;
/* 1100 dec. is equivalent to 7.2A, max. current */
maxjas[0] = maxjas[0]*DEG_PULSES;
maxjas[1] = maxjas[1]*DEG_PULSES;
maxjae[0] = maxjae[0]*DEG_PULSES;
maxjae[1] = maxjae[1]*DEG_PULSES;
}

void control_law_para()
{
  char  claw;
int i, j;
claw = ChanInChar(FromGrip);
switch(claw)
{
    case PD_TORQUE_LOOP: /* torque loop - estimate inertia para */
    ChanIn(FromGrip, torqloopbuf, 129);
    sscanf(torqloopbuf, "%s %s %s %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f
&Ja[0], &Ja[1], &Ja[2], &Ja[3], &Ja[4], &Ba[0],
&Ba[1], &Ba[2], &Ba[3], &Ba[4]);
sscanf(comptorqbuf, "%f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f="%f", &kp11, &kp22, &kp33, &kv11, &kv22, &kv33,
&kg1, &kg2, &kg3, &alpha22, &alpha23, &alpha32, &alpha33);
strcpy (fbtor2, "of");
break;
case NN_CTORQ:
    #ifdef NEURAL    /* defined in cont.mak */
    neuralnet();    /* in contnn.c */
    #endif
    strcpy(fbtor2, "of");
    /* flag to turn user's control on/off */
    break;
case POSITION_CONTROL:  /* position control */
    strcpy(fbtor2, "on");
    break;
}
    /* end of claw switch */
ChanOut(ToGrip, fbtor2, 3);
ChanOutChar(ToServo, choice);  /* start comm. with SERVO */
for(i=0; i<3; i++)  /* run user selected joint */
    ChanOut(ToServo, joint[i], 3);
ChanOut(ToServo, fbtor2, 3);
}
B.6.6  contini.c

#include <conc.h>
#include "e:\exprment\curntorq\include\xputer.h"
Channel *ToFti, *FromFti;
Channel *ToGrip, *FromGrip;
Channel *ToServo, *FromServo;
Channel *ADT_CHAN_IN, *ADT_CHAN_OUT;
extern char choice;
int ini_atd = 1;

int main(void)
{
  ToFti = LINK1OUT;
  FromFti = LINK1IN;
  ToServo = LINK0OUT;
  FromServo = LINK0IN;
  ToGrip = LINK2OUT;
  FromGrip = LINK2IN;
  ADT_CHAN_OUT = LINK3OUT;
  ADT_CHAN_IN = LINK3IN;
  atdb();
  ini_atd = 0;
  while(1)
  {
    choice=ChanInChar(FromGrip);
    switch(choice)
    {
    case JOINT_PATH:  /* select joint and path; ini. a/d */
      initorq();
    }
joint_path();
break;

case RUN_PARAMETERS: /* init. run para. */
run_parameters();
break;

case CONTROL_LAW_PARA:
itorq();
control_law_para();
break;

case REAL_TIME_EXP: /* start exp... read position */
real_time_exp();
break;

case STOP_EXP: /* exit exp. */
exit(1);
break;

default:
break;

}
Appendix C

Graphical Results of Real-time Experiments on the CRS A460 Robot
Payload = 5.5 Lbf
Max. Velocity = 0.1 rad/s

Figure C.1: Position history of joint-1 under computed torque control. for motion aided by gravity
Payload = 5.5 Lbf
Max. Velocity = 0.1 rad/s

Joint #1: Desired & Actual Position Trajectory

Joint #1: Computed Torque Command

Joint #1: Computed Current Command

Joint #1: Position Error

Figure C.2: Position history of joint-1 under PD control for motion aided by gravity
Figure C.3: Position history of joint-1 under PD control with gravity compensation for motion aided by gravity.
Figure C.4: Position history of joint-1 under computed torque control, for motion against gravity.

