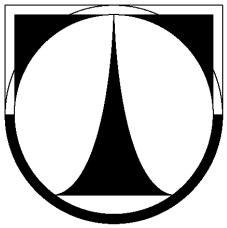
Optimal Control of Robot Manipulators



Ing. Abbas Chatraei

Institute of Mechatronics and Computer Engineering

Faculty of Mechatronics, Informatics and Interdisciplinary Studies

TECHNICAL UNIVERSITY OF LIBEREC

A thesis submitted for the degree of

Doctor of Philosophy

Supervisor: Doc. Mgr. Ing. Václav ZÁDA, CSc.

To my wife, **Saeideh** & my son, **Amirhossein**

for their patience, support and encouragement throughout my Ph.D research

& my new born daughter, **Raha**

&

my **parents in law** and my **parents**

who always gave me their support and encouragement once I required it.

Acknowledgements

I would feel deeply indebted to a number of people during my study.

First of all, I would like to express my sincere gratitude to my supervisor, Mr. Prof. **Václav Záda**, for his guidance, assistance and invaluable professional support and for his approaches to problems which considerably influence mine. I'd also like to thank Mr. Doc. Hlava, Erhart and Modrlak for their helpful guidances while I was educating in this university.

I wish also to thank Mr. Prof. Aleš Richter who always cooperated with me to solve my problems. In addition, I would like to thank Mr. Dr. Ing. Mohsen Gharazi and his wife and Mr. Saeid Ebrahimi as well who supposed me as a member of their families to solve my problems in this country. Eventually, I would like to express my thanks to Mr. Ing. Ladislav Ševčik and all of my friends and staff in TUL for their great friendship.

Abstract

In the recent decades, modeling and designing the various kinds of control schemes for robot manipulators provide several challenges. This dissertation takes into account the problem of finding optimal control inputs and desired optimal trajectories (optimal dynamic motion planning) for open-chain robot manipulators whose dynamics are highly nonlinear and coupled. In doing so, we require a precise dynamic model of the considered system. Thus we develop an experimental identification procedure to estimate the dynamic and friction parameters of our main case study, i.e. the KUKA robot available in the robotic laboratory of Mechatronic faculty of TUL.

Next, this work proposes two new methods to solve the problem mentioned above. In the first proposed method, the unconstrained optimal control problem of robot manipulators is solved with presenting a completely innovative method which yields a global optimal solution for this problem. Our first method finds the optimal solution neither using calculus of variations (indirect methods) nor direct methods, but with a completely different approach obtains the global optimal solution of the considered cost functional. This method has been presented under a theorem with a detailed proof during which we have shown the application of this method in the case of robot manipulators for both point to point motion and trajectory tracking tasks. However, this method has a restriction which can not support the physical constraints on a robot manipulator. Eventually, the proposed method which is a model-based controller is extended into a more general case in which an exact model of the robot is not available, namely designing an *adaptive optimal control* scheme for robot manipulators.

The second proposed method is a combined optimal control method which solves the constrained optimal control problem of robot manipulators. This

approach includes iterative linearization (IL), iterative learning control (ILC) and parametric optimization (PO). In this method, it is assumed that the robot is performing a repeated task such as pick and place parts in an assembly line. Accordingly, in each trial which the robot performs the task, a linear time varying (LTV) version of its nonlinear dynamic model is obtained (using IL) and at the same time an optimal control input for this LTV is computed by parametric optimization method. The optimal solution of each trial is stored in memory of the system to compute the optimal solution in the next trial (ILC). This procedure is repeated so that after a finite number of trials, the sequence of optimal solution of LTVs converge to the robot's optimal control inputs (joint torques or actuator voltages). Eventually, the proposed algorithm is applied into all kinds of the standard manipulator structures (e.g. SCARA, spherical, cylindrical and angular (Puma 560, ABB IRB140 and KUKA) robots). In these case studies the various comparisons are made between the proposed method with the other methods such as direct multiple shooting and spline-based optimal control methods. Briefly, these comparisons yield some efficient capabilities and results for this proposed method which have been presented in chapter 5, in detail.

Keywords: Robot manipulator, Adaptive global optimal control, Robot identification, Excitation trajectory, State dependent coefficient.

Contents

Contents v

List of Figures ix

List of Tables xii

Nomenclature xiii

1 Introduction 1

1.1 Motivation . . . 2
1.2 Background . . . 3
1.2.1 Robot Kinematics . . . 3
1.2.2 Robot Dynamics and Identification . . . 4
1.2.3 Optimal Control of Robot manipulators . . . 4
1.3 Contributions of the Thesis . . . 7
1.4 Outline . . . 9

2 Robot Kinematics, Dynamics and Identification 10

2.1 Kinematics . . . 10
2.1.1 Robot Kinematics . . . 12
2.1.2 Verification of KUKA Inverse Kinematics Model . . . 17
2.2 Dynamics of Robot Manipulators . . . 19
2.2.1 Euler-Lagrange Formulation . . . 20
2.2.2 Newton-Euler Formulation . . . 23
2.2.3 Robot Dynamics Modeler . . . 25
2.2.4 Verifying the Correctness of KUKA Model . . . 26

2.3 Robot Identification . . . 27
2.3.1 Regression Model . . . 29
2.3.2 Identification of Friction Parameters . . . 30
2.3.3 Base Inertial Parameters (Identifiable Parameters) . . . 30
2.3.4 Regression Model of the KUKA Robot . . . 32
2.3.5 Excitation Trajectory . . . 33
2.3.6 Estimation of Dynamic Parameters by Weighted Least Square (WLS) Method . . . 34
2.3.7 Simotion Control System (SCS) . . . 36
2.3.8 Validation . . . 40

3 Optimal Control of Robot Manipulators 43
3.1 Introduction . . . 43
3.2 Optimal Control Problem and Various Performance Criteria . . . 44
3.3 Different Approaches for Solving Optimal Control Problems . . . 46
3.3.1 (Discrete) Dynamic Programming Method . . . 46
3.3.2 Hamilton-Jacobi-Bellman Method (Continuous Dynamic Programming) . . . 47
3.3.3 Indirect Methods . . . 47
3.3.4 Direct methods . . . 48
3.4 Optimal Control Problem of Open-Chain Robot Arms . . . 49
3.4.1 State Space Representation of Robot Manipulators . . . 49
3.4.2 Formulation of Robot OCP . . . 50
3.5 Constrained Nonlinear Programming (NLP) . . . 51
3.6 Parametric Optimization (Spline-Based Optimal Control) . . . 53
3.7 Multiple Shooting Method . . . 57
3.8 Iterative Learning Control (ILC) . . . 58

4 First Proposed Method 62
4.1 Introduction . . . 62
4.2 Preliminary Discussion . . . 63
4.3 New Proposed Theorem . . . 66
4.3.1 Asymptotic Behavior . . . 69

4.3.2 Estimation of Precision . . . 70
4.3.3 Solved Example . . . 70
4.4 Realization 1 . . . 71
4.5 Realization 2 . . . 72
4.5.1 Option of Matrix T . . . 73
4.5.2 Solution of e_i . . . 76
4.5.3 Global Optimal Feedback Control . . . 76
4.6 Case Studies for Global Optimal Controller . . . 78
4.6.1 Vertical Two Links Robot Manipulator . . . 79
4.6.2 KUKA Robot . . . 84
4.7 Adaptive Optimal Control of Robot Manipulator . . . 86
4.8 Case Studies for Adaptive Optimal Controller . . . 90
4.8.1 A Simple Linear System . . . 90
4.8.2 Vertical Two Links Robot Manipulator . . . 94
4.8.3 SCARA Robot . . . 97

5 Second Proposed Method 101
5.1 Introduction . . . 101
5.2 Iterative Linearization . . . 102
5.2.1 Some Preliminary Definitions . . . 103
5.2.2 Successive Approximations of Nonlinear Dynamic Systems . . . 104
5.3 Main Proposed Method for Solving OCP of Robot Arms . . . 108
5.3.1 Cubic Spline Interpolation Consistent with Proposed Method (Trajectory Planning) . . . 108
5.3.2 Optimal Control Problem of LTV Systems . . . 113
5.3.3 Proposed Algorithm . . . 116
5.4 Case studies . . . 121
5.4.1 SCARA Robot . . . 121
5.4.2 Spherical Robot (Stanford Arm) . . . 132
5.4.3 Cylindrical Robot . . . 142
5.4.4 Puma 560 Robot Manipulator . . . 150
5.4.5 ABB IRB140 Robot Manipulator . . . 154
5.4.6 KUKA Robot . . . 158

5.5 A Particular Comparison 162

6 Conclusion **165**

Appendix A: Regression Model of KUKA ROBOT **168**

Appendix B: PUMA 560 ROBOT Dynamics **172**

References **174**

List of Figures

2.1 **Describing the coordinate frames with respect to each other** . 12
2.2 **Robot manipulator** 13
2.3 **MDH frames of joints i and $i-1$** 14
2.4 **IR** 364/10 **KUKA Robot** 15
2.5 **Local frames of KUKA robot according to MDH convention** . 16
2.6 **Geometry model of the KUKA robot** 18
2.7 **Robot writing task together with velocity profile** 18
2.8 **Corresponding trajectories of writing task** 19
2.9 **Verifying the correctness of KUKA inverse kinematics solution** 20
2.10 **Force and moments in RNE formulation** 23
2.11 **Robot Dynamics Modeler (RDM) GUI** 26
2.12 **The torque computed with arbitrary parameters for writing task** 27
2.13 **Correctness of inverse dynamic equations of KUKA robot** . . . 28
2.14 **Excitation trajectories of KUKA robot** 34
2.15 **Estimated base dynamic parameters θ_{B1} to θ_{B15} of KUKA robot** 37
2.16 **Estimated friction parameters of KUKA robot** 38
2.17 Simotion Control System (SCS) 39
2.18 **Validation trajectory to check the estimated model** 40
2.19 **Validation results for actual and estimated Γ_1** 41
2.20 **Validation results for actual and estimated Γ_2** 41
2.21 **Validation results for actual and estimated Γ_3** 42

3.1 **A typical spline function** 55
3.2 **A generic B-spline basis and curve** 56
3.3 **Piecewise constant control (Multiple shooting method)** 59

3.4 A Typical PID iterative learning control scheme 60

4.1 Optimal feedback control schematic 78
4.2 Vertical two links robot manipulator 79
4.3 Optimal profiles of vertical two links robot manipulator 82
4.4 Optimal trajectories of KUKA robot obtained by the proposed method in set-point regulating case 85
4.5 Optimal trajectories of KUKA robot obtained by the proposed method in trajectory tracking case 87
4.6 General structure of adaptive optimal controller of robot manipulators 88
4.7 A simple linear system 90
4.8 Adaptive optimal control system for linear system 91
4.9 Optimal point-to-point motion of linear system 92
4.10 Estimated parameters of linear system for point-to-point motion 92
4.11 Optimal motion of linear system in the case of motion tracking 93
4.12 Estimated parameters of linear system for motion tracking case 93
4.13 Optimal trajectory of two links robot obtained by applying AOC 94
4.14 Estimated parameters of two links robot 96
4.15 SCARA robot 97
4.16 Estimated parameters of SCARA robot 99
4.17 Desired and optimal trajectories of joints 1 and 2 of SCARA robot 100
4.18 Desired and optimal trajectories of joints 3 and 4 of SCARA robot 100

5.1 Fitting a spline to data points 110
5.2 Flowchart of the proposed method 120
5.3 SCARA robot 122
5.4 Optimal profiles of SCARA robot by multiple shooting method 126
5.5 Optimal profiles of SCARA robot by spline based optimal control method 127
5.6 Optimal profiles of SCARA robot by proposed method 128

5.7 Sequence of errors in the successive trials in proposed method for SCARA robot . . . 130
5.8 Spherical (Stanford) robot . . . 133
5.9 Optimal profiles of spherical (Stanford) robot by multiple shooting method . . . 137
5.10 Optimal profiles of spherical (Stanford) robot by spline based optimal control method . . . 138
5.11 Optimal profiles of spherical (Stanford) robot by proposed method 139
5.12 Sequence of errors in the successive trials in proposed method for spherical (Stanford) robot . . . 140
5.13 Three DOF cylindrical robot . . . 142
5.14 Optimal profiles of cylindrical robot obtained by multiple shooting method . . . 145
5.15 Optimal profiles of cylindrical robot obtained by spline based optimal control method . . . 147
5.16 Optimal profiles of cylindrical robot obtained by proposed method 148
5.17 Error profiles obtained for cylindrical robot in various trials . 149
5.18 Puma 560 robot manipulator . . . 150
5.19 Optimal profiles of Puma 560 robot manipulator . . . 153
5.20 Sequence of errors in successive trials obtained for Puma 560 robot . . . 153
5.21 ABB IRB140 robot manipulator . . . 154
5.22 Optimal joint position, velocity and torques of ABB IRB140 robot . . . 157
5.23 Successive errors obtained for ABB IRB140 robot joints . . . 158
5.24 Optimal profiles of KUKA robot . . . 161
5.25 Successive errors of KUKA robot joints . . . 161
5.26 Optimal profiles for unconstrained OCP of vertical two links robot obtained by second proposed method . . . 163
5.27 Optimal profiles for unconstrained OCP of vertical two links robot obtained by multiple shooting method . . . 164

List of Tables

2.1 MDH parameters of KUKA robot 15
2.2 Arbitrary dynamic parameters of KUKA robot to validate its dynamic model 26
2.3 The value of the base dynamic parameters and their standard deviations of KUKA robot 36
2.4 The value of the estimated friction parameters and their standard deviations of KUKA robot 36

4.1 Some typical function spaces 80

5.1 Denavit-Hartenberg parameters of SCARA robot 121
5.2 Dynamic parameters of SCARA robot 123
5.3 Optimal data of SCARA robot obtained from multiple shooting method 127
5.4 Optimal data of SCARA robot obtained from spline based optimal control method 128
5.5 Optimal data of SCARA robot obtained by applying proposed method 131
5.6 Denavit-Hartenberg parameters of spherical (Stanford) robot 133
5.7 Dynamic parameters of spherical (Stanford) robot 134
5.8 Optimal data of spherical (Stanford) robot obtained from multiple shooting method 136
5.9 Optimal data of spherical (Stanford) robot obtained from spline based optimal control method 137
5.10 Optimal data of spherical (Stanford) robot obtained by applying proposed method 141
5.11 Optimal data of cylindrical robot obtained from multiple shooting method 146

5.12 Optimal data of cylindrical robot obtained from spline based optimal control method 146
5.13 Optimal data of cylindrical robot obtained by applying the proposed method 148
5.14 DH parameters of Puma 560 robot manipulator 151
5.15 DH parameters of puma 560 robot manipulator 151
5.16 Joint position and velocity constraints of Puma 560 robot 152
5.17 Initial and final conditions of Puma 560 robot 152
5.18 Optimal data of optimal control of Puma 560 robot 152
5.19 Denavit-Hartenberg parameters of ABB IRB140 robot 155
5.20 Joint position and velocity constraints of ABB IRB140 robot 157
5.21 Initial and final conditions of ABB IRB140 robot 157
5.22 Optimal data of optimal control of ABB IRB140 robot 159
5.23 Joint position, velocity and acceleration constraints of KUKA robot . . 159
5.24 Optimal data obtained for KUKA robot 160
5.25 Optimal data of vertical two links robot obtained from second proposed and multiple shooting methods 163

Chapter 1

Introduction

Modern industrial robots are electro-mechanical systems whose history dates back a few recent decades. However, antecedents of robots which are mechanisms and linkages, have a history of a few centuries. The word robot is Slave in origin, related to the words for work and workers. This word firstly was introduced by Karel Capek, a Czech novelist, in his novel "Rossum's Universal Robots" in 1920. In this novel an engineer made machines that were modeled on human beings. The first industrial robot was manufactured by George Charles Devol, in 1954 called Unimation. After that various universities and companies designed and built different industrial robots such as Puma 560, Stanford, SCARA and etc. Generally, industrial robots are an inspiration of human skeleton and actually they were built to perform intolerable and hazardous task instead of human being. Like human body, the members of each open chain manipulator include: waist, shoulder, elbow, wrist and end effector (hand). In fact, these are the joints of the robot by which the robot links are connected to each other. For studying the robots, it requires to consider some interdisciplinary fields consist of kinematics, statics and dynamics, identification, trajectory planning, motion control, force control, artificial intelligence and so on which each of these subjects are subset of mechanical, electrical and computer sciences.

1.1 Motivation

For a manipulation task carried out by a robot manipulator, there are many paths along which the robot can move. However, the robot should accomplish the desired task in an optimal manner with respect to an appropriate performance criterion. This is the subject considered in this study to solve the well known "optimal control problem" (OCP) for robot manipulators. In other words, the main goal in this thesis is to derive the robot dynamics and then present the new method(s) to obtain the optimal control inputs (joint forces/torques) and generate the optimal trajectories for an open chain robot manipulator. Therefore, the considered objectives of this thesis can be categorized in three main branches:

1. Kinematic and dynamic modeling of the robot arms in closed-form and developing a robot identification procedure to estimate the dynamic and friction parameters of this robot;

2. Solving unconstrained OCP of robot manipulators;

3. Solving constrained OCP of robot manipulators.

The first subject examined in this thesis is to derive the kinematic and dynamic models of robot manipulators in closed-form and developing an identification experiment. In fact, one of the necessary tools to design an optimal controller is possessing a real dynamic model of the system in question. Thus, these subjects is addressed in this thesis generally under the name "robot modeling".

In the second subject, we consider the unconstrained OCP of robot manipulators. The main motivation in this part of our study is to present a method which yields a global optimal solution to the considered unconstrained OCP. Most existing methods such as indirect and direct methods solve this problem by satisfying some necessary conditions which eventually obtain a local optimal solution. We present a completely innovative approach under which a global optimal solution is obtained to the unconstrained OCP of robot manipulators. This method is addressed under a theorem in which we define a cost functional and then by a detailed proof we obtain the minimum value of this functional as well as during this proof we show how this method can be applied in the case of robot manipulators.

The third subject considered in this thesis is to solve the OCP of robot manipulators by considering the physical constraints on a real robot. Most existing methods consider the robot dynamics to solve the OCP. These dynamics have a highly nonlinear and coupled form so that the resulted OCP is too difficult to solve. The second proposed method in this thesis is a compound optimal control scheme whose sub-methods are iterative linearization, iterative learning control and parametric optimization. In this method, which is suitable for robots performing the repeated tasks, a sequence of optimal trajectories is obtained in successive trials so that the limit of this sequence is the optimal solution of the considered time-energy OCP. This method is applied into all types of standard robot arm structures (i.e. SACRA, cylindrical, spherical and revolute articulated robot arms) and a comprehensive comparison is made between the proposed and two other methods, namely multiple shooting and spline based optimal control methods.

1.2 Background

1.2.1 Robot Kinematics

Kinematics of robot manipulators is the study of the motion robot links without considering the forces and torques which cause it. The kinematic model of each robot can be described by four quantities for each robot links. These quantities are called Denavit-Hartenberg parameters which can be obtained in two ways : Denavit-Hartenberg (DH) notation and modified Denavit-Hartenberg (MDH) notation [35; 55]. In general, robot kinematics involve two parts: *direct* kinematics and *inverse* kinematics. In direct kinematics, one obtains the relationship between the individual joint variables of the robot manipulator and the position and orientation of the tool or end-effector. In fact, one wishes to determine how the end-effector position and orientation vary in terms of the joint variables [66]. In contrast, inverse kinematics addresses the finding of joint variables given the position and orientation of the end-effector.

1.2.2 Robot Dynamics and Identification

In contrast with kinematic equations that describe the motion of the robot without taking into account the forces and torques producing it, the dynamic equations explicitly represent the relationship between force and motion. In the case of robot manipulators, these equations are used to design the control system of the robot and to simulate the robot by computer. The dynamic equations of the robot can be derived by use of either Lagrange's equations analytically or recursive Newton-Euler equations numerically [58; 70].

As with the robot kinematics problem, the robot dynamics problem is divided into two parts: *direct* dynamics and *inverse* dynamics problems. In direct dynamics the goal is to calculate the generalized coordinates given the forces and/or torques of the robot joints. The objective in inverse dynamics problem is to compute the torque/force of each joint given the joint positions, velocities and accelerations. The Newton-Euler formulation is an efficient linear recursive approach for inverse dynamics [38].

There is a set of dynamic parameters including the mass, inertia and location of mass center of each robot link as well as the viscous and Coulomb friction parameters in the dynamic equations of the robot. These parameters usually are not provided by robot manufacturers and robot researchers have to measure these parameters themselves. This task is performed under a procedure called *robot identification* by which dynamic and friction parameters of the robot are estimated by means of either off-line or on-line methods [25; 45].

1.2.3 Optimal Control of Robot manipulators

With growing applications of robot manipulators in industrial factories, one of the key features which has been considered is high productivity with low energy consumption as much as possible. One of the factors in increasing the productivity is using industrial robots which perform industrial tasks with high speed and accuracy. There are two ways for robots to work in higher speed. The first way is that the robot manufacturers use more powerful actuators that drive the robots to move at higher speed. However, this method causes bigger actuators which in turn the robots need to consume more energy. Thus, this method is not logical and economical. The second way is to design a controller for the robot manipulators to perform the respective task in minimum time

and minimum energy consumption. Hence, designing such controllers is the goal in the optimal control technique. The OCP of the robot manipulators can be described as follows:

Consider an n-axes robot manipulator whose actuators are DC motors and in a desired task it must move from an initial configuration to a final one. Find the optimal control inputs (voltages) for the motors of the robot and optimal robot trajectories so that a performance criterion characterized the energy consumption by the robot or traversal time from initial configuration to final one (or a compound of energy consumption and traversal time) is minimized without exceeding the preassigned maximum velocities, accelerations and jerks for all joints.

In general, there are three main approaches to solve the (continuous or discrete) OCP of nonlinear dynamical systems [36]:

- indirect methods (variational approaches),
- dynamic programming,
- direct methods.

Indirect methods obtain the solution of the considered constrained OCP by solving a two-point boundary-value (TPBV) problem constructed from optimality conditions represented by Pontryagin's maximum principle, adjoint equations and the transversality (boundary) conditions [74].

Dynamic programming employs the principle of optimality which finally leads to the Hamiltonian-Jacobi-Bellman (HJB) equation [15; 77]. It actually is a partial differential equation which must be solved to obtain the solution of OCP [67].

Direct methods are those that solve the OCP by minimizing the considered cost functional (performance criterion) directly versus the indirect methods. In fact, these methods transfer the OCP into a nonlinear programming problem (NLP) which can be solved by different numerical optimization methods such as SQP (Sequential Quadratic programming) [17; 51]. There are, however, other methods that can be used to solve the OCP such as *nonlinear model predictive control (NMPC)*. NMPC is a modern control strategy which actually solves the OCPs in closed loop feedback form [62].

Early the time-OCP of robot manipulators was solved by indirect methods. In fact, this problem was seriously considered by Bobrow in his Ph.D dissertation [21].