Payload = 5.5 Lbf
Max. Velocity = 0.1 rad/s
Payload = 5.5 Lbf
Max. Velocity = 0.1 rad/s

Figure C.5: Position history of joint-1 under PD control, for motion against gravity
Payload = 5.5 Lbf
Max. Velocity = 0.1 rad/s

Joint #1: Desired & Actual Position Trajectory

Joint #1: Position Error

Joint #1: Computed Torque Command

Joint #1: Computed Current Command

Figure C.6: Position history of joint-1 under PD control with gravity compensation for motion against gravity
Joint #1: Desired & Actual Position Trajectory

Joint #1: Position Error

Joint #1: Computed Torque Command

Joint #1: Computed Current Command

Figure C.7: Position history of joint-1 under computed torque control at a peak joint-speed of 0.1 rad/s and a payload of 0.5Lb
Figure C.8: Position history of joint-1 under computed torque control, at a peak joint-speed of 1 rad/s and a payload of 0.5Lb.
Figure C.9: Position history of joint-1 under computed torque control, at a peak joint-speed of 3 rad/s and a payload of 0.5Lb
Figure C.10: Position history of joint-1 under computed torque control, at a peak joint-speed of 0.1 rad/s and a payload of 1.5Lb
Figure C.11: Position history of joint-1 under computed torque control. at a peak joint-speed of 1 rad/s and a payload of 1.5Lb
Figure C.12: Position history of joint-1 under computed torque control, at a peak joint-speed of 3 rad/s and a payload of 1.5Lb
Joint #1: Desired & Actual Position Trajectory

Joint #1: Position Error

Joint #1: Computed Torque Command

Joint #1: Computed Current Command

Figure C.13: Position history of joint-1 under computed torque control, at a peak joint-speed of 0.1 rad/s and a payload of 2.5Lb
Figure C.14: Position history of joint-1 under computed torque control, at a peak joint-speed of 1 rad/s and a payload of 2.5Lb
Figure C.15: Position history of joint-1 under computed torque control. at a peak joint-speed of 3 rad/s and a payload of 2.5Lb
Figure C.16: Position history of joint-1 under computed torque control, at a peak joint-speed of 0.1 rad/s and a payload of 3.5Lb
Figure C.17: Position history of joint-1 under computed torque control, at a peak joint-speed of 1 rad/s and a payload of 3.5Lb
Figure C.18: Position history of joint-1 under computed torque control, at a peak joint-speed of 3 rad/s and a payload of 3.5Lb
Figure C.19: Position history of joint-1 under computed torque control. at a peak joint-speed of 0.1 rad/s and a payload of 4.5Lb
Figure C.20: Position history of joint-1 under computed torque control. at a peak joint-speed of 1 rad/s and a payload of 4.5Lb
Figure C.21: Position history of joint-1 under computed torque control, at a peak joint-speed of 3 rad/s and a payload of 4.5Lb
Figure C.22: Position history of joint-1 under computed torque control, at a peak joint-speed of 0.1 rad/s and a payload of 5.5Lb
Figure C.23: Position history of joint-1 under computed torque control. at a peak joint-speed of 1 rad/s and a payload of 5.5Lb
Figure C.24: Position history of joint-1 under computed torque control. at a peak joint-speed of 3 rad/s and a payload of 5.5Lb
Joint #2: Desired & Actual Position Trajectory

Joint #2: Position Error

Joint #2: Computed Torque Command

Joint #2: Computed Current Command

Figure C.25: Position history of joint-2 under computed torque control for motion against gravity
Joint #2: Desired & Actual Position Trajectory

Payload = 5.5 Lbf
Max. Velocity = 0.1 rad/s

Joint #2: Computed Torque Command

Joint #2: Computed Current Command

Figure C.26: Position history of joint-2 under independent-joint PD control for motion against gravity
Payload = 5.5 Lbf
Max. Velocity = 0.1 rad/s

Figure C.27: Position history of joint-2 under PD control with gravity compensation for motion against gravity
Payload = 5.5 Lbf
Max. Velocity = 0.1 rad/s

Joint #2: Desired & Actual Position Trajectory

Joint #2: Position Error

Joint #2: Computed Torque Command

Joint #2: Computed Current Command

Figure C.28: Position history of joint-2 under computed torque control for motion aided by gravity
Payload = 5.5 Lbf
Max. Velocity = 0.1 rad/s

Joint #2: Desired & Actual Position Trajectory

Joint #2: Position Error

Joint #2: Computed Torque Command

Joint #2: Computed Current Command

Figure C.29: Position history of joint-2 under independent-joint PD control for motion aided by gravity
Joint #2: Desired & Actual Position Trajectory

Joint #2: Position Error

Joint #2: Computed Torque Command

Joint #2: Computed Current Command

Figure C.30: Position history of joint-2 under PD control with gravity compensation for motion aided by gravity.

Payload = 5.5 Lbf
Max. Velocity = 0.1 rad/s
Figure C.31: Position history of joint-2 under computed torque control, at a peak joint-speed of 0.1 rad/s and a payload of 0.5Lb
Figure C.32: Position history of joint-2 under computed torque control, at a peak joint-speed of 1 rad/s and a payload of 0.5Lb
Figure C.33: Position history of joint-2 under computed torque control, at a peak joint-speed of 3 rad/s and a payload of 0.5Lb
Figure C.34: Position history of joint-2 under computed torque control. at a peak joint-speed of 0.1 rad/s and a payload of 1.5Lb
Figure C.35: Position history of joint-2 under computed torque control, at a peak joint-speed of 1 rad/s and a payload of 1.5Lb
Figure C.36: Position history of joint-2 under computed torque control. at a peak joint-speed of 3 rad/s and a payload of 1.5Lb
Figure C.37: Position history of joint-2 under computed torque control, at a peak joint-speed of 0.1 rad/s and a payload of 2.5Lb
Figure C.38: Position history of joint-2 under computed torque control, at a peak joint-speed of 1 rad/s and a payload of 2.5Lb
Figure C.39: Position history of joint-2 under computed torque control. at a peak joint-speed of 3 rad/s and a payload of 2.5Lb
Joint #2: Desired & Actual Position Trajectory

Joint #2: Position Error

Joint #2: Computed Torque Command

Joint #2: Computed Current Command

Figure C.40: Position history of joint-2 under computed torque control, at a peak joint-speed of 0.1 rad/s and a payload of 3.5Lb
Figure C.41: Position history of joint-2 under computed torque control, at a peak joint-speed of 1 rad/s and a payload of 3.5Lb
Figure C.42: Position history of joint-2 under computed torque control. at a peak joint-speed of 3 rad/s and a payload of 3.5Lb
Figure C.43: Position history of joint-2 under computed torque control, at a peak joint-speed of 0.1 rad/s and a payload of 4.5Lb
Figure C.44: Position history of joint-2 under computed torque control. at a peak joint-speed of 1 rad/s and a payload of 4.5Lb
Figure C.45: Position history of joint-2 under computed torque control, at a peak joint-speed of 3 rad/s and a payload of 4.5Lb
Payload = 5.5 Lbf
Max. Velocity = 1 rad/s

Joint #2: Desired & Actual Position Trajectory
Joint #2: Position Error
Joint #2: Computed Torque Command
Joint #2: Computed Current Command

Figure C.46: Position history of joint-2 under computed torque control, at a peak joint-speed of 1 rad/s and a payload of 5.5Lb
Payload = 5.5 Lbf
Max. joint velocity = 3 rad/s

Figure C.47: Position history of joint-2 under computed torque control. at a peak joint-speed of 3 rad/s and a payload of 5.5Lb
Figure C.48: Position history of joint-3 under computed torque control for motion aided by gravity
Payload = 5.5 Lbf
Max. Velocity = 0.1 rad/s

Figure C.49: Position history of joint-3 under independent-joint PD control for motion aided by gravity
Figure C.50: Position history of joint-3 under PD control with gravity compensation for motion aided by gravity
Payload = 5.5 Lbf
Max. Velocity = 0.1 rad/s

Figure C.51: Position history of joint-3 under computed torque control for motion against gravity
Figure C.52: Position history of joint-3 under independent-joint control for motion against gravity
Payload = 5.5 Lbf
Max. Velocity = 0.1 rad/s