He proposed a new method in which a modified Pontryagin's minimum principle is used to calculate the optimal control torque of each joint of a robot manipulator. In this method a so-called position-velocity phase-plane algorithm is introduced according to which the desired path is parameterized by an arc length parameter "s" and then the robot dynamics is represented in terms of this parameter s which reduces the robot dynamics to second order. Then the resulted second order OCP is solved in phase plane easily. After this proposed method, some other researches were presented by which a set of improvements were applied to the Bobrow's first work [20; 83], for instance considering the singularity problem appeared in OCP [81; 84]. Several other works then parameterized the desired path and applied a parametric optimization algorithm to obtain the time optimal trajectories which avoid obstacles [39]. Some other studies added a energy term to the cost function [82] and also actuator dynamics [71].

Dynamic programming method also was employed to obtain the minimum-time optimal trajectories [8; 37; 85]. In [85], the Bobrow's method was used to solve the OCP of robot manipulators, but for computing the optimal controls, a dynamic programming algorithm has been developed to derive the reduced set of second order differential equations in terms of path parameter and in this way they loosed the problem of "curse of dimensionality" appeared in dynamic programming method.

Although two above methods have been used successfully in many applications, but they have been replaced by direct methods in recent years. The basic idea of this method, in the case of robot manipulators, is that the joint trajectories are approximated by a parametric function such as spline functions and then using a nonlinear programming, the optimal values of the parameters in used approximating function are achieved. One of the main advantages to this approximations is that usually the resulted parametric optimization problem has a feasible solution. Many researchers presented different approaches to generate the optimal joint trajectories. Among these works, polynomial cubic spline functions and B-splines have been used in many studies [19; 42; 48; 87; 88; 95]. In [19], B-spline functions were used to parameterize the joint motions and derive a general optimization technique for robots using Denavit-Hartenberg parameters of the robot and the full robot dynamics. In [87], the cubic spline trajectories is used to converting the OCP into a finite dimensional optimization problem by considering maximum values of velocity, acceleration and jerk for all robot joints. Point to point trajectory parametrization was performed in [88] by means of

cubic B-splines. [95] proposes a method to obtain a global solution to OCP of robot manipulators. Incorporating Both acceleration and jerk as the objectives is considered in [48]. Eventually in [42] an efficient algorithm is proposed to solve the OCP by using polynomial cubic spline functions. There is also another sub-part of the direct methods called *shooting methods*. These kinds of methods, which include single shooting, collocation and multiple shooting methods, use a constant piecewise function to parametrize the control inputs (robot joint's torques and forces) of the system. These methods have been considered in [22; 36], in detail.

In most studies mentioned above, the obtained optimal solution is a local one whereas a little number of researches is found on obtaining the global optimal solution to the OCP of robot manipulators. In the our first proposed method we consider this subject so that it yields a global optimal solution to the considered unconstrained OCP in a completely innovative and different approach versus of the methods mentioned above. In the second proposed method we present a combined optimal control method through which the constrained OCP of robot manipulators is solved. In this approach which is appropriate for robots performing the repeated tasks, in each repetition (trial) a linearized version of robot dynamics is obtained and using the parametric optimization method the OCP is solved in each trial so that after a finite number of trials, the sequence of optimal trajectories converge to the optimal trajectory of the original dynamic system (robot dynamics). This method is applied into all standard types of robot manipulators, i.e. SCARA, spherical, cylindrical and angular robot arms.

1.3 Contributions of the Thesis

The first subject dealt with in this thesis, is modeling of open-chain robot arms. As a main case study, the direct and indirect kinematic models of a KUKA robot available in the robotic laboratory of Mechatronic faculty of TUL are obtained in closed-form through a MDH notation. The dynamic model of this robot are derived by applying a Euler-Lagrange formulation in closed-form as well. These models are verified through a writing task accomplished by the robot in the simulation environment (robotic toolbox of MATLAB) by considering some arbitrary values for dynamic parameters of the robot. Of course, in this thesis the dynamic equations of all standard manipulator

structures are derived in the different chapters.

Then, so as to obtain a dynamic model of the considered KUKA robot an experimental identification is developed through which the dynamic and friction parameters of this industrial robot are estimated. But before performing this task, one require to obtain a mathematical model of the robot so-called "regression model" which is linear in terms of the dynamic parameters of the robot. Thus, it is shown that how the regression model is obtained for the KUKA robot through an QR decomposition method to use in the next step of identification. Also from other key issues in the robot identification is to design an appropriate excitation trajectory along which the robot should be moved during the identification experiment. The practical and theoretical subjects to obtain this trajectory through an optimization problem are addressed and the best choice of the cost function for this problem in the case of the KUKA robot is investigated. Then, the dynamic and friction parameters of this robot are estimated by applying a weighted least square method. It is worth to note that the above procedure to estimate the dynamic parameters of the KUKA robot can be applied into any angular robot manipulator.

In this thesis two proposed methods are introduced to solve the OCP of robot manipulators. The first method solves this problem through a completely innovative manner unlike the existing methods which solve this problem via calculus of variations (indirect method), direct and dynamic programming methods. Most of these approaches present a local optimal solution through satisfying a series of necessary conditions. In contrast with these methods, the proposed approach results in a global optimal solution for the considered cost functional. This method is presented under a theorem and during its proof which is given in detail, the application of the theorem in the case of robot manipulators is explained. From the other features of this method is that it can be used in both point to point motion tasks (e.g. pick and place parts or spot welding tasks) and trajectory tracking tasks such as painting or welding tasks. However, the proposed method has a restriction so that it can not support the physical constraints on the robot. Eventually, the proposed method which is a model-based controller is extended into a more general case in which an exact model of the robot is not available, namely designing an *adaptive optimal control* scheme for robot manipulators.

The second proposed method addressed in this thesis is a compound optimal control

method through which the constrained time-energy OCPs of robot manipulators can be solved. The sub-methods used are iterative linearization, parametric optimization and iterative learning control methods. Let a robot which is performing a repeated task then the proposed method solves the OCP of this robot during a finite number of repetitions (trials). Accordingly, an efficient and robust numerical algorithm is developed to accomplish the procedure of this method. This algorithm is capable to support any kind of constraints and cost functional whether linear or nonlinear. It also generates the smooth trajectories for the robot to avoid exciting the resonance modes of the robot dynamics. Another important feature of this proposed method is that the optimal solution is obtained through a finite number of trials, hence the load of computation is partitioned on these trials so that it can be used as an on-line optimal control method for robot manipulators which is performing repeated tasks.

This thesis eventually presents a comprehensive optimal results by applying the proposed method into all standard types of the manipulator structures including SCARA, spherical (Stanford), cylindrical and angular manipulators. Then the advantages and disadvantages of the proposed method are compared with multiple shooting and spline based optimal control methods through solving the OCP of the SCARA, spherical and cylindrical robots. Eventually, the optimal trajectories of three well known industrial robots, i.e. Puma 560, ABB IRB140 and KUKA robot are presented using the proposed method.

1.4 Outline

This dissertation has six chapters as follows: chapter 1 is an introduction to robot kinematics, dynamics and identification, as well as the OCP. Chapter 2 deals with the subjects such as robot kinematics, dynamics and identification, in detail. In this chapter the results of KUKA robot identification is presented. Chapter 3 addresses the subjects relating to the OCP formulation and the different methods to solve this problem. In chapter 4, we present our first proposed method to solve the unconstrained OCP of robot manipulators. Then, the proposed method to solve the constrained OCP of robot manipulators is dealt with in chapter 5. Eventually, in chapter 6 the concluding remarks are presented.

Chapter 2

Robot Kinematics, Dynamics and Identification

Kinematics is the science of describing the motion of an object without considering the external forces exerted on this object. In fact, kinematics deals with computing the position, velocity and acceleration of a body. In contrast, dynamics is the study of the factors caused the motion of a body, i.e. forces and torques.

In this chapter, we will first address the kinematics of the robot manipulators by which the relation between pose (position and orientation) of the end-effector and configuration of the robot can be obtained. We then deal with dynamics of the robot in which the relations between joint positions, velocities, accelerations and torques are obtained. Furthermore, the system identification is discussed to estimate the friction and inertial parameters of the robot. All subjects mentioned above are used to derive an exact dynamics of a KUKA industrial robot available in robotic laboratory in mechatronics faculty of TUL.

2.1 Kinematics

A rigid body is an object in which the distance between particles is constant and invariance in time regardless of the external forces exerted on it. A robot manipulator consists of a chain of rigid bodies, known as *links of the robot*, connected by revolution or prismatic joints driven by actuators. The first link of this chain is known as the

base link fixed to the ground, while the end-effector (tool) is mounted on the last link. This tool can be any device intended to manipulate objects (grippers) or to transform them (welding or paint tools).

In order to study the motion of this mechanical structure, it is necessary to consider the motion of all links relative to the other links. For this reason, a coordinate frame is allocated to each link and the position and orientation of each frame is computed with respect to other frames. Therefore, in this way we are able to describe the position and orientation of an object intended to manipulate by the robot. Let us now consider point p , as shown in Figure 2.1, whose coordinates relative to frame B are $p_B = (x_B, y_B, z_B)$. For computing the coordinates of this point with respect to the frame A, we use the following coordinate transformation:

$$p_A = Rp_B + d, \tag{2.1}$$

where $R \in \mathbb{R}^{3\times3}$ represents the rotation part of the transformation while $d \in \mathbb{R}^3$ is its translation part. The matrix R has two basic attributes:

- it is an *orthogonal* matrix, that is, $R^T R = I$ or $R^T = R^{-1}$,
- its determinant is $+1$ ($det(R) = +1$).

The set of 3×3 rotation matrices R which have two proprieties mentioned above, is known as *special orthogonal* matrices set ($SO\,(3)$). In fact, the matrix R describes the orientation of the frame B w.r.t frame A which can be presented by Euler angles or quaternions.

Notice that, the transformation (2.1) is not linear, so with defining the homogeneous coordinates $p = (x, y, z, 1)$, it can be modified into a linear transformation as follows:

$$\begin{bmatrix} p_A \\ 1 \end{bmatrix} = \begin{bmatrix} R & d \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} p_B \\ 1 \end{bmatrix} = T \begin{bmatrix} p_B \\ 1 \end{bmatrix}, \tag{2.2}$$

where $T \in \mathbb{R}^{4\times4}$ is called *homogeneous transformation matrix* and includes both rotation and translation parts in a single matrix. The set of 4×4 homogeneous transfor-

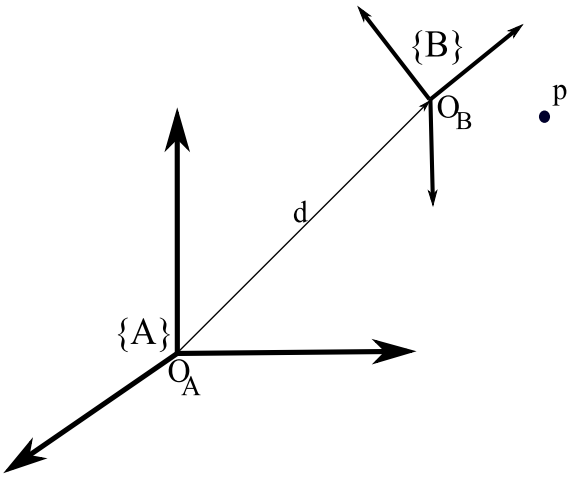


Figure 2.1: **Describing the coordinate frames with respect to each other**

mation matrices T is called *special Euclidean* set $SE(3)$, i.e,

$$SE(3) = \left\{ T = \begin{bmatrix} R & d \\ 0 & 1 \end{bmatrix} \mid R \in SO(3), d \in \mathbb{R} \right\}. \tag{2.3}$$

2.1.1 Robot Kinematics

The problem of robot manipulator kinematics is divided into two parts:

- *direct (forward) kinematics*: given an n-axes manipulator, where n is the number of robot's degrees of freedom. For this robot, the joint angle vector is defined as $\mathbf{q}(\mathbf{t}) = (q_1(t), q_2(t), \ldots, q_n(t))$. The objective in the direct kinematics is to compute the position and orientation of the end-effector in terms of elements of this vector.

- *inverse kinematics*: in the inverse kinematics we are interested to calculate the joint angles in terms of the end-effector position and orientation.

This section deals with these two problems for serial robot manipulators. Let us now consider an n-axes robot manipulator, as shown in Figure 2.2, constituted of $n+1$ links $L_0, L_1, \ldots, L_n$ connected by n joints $J_1, J_2, , \ldots, J_n$, where the base link (link 0) has been fixed to the ground. As shown in this figure, a coordinate frame is adopted to each link so that the position and orientation of the last link, i.e., the pose of

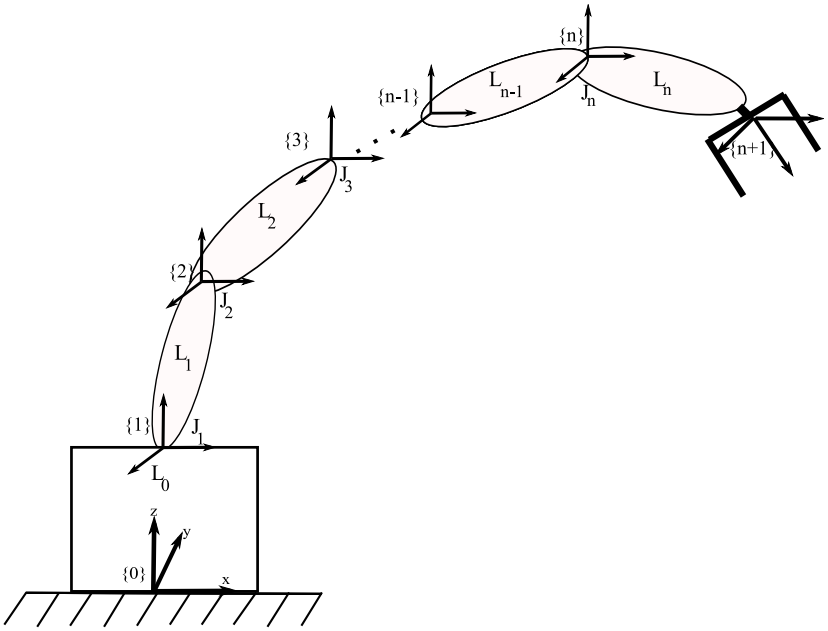


Figure 2.2: **Robot manipulator**

frame $\{n\}$, relative to the base link frame $\{0\}$ is described by the following coordinate transformation:

$$^{0}T_{n} = {}^{0}A_{1}\,{}^{1}A_{2}\,\ldots\,{}^{n-1}A_{n}, \tag{2.4}$$

where $^{i-1}A_i$ denotes the homogeneous transformation of frame i relative to frame $i-1$. In order to describe the relationship between coordinate frames of a robot we can use either Denavit-Hartenberg (DH) notation or modified Denavit-Hartenberg (MDH) one [55] which we will use the latter to obtain the kinematic model of the KUKA robot. In the MDH notation, the origin of the frame i corresponding to the link i, is placed in the intersection of the axis of the joint i and common perpendicular axis of the joint i and axis of the joint $i+1$. In addition, the axis z of this frame is along the axis of the joint i and x axis is along the common perpendicular axis of the joint i and axis of the joint $i+1$. Eventually, the y axis of frame i is specified according to right hand rule. Then, the transformation matrix $^{i-1}A_i$ describing frame i w.r.t $i-1$ is given as

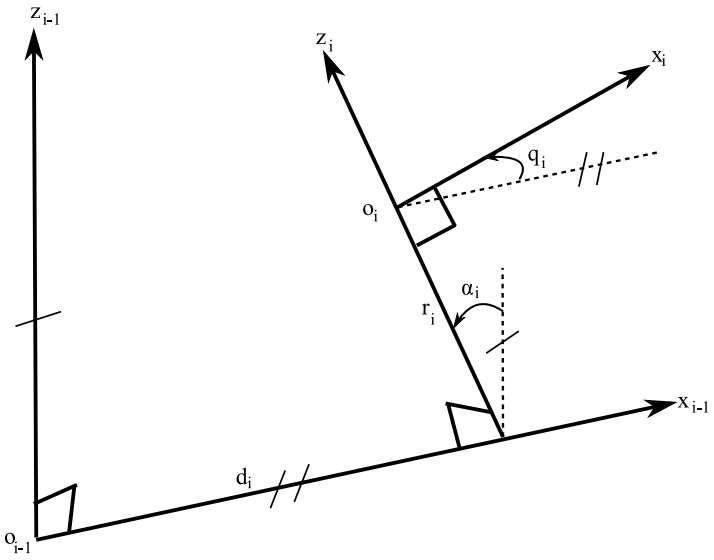


Figure 2.3: **MDH frames of joints i and $i-1$**

follows:

$$ {}^{i-1}A_i = \begin{bmatrix} Cq_i & -Sq_i & 0 & d_i \\ C\alpha_i Sq_i & C\alpha_i Cq_i & -S\alpha_i & -r_i S\alpha_i \\ S\alpha_i Sq_i & S\alpha_i Cq_i & C\alpha_i & r_i C\alpha_i \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{2.5} $$

where S and C stand for functions "sin" and "cos" as well as d_i, α_i, q_i and r_i are MDH parameters of the link i of the robot which can be obtained as follows: (see Figure 2.3)

- q_i : angle between axes x_{i-1} and x_i about z_i,
- α_i : angle between axes z_{i-1} and z_i about x_{i-1},
- d_i : distance between z_{i-1} and z_i along x_{i-1},
- r_i : distance between x_{i-1} and x_i along z_i.

Let us now solve the kinematics problem of a KUKA robot manipulator, as shown in Figure 2.4, which has six degrees of freedom. Generally, in industrial robot manipulators the first three joints (major joints) are responsible for tunning the position of

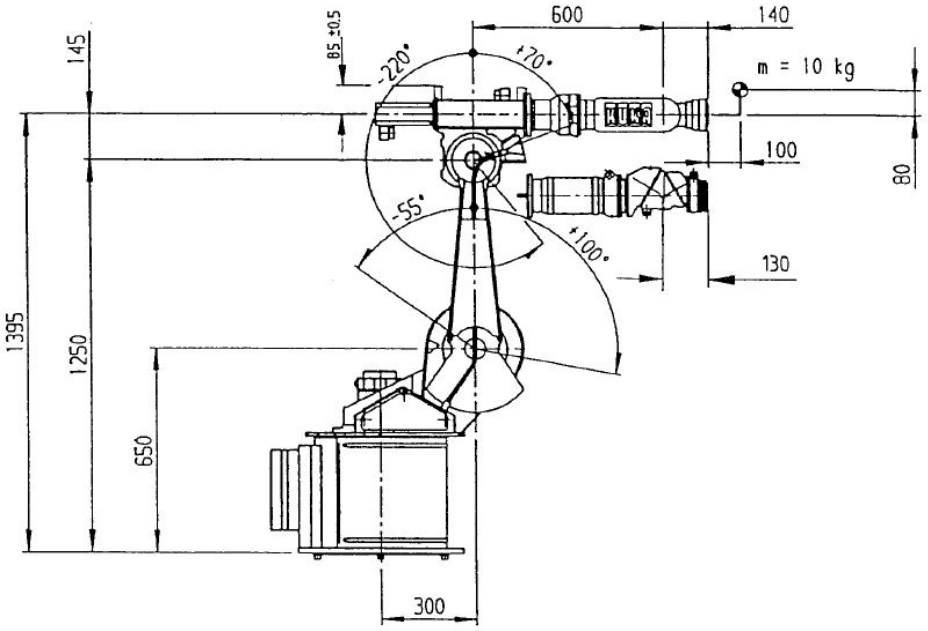


Figure 2.4: **IR** 364/10 **KUKA Robot**

the end-effector, while the last three joints are used to set the orientation of the end-effector. In the optimal control problem of robot manipulators, usually the torque of the joints which are used to set the position of the end-effector, are subjected. Therefore, we assume that the last three joints of the robot are fixed in their home positions, i.e. $q_4 = q_5 = q_6 = 0$. According to MDH convention, the desired frames are assigned to each link of the KUKA robot as depicted in Figure 2.5 and the MDH parameters are obtained as given in Table 2.1. In addition, the transformation matrices of the robot are obtained as follows:

Table 2.1: MDH parameters of KUKA robot

i	α_i	d_i	q_i	r_i
1	0	0	q_1	0.65
2	$-\pi/2$	0.3	q_2	0
3	0	0.6	q_3	0
4	$-\pi/2$	0.145	$q_4 = 0$	0.6
5	$\pi/2$	0	$q_5 = 0$	0
6	$-\pi/2$	0	$q_6 = 0$	0.14

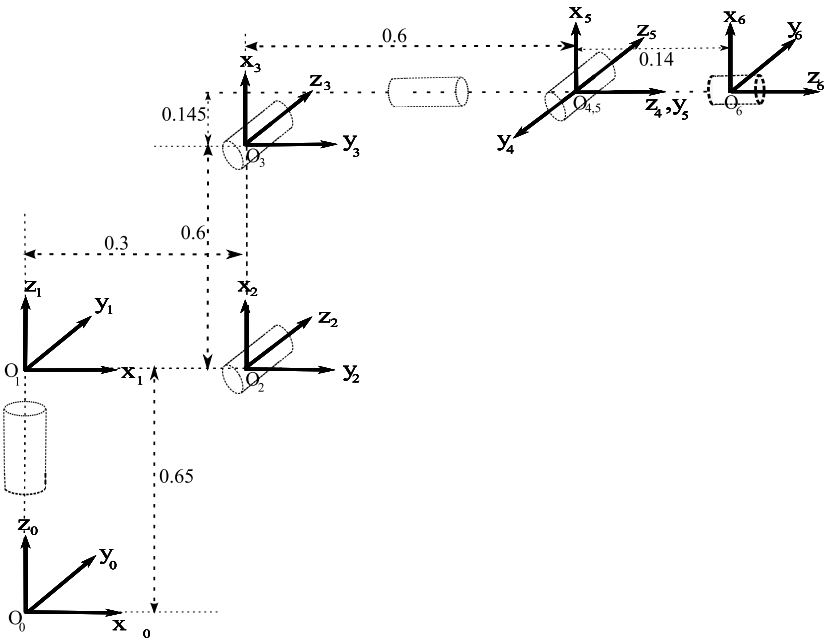


Figure 2.5: **Local frames of KUKA robot according to MDH convention**

$$
{}^{0}A_1 = \begin{bmatrix} C1 & -S1 & 0 & 0 \\ S1 & C1 & 0 & 0 \\ 0 & 0 & 1 & 0.65 \\ 0 & 0 & 0 & 1 \end{bmatrix}, {}^{1}A_2 = \begin{bmatrix} S2 & C2 & 0 & 0.3 \\ 0 & 0 & 1 & 0 \\ C2 & -S2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},
$$
$$
{}^{2}A_3 = \begin{bmatrix} C3 & -S3 & 0 & 0.6 \\ S3 & C3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, {}^{3}A_4 = \begin{bmatrix} 1 & 0 & 0 & 0.145 \\ 0 & 0 & 1 & 0.6 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{2.6}
$$
$$
{}^{4}A_5 = \begin{bmatrix} 1 & 0 & 0 & 0.6 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, {}^{5}A_6 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0.14 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.
$$

By computing the general transformation matrix ${}^{0}T_6$ by referring to (2.4), the kinematics model of KUKA can be obtained by considering the last elements of the rows

1, 2 and 3 of ${}^0T_6$ which are as follows:

$$\begin{aligned} p_x &= C1\left(0.74C23 + 0.145S23 + 0.6S2 + 0.3\right), \\ p_y &= S1\left(0.74C23 + 0.145S23 + 0.6S2 + 0.3\right), \\ p_z &= 0.145C23 - 0.74S23 + 0.6C2 + 0.65, \end{aligned} \tag{2.7}$$

where p_x, p_y and p_z are the coordinates of origin of the last frame $\Re_6$ with respect to the base frame $\Re_0$ as well as $S23$ and $C23$ stand for $\sin\left(q_2+q_3\right)$ and $\cos\left(q_2+q_3\right)$, respectively.