Joint #3: Desired & Actual Position Trajectory

Joint #3: Position Error

Joint #3: Computed Torque Command

Joint #3: Computed Current Command

Figure C.53: Position history of joint-3 under PD control with gravity compensation for motion against gravity
Figure C.54: Position history of joint-3 under computed torque control. at a peak joint-speed of 0.1 rad/s and a payload of 0.5Lb
Figure C.55: Position history of joint-3 under computed torque control, at a peak joint-speed of 1 rad/s and a payload of 0.5Lb
Figure C.56: Position history of joint-3 under computed torque control. at a peak joint-speed of 3 rad/s and a payload of 0.5Lb
Figure C.57: Position history of joint-3 under computed torque control, at a peak joint-speed of 0.1 rad/s and a payload of 1.5Lb
Figure C.58: Position history of joint-3 under computed torque control. at a peak joint-speed of 1 rad/s and a payload of 1.5Lb
Figure C.59: Position history of joint-3 under computed torque control, at a peak joint-speed of 3 rad/s and a payload of 1.5 Lb
Figure C.60: Position history of joint-3 under computed torque control, at a peak joint-speed of 0.1 rad/s and a payload of 2.5Lb
Figure C.61: Position history of joint-3 under computed torque control. at a peak joint-speed of 1 rad/s and a payload of 2.5Lb
Figure C.62: Position history of joint-3 under computed torque control, at a peak joint-speed of 3 rad/s and a payload of 2.5Lb
Figure C.63: Position history of joint-3 under computed torque control. at a peak joint-speed of 0.1 rad/s and a payload of 3.5Lb
Joint #3: Desired & Actual Position Trajectory

Joint #3: Position Error

Joint #3: Computed Torque Command

Joint #3: Computed Current Command

Figure C.64: Position history of joint-3 under computed torque control, at a peak joint-speed of 1 rad/s and a payload of 3.5Lb
Figure C.65: Position history of joint-3 under computed torque control, at a peak joint-speed of 3 rad/s and a payload of 3.5Lb
Figure C.66: Position history of joint-3 under computed torque control. at a peak joint-speed of 0.1 rad/s and a payload of 4.5Lb
Figure C.67: Position history of joint-3 under computed torque control, at a peak joint-speed of 1 rad/s and a payload of 4.5Lb
Figure C.68: Position history of joint-3 under computed torque control. at a peak joint-speed of 3 rad/s and a payload of 4.5Lb
Payload = 5.5 Lbf
Max. Velocity = 1 rad/s

Figure C.69: Position history of joint-3 under computed torque control at a peak joint-speed of 1 rad/s and a payload of 5.5 Lb
Joint #3: Desired & Actual Position Trajectory

Joint #3: Position Error

Joint #3: Computed Torque Command

Joint #3: Computed Current Command

Figure C.70: Position history of joint-3 under computed torque control at a peak joint-speed of 3 rad/s and a payload of 5.5 Lb
Table C.1: Joint-1 peak position-error in degrees
Max. Acceleration = 12.56 rad/s²
Payload = 2.49Kg (5.5Lb)

<table>
<thead>
<tr>
<th>Controller</th>
<th>Max. Joint-Velocity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3.0 rad/s</td>
</tr>
<tr>
<td>Comp. torque control</td>
<td>0.10</td>
</tr>
<tr>
<td>PID control</td>
<td>0.55</td>
</tr>
<tr>
<td>PID + gravity comp.</td>
<td>0.55</td>
</tr>
</tbody>
</table>

Table C.2: Joint-2 peak position-error in degrees
Max. Acceleration = 12.56 rad/s²
Payload = 2.49Kg (5.5Lb)

<table>
<thead>
<tr>
<th>Controller</th>
<th>Max. Joint-Velocity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3.0 rad/s</td>
</tr>
<tr>
<td>Comp. torque control</td>
<td>0.08</td>
</tr>
<tr>
<td>PID control</td>
<td>2.0</td>
</tr>
<tr>
<td>PID + gravity comp.</td>
<td>1.6</td>
</tr>
</tbody>
</table>