Let us now consider the inverse kinematics problem of this robot. In order to solve this problem, we use the geometry model of the robot as shown in Figure 2.6. Based on this figure, the inverse kinematics equations of this robot are obtained as:

$$\begin{aligned} q_1 &= Atan2(p_y, p_x), \\ q_2 &= Atan2\left(D_1, \sqrt{1-D_1^2}\right) - Atan2\left(D_2, \sqrt{1-D_2^2}\right), \\ q_3 &= Atan2\left(\sqrt{1-D_3^2}, D_3\right) - Atan2\left(D_4, \sqrt{1-D_4^2}\right), \end{aligned} \tag{2.8}$$

where

$$\begin{aligned} D_1 &= \frac{d^2 + 0.6^2 - 0.74^2 - 0.145^2}{2 \times 0.6 \times d}, \; D_2 = \frac{p_z - 0.65}{d}, \\ D_3 &= \frac{d^2 - 0.6^2 - 0.74^2 - 0.145^2}{2 \times 0.6 \times \sqrt{0.74^2 + 0.145^2}}, \; D_4 = \frac{0.74}{\sqrt{0.74^2 + 0.145^2}}, \end{aligned} \tag{2.9}$$

and $d = \sqrt{\left(p_x - 0.3\cos\left(q_1\right)\right)^2 + \left(p_y - 0.3\sin\left(q_1\right)\right)^2 + \left(z - 0.65\right)^2}$.

2.1.2 Verification of KUKA Inverse Kinematics Model

Let us now consider a writing task, as shown in Figure 2.7, which is to be carried out by the robot in the vertical $x-z$ plane. In this figure, the traversed path by end-effector of the robot in $x-z$ plane, x and z profiles together with velocity profile of the end-effector are provided. Notice that the value of y equals $0.4m$ for the given writing task and the sample time is $5ms$. The joint trajectories $q_1\left(t\right), q_2\left(t\right)$ and $q_3\left(t\right)$, corresponding to the given writing task are obtained by means of equations (2.8) and

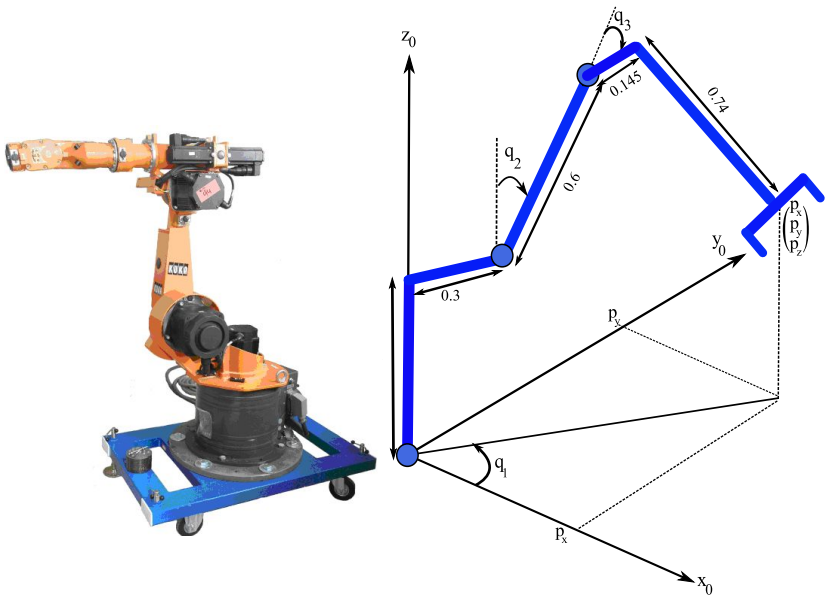


Figure 2.6: **Geometry model of the KUKA robot**

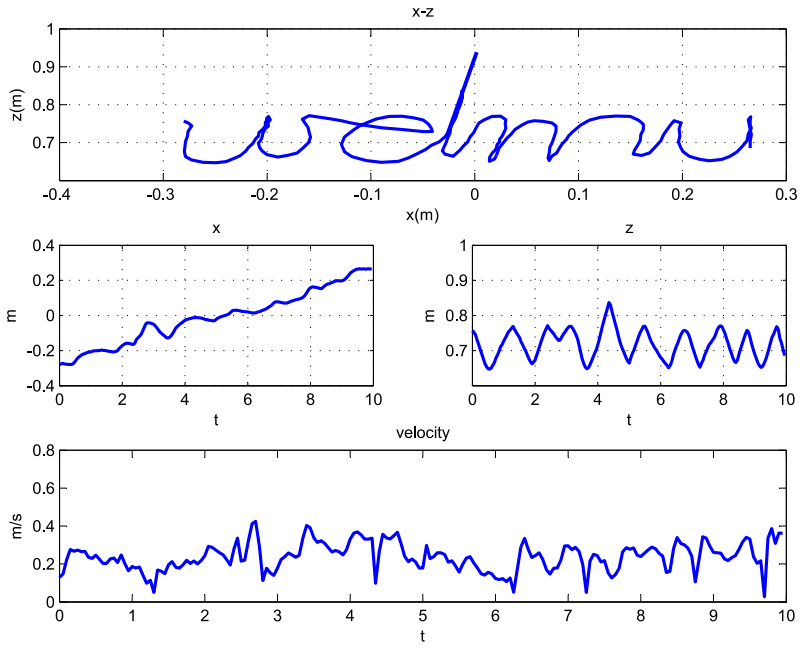


Figure 2.7: **Robot writing task together with velocity profile**

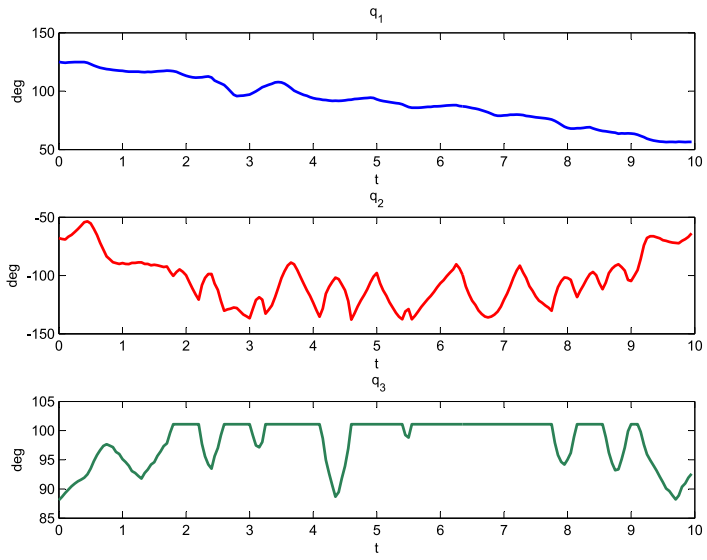


Figure 2.8: **Corresponding trajectories of writing task**

(2.9), as shown in Figure 2.8.

In order to verify the correctness of the inverse kinematics equation (2.8) and (2.9), we used the MATLAB Robotic Toolbox (MRT). In this toolbox we first created a robot object having the KUKA specifications and using functions *fkine* and *ikine* the inverse kinematics trajectories corresponded to the writing task are obtained. As depicted in Figure 2.9, the difference between MRT trajectories and trajectories shown in Figure 2.8 are obtained. As seen, there is only round-off error which shows the correctness of the inverse kinematics model of KUKA robot given in (2.8).

2.2 Dynamics of Robot Manipulators

Robot dynamics deals with the equations of robot manipulator motions which describe the dynamic behavior of the robot and most of the researches are nearly made by such equations; since they are useful to simulate a robot by computer as well as to design the control systems for the robots. In this section, the dynamics of open chain robot manipulators are considered which includes forward and inverse dynamics. In order to derive dynamic equations of motion of a robot arm, usually two conventional

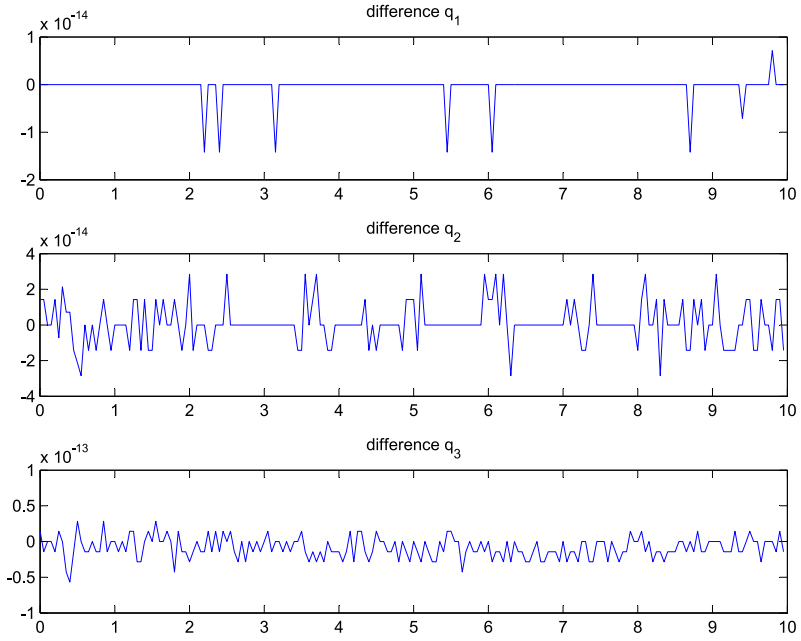


Figure 2.9: **Verifying the correctness of KUKA inverse kinematics solution**

formulation approaches can be used: Euler-Lagrange (EL) and Newton-Euler (NE) formulations. The EL formulation generates the dynamic equations analytically, while the NE approach derives the motion equations by use of recursive relations [66].

2.2.1 Euler-Lagrange Formulation

The dynamic model of a robot manipulator can be expressed in matrix form by the following equation:

$$\mathbf{M}\left(\mathbf{q}\left(t\right)\right)\ddot{\mathbf{q}}\left(t\right)+\mathbf{C}\left(\mathbf{q}\left(t\right),\dot{\mathbf{q}}\left(t\right)\right)\dot{\mathbf{q}}\left(t\right)+\mathbf{G}\left(\mathbf{q}\left(t\right)\right)+\mathbf{F}\left(\dot{\mathbf{q}}\right)=\boldsymbol{\tau}\left(t\right), \tag{2.10}$$

where

- $\mathbf{q}\left(t\right),\dot{\mathbf{q}}\left(t\right)$ and $\ddot{\mathbf{q}}\left(t\right)$ are $n\times 1$ vectors of joint variables, velocities and accelera-

tions, respectively,

$$\begin{aligned}\mathbf{q}(t) &= (q_1(t), q_2(t), \ldots, q_n(t))^T, \\ \dot{\mathbf{q}}(t) &= (\dot{q}_1(t), \dot{q}_2(t), \ldots, \dot{q}_n(t))^T, \\ \ddot{\mathbf{q}}(t) &= (\ddot{q}_1(t), \ddot{q}_2(t), \ldots, \ddot{q}_n(t))^T.\end{aligned} \tag{2.11}$$

- $\boldsymbol{\tau}(t)$ is an $n \times 1$ vector of generalized torques, expressed as

$$\boldsymbol{\tau}(t) = (\tau_1(t), \tau_2(t), \ldots, \tau_n(t))^T. \tag{2.12}$$

- $\mathbf{M}(\mathbf{q}(t))$ is the *acceleration-related symmetric matrix:*

$$M_{ik} = \sum_{j=\max(i,k)}^{n} Tr\left(\mathbf{U}_{jk}\mathbf{J}_j\mathbf{U}_{ji}^T\right) \quad i,\ k\ =\ 1, 2, \ldots, n. \tag{2.13}$$

- Vector $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ includes *Coriolis and centrifugal terms:*

$$\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) = (C_1, C_2, \ldots, C_n)^T, \tag{2.14}$$

where $C_i = \sum_{k=1}^{n}\sum_{m=1}^{n} c_{ikm}\dot{q}_k\dot{q}_m, i = 1, 2, \ldots, n$ and

$$c_{ikm} = \sum_{j=\max(i,k,m)}^{n} \mathrm{Tr}\left(\mathbf{U}_{jkm}\mathbf{J}_j\mathbf{U}_{ji}^T\right) \quad i, k, m = 1, 2, \ldots, n. \tag{2.15}$$

- Vector $\mathbf{G}(\mathbf{q})$ contains *gravity terms*

$$\begin{aligned}\mathbf{G}(\mathbf{q}) &= (G_1, G_2, \ldots, G_n)^T, \\ G_i &= \sum_{j=1}^{n}\left(-m_j\mathbf{g}^T\mathbf{U}_{ji}\,\overline{\mathbf{r}}_j\right) \quad i = 1, 2, \ldots, n,\end{aligned} \tag{2.16}$$

where $\overline{\mathbf{r}}_i = (\overline{x}_i,\ \overline{y}_i,\ \overline{z}_i, 1)^T$.

- Vector $\mathbf{F}(\dot{\mathbf{q}})$ contains *friction terms*

$$\mathbf{F}(\dot{\mathbf{q}}) = \mathbf{F}_v \dot{\mathbf{q}} + \mathbf{F}_c \,\mathrm{SGN}(\dot{\mathbf{q}}). \tag{2.17}$$

Notice that, in the equations (2.10) to (2.17), the following matrices have been used:

$$\mathbf{U}_{ij} = \begin{cases} {}^0\mathbf{A}_{j-1}\mathbf{Q}_j\,{}^{j-1}\mathbf{A}_i & \text{for } j \le i \\ 0 & \text{for } j > i \end{cases},$$

$$\mathbf{U}_{ijk} = \begin{cases} {}^0\mathbf{A}_{j-1}\mathbf{Q}_j\,{}^{j-1}\mathbf{A}_{k-1}\mathbf{Q}_k\,{}^{k-1}\mathbf{A}_i & i \ge k \ge j \\ {}^0\mathbf{A}_{k-1}\mathbf{Q}_k\,{}^{k-1}\mathbf{A}_{j-1}\mathbf{Q}_j\,{}^{j-1}\mathbf{A}_i & i \ge j \ge k \\ 0 & i < j \ \text{ or } \ i < k \end{cases}, \tag{2.18}$$

$$\mathbf{J}_i = \begin{bmatrix} \frac{-I^i_{xx}+I^i_{yy}+I^i_{zz}}{2} & I^i_{xy} & I^i_{xz} & m_i\overline{x}_i \\ I^i_{xy} & \frac{I^i_{xx}-I^i_{yy}+I^i_{zz}}{2} & I^i_{xy} & m_i\overline{y}_i \\ I^i_{xz} & I^i_{yz} & \frac{I^i_{xx}+I^i_{yy}-I^i_{zz}}{2} & m_i\overline{z}_i \\ m_i\overline{x}_i & m_i\overline{y}_i & m_i\overline{z}_i & m_i \end{bmatrix},$$

$$Q_i = \begin{cases} \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} & \text{for revolute joints,} \\ \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} & \text{for prismatic joints,} \end{cases} \tag{2.19}$$

$$\mathbf{g} = (0, 0, -g, 0), \; g = 9.81,$$

where ${}^{i-1}A_i$ is the coordinate transformation from frame $\{i\}$ to frame $\{i-1\}$, represented in the last section. I^i_{xx}, I^i_{yy}, I^i_{zz} are principal moments and I^i_{xy}, I^i_{xz}, I^i_{yz} denote products of inertia of link i. $\overline{x}_i$, $\overline{y}_i$ and $\overline{z}_i$ are coordinates of mass center of the link

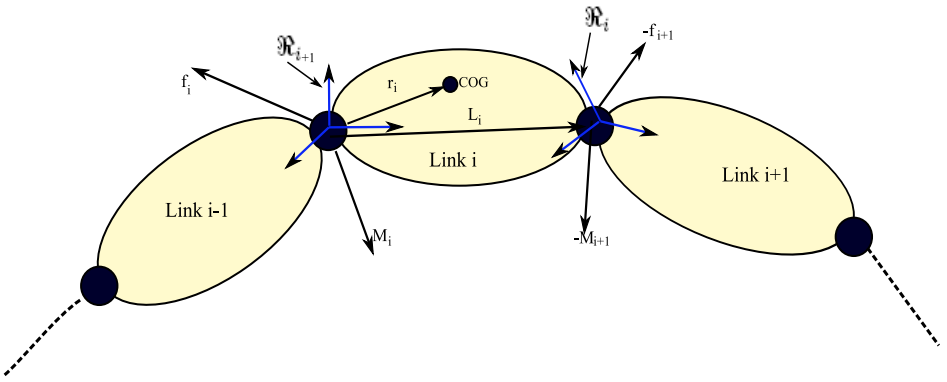


Figure 2.10: **Force and moments in RNE formulation**

i and m_i is mass of the same link. As a result, according to the equations above, the dynamic model of a robot manipulator is obtained which has a highly nonlinear and coupled form. These equations can be used to derive the dynamic model of KUKA robot.

2.2.2 Newton-Euler Formulation

Let us consider the recursive Newton-Euler (RNE) formulation to derive the dynamics model of the KUKA robot. RNE algorithm has a numerical nature in which during two iterative procedures, the dynamic equations of the robot are obtained. In the first iteration called *forward iteration*, the angular velocity and acceleration of the robot links are calculated from link 0 to link n while in the second iteration (*backward iteration*) the set of forces and torques interacted between adjacent links from last link to the link 0 are computed. Before dealing with RNE algorithm, let us define some principal notations as follows (see Figure 2.10):

- τ_i : torque applied to joint i;
- τ_{fi} : friction torque acting on joint i;
- z_i : unit vector along axis of joint i;

- ${}^{i}I_i$: inertia tensor of link i w.r.t its frame, i.e. $\Re_i$, expressed as

$$ {}^{i}I_i = \begin{bmatrix} I_{xxi} & I_{xyi} & I_{xzi} \\ I_{xyi} & I_{yyi} & I_{yzi} \\ I_{xzi} & I_{yzi} & I_{zzi} \end{bmatrix}; $$

- m_i : mass of link i;
- $\mathbf{MS}_i$: first moments of inertia of link i about the origin of the its frame, i.e. $\Re_i$;
- $\mathbf{F}_i$: resultant of external forces on link i;
- ${}^{j}\mathbf{P}_i$: origin of frame $\Re_i$ w.r.t frame $\Re_j$;
- F_{ci} : Coulomb friction parameter of joint i;
- F_{vi} : viscous friction parameter of joint i;
- M_i moment of external force on link i about its COG;
- f_i force exerted on link i by link $i-1$;
- n_i moment about origin of $\Re_i$ exerted on link i by link $i-1$;
- I_{ai} : moment of inertia of the rotor of actuator i and of its transmission system referred to the joint side.
- ${}^{i-1}R_i$: rotation submatrix of transformation matrix ${}^{i-1}A_i$
- $\sigma_i = \begin{cases} 0 & \text{if joint } j \text{ is revolute} \\ 1 & \text{otherwise;} \end{cases}$
- if $a = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} \Rightarrow \hat{a} = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix}$.

RNE algorithm:

Forward iteration: let $i = 1, 2, \ldots, n$

$$
\begin{aligned}
&{}^{i}\omega_{i-1} = {}^{i}R_{i-1}\ {}^{i-1}\omega_{i-1},\\
&{}^{i}\omega_{i} = {}^{i}\omega_{i-1} + \bar{\sigma}_i\ \dot{q}_i\ {}^{i}z_i,\\
&{}^{i}\dot{\omega}_{i-1} = {}^{i}R_{i-1}\ {}^{i-1}\dot{\omega}_{i-1} + \bar{\sigma}_i \left(\ddot{q}_i\ {}^{i}z_i + {}^{i}\omega_{i-1} \times \dot{q}_i\ {}^{i}z_i\right),\\
&{}^{i}\mathbf{U}_i = {}^{i}\hat{\dot{\omega}}_i + {}^{i}\hat{\omega}_i\ {}^{i}\hat{\omega}_i,\\
&{}^{i}\dot{\mathbf{V}}_{i-1} = {}^{i}R_{i-1} \left({}^{i-1}\dot{\mathbf{V}}_{i-1} + {}^{i-1}\mathbf{U}_{i-1}\ {}^{i-1}\mathbf{P}_i\right) + \sigma_i \left(\ddot{q}_i\ {}^{i}z_i + 2\ {}^{i}\omega_{i-1} \times \dot{q}_i\ {}^{i}z_i\right),\\
&{}^{i}F_i = m_i\ {}^{i}\dot{\mathbf{V}}_{i-1} + {}^{i}\mathbf{U}_i\ {}^{i}\mathbf{MS}_i,\\
&{}^{i}\mathbf{M}_{i-1} = {}^{i}\mathbf{I}_{i-1}\ {}^{i}\dot{\omega}_{i-1} + {}^{i}\omega_{i-1} \times \left({}^{i}\mathbf{I}_{i-1}\ {}^{i}\omega_{i-1}\right) + {}^{i}\mathbf{MS}_i \times {}^{i}\dot{\mathbf{V}}_i,
\end{aligned}
\tag{2.20}
$$

with initial values: ${}^{0}\omega_0 = \mathbf{0}$, ${}^{0}\dot{\omega}_0 = \mathbf{0}$ and ${}^{0}\dot{\mathbf{V}}_0 = \begin{bmatrix} 0 & 0 & 9.81 \end{bmatrix}$.

Backward iteration, let $i = n, \ldots, 1$

$$
\begin{aligned}
&{}^{i}\mathbf{f}_i = {}^{i}\mathbf{F}_i + {}^{i}\mathbf{f}_{i+1},\\
&{}^{i-1}\mathbf{f}_i = {}^{i-1}\mathbf{R}_i\ {}^{i}\mathbf{f}_i,\\
&{}^{i}\mathbf{n}_i = {}^{i}\mathbf{M}_i + {}^{i}\mathbf{R}_{i+1}\ {}^{i+1}\mathbf{n}_{i+1} + {}^{i}\mathbf{P}_{i+1} \times {}^{i}\mathbf{f}_{i+1},\\
&\tau_i = \left(\sigma_i\ {}^{i}\mathbf{f}_i + \bar{\sigma}_i\ {}^{i}\mathbf{n}_i\right)^T\ {}^{i}z_i + \tau_{fi} + I_{ai}\ \ddot{q}_i,\\
&\tau_{fi} = F_{ci}\ \mathrm{sign}\left(\dot{q}_i\right) + F_{vi}\ \dot{q}_i.
\end{aligned}
\tag{2.21}
$$

In the subsequent subsections, we will present a Graphical User Interface (GUI) of MATLAB to derive the dynamic equations of a three or six degrees of freedom robot manipulator.

2.2.3 Robot Dynamics Modeler

One of the useful tools in MATLAB is to create Graphical User Interface (GUI) which enables programmers to design an interactive form that enables user to work with an algorithm easily. Therefore, we used this capability to develop a Robot Dynamics Modeler (RDM) GUI , as shown in Figure 2.11, by which the user can derive the dynamic equations of an open chain robot manipulator. In this GUI whose main program is the relations represented in the previous subsections (EL or RNE algorithms) , the user can select either 3 or 6 degrees of freedom robot and after entering the MDH Parame-

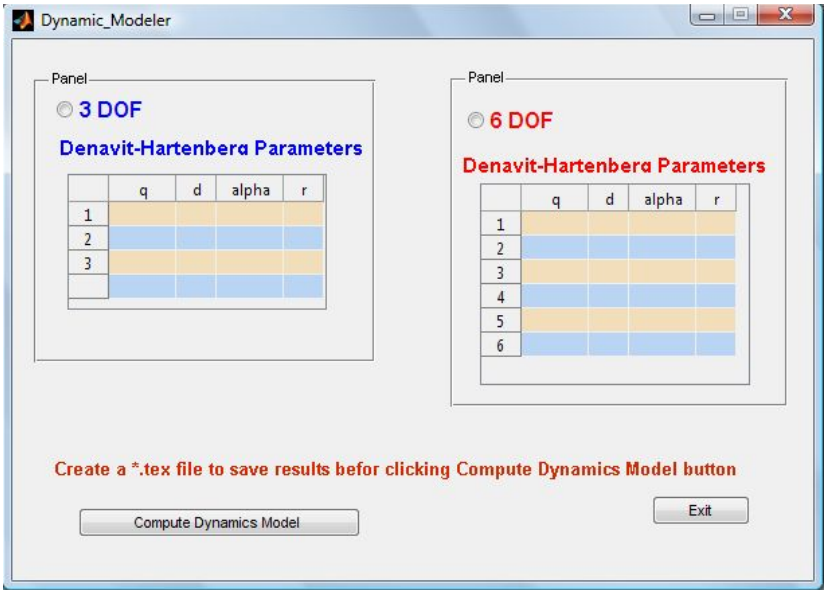


Figure 2.11: **Robot Dynamics Modeler (RDM) GUI**

ters and clicking the "Compute Dynamic Model" button, the dynamic equations of the robot are provided and written into some *.txt files saved in the desktop of computer that each file contains the dynamic equation of each joint of the robot. These equations are simplified by means of some MATLAB commands such as *simple* or *simplify*.

2.2.4 Verifying the Correctness of KUKA Model

As with verifying the inverse kinematics model correctness, let us now verify the dynamic KUKA model obtained by RNE. We reused the writing task given in Figure 2.7 as well as some arbitrary dynamic parameters for the robot as listed in Table 2.2.

Table 2.2: Arbitrary dynamic parameters of KUKA robot to validate its dynamic model

Link	m_i	Center of mass pos.	I_{xxi}	I_{yyi}	I_{zzi}	I_{xyi}	I_{yzi}	I_{xzi}
1	0	$(0, 0, 0.4)$	0	0.35	0	0	0	0
2	17.4	$(0.15, 0, 0)$	0.13	0.524	0.539	0	0	0
3	8	$(0.3, 0, 0)$	0.066	0.086	0.0125	0	0	0
4	0	$(0, 0.2, 0)$	0	0	0	0	0	0
5	0	$(0, 0, 0.05)$	0	0	0	0	0	0
6	0	$(0, 0, 0)$	0	0	0	0	0	0

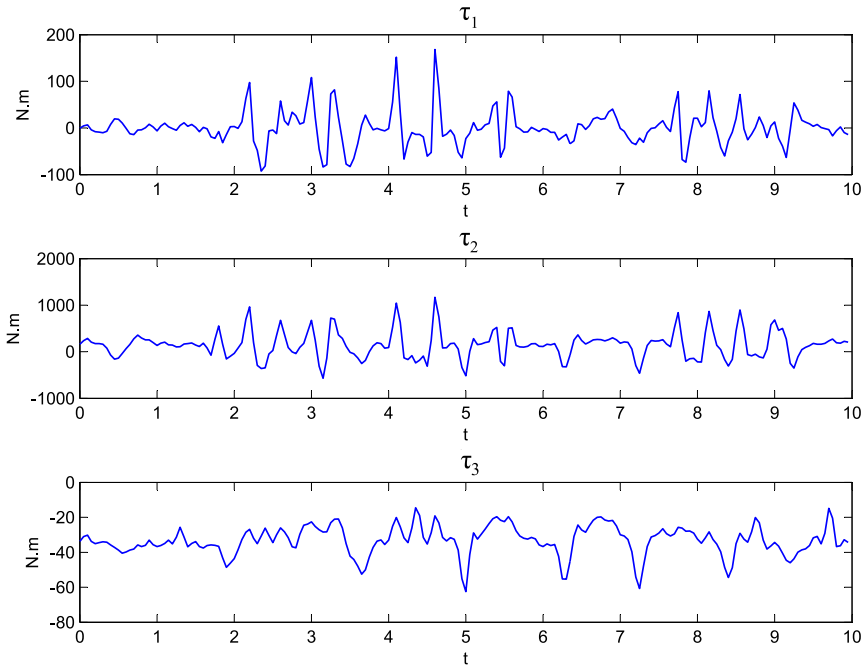


Figure 2.12: **The torque computed with arbitrary parameters for writing task**

These data are input into dynamic model and the required torques to perform the writing task are computed, as shown in Figure 2.12. These torques also are computed by Robotic Toolbox of MATLAB using function *rne* which solves the inverse dynamic of robots by RNE method. Eventually, we obtained the difference between two series of torques as depicted in Figure 2.13 which illustrates the correctness of our KUKA model.

2.3 Robot Identification

In order to design an model-based control system or a computer simulator of a robot manipulator, we require a precise dynamic model of this robot. Robot manufacturers usually don't provide dynamic and friction parameters of the robot manipulators, such as mass, moments of inertia and center of mass (COF) location of each link as well as viscous and Coulomb friction parameters. Hence, the researchers should measure these parameters themselves thorough one of the following approaches:

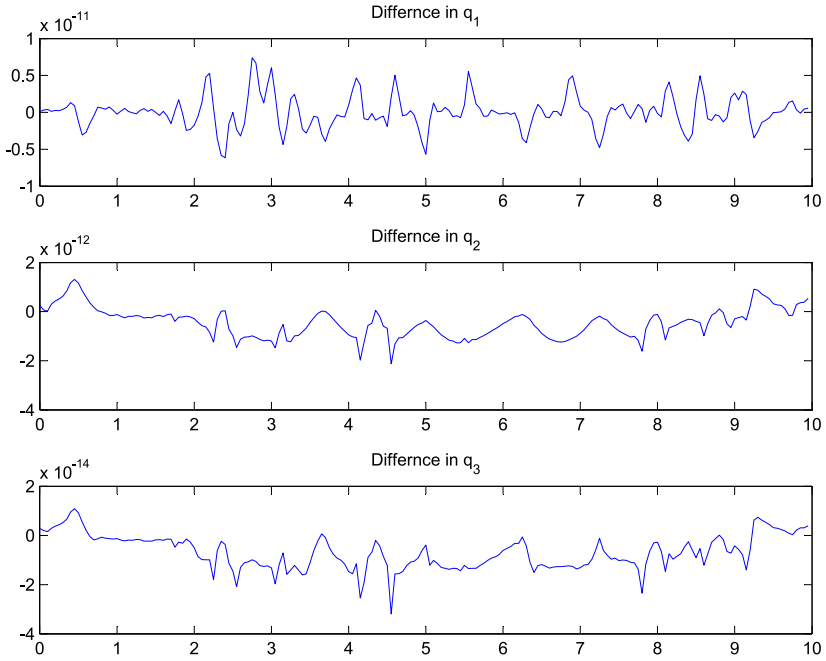


Figure 2.13: **Correctness of inverse dynamic equations of KUKA robot**

- **Physical experiments**, thorough which the robot is disassembled and these parameters are measured by physical measurement devices or special experiments [6].

- **CAM/CAD softwares**, by which a $3D$ model of the robot is developed by some special software packages which calculates the inertia parameters of the robot.

- **Experimental identification**, which deals with estimation of robot dynamic and friction parameters by means of commanding the robot with a particular trajectory, known as *excitation trajectory*, and then measure the torque of each link. These data are used in a statistical framework, such as least square or maximum-likelihood method, to estimate the robot model parameters. In the robot identification, there are three types of parameters: 1- fully identifiable 2- identifiable in linear combinations 3- completely unidentifiable. In fact, during the identification experiment, some dynamic parameters can not be identified because of restricted motion near the base link, while the rest ones can be identified by the form 1 or 2. There are a series of literature that consider the robot identi-

fication problem (RIP) [25; 50; 68; 90] which some of them have been considered as on-line identification like [50] whereas others used off-line methods such as [90]. In these papers, the RIP has been solved by one of the methods: Least square (LS) [61], weighted least square (WLS) [40; 44], Extended Kalman filter (EKF) [45], maximum likelihood or batch adaptive techniques. It must be noted that one of the interesting methods, has been presented in [25] (and also in [50]) in which the *iterative learning control* (ILC) has been used to solve the RIP. In other words, using the capabilities existed in ILC for controlling the robots, it is possible to combine the ILC and identification experiment so that after a finite numbers of trials, the dynamic parameters of the robot can be identified. However, after some researches we found that off-line methods provide us more precise results. Thus, for KUKA robot we used the WLS method to estimate its dynamic and friction parameters which its details are presented in the subsequent subsections.

2.3.1 Regression Model

From identification point of view, there is a helpful property in dynamic equations of the robot which are linear in terms of dynamic and friction parameters of the robot. This property leads us to use the linear regression method to estimate the dynamic and friction parameters. In other words, the dynamics model obtained from equation (2.10) can be expressed as follows:

$$\boldsymbol{\tau} = \boldsymbol{\tau_d} + \boldsymbol{\tau_f} = \mathbf{Y}\left(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}\right) \cdot \boldsymbol{\theta_d} + \mathbf{F}\left(\dot{\mathbf{q}}\right) \cdot \boldsymbol{\theta_f}, \tag{2.22}$$

where $\mathbf{Y} \in \mathbb{R}^{n \times m_d}$ $(m_d = 11n)$ is a matrix whose entries are nonlinear functions of $\mathbf{q}, \dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$ as well as $\mathbf{F} \in \mathbb{R}^{n \times m_f}$ $(m_f = 2n)$ is a function of joint velocities $\dot{\mathbf{q}}$. Furthermore, $\boldsymbol{\theta_d} \in \mathbb{R}^{11n \times 1}$ is a vector whose elements are masses, inertial parameters and Cartesian location of the mass center of robot's links and $\boldsymbol{\theta_f} \in \mathbb{R}^{2n}$ is a vector contains the friction parameters of the robot's joints as follows:

$$\begin{aligned} \boldsymbol{\theta_d} &= \left[\boldsymbol{\theta}_{d,1}\boldsymbol{\theta}_{d,2}\cdots\boldsymbol{\theta}_{d,n}\right]^T, \\ \boldsymbol{\theta_f} &= \left[\boldsymbol{\theta}_{f,1}\boldsymbol{\theta}_{f,2}\cdots\boldsymbol{\theta}_{f,n}\right]^T, \end{aligned} \tag{2.23}$$

with

$$\begin{aligned} \boldsymbol{\theta_{d,i}} &= \left[I_{xxi}, I_{xy_i}, I_{xzi}, I_{yy_i}, I_{yz_i}, I_{zzi}, m_i\overline{x}_i, m_i\overline{y}_i, m_i\overline{z}_i, m_i, I_{ai}\right]^T, \\ \boldsymbol{\theta_{f,i}} &= \left[\ F_{ci}, F_{vi}\right]^T. \end{aligned} \tag{2.24}$$

2.3.2 Identification of Friction Parameters

As stated above, the friction torque on jth joint can be given by

$$\tau_{fj} = F_{cj}\ sign\left(\dot{q}_j\right) + F_{vj}\dot{q}_j, \tag{2.25}$$

where F_{cj} is Coulomb friction parameter and F_{vj} is the viscose friction one. The friction parameters of each joint are estimated separately by applying a constant velocity motion (for instance, a sinusoid motion with constant velocity) on each joint while other joints are rest in their current positions. Thus, if only one joint, say joint j, of the robot is allowed to move, the dynamic equation of the system can be represented as :

$$J_j\ddot{q}_j + h_j \sin q_j = \tau_j - \tau_{fj}, \tag{2.26}$$

where h_j is a coefficient of the gravity term, J_j is the equivalent moment of inertia of the joint and τ_{fj} is obtained according to (2.25). Note that if the joint axis is parallel with the direction of gravity then $h_j = 0$ and otherwise J_j together with h_j should be identified with F_{cj} and F_{vj}. Therefore, the friction parameters of each joint of the robot can be estimated by collecting its position and torque data. In fact, these parameters are estimated using weighted least square method which is explained later in detail.

2.3.3 Base Inertial Parameters (Identifiable Parameters)

The minimum set of necessary parameters to calculate the dynamic model of a robot is called *base inertial parameters* or identifiable parameters. This set is obtained by eliminating some parameters in matrix $\boldsymbol{\theta}$ which have no effect on the robot dynamics or by grouping some of them. In order to obtain such set, the following procedure requires to be done.

After determining the friction parameters, the dynamics model (2.22) can be rewritten as follows:

$$\boldsymbol{\tau_d} = \boldsymbol{\tau} - \boldsymbol{\tau_f} = \mathbf{Y}\boldsymbol{\theta_d}. \tag{2.27}$$

Let us now apply the model (2.27) at ℓ time instances (with $\ell \gg 11n$) on a trajectory which leads to the following equation (regression model):

$$\boldsymbol{\Gamma} = \mathbf{W}\boldsymbol{\theta_d}, \tag{2.28}$$

where $\mathbf{W} \in \mathbb{R}^{(\ell n)\times m_d}$ is called the *observation matrix*, or regressor, represented as follows:

$$\mathbf{W} = \begin{bmatrix} \mathbf{Y}\left(\mathbf{q}\left(t_1\right), \dot{\mathbf{q}}\left(t_1\right), \ddot{\mathbf{q}}\left(t_1\right)\right) \\ \vdots \\ \mathbf{Y}\left(\mathbf{q}\left(t_\ell\right), \dot{\mathbf{q}}\left(t_\ell\right), \ddot{\mathbf{q}}\left(t_\ell\right)\right) \end{bmatrix}, \tag{2.29}$$

and $\boldsymbol{\Gamma} = \left[\boldsymbol{\tau_d}\left(t_1\right), \ldots, \boldsymbol{\tau_d}\left(t_\ell\right)\right]^T$.

Let us now consider the following rules to obtain the base inertial parameter set:

The first rule states that those parameters in $\boldsymbol{\theta_d}$ whose correspondent columns in matrix $\mathbf{W}$ (equation (2.28)) are zero have no effect on robot dynamics , i.e. if we denote jth column of $\mathbf{W}$ as $\mathbf{W}^j$, then $\mathbf{W}^j = 0$ shows that the corresponding parameter in $\boldsymbol{\theta_d}$ has no effect on the robot dynamics, and therefore such column and parameter can be eliminated.

The second rule used to obtain the base inertial parameter set is applied by grouping some parameters in vector $\boldsymbol{\theta_d}$. These grouped parameters are obtained by checking the rank of matrix $\mathbf{W}$. If it is a rank-deficient matrix then by dividing this matrix into two parts so that equation (2.28) can be rewritten as:

$$\boldsymbol{\Gamma} = \begin{bmatrix} \mathbf{W_1} & \mathbf{W_2} \end{bmatrix} \begin{bmatrix} \boldsymbol{\theta_{d1}} \\ \boldsymbol{\theta_{d2}} \end{bmatrix}, \tag{2.30}$$

where part $\mathbf{W_1}$ contains independent columns of $\mathbf{W}$ while dependent columns of $\mathbf{W}$ are contained in $\mathbf{W_2}$ as well as, $\boldsymbol{\theta_{d1}}$ and $\boldsymbol{\theta_{d2}}$ are the corresponding parameters of matrices

$\mathbf{W_1}$ and $\mathbf{W_2}$, respectively. Thus, two matrices $\mathbf{W_1}$ and $\mathbf{W_2}$ can be related as

$$\mathbf{W_2} = \mathbf{k}\mathbf{W_1}, \tag{2.31}$$

where $\mathbf{k}$ is a constant matrix. Therefore, the parameters $\boldsymbol{\theta_2}$ can be grouped with $\boldsymbol{\theta_1}$ by considering equation (2.30) as follows:

$$\boldsymbol{\Gamma} = \mathbf{W_1}\left[\boldsymbol{\theta_{d1}} + \mathbf{k}\boldsymbol{\theta_{d2}}\right] = \mathbf{W_1}\boldsymbol{\theta_B}, \tag{2.32}$$

where $\boldsymbol{\theta_B}$ is the grouping parameter vector which contains the base inertial parameters set.

It can be shown that the number of base inertial parameters is equal the rank of matrix $\mathbf{W}$ and base parameters are corresponding to independent columns of such matrix. Note that in order to obtain the independent columns of matrix $\mathbf{W}$, the QR decomposition can be used as follows:

$$\mathbf{Q^T W} = \begin{bmatrix} \mathbf{R} \\ \mathbf{0} \end{bmatrix}, \tag{2.33}$$

where $\mathbf{Q}$ is an orthogonal matrix and $\mathbf{R}$ is an upper-triangle matrix. Let us now consider a lemma in linear algebra which states the independent columns of matrix $\mathbf{W}$ correspond to the zero elements on diagonal of the matrix $\mathbf{R}$. Therefore, in this way we can obtain the base inertial parameters set of the robot dynamics.

2.3.4 Regression Model of the KUKA Robot

According to the procedure stated in the previous subsection, the regression model of the KUKA robot has the following form

$$\boldsymbol{\Gamma} = \mathbf{W_1}\boldsymbol{\theta_B}, \tag{2.34}$$

where the elements of the matrix $\mathbf{Y_1} \in \mathbb{R}^{3\times 15}$ and vector $\boldsymbol{\theta_B} \in \mathbb{R}^{15\times 1}$ are given in Appendix A. It means that the base parameter set of KUKA robot has 15 elements which together with friction parameters this number equals 21.

2.3.5 Excitation Trajectory

In each system identification an excitation trajectory is required to be obtained. In the case of robot identification, it is a special trajectory along which the robot must move during the identification experiment. Actually it is a trajectory which "excites" all dynamics of the robot as well as, the sensitivity of the least square method, which is used to estimate the base dynamic parameters, with respect to noise and model errors can be minimized along this trajectory [5; 26].

Excitation trajectory for a robot identification can be obtained by solving an optimization problem. In fact, in the least square methods it is proved that to attenuate the effect of the noise in the computations, the condition number of the *observation matrix* given in (2.32), i.e. $\mathbf{W_1}$ must be minimized. Hence, the excitation trajectory is obtained by the follows optimization problem:

$$\min \; J_c = cond\left(\mathbf{W_1}\right), \tag{2.35}$$

where cond stands for condition number and subject to joint position, velocity and acceleration constraints. In order to solve such problem, a parametric optimization method can be used. It means that the following finite Fourier series have been used to parameterize the nonlinear expression existed in J_c.

$$\begin{aligned} q_{d,i} &= q_{0,i} + \sum_{j=1}^{N_i} \frac{1}{j\omega} \left[a_{i,j} \sin\left(j\omega t\right) - b_{i,j} \cos\left(j\omega t\right)\right], \\ \dot{q}_{d,i}\left(t\right) &= \sum_{j=1}^{N_i} a_{i,j} \cos\left(j\omega t\right) + b_{i,j} \sin\left(j\omega t\right), \\ \ddot{q}_{d,i}\left(t\right) &= \sum_{j=1}^{N_i} \left(j\omega\right) \left(-a_{i,j} \sin\left(j\omega t\right) + b_{i,j} \cos\left(j\omega t\right)\right), \end{aligned} \tag{2.36}$$

where ω is a given value and the coefficients $q_{0,i}, a_{i,j}$ and $b_{i,j}$ should be calculated through optimization procedure. Therefore, the infinite dimension optimization problem is converted into a finite one and the Fourier coefficients are computed after solving such problem [90].

After solving the optimization problem mentioned above for the KUKA robot by function *fmincon* in MATLAB, the calculated minimum value of condition number

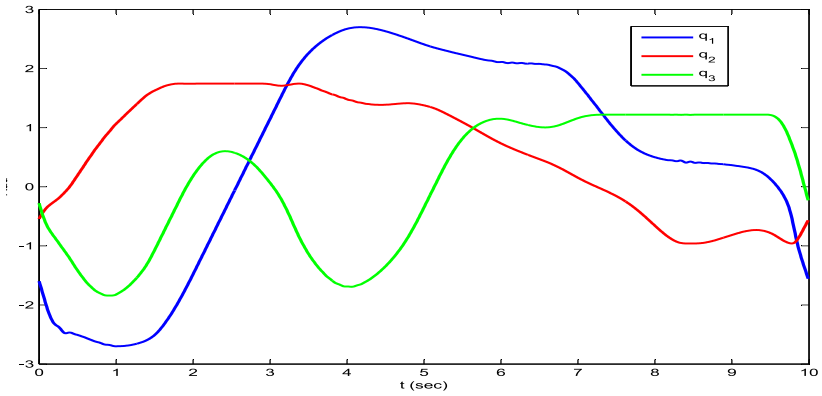


Figure 2.14: **Excitation trajectories of KUKA robot**

became 6.77. Figure 2.14 shows the excitation trajectories for KUKA robot after solving the optimization problem, given $\omega = 2\pi \times 0.1$ and $t \in [0, 10]$ as well as $N_1 = N_2 = N_3 = 100$.