Table C.3: Joint-3 peak position-error in degrees
Max. Acceleration = 12.56 rad/s²
Payload = 2.49Kg (5.5Lb)

<table>
<thead>
<tr>
<th>Controller</th>
<th>Max. Joint-Velocity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3.0 rad/s</td>
</tr>
<tr>
<td>Comp. torque control</td>
<td>0.11</td>
</tr>
<tr>
<td>PID control</td>
<td>2.0</td>
</tr>
<tr>
<td>PID + gravity comp.</td>
<td>2.0</td>
</tr>
</tbody>
</table>
Table C.4: Joint-1 peak position-error in degrees, under computed torque-control
Max. Acceleration = $12.56 \text{ rad/s}^2$

<table>
<thead>
<tr>
<th>Max. velocity</th>
<th>Payload</th>
<th>2.49 Kg</th>
<th>2.03 Kg</th>
<th>1.59 Kg</th>
<th>1.13 Kg</th>
<th>0.68 Kg</th>
<th>0.23 Kg</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 rad/s</td>
<td>0.08</td>
<td>0.08</td>
<td>0.09</td>
<td>0.09</td>
<td>0.09</td>
<td>0.09</td>
<td></td>
</tr>
<tr>
<td>1 rad/s</td>
<td>0.032</td>
<td>0.028</td>
<td>0.041</td>
<td>0.034</td>
<td>0.033</td>
<td>0.03</td>
<td></td>
</tr>
<tr>
<td>0.1 rad/s</td>
<td>0.042</td>
<td>0.04</td>
<td>0.04</td>
<td>0.045</td>
<td>0.045</td>
<td>0.038</td>
<td></td>
</tr>
</tbody>
</table>

Table C.5: Joint-2 peak position-error in degrees, under computed torque-control
Max. Acceleration = $12.56 \text{ rad/s}^2$

<table>
<thead>
<tr>
<th>Max. velocity</th>
<th>Payload</th>
<th>2.49 Kg</th>
<th>2.03 Kg</th>
<th>1.59 Kg</th>
<th>1.13 Kg</th>
<th>0.68 Kg</th>
<th>0.23 Kg</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 rad/s</td>
<td>0.08</td>
<td>0.075</td>
<td>0.075</td>
<td>0.08</td>
<td>0.085</td>
<td>0.1</td>
<td></td>
</tr>
<tr>
<td>1 rad/s</td>
<td>0.035</td>
<td>0.035</td>
<td>0.03</td>
<td>0.03</td>
<td>0.033</td>
<td>0.04</td>
<td></td>
</tr>
<tr>
<td>0.1 rad/s</td>
<td>0.048</td>
<td>0.03</td>
<td>0.03</td>
<td>0.025</td>
<td>0.03</td>
<td>0.03</td>
<td></td>
</tr>
</tbody>
</table>

Table C.6: Joint-3 peak position-error in degrees, under computed torque-control
Max. Acceleration = $12.56 \text{ rad/s}^2$

<table>
<thead>
<tr>
<th>Max. velocity</th>
<th>Payload</th>
<th>2.49 Kg</th>
<th>2.03 Kg</th>
<th>1.59 Kg</th>
<th>1.13 Kg</th>
<th>0.68 Kg</th>
<th>0.23 Kg</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 rad/s</td>
<td>0.068</td>
<td>0.06</td>
<td>0.065</td>
<td>0.07</td>
<td>0.07</td>
<td>0.075</td>
<td></td>
</tr>
<tr>
<td>1 rad/s</td>
<td>0.06</td>
<td>0.06</td>
<td>0.06</td>
<td>0.06</td>
<td>0.07</td>
<td>0.07</td>
<td></td>
</tr>
<tr>
<td>0.1 rad/s</td>
<td>0.06</td>
<td>0.065</td>
<td>0.065</td>
<td>0.06</td>
<td>0.068</td>
<td>0.07</td>
<td></td>
</tr>
</tbody>
</table>
IMAGE EVALUATION
TEST TARGET (QA-3)