2.3.6 Estimation of Dynamic Parameters by Weighted Least Square (WLS) Method

In order to estimate the base dynamic parameters of the robot, the most common tool is WLS method. After substituting the data collected in the identification experiment in the equation (2.32), the following equation called *identification model* of the robot is obtained:

$$\mathbf{\Gamma} = \mathbf{W_1}\boldsymbol{\theta_B} + \boldsymbol{\varepsilon}, \tag{2.37}$$

where $\boldsymbol{\varepsilon}$ is the $(\ell n \times 1)$ error vector. According to the LS method the estimated base parameters are calculated as the solution of the following optimization problem:

$$\hat{\boldsymbol{\theta}}_{\boldsymbol{B}} = \min_{\boldsymbol{\theta_B}} \|\boldsymbol{\varepsilon}\|^2 . \tag{2.38}$$

The solution of such LS problem can be obtained in the closed form, provided that the observation matrix $\mathbf{W_1}$ is full rank:

$$\hat{\boldsymbol{\theta}}_{\boldsymbol{B}} = \mathbf{W_1}^{+}\,\mathbf{\Gamma} \tag{2.39}$$

where $\mathbf{W_1}^+ = \left(\mathbf{W_1}^T \mathbf{W_1}\right)^{-1} \mathbf{W_1}^T$ is called *pseudo-inverse* matrix. This solution is not so accurate due to noise existed on the measured data, even though they are filtered. Since we can not design an ideal filter and some data are lose in the filtering step. Hence, the WLS can be used to obtain more accurate estimated parameters which is the solution of equation (2.37) according to which the pseudo-inverse matrix is converted to the following one:

$$\mathbf{W_1}^+ = \left(\mathbf{W_1}^T H \mathbf{W_1}\right)^{-1} \mathbf{W_1}^T H, \tag{2.40}$$

where matrix H is a diagonal weighted matrix whose diagonal elements are determined as follows:

$$\Sigma = \left[\frac{1}{\hat{\sigma}_\varepsilon^1} \cdots \frac{1}{\hat{\sigma}_\varepsilon^\ell}\right], \tag{2.41}$$

where $\hat{\sigma}_\varepsilon^j$ is the error standard deviation in the row j in the equation (2.37) which can be calculated as follows [40; 44]:

$$\hat{\sigma}_\varepsilon^j = \frac{\|\varepsilon^j\|}{\sqrt{\ell - \text{length}\left(\theta_B\right)}}. \tag{2.42}$$

Notice that we filtered data obtained from the measurement tool (measurement part of the Simotion control system, explained in the next subsection) with a *low pass butter worth filter* with the cutoff frequency equals $10Hz$. Eventually, applying the following recursive algorithm, the solution of WLS can be obtained [40]:

$$\hat{\theta}_B\left(k+1\right) = \hat{\theta}_B\left(k\right) + P\left(k+1\right) W_1^T\left(k+1\right)\left(\mathbf{\Gamma}\left(k+1\right) - W_1\left(k+1\right)\hat{\theta}_B\left(k\right)\right), \tag{2.43}$$

where

$$P\left(k+1\right) = P\left(k\right) - P\left(k\right) W_1^T\left(k+1\right) \left(I + W_1\left(k+1\right) P\left(k\right) W_1^T\left(k+1\right)\right)^{-1} W_1\left(k+1\right) P\left(k\right). \tag{2.44}$$

The results of estimated parameters by recursive algorithm mentioned above, have been shown in Figure 2.15. The steady values of dynamic parameters and their standard

deviations are given in Table 2.3. Furthermore, the estimated friction parameters are given in Figure 2.16 and their final values together with standard deviations are given in Table 2.4.

Table 2.3: The value of the base dynamic parameters and their standard deviations of KUKA robot

Parameter	Estimated value	$\sigma_{\hat{\theta}}$
θ_{B1}	30.25	0.25
θ_{B2}	13.45	0.92
θ_{B3}	-2.85	0.57
θ_{B4}	12.15	0.88
θ_{B5}	-1.65	0.27
θ_{B6}	95.76	0.89
θ_{B7}	0.153	0.14
θ_{B8}	-35.88	0.65
θ_{B9}	8.152	0.47
θ_{B10}	4.2	0.25
θ_{B11}	-0.446	0.15
θ_{B12}	-3.7	0.27
θ_{B13}	16.4	0.35
θ_{B14}	4.72	0.57
θ_{B15}	-4.46	0.64

Table 2.4: The value of the estimated friction parameters and their standard deviations of KUKA robot

Parameter	Estimated value	$\sigma_{\hat{\theta}}$
F_{c1}	15.2	0.57
F_{v1}	46.5	0.66
F_{c2}	18.56	0.48
F_{v2}	71.74	0.88
F_{c3}	19.85	0.54
F_{v3}	38.84	0.68

2.3.7 Simotion Control System (SCS)

The original control unit of the KUKA robot which is used for this experiment is no longer used as a control system. For the purpose of such experiments was equipped

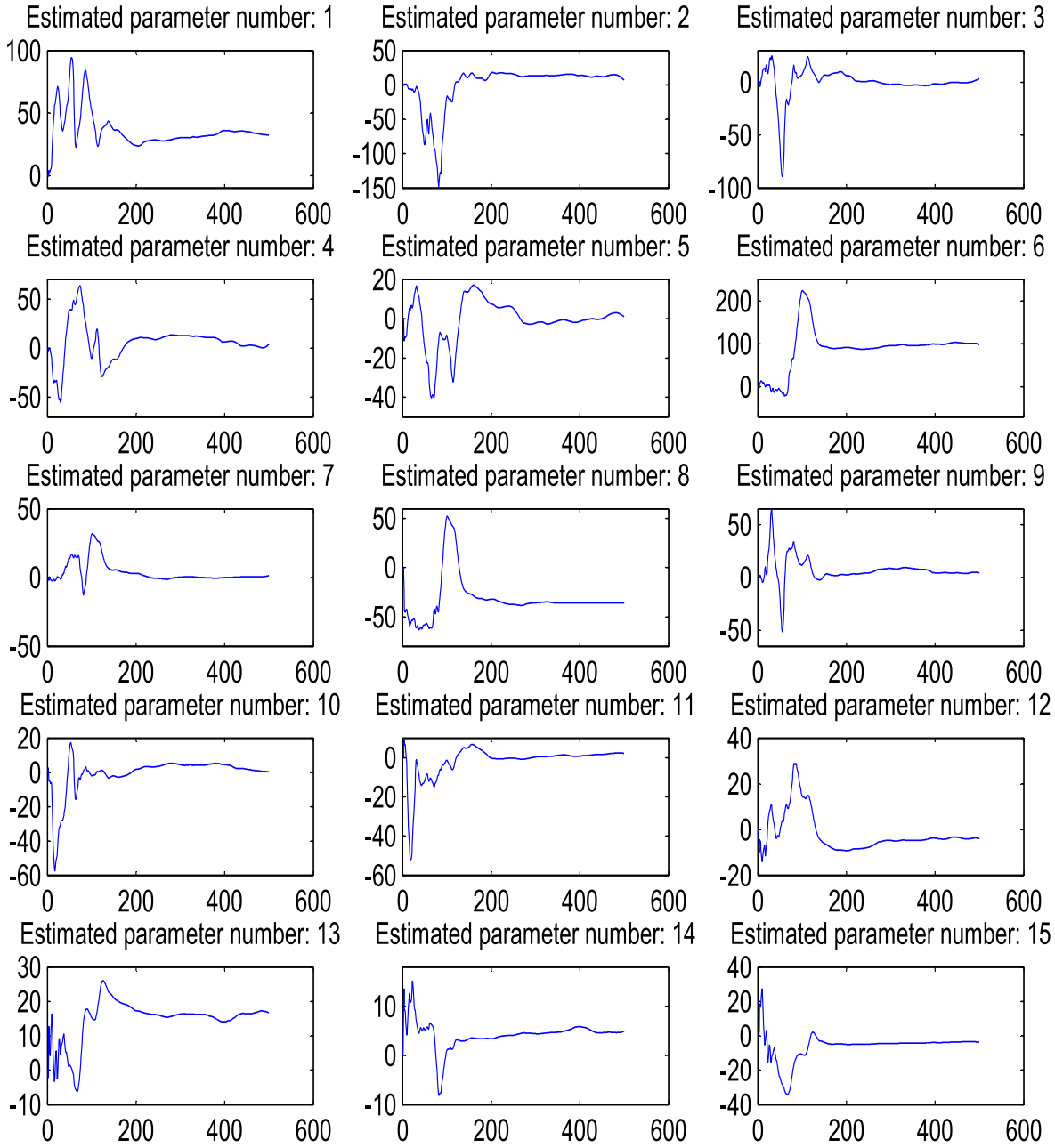


Figure 2.15: **Estimated base dynamic parameters θ_{B1} to θ_{B15} of KUKA robot**

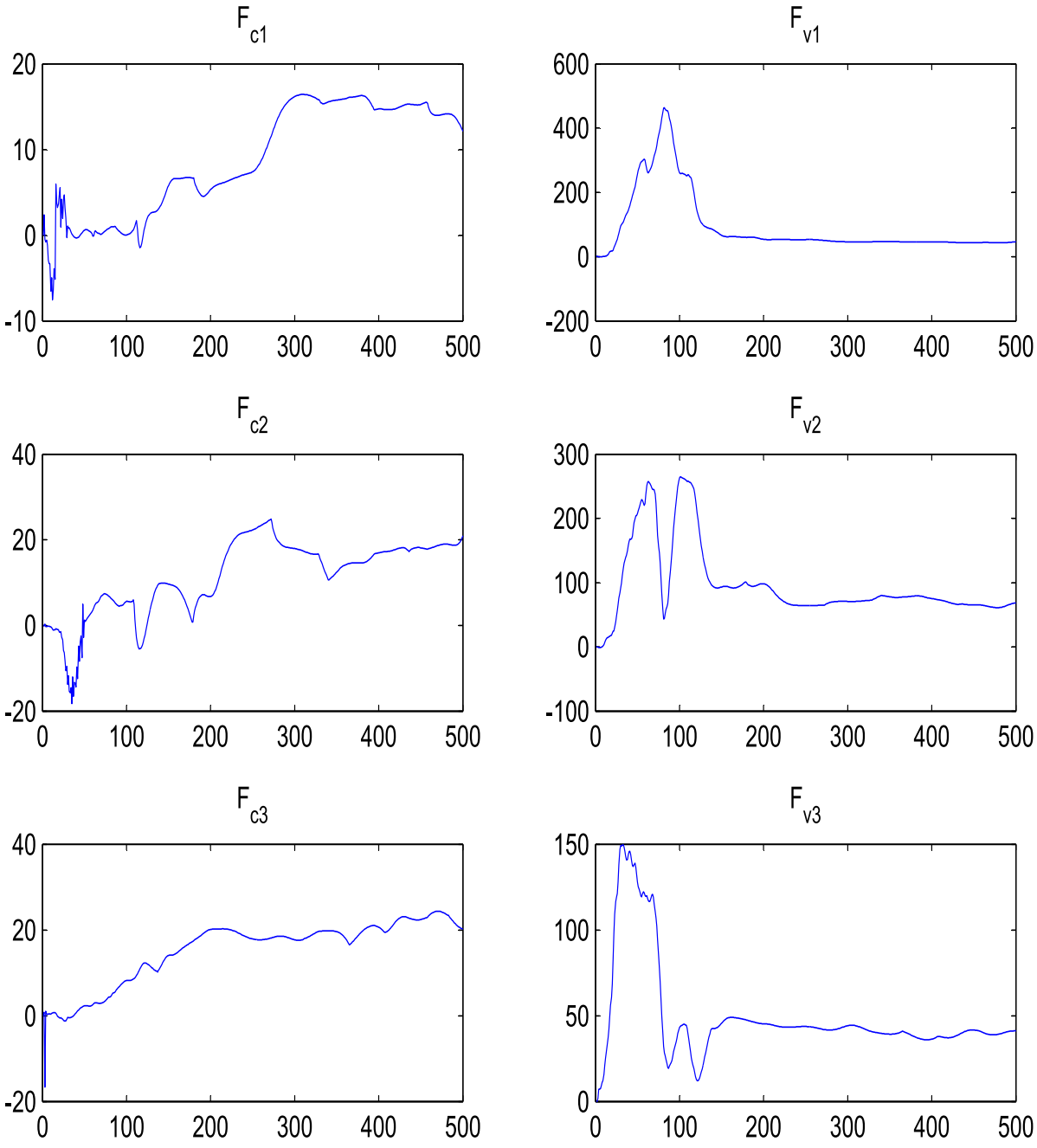


Figure 2.16: **Estimated friction parameters of KUKA robot**

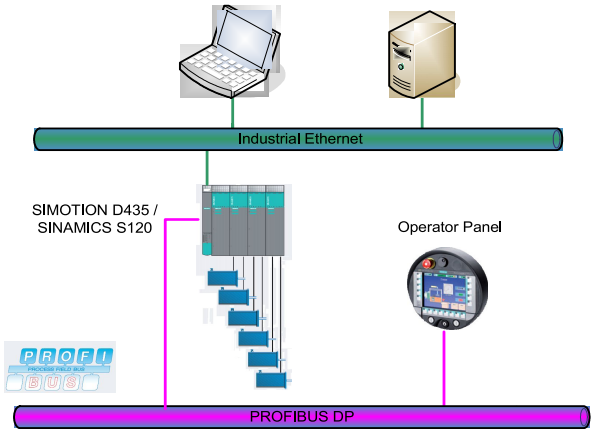


Figure 2.17: Simotion Control System (SCS)

with a new control system consisting of the standard features of the Siemens Industrial Automation. The advantages of this system are mainly in its openness, which allows both to monitor any variable with which the system works (actual current, speed, position values of the axes etc.), but also allows influencing the position cascade control structure of each axis. The last but not least the system gives the possibility to implement kinematics and dynamic model of the mechanism.

The power level is formed by the servo drive SINAMICS S120 that performs control of each axis based on cascade position control (see Figure 2.17). As a superior control unit is used the motion controller SIMOTION D435, which enables higher positioning control method, such as leading a position of the axes along the pre-defined position profiles, synchronization of the axes on the each other, but also gives scope for creating of the control code such as relations for kinematics or dynamics of the mechanism.

As mentioned above SIMOTION controller allows to control the position of individual axes in accordance with pre-defined position profiles and simultaneously record physical quantities such as actual torque, speed, position of these axes, together with their timestamps. These measurements can be viewed by the software scope or exported to the data file and process them by means of other software as MATLAB. This has been advantageously used in our experiments.

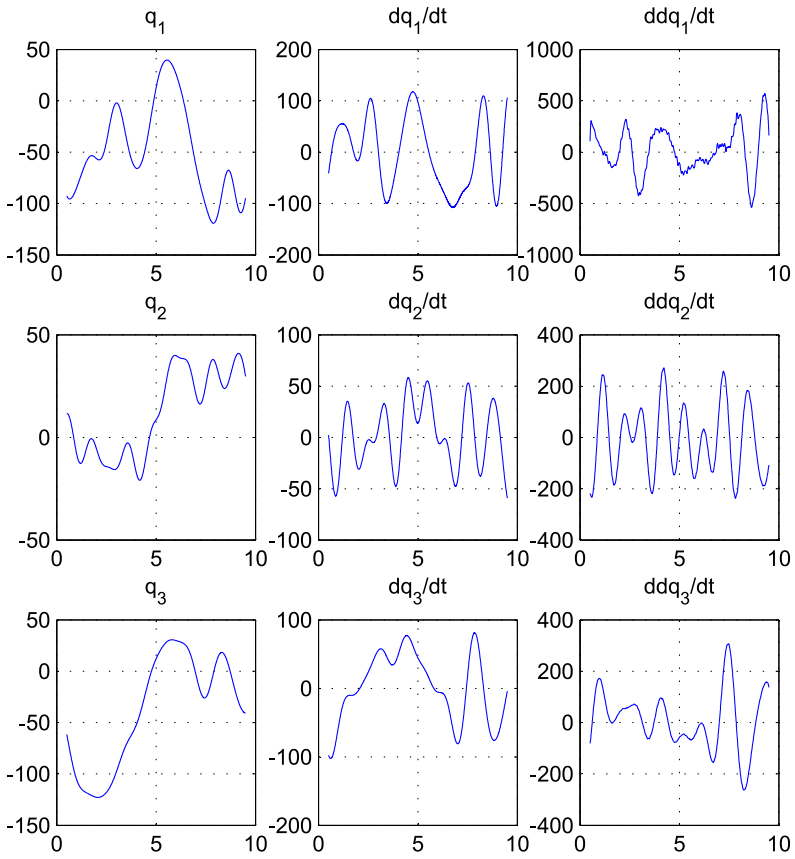


Figure 2.18: **Validation trajectory to check the estimated model**

2.3.8 Validation

The accuracy of the estimated parameters can be validated by some different trajectory shown in Figure 2.18. In doing so, we command the robot to move along these trajectories and then measure the torques by both SCS and estimated model. Figures 2.19 to 2.21 show the torques of the first three joints of the KUKA robot obtained from SCS and our estimated model, respectively. The average values of the error between these torques are 2.66%, 1.04% and 7.43%, respectively.

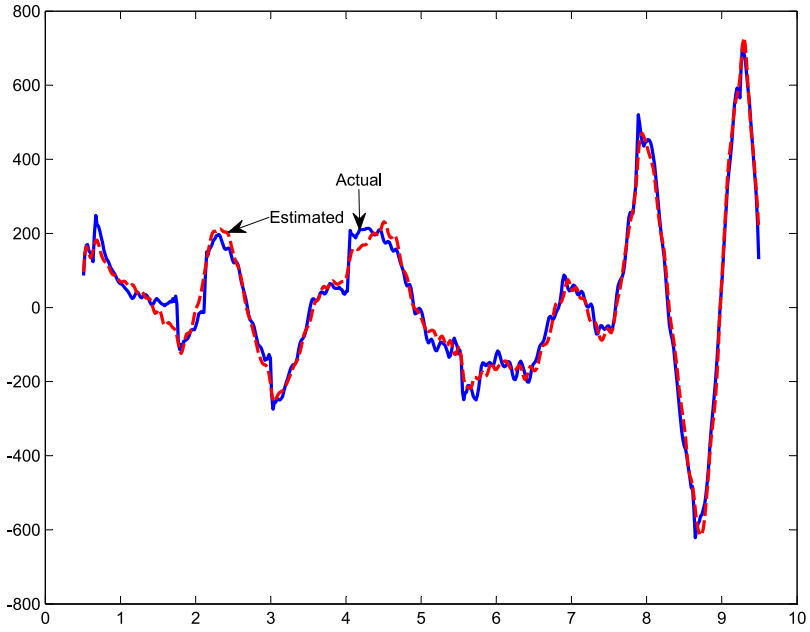


Figure 2.19: **Validation results for actual and estimated Γ_1**

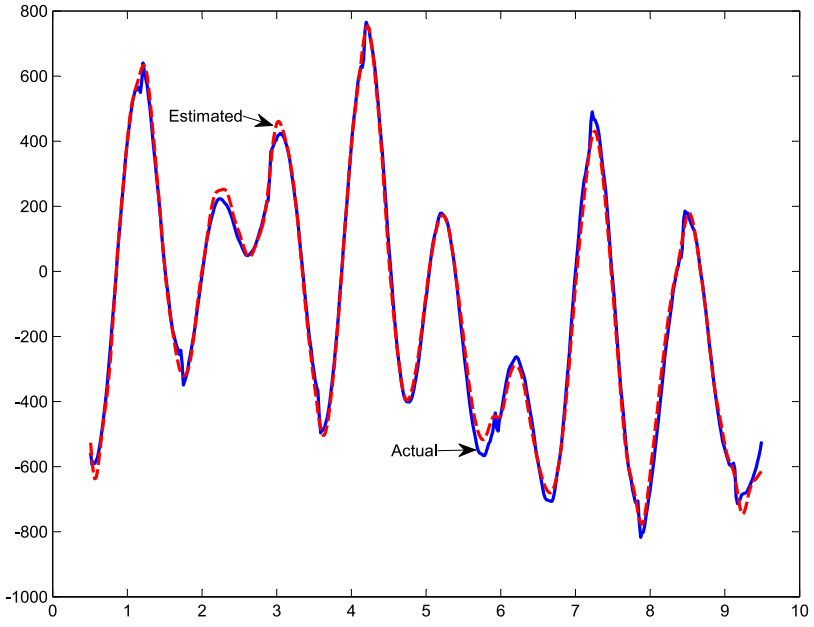


Figure 2.20: **Validation results for actual and estimated Γ_2**

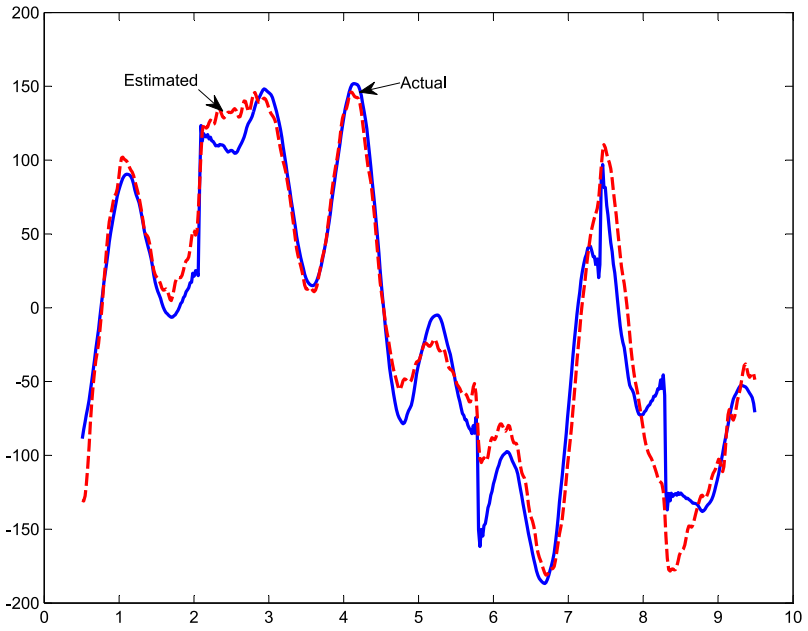


Figure 2.21: **Validation results for actual and estimated Γ_3**

Chapter 3

Optimal Control of Robot Manipulators

3.1 Introduction

In both classical and modern control theories, the objective is to obtain the control input(s) (or control law(s)) for dynamic systems so that the output(s) of the system can follow the desired input(s) (tracking problem) or reaches to the constant desired input(s) (regulating problem). In recent decades, some advanced control strategies have been introduced to design control systems with better performance such as optimal control, robust control, adaptive control, iterative learning control and so on.

Optimal control theory is one of the interesting subjects that was seriously researched by Pontryagin [74]. Roughly speaking, optimal control which is an extension of calculus of variations, deals with designing a control input for a dynamic system so that a performance criterion is minimized without exceeding the constraints on the system. It has a lot of applications in the different scopes such as designing of control system for spacecrafts [53], robot manipulators, chemical processes [76] and so on [46]. In fact, in these systems the power (energy or fuel) consumption and elapsed time for executing a task are very important; therefore, designing an optimal control system is the best choice.

The application of this kind of control methodology can be appropriate in controlling of mechanical robots. For instance, for increasing the productivity of a factory

in which there are a few industrial robots, it is necessary that robots operate as fast as possible. One solution is using more powerful actuators that drive the robots to move at higher speed. However, this method causes bigger actuators which in turn the robots need to consume more energy. Thus, this method is not so economical. The other possibility is to control the robots optimally. That is, they operate so that a performance criterion characterized the traversal time or energy consumption or both can be minimized at the same time. This topic has taken the time of many researchers to design the optimal controller for dynamic systems and specifically mechanical robot manipulators. In this chapter, the OCP of manipulators and some important and efficient methods to solve the OCP of robot manipulators shall be defined and discussed.

3.2 Optimal Control Problem and Various Performance Criteria

First, let us formulate an OCP:

P1[1] : Given the following dynamical system, described by a differential equation in the state space:

$$\dot{\mathbf{x}} = \mathbf{f}\left(\mathbf{x}\left(t\right), \mathbf{u}\left(t\right), t\right), \quad \mathbf{x}\left(t_0\right) = \mathbf{x_0}. \tag{3.1}$$

Find an admissible optimal control $\mathbf{u}^* : [t_0, t_f] \rightarrow \Omega \subseteq \mathbb{R}^m$ so that the dynamic system (3.1) is transfered from the initial state $\mathbf{x}\left(t_0\right) = \mathbf{x_0}$ into an admissible final state $\mathbf{x}\left(t_f\right) \in S \subseteq \mathbb{R}^n$ and the corresponding state trajectory $\mathbf{x}\left(.\right)$ satisfies the state constraint $\mathbf{x}\left(t\right) \in \Omega_x\left(t\right) \subseteq \mathbb{R}^n$ at all the time instances of interval $t \in [t_0, t_f]$ as well as the cost functional

$$J = \phi\left(\mathbf{x}\left(t_f\right), t_f\right) + \int_{t_0}^{t_f} L\left(\mathbf{x}\left(t\right), \mathbf{u}\left(t\right), t\right) dt \tag{3.2}$$

is minimized. Notice that t_f and $\mathbf{x}\left(t_f\right)$ can be either fixed or free, either one or both of them. Besides, in OCPs attention must be paid to existence and uniqueness of its solution. Moreover, bear in mind that the optimal control can be in closed-loop or

[1] In the next sections, we refer this problem as "P1"

open-loop forms; hence, if $\mathbf{u}^*(t) = \mathbf{e}(\mathbf{x}(t), t)$; then the optimal control is closed-loop and if $\mathbf{u}^*(t) = \mathbf{e}(\mathbf{x}(t_0), t)$, it is an open-loop.

As the problem "P1" indicates, the performance criterion is the distinguished point than the other control strategies. Therefore, it is convenient to be examined more. In the sequel, different types of performance criteria are listed [52; 56]:

- **Minimum-time problem**: In the problem "P1", the system is supposed to transfer from initial position (state) $\mathbf{x}(t_0)$ to some admissible final position (state) $\mathbf{x}(t_f)$ in minimum time; So

$$J = t_f - t_0 = \int_{t_0}^{t_f} dt. \tag{3.3}$$

- **Terminal control problem**: In the problem "P1", if the objective is to minimize the difference between final state of the system and its desired value evaluated in t_f, i.e. $\mathbf{r}(t_f)$, then

$$J = \|\mathbf{x}(t_f) - \mathbf{r}(t_f)\|_H^2 = [\mathbf{x}(t_f) - \mathbf{r}(t_f)]^T H [\mathbf{x}(t_f) - \mathbf{r}(t_f)], \tag{3.4}$$

  which H is a positive semi-definite matrix (very often as $diag(h_{ii})$).

- **Minimum-control-effort problem**: In problem "P1", consider that the system is forced to change its state from initial position (state) $\mathbf{x}(t_0)$ to some admissible final position (state) $\mathbf{x}(t_f)$ with minimum consumption of energy; thus

$$J = \int_{t_0}^{t_f} \left[\mathbf{u}^T(t) R\mathbf{u}(t)\right] dt = \int_{t_0}^{t_f} \|\mathbf{u}(t)\|_R^2 dt, \tag{3.5}$$

  which R is a Hermitian symmetric positive definite matrix.

- **Tracking and minimum-control-effort problem**: In problem "P1", if the system is to be excited to follow the desired trajectory $\mathbf{r}(t)$, it is necessary that the deviation of the state trajectory of the system from this desired (reference) trajectory is minimized in all the time instances; that is:

$$J = \|\mathbf{x}(t_f) - \mathbf{r}(t_f)\|_H^2 + \int_{t_0}^{t_f} \left[\|\mathbf{x}(t) - \mathbf{r}(t)\|_{Q(t)}^2 + \|\mathbf{u}(t)\|_{R(t)}^2\right] dt, \tag{3.6}$$

where $Q(t)$ is a real symmetric $n \times n$ positive semi-definite matrix, $R(t)$ is a real symmetric $m \times m$ hermitian positive definite matrix and H is the real symmetric $n \times n$ positive semi-definite matrix.

- **General Case:** The general form of cost functional can be as follows

$$J = \varphi(\mathbf{x}(t_f), t_f) + \int_{t_0}^{t_f} L(\mathbf{x}(t), \mathbf{u}(t), t)\, dt, \tag{3.7}$$

where φ and L are a nonlinear function of their variables.

3.3 Different Approaches for Solving Optimal Control Problems

The different kinds of OCPs can generally fall into two categories: discrete and continuous which, in turn, each of which has several subcategories. On the other hand, this categorization can be done on the basis of how the OCOs are solved: analytically or numerically. In general, there are three main approaches to solve the OCPs of dynamical systems:

- dynamic programming method
- indirect method
- direct method

These methods will be considered in the next subsections.

3.3.1 (Discrete) Dynamic Programming Method

In this method, with discretization of the system's dynamic equations and quantizing control input and state variables and then applying the *principle of optimality*, we can derive a recurrence equation called *Bellman's recurrence equation* by which the optimal policy (control law) can be obtained [16; 56].

3.3.2 Hamilton-Jacobi-Bellman Method (Continuous Dynamic Programming)

Hamilton-Jacobi-Bellman (HJB) equation is a partial differential equation by which the solution of the OCPs can be obtained [64]. In the case of problem P1 given in section 5.3.2, first a function called *Hamiltonian* is defined as follows:

$$H\left(\mathbf{x}\left(t\right),\mathbf{u}\left(t\right),J_{\mathbf{x}}^{*},t\right)=L\left(\mathbf{x}\left(t\right),\mathbf{u}\left(t\right),t\right)+J_{\mathbf{x}}^{*T}\left(\mathbf{x}\left(t\right),t\right)\left[\mathbf{f}\left(\mathbf{x}\left(t\right),\mathbf{u}\left(t\right),t\right)\right]. \tag{3.8}$$

In order to obtain the optimal control $u^{*}\left(t\right)$, it is proved that J^{*} must be obtained from the following partial differential equation so-called HJB equation

$$\begin{cases} 0=J_{t}^{*}\left(\mathbf{x}\left(t\right),t\right)+H\left(\mathbf{x}\left(t\right),\mathbf{u}^{*}\left(\mathbf{x}\left(t\right),J_{\mathbf{x}}^{*},t\right),J_{\mathbf{x}}^{*},t\right) \\ \text{Boundary condition:}\quad J^{*}\left(\mathbf{x}\left(t_{f}\right),t_{f}\right)=\phi\left(\mathbf{x}\left(t_{f}\right),t_{f}\right) \end{cases}. \tag{3.9}$$

Notice that HJB equation is a time continuous analogous of the Bellman's recurrence equation given in dynamic programming method.

3.3.3 Indirect Methods

Generally, *Calculus of Variations* (variational techniques) is a field of mathematics which handles the functionals instead of ordinary functions. As mentioned earlier, the most tasks in optimal control are to optimize an integral function characterized the time or consumed energy by the system. The Lagrange's equation in calculus of variations is used for solving the unconstrained OCPs while the *Pontryagin's minimum principle* can be used to solve the constrained OCPs [46]. Since we face with constrained OCPs for mechanical systems, the Pontryagin's minimum principle is dealt with more more. This principle is expressed under the following definition and theorem:

Definition: $H(\mathbf{x},\mathbf{u},\lambda,t)=L\left(\mathbf{x}\left(t\right),\mathbf{u}\left(t\right),t\right)+\lambda^{T}\left(t\right)\ \mathbf{f}\left(\mathbf{x}\left(t\right),\mathbf{u}\left(t\right),t\right)$ is called *Hamiltonian* where $\lambda^{T}\left(t\right)\in\mathbb{R}^{n}$.

Theorem (Pontryagin's Minimum Principle):
If $\mathbf{f}\left(\mathbf{x}\left(t\right),\mathbf{u}\left(t\right),t\right)$ is continuous in $(\mathbf{x},\mathbf{u},t)$ and the derivatives $\frac{\partial}{\partial t}\mathbf{f}$ and $\bigtriangledown_{\mathbf{x}}\mathbf{f}$ exist and are continuous in $(\mathbf{x},\mathbf{u},t)$, then the necessary conditions that control $\mathbf{u}^{*}$ be an optimal

control are

$$\begin{gathered}\dot{\mathbf{x}}^{*}(t)=\frac{\partial H}{\partial \lambda}\left(\mathbf{x}^{*}(t),\mathbf{u}^{*}(t),\lambda^{*}(t),t\right),\\ \dot{\lambda}^{*}(t)=-\frac{\partial H}{\partial \mathbf{x}}\left(\mathbf{x}^{*}(t),\mathbf{u}^{*}(t),\lambda^{*}(t),t\right),\\ H\left(\mathbf{x}^{*}(t),\mathbf{u}^{*}(t),\lambda^{*}(t),t\right)\leq H\left(\mathbf{x}^{*}(t),\mathbf{u}(t),\lambda^{*}(t),t\right), \qquad .\\ \text{for all } \mathbf{u}(t)\in\Omega,\\ \left[\frac{\partial \phi}{\partial \mathbf{x}}\left(\mathbf{x}^{*}(t_f)-\lambda^{*}(t_f)\right)\right]^{T}\delta\mathbf{x}_f+\left[H\left(\mathbf{x}^{*}(t_f),\mathbf{x}^{*}(t_f),\mathbf{x}^{*}(t_f),t_f\right)+\frac{\partial \phi}{\partial t}\left(\mathbf{x}^{*}(t_f),t_f\right)\right]\end{gathered} \tag{3.10}$$

The necessary conditions (3.10) result in a *two-point boundary-value (TPBV) problem* whose solution is very difficult to obtain and very often is analytically impossible. Therefore, the numerical methods such as steepest descent, quasilinearization or gradient projection methods can be used for solving them [46; 52; 56; 75]. One limitations of these numerical methods is that it is necessary to guess an appropriate initial solution which is usually a cumbersome task.

3.3.4 Direct methods

The basic idea in the direct methods is that the solution of OCP is obtained by directly minimizing the cost functional given in OCP. In doing so, firstly the infinite dimensional OCP is converted into a finite nonlinear programming problem (NLP) via a parameterization procedure and then this NLP can be solved using the existing standard methods like sequential quadratic programming (SQP). Note that direct methods can be categorized as follows [22; 36]:

- Single shooting
- Collocation
- Multiple shooting
- Spline based optimal control

In the next sections we will firstly formulate the OCP of robot manipulators and then we shall explain the multiple shooting and spline based optimal control methods in the case of robot manipulators which will be used in the chapter 5 to compare with the proposed method.

3.4 Optimal Control Problem of Open-Chain Robot Arms

In this section, we formulate the OCP of robot manipulators. In doing so, we require the state space representation of the dynamic model of the robot. In the following subsections these subjects are presented.

3.4.1 State Space Representation of Robot Manipulators

As represented in the previous chapter, the motion equations of an n-axes robot manipulator can be expressed as

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) + F(\dot{q}) = \tau \tag{3.11}$$

where q is the $n \times 1$ vector of joint displacements, $\dot{q}$ is the $n \times 1$ vector of joint velocities, τ is the $n \times 1$ vector of input torques, $M(q)$ is the $n \times n$ symmetric positive definite inertia matrix, $C(q,\dot{q})$ is $n \times n$ matrix of centripetal and Coriolis torques, $G(q)$ is the $n \times 1$ vector of gravitation torques obtained as the gradient of the robot's potential energy due to gravity, and $F(\dot{q})$ is the $n \times 1$ friction torques [58]. The robot dynamics (3.11) can be rewritten as follows

$$M(q)\ddot{q} + N(q,\dot{q}) = \tau, \tag{3.12}$$

where

$$N(q,\dot{q}) = C(q,\dot{q})\dot{q} + G(q) + F(\dot{q}). \tag{3.13}$$

In order to study the dynamic systems, it is better to represent them in the state space. Thus, because $M(q)$ is invertible, the equation (3.12) can be written as

$$\ddot{q} = -M^{-1}(q)N(q,\dot{q}) + M^{-1}(q)\tau. \tag{3.14}$$

Then by defining the position/velocity state $x \in \mathbb{R}^{2n}$ as

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} q \\ \dot{q} \end{bmatrix}, \tag{3.15}$$

the state space representation of (3.14) will be in the form

$$\dot{x} = \begin{bmatrix} x_2 \\ -M^{-1}(x_1) N(x_1, x_2) \end{bmatrix} + \begin{bmatrix} 0 \\ M^{-1}(x_1) \end{bmatrix} \tau. \tag{3.16}$$

Therefore, equation (3.16) can be written as

$$\dot{x} = f(x) + g(x) \tau, \tag{3.17}$$

where

$$f(x) = \begin{bmatrix} x_2 \\ -M^{-1}(x_1) N(x_1, x_2) \end{bmatrix}, \tag{3.18}$$

and

$$g(x) = \begin{bmatrix} 0 \\ M^{-1}(x_1) \end{bmatrix}. \tag{3.19}$$

3.4.2 Formulation of Robot OCP

First of all, It is worth to be noted that the cost functional used in the OCP of robots can be one of the following cases: [11; 14; 57; 89]:

- minimum time

$$J_c = \int_0^{t_f} dt = t_f \qquad t_f \text{ is free}, \tag{3.20}$$

- minimum energy consumption

$$J_c = \int_0^{t_f} \sum_{i=0}^{n} (\tau_i(t))^2 \, dt \qquad t_f \text{ is fixed}, \tag{3.21}$$

- minimum power consumption

$$J_c = \int_0^{t_f} \sum_{i=0}^{n} (\dot{q}_i(t) \tau_i(t))^2 \, dt \qquad t_f \text{ is fixed}, \tag{3.22}$$

Hence, the OCP of robot manipulators can be formulated as follows:

$$\begin{aligned} &\min_{\tau} J_c \\ &\textbf{subject to} \end{aligned} \tag{3.23a}$$

$$\text{robot dynamics:} \qquad \dot{x} = f\left(x\left(t\right)\right) + g\left(x\left(t\right)\right)\tau\left(t\right), \tag{3.23b}$$

$$\text{constraints:} \begin{cases} \textit{torque constraints:} & |\tau_i| \leq \tau_{i,max} \quad , \quad i = 1, 2, \ldots, n \\ \textit{position constraints:} & |q_i| \leq q_{i,max} \quad , \quad i = 1, 2, \ldots, n \\ \textit{velocity constraints:} & |\dot{q}_i| \leq \dot{q}_{i,max} \quad , \quad i = 1, 2, \ldots, n \\ \textit{position boundary conditions:} & q_i\left(0\right) = q_{i0} \quad , \quad i = 1, 2, \ldots, n \\ \textit{velocity boundary conditions:} & \dot{q}_i\left(0\right) = \dot{q}_{i0} \quad , \quad i = 1, 2, \ldots, n \\ \textit{end conditions:} & q_i\left(t_f\right) = q_{if} \quad , \quad i = 1, 2, \ldots, n \\ \textit{path constraint:} & \mathbf{h}(\mathbf{q}) = 0. \end{cases} \tag{3.23c}$$

Usually, the OCP (3.23) is given rise to a constrained nonlinear programming which will be dealt with in the next subsection.

3.5 Constrained Nonlinear Programming (NLP)

As explained earlier, the direct methods convert the OCPs into a constrained nonlinear programming. The NLP can be represented as follows [13; 18],

$$\begin{aligned} &\min_{x \in X} f\left(x\right) \\ &\text{subject to} \\ &G_i\left(x\right) = 0 \ , \ i = 1, 2, \ldots, m_e, \\ &G_i\left(x\right) \leq 0 \ , \ i = m_e + 1, \ldots, m, \end{aligned} \tag{3.24}$$

where $X \subset \mathbb{R}^n$, $x \in X$ includes designed parameters, $f : X \to \mathbb{R}$ and $G : X \to \mathbb{R}^m$ are continuous functions. The function f is called *cost (objective, criterion)* function. Moreover, each of $G_i\left(x\right),\ i = 1, 2, \ldots, m_e$ is called *inequality* constraint, while each

of $G_i(x)$, $i = m_e + 1, 2, \ldots, m$ is called an *equality* constraint. In addition, a vector $x \in X$ satisfying all the constraints is called a *feasible* solution to the problem; the collection of all such points forms the feasible region. The goal in the NLP problem (3.24) is to find a feasible point x^* so that $f(x) \geq f(x^*)$ for each feasible point x. In the sequel, the method to solve NLP problem is given.

Let us first define a new function called *Lagrangian*:

$$L(x, \lambda) = f(x) + \sum_{i=1}^{m} \lambda_i \cdot G_i(x), \tag{3.25}$$

where λ_is are called *Lagrange's multipliers*. Therefore, the optimal solution of the constrained NLP (3.24) is obtained by satisfying the following equations referred to as the so-called *Karush-Kuhn-Tucker* (KKT) equations:

$$\begin{aligned} &\nabla L(x^*, \lambda^*) = 0, \\ &\lambda_i G_i(x^*) = 0\ ,\ i = 1, 2, \ldots, m_e, \\ &\lambda_i \geq 0\ ,\ i = m_e + 1, \ldots, m, \end{aligned} \tag{3.26}$$

where ∇ is the gradient operator.

It must be noted that (KKT) equations (3.26) are just the necessary conditions for obtaining a local optimal solution to the NLP (3.24). However, if both functions $f(x)$ and $G(x)$ are convex, the KKT conditions are necessary and sufficient for finding a global optimum point. Usually the solution of KKT equations is obtained numerically by iterative methods. One of the conventional iterative methods is *Sequential Quadratic Programming* (SQP) which is expressed as

$$x_{k+1} = x_k + \Delta x_k, \tag{3.27}$$

$$\lambda_{k+1} = \lambda_k^{QP}, \tag{3.28}$$

where Δx_k and λ_k^{QP} are the solutions of the following *quadratic problem* (QP):

$$\begin{aligned} &\min_{\Delta x} \left[\frac{1}{2}\Delta x^T A_k \Delta x + \nabla f(x_k)\,\Delta x\right], \\ &\text{subject to} \\ &G_i(x_k) + \nabla G_i^T(x_k)\,\Delta x = 0\ ,\ i = 1, 2, \ldots, m_e, \\ &G_i(x_k) + \nabla G_i^T(x_k)\,\Delta x \geq 0\ ,\ i = m_e + 1, \ldots, m, \end{aligned} \tag{3.29}$$

where A_k is the approximation of the Hessian of the Lagrangian, namely

$$A_k \approx \nabla^2 L(x_k, \lambda_k)\,, \tag{3.30}$$

and ∇G_i is the constant Jacobian. Note that there are different methods for computing approximate Hessian, for example, *Gauss-Newton* method.

In MATLAB, there is some toolbox called Optimization toolbox which supports various kinds of optimization problem. For example the command *fmincon()* is the one which solves the constrained NLP [63].

3.6 Parametric Optimization (Spline-Based Optimal Control)

As explained in section 3.3, the different subbranches of direct method can be used to solve the OCPs. In particular, in the resent decades most of the researches have used these methods to solve the OCP of robot manipulators [27; 30; 69; 92]. The basic idea in the spline-based optimal control method is that the state or control trajectory of the system is approximated by some kind of parameterized approximating functions such as spline functions (cubic splines, B-splines or Bezier splines) which actually the objective is to convert the infinite dimensional OCP into a finite dimensional one. In the case of robot manipulators, the chosen trajectory should have sufficient smoothness properties; since it give rise to avoid the excitation of the mechanical resonance modes of the manipulator which damage the actuator of the robot. If the acceleration of the used trajectory is continuous function so that the value of the jerk (derivative of acceleration) is bounded, the resonance modes are not excited. This feature can be

found in the cubic spline functions or B-splines.

Let us now suppose a robot manipulator modeled as

$$M(q)\ddot{q} + C(q,\dot{q}) + F(\dot{q}) + G(q) = \tau, \tag{3.31}$$

where q and τ are n-vectors of joint variables and of generalized forces, respectively. $M(q)$ is the inertia matrix, $C(q,\dot{q})$ the Coriolis/centripetal vector, $F(\dot{q})$ the friction vector and $G(q)$ the gravity vector. After some simplifications, equation (3.31) can be expressed as

$$M(q)\ddot{q} + N(q,\dot{q}) = \tau, \tag{3.32}$$

where $N(q,\dot{q}) = C(q,\dot{q}) + F(\dot{q}) + G(q)$.

Note that the state, control and boundary constraints and also cost functional can be considered as

$$\text{cost functional:} \qquad J = \phi(q(T),\dot{q}(T)) + \int_0^T L(q(t),\dot{q}(t),\tau(t))\,dt, \tag{3.33}$$

$$\text{constraints:} \qquad \begin{cases} q^i_{min} \le q^i(t) \le q^i_{max},\ i = 1,2,\ldots,n, \\ \tau^i_{min} \le \tau^i(t) \le \tau^i_{max},\ i = 1,2,\ldots,n, \\ q^i(0) = q^i_0,\ q^i(T) = q^i_T,\ i = 1,2,\ldots,n. \end{cases} \tag{3.34}$$

Let us now consider a cubic spline function shown in Figure 3.1 as

$$S(t) = \begin{cases} s_1(t) & t_1 \le t < t_2, \\ s_2(t) & t_2 \le t < t_3, \\ \vdots & \vdots \\ s_{N-1}(t) & t_{N-1} \le t < t_N, \end{cases} \tag{3.35}$$

where s_i is a third degree polynomial defined by

$$s_i(t) = a_i(t-t_i)^3 + b_i(t-t_i)^2 + c_i(t-t_i) + d_i, \tag{3.36}$$

for $i = 1,2,\ldots,N-1$. So as to achieve a smooth trajectory, the following conditions should be met:

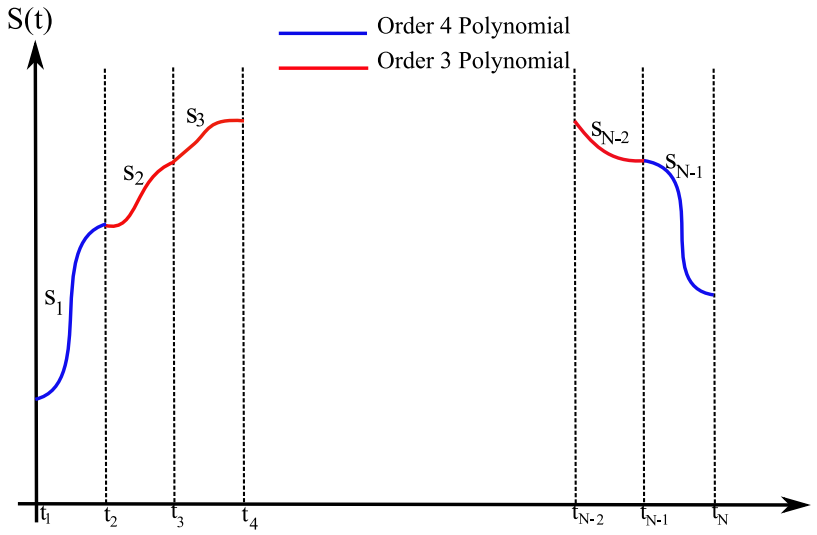


Figure 3.1: **A typical spline function**

1. $s_1(t_1) = q_0,\ s_{N-1}(t_N) = q_T,$
2. $\dot{s}_1(t_1) = \dot{q}_0,\ \dot{s}_{N-1}(t_N) = \dot{q}_T,$
3. $\ddot{s}_1(t_1) = 0,\ \ddot{s}_{N-1}(t_N) = 0,$
4. $s_i(t_{i+1}) = s_{i+1}(t_{i+1})$ for $i = 1, 2, \ldots, N-2,$
5. $\dot{s}_i(t_{i+1}) = \dot{s}_{i+1}(t_{i+1})$ for $i = 1, 2, \ldots, N-2,$
6. $\ddot{s}_i(t_{i+1}) = \ddot{s}_{i+1}(t_{i+1})$ for $i = 1, 2, \ldots, N-2.$

It is clear by satisfying the above conditions, some of the parameters a_i, b_i, c_i and d_i are dependent while others are independent. The number of independent parameters is calculated by $m = N + 3 - \alpha$ where α is the number of boundary conditions in each joint position of the robot which usually $\alpha = 4$, then $m = N - 1$ [95]. Substituting equation (3.35) and its first and second derivative into dynamics (3.32), converts it into a set of parametric differential equations by which the parametric control inputs are obtained in terms of parameters of a_i, b_i, c_i and d_is. Then, with substituting the parametric states and control inputs into the cost functional (3.33), it is obtained as $J(a_i^1, a_i^2, \ldots, a_i^n)$ where $i = 1, 2, \ldots, N-1$ and n is the robot's degree of freedom.

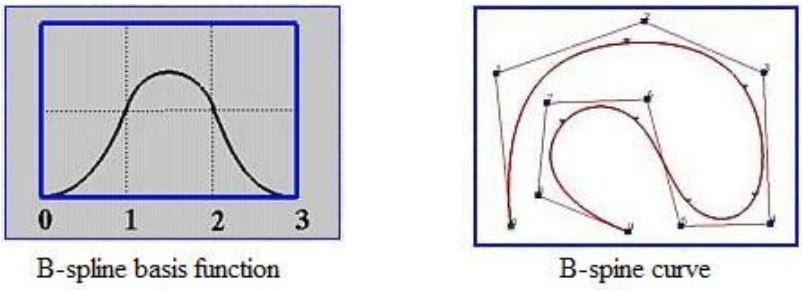


Figure 3.2: **A generic B-spline basis and curve**

Hence, in this way the original infinite problem is converted into a finite NLP which can then be solved by the common methods for solving NLPs such as SQP or genetic algorithms.

As stated above, B-spline functions can also be used to convert the OCPs into parametric optimization problems. In the sequel, a brief explanation concerning this kind of spline functions is brought. In general, they have the following form

$$q(t) = \sum_{i=0}^{n} N_{i,p}(t) P_i, \tag{3.37}$$

where $N_{i,p}$s are B-spline basis functions of degree p and P_i's are referred to as the *control points* while n is the number of these control points. The B-spline basis functions are defined recursively as

$$\begin{aligned} N_{i,0}(t) &= \begin{cases} 1 & t_i \leq t \leq t_{i+1} \\ 0 & \text{otherwise} \end{cases}, \\ N_{i,p}(t) &= \frac{t - t_i}{t_{i+p} - t_i} N_{i,p-1}(t) + \frac{t_{i+p+1} - t}{t_{i+p+1} - t_{i+1}} N_{i+1,p-1}(t). \end{aligned} \tag{3.38}$$

For instance if $p = 4$, then cubic B-spline curves is obtained. Figure (3.2) shows some generic B-spline basis and curves. Note that in this case the control points are used as parameters in nonlinear programming. Moreover, the most important feature of the B-spines is that the variation a control point does not change shape of the whole B-spline curve. It means that the influence of control points is local not global.

3.7 Multiple Shooting Method

Multiple shooting method, as mentioned earlier, is one of the subbranches of direct methods [24; 36]. In this section we attempt to consider the algorithm used in this method.

Let us now formulate the OCP of robot manipulators as follows:

$$\min_{\tau} J, \tag{3.39}$$

where $J \in \mathbb{R}$ is defined as

$$J = \phi\left(x\left(T\right)\right) + \int_0^T L\left(x\left(t\right), \tau\left(t\right)\right) dt, \tag{3.40}$$

subject to

$$\text{dynamic equation of robot:} \qquad \dot{x} = f\left(x\right) + g\left(x\right)\tau, \tag{3.41}$$

$$\text{boundary conditions} x\left(0\right) = x_0,\ x\left(T\right) = x_T, \tag{3.42}$$

$$\text{constraints:} \qquad \begin{cases} x_{min} \leq x\left(t\right) \leq x_{max}, \\ \tau_{min} \leq \tau\left(t\right) \leq \tau_{max}, \\ h\left(x\left(T\right)\right) = 0. \end{cases} \tag{3.43}$$

The procedure in which multiple shooting method solves the OCP includes the following steps:

- Step 1:

  Partition the time interval $[0, T]$ to N segments (see Figure 3.3).

- Step 2:

  Consider a piecewise control history for $\tau\left(t\right)$ so that in the subinterval $[t_i, t_{i+1}]$ it is

  $$\tau\left(t\right) = a_i. \tag{3.44}$$

- Step 3:

  Solve the dynamics of the system independently on each subinterval, considering

an assumed initial condition

$$\begin{aligned} \dot{x}_i(t) &= f(x_i(t)) + g(x_i)\, a_i, \\ x_i(t) &= b_i. \end{aligned} \tag{3.45}$$

Therefore, in this step, we obtain a sequence of state trajectories $\{x_i(t; a_i, b_i)\}_{i=0}^{N-1}$.

- Step 4: Calculate the cost functional (numerically) in each subinterval by trajectory obtained on that subinterval, i.e.

$$K_i = \int_{t_i}^{t_{i+!}} L(x_i, a_i)\, dt. \tag{3.46}$$

- Step 5: Obtain the sequence of continuity conditions in each grid point, i.e., $\{b_{i+1} = x_i(t_{i+1}; a_i, b_i)\}_{i=1}^{N-1}$.

- Step 6: Solve the following NLP obtained from the above steps

$$\begin{aligned} &\min_{a_i, b_i} J = \phi(b_N) + \sum_{i=0}^{N-1} K_i, \\ &\text{subject to} \\ &b_0 - x_0 = 0 \\ &b_{i+1} = x_i(t_{i+1}; a_i, b_i),\ i = 0, 1, \ldots, N-1, \\ &x_{min} \leq b_i \leq x_{max} = 0,\ i = 1, 2, \ldots, N, \\ &\tau_{min} \leq a_i \leq \tau_{max} = 0,\ i = 0, 2, \ldots, N-1, \\ &h(b_N) = 0. \end{aligned} \tag{3.47}$$

3.8 Iterative Learning Control (ILC)

In this section, the iterative learning control method which is one of the modern control techniques for robot arms is dealt with. Let us consider the robots which perform some special task repeatedly; for instance, those which operate in assembly lines to pick and

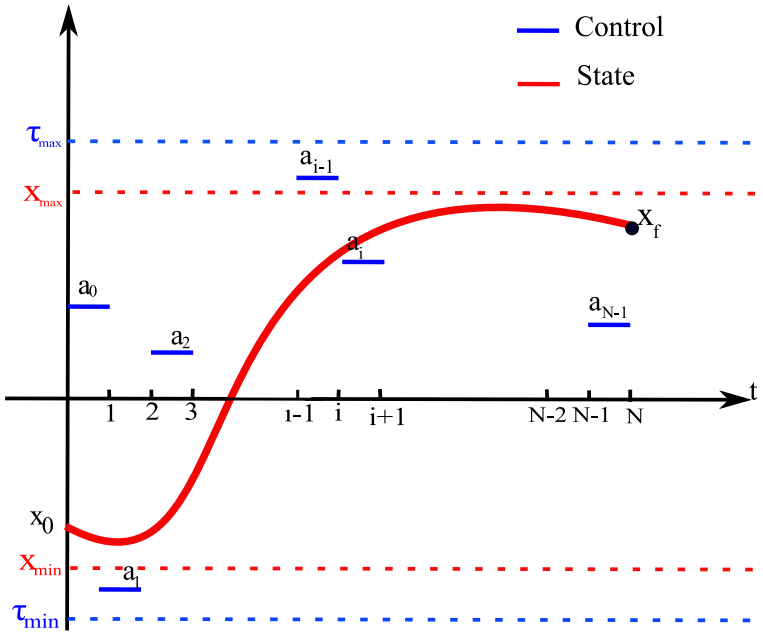


Figure 3.3: **Piecewise constant control (Multiple shooting method)**

place parts. In order to control such robots, firstly Arimoto in [4] proposed a new method so-called ILC. The basic idea in this method is based on learning [3; 4; 93; 94]. Let us consider a perceptible example in which a basketball player should exercise many times to be able enter the ball in basket from penalty point and the player tries to improve his or her performance in each time to throw the ball in a correct path to put it into the basket. In fact, this procedure is also used in ILC schemes. For instance, for controlling a robot in a learning manner, it learns to perform a task with high performance in some finite time interval through several learning courses (trials) and in each new trial, it improves its performance by correcting the previous errors such that after a series of trials, the error sequence converges to zero and the robot finds the best performance in the sense of tracking problem. The proposers of this scheme believe that for controlling the repeated systems, ILC requires very little information about dynamic model of the system, i.e., there is no need to an exact model of the system.

The structure of an ILC is shown in Figure 3.4 according to which the control input of the system in the current trial (step) $u_{k+1}(t)$ is updated by considering the previous value of the error $e_k(t)$ which has been stored in memory of the system. The

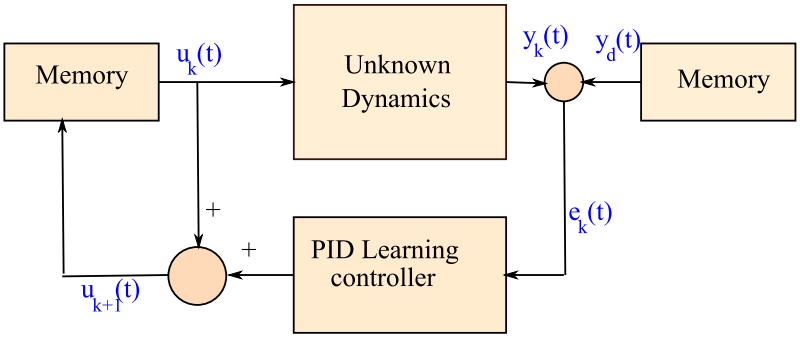


Figure 3.4: **A Typical PID iterative learning control scheme**

previous error $e_k(t)$ equals the difference between desired output $y_d(t)$ and current actual output $y_k(t)$. For a $PID-ILC$, the control input for the next trial is given by

$$u_{k+1}(t) = u_k(t) + \Gamma_P \cdot e_k(t) + \Gamma_D \cdot \frac{\mathrm{d}e_k(t)}{\mathrm{d}t} + \Gamma_I \cdot \int_0^t e_k(t)\, dt, \tag{3.48}$$

where Γ_P, Γ_D and Γ_I are called *learning gains* which must be determined and T is final time. It must also be noted that the system is homed after finishing each trial, i.e. the initial condition for all trials is the same.

In [4], for a class of nonlinear systems, i.e. equation (3.49) which also includes dynamics of the robots, an D iterative learning control scheme has been proposed (refer equation (3.48) and Figure 3.4 with $\Gamma_P = \Gamma_I = 0$). In this work, it is assumed that the state space representation of the system is obtained as follows:

$$\begin{cases} \dot{x} = f(t, x(t)) + Bu, \\ y = Cx, \end{cases} \tag{3.49}$$

where $x, f \in R^n$, $u, y \in R^r$, $C \in R^{r\times n}$ and $B \in R^{n\times r}$, as well as it is assumed that $x_k(0) = x_0$. In this paper, it has been proved that the following conditional relation must be satisfied so that it can be used to compute the D learning gain of the ILC controller, namely

$$\|I_r - CB\Gamma_D\|_\infty < 1, \quad \text{for all } t \in [0, T], \tag{3.50}$$

where $\|\cdot\|_\infty$ is infinity norm which for a vector $a = [a_1, a_2, \ldots, a_n]^T$, for example, this

norm is defined as follows

$$\|a\|_{\infty} = \max\left(|a_1|, |a_2|, \ldots, |a_n|\right). \tag{3.51}$$

Arimmoto and his co-workers in this paper showed that a robot manipulator has a similar dynamics given by (3.49) which is obtained by the Hamiltonian mechanics and then applied the above procedure to control a robot manipulator whose end-effector is tracking a desired path. In the recent years, there exists a series of researches whose authors have combined the ILC and optimal strategies like those presented in [2; 65; 91].

Chapter 4

First Proposed Method

4.1 Introduction

In order to solve the OCP of robot manipulators, our studies are generally divided into two main parts. The first part is presented in this chapter and the second one will be dealt with in the next chapter. What we present in this chapter is a new method which solves the optimal control problem of robot manipulators globally. Usually the existing methods result in a local optimal solution for this problem, obtained by fulfilling a series of necessary conditions such as those presented in Ponntryagin's maximum principle or necessary KKT conditions in direct methods, as discussed in the previous chapter.

Our first proposed method solves the unconstrained optimal control problem of robot manipulators by a completely innovative approach without using the calculus of variations (indirect method), direct methods or dynamic programming approach. These methods, as explained in the previous chapter, yield a local optimal solution for the considered problem and actually they satisfy some necessary conditions to find the stationary point of the considered cost functions. Unlike these methods, the proposed method in this chapter results in a global optimal solution for the considered OCP. In addition, this method can be used for both set-point regulating tasks (e.g. pick and place parts or spot welding tasks) and trajectory tracking tasks such as painting or welding tasks. However, the proposed method has a limitation so that it can not support the physical constraints on robot manipulators. Instead, it can be used as an on-line optimal control algorithm which produces the optimal solution without

performing any kind of optimization algorithms which require time to find the optimal solution.

In this chapter, we first present some preliminary definitions and theorems from theory of control and then we shall address our proposed method under a new theorem. We shall prove this theorem in detail and present the necessary discussions regarding its various parts. After that, the procedure of realization this theorem in the case of robot manipulators will be considered. Eventually, the proposed method is applied into some case studies.

4.2 Preliminary Discussion

As shown in the previous chapter, the equations of robot motions are in the form

$$M(q)\ddot{q} + N(q,\dot{q}) + G(q) = u, \tag{4.1}$$

where q, $\dot{q}$, $\ddot{q}$ and u are the n-vectors of joint displacements, velocities, accelerations and torques, respectively, $M(q)$ is the $n \times n$ symmetric positive definite inertia matrix, $N(q,\dot{q})$ is $n \times n$ matrix of centripetal and Coriolis terms and $G(q)$ is the n-vector of gravitation torques obtained as the gradient of the robot's potential energy due to gravity. Generally, robot dynamics (4.1) may be rewritten in the form

$$F(q,\dot{q},\ddot{q}) = u. \tag{4.2}$$

On the other hand, in mathematics usually the following nonlinear differential equations are dealt with

$$\dot{x} = f(x), \tag{4.3}$$

or in the control theory the following dynamic systems are considered

$$\dot{x} = f(x,u). \tag{4.4}$$

Namely, a general differential equation can be supported in the form

$$H(\dot{x},x,u) = 0 \tag{4.5}$$

which can be solved owing to $\dot{x}$.

However, the physical laws lead to the form (4.2) that corresponds the solvability of (4.5) owing to u. Often it is much more better to consider the system dynamics (4.2). Therefore, if we have any trajectory $q = q(t)$ for disposition as well as dynamics (4.2) is an accurate equation of motion, then by substitution of

$$q = q(t) \tag{4.6}$$

into (4.2) we obtain a control law as

$$u(t) = F(q(t), \dot{q}(t), \ddot{q}(t)). \tag{4.7}$$

In order to represent the robot dynamics (4.1) as (4.4), the following classical transformation can be applied

$$x_1 = q,\ x_2 = \dot{q} = \dot{x}_1,\ x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \equiv \begin{pmatrix} q \\ \dot{q} \end{pmatrix}.$$

Then, (4.1) can be rewritten as

$$M(x_1)\dot{x}_2 + N(x)x_2 + G(x_1) = u.$$

Since M is non-singular we have

$$\begin{aligned} \dot{x}_1 &= x_2, \\ \dot{x}_2 &= -M^{-1}N x_2 - M^{-1}G + M^{-1}u, \end{aligned} \tag{4.8}$$

which corresponds $\dot{x} = f(x, u)$.

There are usually two types of cost functionals of the form

$$J_1 = \varphi(q(t_f), \dot{q}(t_f), t_f) + \int_0^{t_f} f_0(q, \dot{q}, \ddot{q})\, dt \tag{4.9}$$

and

$$J_2 = \varphi(q(t_f), \dot{q}(t_f), t_f) + \int_0^{t_f} f_{00}(q, \dot{q}, \ddot{q}, u)\, dt. \tag{4.10}$$

However, substitution (4.1) or (4.2) into (4.10) yields functional (4.9). Therefore, we may solve the problem of minimizing cost functional (4.9) which is the problem of *calculus of variations*. It, of course, may be used whenever that there are no constraints on the state x and the control u. Hence, for obtaining the optimal trajectories it is necessary to minimize J_1 from (4.9) whose special type is quadratic functionals. Similarly as quadratic equations, we can try to alter (modify) the criterion (functional) into more suitable form.

Let us now review some definitions used in the subsequent sections.

Definition 1. *The matrix A of type $n \times n$ is called stable matrix if and only if all its eigenvalues have negative real parts.*

Definition 2. *The real symmetric $n \times n$ matrix A is said to be positive definite if and only if $x^T Ax$ is positive definite, namely,*

$$x^T Ax \geq 0 \quad \textit{and} \quad \left(x^T Ax = 0 \Leftrightarrow x = 0\right).$$

Definition 3. *The matrices A and B are called similar, if and only if there is a non-singular matrix T, such that*

$$A = T^{-1}BT \ \ (\textit{that is iff} \ \ TA = BT). \tag{4.11}$$

Definition 4. *The matrices A and B are called congruent, if and only if there is a matrix P, such that*

$$B = P^T AP. \tag{4.12}$$

In addition, there is a well known theorem which is very often used in the theory of control:

Theorem. Let P be a given stable matrix and let S be any positive definite symmetric matrix. Then there exists a unique matrix X such that

$$XP + P^T X = -S. \tag{4.13}$$

Moreover, the matrix X is symmetric and positive definite. This theorem will play a key role in the different parts of the proposed method. The proof of this theorem will be omitted, since belongs into well known theorems.

4.3 New Proposed Theorem

In this section we present our novel approach which solves the unconstrained optimal control problem of robot manipulators globally. This new method will be presented under the following theorem and its application for robot manipulators is considered in the subsequent sections.

Theorem. Let A, B be positive definite matrices and let they be congruent, such that $B = P^T AP$. Suppose P is stable and $P^T A = AP$. Then the criterion

$$J = \int_0^\infty \left(\dot{\xi}^T A \dot{\xi} + \xi^T B \xi \right) dt \tag{4.14}$$

has the global minimum value

$$J_{min} = \frac{1}{2} \xi^T (0)\, C\, \xi (0)$$

on the set of differentiable curves $\xi(t)$ such that $\lim_{t\to\infty} \xi(t) = 0$. The optimal solution is $\xi(t) = e^{Pt}\xi(0)$. The matrix C is $-2AP$.

Proof.

$$J = \int_0^\infty \left\{ \dot{\xi}^T A \dot{\xi} + \left[\left(P\xi - \dot{\xi} \right) + \dot{\xi} \right]^T A \left[\left(P\xi - \dot{\xi} \right) + \dot{\xi} \right] \right\} dt =$$
$$\int_0^\infty \left\{ \dot{\xi}^T A \dot{\xi} + \left(P\xi - \dot{\xi} \right)^T A \left(P\xi - \dot{\xi} \right) + \left(P\xi - \dot{\xi} \right)^T A \dot{\xi} + \dot{\xi}^T A \left(P\xi - \dot{\xi} \right) + \dot{\xi}^T A \dot{\xi} \right\} dt =$$
$$2 \int_0^\infty \dot{\xi}^T A\, \dot{\xi}\, dt + \int_0^\infty \left(P\xi - \dot{\xi} \right)^T A \left(P\xi - \dot{\xi} \right) dt + \int_0^\infty \left[(P\xi)^T A \dot{\xi} + \dot{\xi}^T A P \xi \right] dt$$
$$- 2 \int_0^\infty \dot{\xi}^T A \dot{\xi}\, dt$$

Thus, we have

$$J = \int_0^\infty \left(P\xi - \dot{\xi} \right)^T A \left(P\xi - \dot{\xi} \right) dt + \int_0^\infty \left(\xi^T P^T A \dot{\xi} + \dot{\xi}^T A P \xi \right) dt.$$

Because $P^T A = AP$, we may rewrite the second integral

$$J = \int_0^\infty \left(P\xi - \dot{\xi}\right)^T A \left(P\xi - \dot{\xi}\right) dt + \int_0^\infty \left(\xi^T AP\dot{\xi} + \dot{\xi}^T AP\xi\right) dt =$$
$$\int_0^\infty \left(P\xi - \dot{\xi}\right)^T A \left(P\xi - \dot{\xi}\right) dt + \int_0^\infty \frac{\mathrm{d}}{\mathrm{d}t}\left(\xi^T AP\xi\right) dt =$$
$$\int_0^\infty \left(P\xi - \dot{\xi}\right)^T A \left(P\xi - \dot{\xi}\right) dt + \left[\xi\left(t\right)^T AP\xi\left(t\right)\right]_0^\infty .$$

But the second term is

$$\left[\xi\left(t\right)^T AP\,\xi\left(t\right)\right]_0^\infty = \lim_{t\to\infty} \xi\left(t\right)^T AP\,\xi\left(t\right) - \xi\left(0\right)^T AP\,\xi\left(0\right) = -\xi\left(0\right)^T AP\,\xi\left(0\right),$$

because it was postulated $\lim_{t\to\infty} \xi\left(t\right) = 0$. Therefore, we obtain

$$J = \int_0^\infty \left(P\xi - \dot{\xi}\right)^T A \left(P\xi - \dot{\xi}\right) dt - \xi\left(0\right)^T AP\xi\left(0\right). \tag{4.15}$$

We can prove $-AP$ is positive definite. In fact, the matrices A and P are given. Let us define $C = -2AP = -AP - P^T A$. Hence $AP = -\frac{1}{2}C$ and $P^T A = -\frac{1}{2}C$. Evidently, C is symmetric, but we don't know if C is positive definite. Obviously,

$$B = P^T AP = P^T\left(-\frac{1}{2}C\right) = -\frac{1}{2}P^T C$$

and

$$B = P^T AP = \left(P^T A\right)P = -\frac{1}{2}CP$$

Thus $CP = P^T C$ and

$$CP + P^T C = -4B \tag{4.16}$$

The matrix P is stable and $4B$ is positive definite. Using the previous theorem together with (4.13) follows that the matrix C exists and only one and that C is symmetric and positive definite in (4.16).

Now, let us consider an equation

$$XP + P^T X = -C$$

and use the previous theorem (with (4.13)) again. We see there is unique matrix X, which is positive definite, but from the definition of C we know

$$AP + P^T A = -C,$$

so it must be $X = A$.

We proved $C = -2AP$ is positive definite and hence (4.15) may be rewritten as

$$J = \int_0^\infty \left(P\xi - \dot{\xi}\right)^T A \left(P\xi - \dot{\xi}\right) dt + \frac{1}{2}\xi\left(0\right)^T C\xi\left(0\right) \tag{4.17}$$

Now it is seen that the minimum of J is

$$J_{min} = \frac{1}{2}\xi\left(0\right)^T C\xi\left(0\right)$$

and is equal zero only for $\xi\left(0\right) = 0$. The necessary and sufficient condition is

$$\dot{\xi} = P\xi \tag{4.18}$$

and this equation can be solved very easy. It must be noted that the solution of (4.18) is the *global minimum* of (4.17).

□

How to solve (4.18):

From $\dot{\xi} - P\xi = 0$ by multiplication we obtain $e^{-Pt}\left(\dot{\xi} - P\xi\right) = 0$ and hence $e^{-Pt}\dot{\xi} - e^{-Pt}P\xi = 0$ and from this we have

$$\frac{d}{dt}\left(e^{-Pt}\xi\right) = 0. \tag{4.19}$$

Integrating (4.19) from 0 to t, we have $e^{-Pt}\xi\left(t\right) = e^{-P0}\xi\left(0\right)$, and so

$$\xi\left(t\right) = e^{Pt}\xi\left(0\right). \tag{4.20}$$

Further, the condition $\lim_{t\to\infty} e^{Pt}\xi\left(0\right) = 0$ is truth, because matrix P is stable and

so

$$\lim_{t\to\infty} \xi(t) = 0$$

Remark. The theorem can be proved by calculus of variation, but it is too difficult to prove the global optimality.

4.3.1 Asymptotic Behavior

It is known that e^{Pt} is a fundamental matrix of the linear system $\dot{\xi} = P\xi$. From theory of ordinary differential equation it is known that every element of e^{Pt} is of the form $\sum_{j=1}^{k} p_i(t)\, e^{\lambda_j t}$, where $p_j(t)$ is a polynomial of degree not more than $(n_j - 1)$, where $\lambda_1, \lambda_2, \ldots \lambda_k$ are the distinct eigenvalues of P and λ_j has multiplicity n_j, such that $n_1 + n_2 + \ldots + n_k = n$. If ρ is chosen such that $\rho > \max_{j=1,2,\ldots,k}(\text{Real}\,\lambda_j)$, then $\left|t^h e^{\lambda_j t}\right| = t^h e^{Re(\lambda_j t)} < e^{\rho t}$ for t large enough, and every term in the sum $\sum p_j(t)\, e^{\lambda_j t}$ is at most $Me^{\rho t}$ $(0 \le t \le \infty)$ for some constant M. Since there are at most n^2 such terms in the matrix e^{Pt}, then

$$\left\|e^{Pt}\right\| \le Ke^{\rho t}$$

for $K = \tilde{M}n^2$, where $\tilde{M}$ is the largest of the n^2 values of M.

We also remark that the constant ρ may be chosen as any number greater than or equal to the largest of $Re\,\lambda_1,\ Re\,\lambda_2, \ldots, Re\,\lambda_n$, whenever every eigenvalue whose real part is equal to this maximum is itself simple. In particular, this is always true if P has no multiple eigenvalues.

The previous discussion was made for arbitrary matrix P. But in our case P is stable, so we can assert the following result:

Lemma.

If all eigenvalues of P have negative real parts, then every solution $e^{Pt}\xi(0)$ of the system

$$\dot{\xi} = P\xi$$

approaches zero as $t \to \infty$. More precisely, there exist constants $\sigma > 0, \bar{K} > 0$ such that

$$\left\|e^{Pt}\xi(0)\right\| \le \bar{K}e^{-\sigma t} \qquad \text{for } 0 \le t \tag{4.21}$$

Proof. We can choose $-\sigma$ $(\sigma > 0)$ as any number larger than the real part of every

eigenvalue ($-\sigma$ plays the role of ρ in the previous discussion). Every solution has the form $e^{Pt}\xi(0)$. Then

$$\left\|e^{Pt}\xi(0)\right\| \leq \left\|e^{Pt}\right\| \|\xi(0)\| \leq K\|\xi(0)\| e^{-\sigma t} = \bar{K}e^{-\sigma t}$$

that is because there is $K > 0$, such that

$$\left\|e^{Pt}\right\| \leq Ke^{-\sigma t} \qquad \text{for } 0 \leq t$$

as was shown earlier. □

4.3.2 Estimation of Precision

If there is given some small $\varepsilon > 0$, then for all $t > -\sigma^{-1} \ln \frac{\varepsilon}{K}$ is

$$\|\xi(t)\| = \left\|e^{Pt}\xi(0)\right\| < \varepsilon. \tag{4.22}$$

That follows from (4.21). Thus, if any control process is from 0 to $t = T$, then for $T > -\sigma^{-1} \ln \frac{\varepsilon}{K}$ this method can be used. If $T \leq -\sigma^{-1} \ln \frac{\varepsilon}{K}$ then it can not be used.

4.3.3 Solved Example

Let be done a functional

$$J = \int_0^\infty \left(\dot{\xi}^2 + \xi^2\right) dt$$

with conditions $\xi(0) = 1, \xi(\infty) = 0$.

a) Let us solve it by classical calculus of variation. If $F = \dot{\xi}^2 + \xi^2$, then the Euler equation is $F_\xi - \frac{\mathrm{d}}{\mathrm{d}t}F\dot{\xi} = 0$, so $\ddot{\xi} - \xi = 0$. The characteristic equation is $\lambda^2 - 1 = 0$, so $\lambda = \pm 1$. The solution is

$$\xi = c_1 e^{-t} + c_2 e^{t}.$$

The boundary conditions are

$$\lim_{n\to\infty} \xi(t) = 0 \Rightarrow c_2 = 0$$
$$\xi(0) = 1 \Rightarrow c_1 = 1;$$

Therefore, we obtain a solution $\xi = e^{-t}$.

b) Let us use the developed method.

$$n = 1,\ A = 1,\ B = 1,\ B = P^T AP \Rightarrow 1 = p^2,$$

for $p = \pm 1$. But P must be stable, so $p = -1$ and hence

$$\xi(t) = e^{Pt}\xi(0) = e^{-t}.$$

4.4 Realization 1

Let the robot is precisely described by the equation

$$M(q)\ddot{q} + N(q, \dot{q}) + G(q) = u, \tag{4.23}$$

and let $q_d(t)$ is any desired trajectory of robot motion.

Let us mark $\Delta q(t) = q(t) - q_d(t)$ the difference between the actual and desired trajectories. Then $\xi \equiv \Delta q,\ \xi(0) = \Delta q(0) = q(0) - q_d(0)$ and from our theory follows

$$\Delta q(t) = e^{Pt}\Delta q(0),$$

because $\Delta\dot{q} = P\Delta q$ (as was $\dot{\xi} = P\xi$), we obtain $\Delta\ddot{q} = P\Delta\dot{q} = P^2\Delta q$ and hence

$$q^*(t) = q_d(t) + e^{Pt}\Delta q(0),\ \dot{q}^*(t) = \dot{q}_d(t) + e^{Pt}\Delta q(0),\ \ddot{q}^*(t) = \ddot{q}_d(t) + e^{Pt}\Delta q(0), \tag{4.24}$$

if we substitute q^*, $\dot{q}^*$, $\ddot{q}^*$ from (4.24) into (4.23), we obtain the optimal control

$$u^*(t) = M(q^*(t))\ddot{q}^*(t) + N(q^*(t), \dot{q}^*(t)) + G(q^*)$$

as the function of t. This is rightful for precisely description equation (4.23), not for incorrect.

Remark 1. Especially, if $q_d(t) = q_d(T) = Cons.$, then $\dot{q}_d = 0,\ \ddot{q}_d = 0$ and we obtain the optimal control to transfer the robot from initial configuration $q(0)$ into final one $q_d(T)$.

Remark 2. In the realization above, we can only set the initial value of joint disposition variable and no any capability exists to set the initial value of the joint velocity and acceleration. In the next discussion we can present a new realization of our theorem to take into account both initial joint disposition and velocity variables but no acceleration.

4.5 Realization 2

For robot equation

$$M\ddot{q} + N\dot{q} + G = u \tag{4.25}$$

let us define a state

$$x = (q_1,\ \dot{q}_1,\ q_2,\ \dot{q}_2, \ldots,\ q_n,\ \dot{q}_n)^T. \tag{4.26}$$

Now define

$$e = x_d - x \tag{4.27}$$

where e is a $m \times 1$ vector with $m = 2n$. Let us study the problem to find a matrix T such that

$$\xi = -T\,e \tag{4.28}$$

where $\xi \in \mathbb{R}^{n\times 1}$ and $T \in \mathbb{R}^{n\times m}$. Our aim is to find a suitable form of matrix T.

From $\dot{\xi} = P\xi$ given in (4.18) we obtain $T\dot{e} = PTe$ and so for $e(0) = e_0$ we have $\xi(0) = -Te_0$. Thus, we achieve the following matrix equation

$$T\dot{e} = PTe \tag{4.29}$$

which will play an important role. The state equation of (4.25) is

$$\dot{x} = f(x) + g(x)\,u \tag{4.30}$$

and so

$$\dot{e} = \dot{x}_d - \dot{x} = \dot{x}_d - f(x) - g(x)\,u,$$

where $x = x_d - e$. The vector equation (4.30) may be written as

$$\begin{aligned} \dot{x}_{i-1} &= x_i \\ \dot{x}_i &= f_i(x) + \sum_{j=1}^{n} g_{ij}(x)\,u_j \end{aligned}$$

for $i = 2, 4, 6, \ldots, m$, or we can write it as

$$\begin{aligned} \dot{x}_{2k-1} &= x_{2k} \\ \dot{x}_{2k} &= f_{2k}(x) + \sum_{j=1}^{n} g_{2k,j}(x)\,u_j \end{aligned} \tag{4.31}$$

for $k = 1, 2, 3, \ldots, n$.

4.5.1 Option of Matrix T

The equation (4.29) can be written as $(m > n)$

$$\begin{bmatrix} \sqsubset T_{11} & T_{12} \sqsupset & T_{13} & T_{14} & T_{15} & \cdots & & T_{1m} \\ T_{21} & T_{22} & \sqsubset T_{23} & T_{24} \sqsupset & T_{25} & \cdots & & T_{2m} \\ T_{31} & T_{32} & T_{33} & T_{34} & \sqsubset T_{35} & \cdots & & T_{3m} \\ \vdots & & & & & & & \\ T_{n1} & T_{n2} & & \cdots & & & \sqsubset T_{n,m-1} & T_{n,m} \sqsupset \end{bmatrix} \begin{bmatrix} \dot{e}_1 \\ \dot{e}_2 \\ \dot{e}_3 \\ \vdots \\ \dot{e}_m \end{bmatrix} =$$

$$\begin{bmatrix} P_{11} & \cdots & P_{1n} \\ & & \\ & \vdots & \\ & & \\ P_{n1} & \cdots & P_{nn} \end{bmatrix} \begin{bmatrix} T_{11} & T_{12} & \cdots & T_{1m} \\ T_{21} & T_{22} & \cdots & T_{2m} \\ T_{31} & T_{32} & \cdots & T_{3m} \\ \vdots & & & \\ T_{n1} & T_{n2} & \cdots & T_{nm} \end{bmatrix} \begin{bmatrix} e_1 \\ e_2 \\ e_3 \\ \vdots \\ e_m \end{bmatrix} \tag{4.32}$$

The matrix T is $n \times m$ and we choose its quasi-diagonal as non-zero elements, others will be zero. The matrix P let be diagonal. Thus the matrix equation (4.32) can be

rewritten in

$$\begin{aligned} T_{11}\dot{e}_1 + T_{12}\dot{e}_2 &= P_{11}T_{11}e_1 + P_{11}T_{12}e_2 \\ T_{23}\dot{e}_3 + T_{24}\dot{e}_4 &= P_{22}T_{23}e_3 + P_{22}T_{24}e_4 \\ &\vdots \\ T_{n,m-1}\dot{e}_{m-1} + T_{nm}\dot{e}_m &= P_{nn}T_{n,m-1}e_{m-1} + P_{nn}T_{nm}e_m \end{aligned} \tag{4.33}$$

From $\dot{x}_{2k-1} = x_{2k}$ in (4.31) we obtain equations

$$\dot{e}_1 = e_2,\ \dot{e}_3 = e_4, \cdots \dot{e}_{m-1} = e_m \tag{4.34}$$

and so we can rewrite (4.33) in the form

$$\begin{aligned} T_{12}\dot{e}_2 &= P_{11}T_{11}e_1 + (P_{11}T_{12} - T_{11})\, e_2 \\ T_{24}\dot{e}_4 &= P_{22}T_{23}e_3 + (P_{22}T_{24} - T_{23})\, e_4 \\ &\vdots \\ T_{nm}\dot{e}_m &= P_{nn}T_{n,m-1}e_{m-1} + (P_{nn}T_{nm} - T_{n,m-1})\, e_n \end{aligned} \tag{4.35}$$

The equation (4.34) and (4.35) we may write

$$\begin{aligned} \dot{e}_{2k-1} &= e_{2k} \\ \dot{e}_{2k} &= \frac{P_{kk}T_{k,2k-1}}{T_{k,2k}} e_{2k-1} + \left(P_{kk} - \frac{T_{k,2k-1}}{T_{k,2k}}\right) e_{2k} \end{aligned} \tag{4.36}$$

for $k = 1, 2, \cdots, n$.
Because here we have a fraction $\frac{T_{k,2k-1}}{T_{k,2k}}$, it will be better to choose

$$T_{k,2k} = 1 \tag{4.37}$$

and then for

$$M_k = \begin{bmatrix} 0 & 1 \\ P_{kk}T_{k,2k-1} & P_{kk} - T_{k,2k-1} \end{bmatrix} \tag{4.38}$$

we may rewrite (4.36) as

$$\begin{pmatrix} \dot{e}_{2k-1} \\ \dot{e}_{2k} \end{pmatrix} = M_k \begin{pmatrix} e_{2k-1} \\ e_{2k} \end{pmatrix} \tag{4.39}$$

How it is with stability?

Let us examine the eigenvalues of M_k as follows

$$\det\left(M_k - \lambda I\right) = \begin{vmatrix} -\lambda & 1 \\ P_{kk}T_{k,2k-1} & P_{kk} - T_{k,2k-1} - \lambda \end{vmatrix} = \lambda^2 - \lambda\left(P_{kk} - T_{k,2k-1}\right) - P_{kk}T_{k,2k-1}$$

The characteristic equation is $\det\left(M_k - \lambda I\right) = 0$, so

$$\left(\lambda - P_{kk}\right)\left(\lambda - T_{k,2k-1}\right) = 0.$$

Here we can write

$$\begin{aligned} \lambda_1 &= P_{kk} \\ \lambda_2 &= -T_{k,2k-1} \end{aligned}$$

$P_{kk}, T_{k,2k-1}$ are real numbers, thus from theory of stability $\lambda_1 < 0, \lambda_2 < 0$ and hence

$$\begin{aligned} P_{kk} &< 0 \\ T_{k,2k-1} &> 0 \end{aligned} \tag{4.40}$$

Therefore, matrix P must be stable which it is our assumption of the previous theorem. The matrix T then has the form

$$T = \begin{bmatrix} T_{11} & 1 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & T_{23} & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & T_{35} & 1 & \cdots & 0 \\ \vdots & & & \vdots & & & & \vdots \\ 0 & & & & & & T_{n,m-1} & 1 \end{bmatrix} \tag{4.41}$$

The numbers $T_{k,2k-1}$ can be chosen arbitrary, but positive.

4.5.2 Solution of e_i

Let $a = P_{kk}T_{k,2k-1}$, $b = P_{kk} - T_{k,2k-1}$. Then characteristic equation of (4.36) is

$$\lambda^2 - b\lambda - a = 0 \Rightarrow \lambda_{1,2} = \frac{b \pm \sqrt{b^2 + 4a}}{2},$$

in which $b^2 + 4a = (P_{kk} - T_{k,2k-1})^2 + 4P_{kk}T_{k,2k-1} = (P_{kk} + T_{k,2k-1})^2 \geq 0$, so

$$\lambda_{1,2} = \frac{1}{2}\left(P_{kk} - T_{k,2k-1} \pm |P_{kk} + T_{k,2k-1}|\right) \Rightarrow \begin{cases} \lambda_1 = P_{kk} \\ \lambda_2 = -T_{k,2k-1} \end{cases}$$

Thus

$$\begin{aligned} e_{2k-1} &= c_{k1}e^{P_{kk}t} + c_{k2}e^{-T_{k,2k-1}t} \\ e_{2k} &= \dot{e}_{2k-1} = P_{kk}c_{k1}e^{P_{kk}t} - T_{k,2k-1}c_{k2}e^{-T_{k,2k-1}t} \end{aligned} \tag{4.42}$$

4.5.3 Global Optimal Feedback Control

From the equation

$$\dot{e} = \dot{x}_d - f(x) - g(x)\,u, \tag{4.43}$$

we have

$$\boxed{g(x)\ u = \dot{x}_d - \dot{e} - f(x), \quad x = x_d - e.} \tag{4.44}$$

Now we can exploit two following ways to derive the optimal control u:

A. Multiplying (4.43) by T

$$T\,g(x)\ u = T\left(\dot{x}_d - f(x)\right) - T\dot{e}, \tag{4.45}$$

and according to (4.29) we will have

$$T\,g(x)\ u = T\left(\dot{x}_d - f(x)\right) - PTe, \tag{4.46}$$

Let us compute the multiplication $T\,g(x)$ in the left hand side of the above

equation:

$$T \cdot g = \begin{bmatrix} T_{11} & 1 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & T_{23} & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & T_{35} & 1 & \cdots & 0 \\ \vdots & & & \vdots & & & & \vdots \\ 0 & & & & & & T_{n,m-1} & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & \cdots & 0 \\ g_{21} & g_{22} & \cdots & g_{2n} \\ 0 & 0 & \cdots & 0 \\ g_{41} & g_{42} & \cdots & g_{4n} \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 \\ g_{m1} & g_{m2} & \cdots & g_{mn} \end{bmatrix} =$$

$$\begin{bmatrix} g_{21} & g_{22} & \cdots & g_{2n} \\ g_{41} & g_{42} & \cdots & g_{4n} \\ \vdots & \vdots & \vdots & \vdots \\ g_{m1} & g_{m2} & \cdots & g_{mn}. \end{bmatrix} \tag{4.47}$$

Let us denote

$$\tilde{g} = \begin{bmatrix} g_{21} & g_{22} & \cdots & g_{2n} \\ g_{41} & g_{42} & \cdots & g_{4n} \\ \vdots & \vdots & \vdots & \vdots \\ g_{m1} & g_{m2} & \cdots & g_{mn}. \end{bmatrix} \tag{4.48}$$

We see $\tilde{g}$ is a square matrix of type $n \times n$ and suppose $det\, \tilde{g} \neq 0$; hence it is a regular matrix and we can solve (4.46)

$$u(t) = (\tilde{g})^{-1} T (\dot{x}_d - f(x)) - (\tilde{g})^{-1} P T e \tag{4.49}$$

Therefore, in this way we obtained an optimal control of our problem by (4.49).

B. Remember, we can use the equation of robot motion (4.25) for establishing of the control $u(t)$. In fact, using (4.36) and (4.27) we obtained $x = x(t)$ and from ((4.26)) we can get $q(t)$ and $\dot{q}(t)$ and by derivative of $\dot{q}$ we have $\ddot{q}(t)$. If we substitute these results into ((4.25)), we obtain the control vector $u = u(t)$.

Remark 2. The method A is more general, because there is used the formula (4.30). So we employ only method A to obtain the optimal control $u(t)$.

Let us now consider two cases:

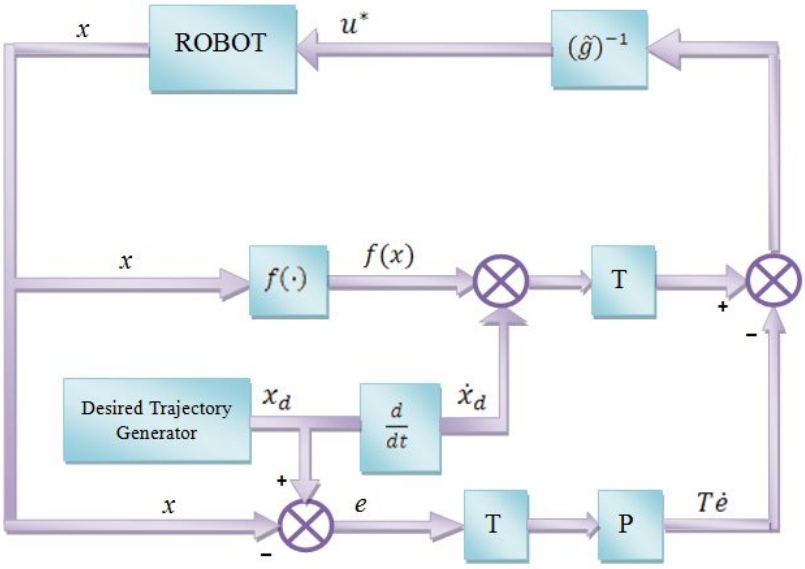


Figure 4.1: **Optimal feedback control schematic**

a. Let be given $e = e(t)$, for example by solving (4.39). Then by (4.27) we are able to find $x(t)$ and by a substitution into (4.49) we obtain the optimal control.

b. Contrarily, let be given an optimal control $u = u^*(t)$. Then by (4.30) we can compute $x(t)$ and then by substitution of these results into (4.43) we obtain a vector $\dot{e}(t)$, from which follows $e(t)$. In this manner we opened a way into the **optimal feedback control** for the precise model of robot (4.30). The feedback control is schematically depicted on the Figure 4.1.

   Really, we can mutually interchange the block ROBOT onto the relation (4.30), because all positions and motions are precisely described by (4.30) about our presumption.

   This schematic can be adjusted onto an adaptive control, if the equation (4.30) is not a precise model of robot motion.

4.6 Case Studies for Global Optimal Controller

In this section we apply the proposed method for some well known robot manipulators

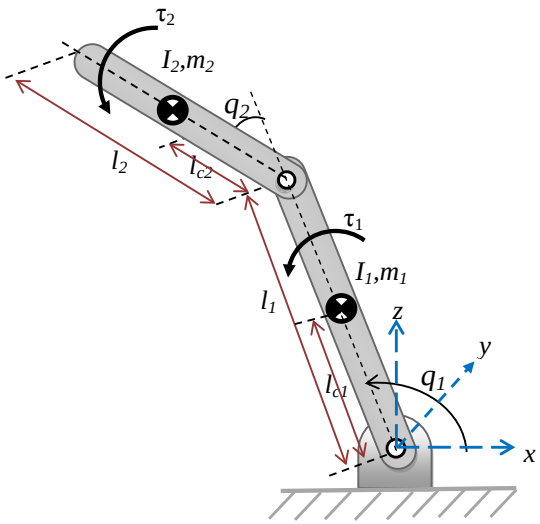


Figure 4.2: **Vertical two links robot manipulator**

4.6.1 Vertical Two Links Robot Manipulator

Consider a physical model of a vertical two-link robot, with each joint equipped with a motor for providing input torque as shown in Figure 4.2 which the following notations are used in this figure.

- q_i the joint angle of joint i
- m_i the mass of link i
- l_i the length of link i
- l_{ci} the location of mass center of link i with respect to coordinate system in joint i.
- I_i the moment of inertia of link i about the axis passed through the mass center and paralleled to the axis y.

The values given in Table 4.1 are used for these variable.

Let us first obtain the dynamic model of this robot by Lagrange-Euler method. Let q_1, q_2 as the general coordinates of this dynamic system. Thus, the kinetic and potential terms of two links are obtained as:

Table 4.1: Some typical function spaces

m_1	m_2	l_1	l_2	l_{c1}	l_{c2}	I_1	I_2
2 kg	1 kg	0.6 m	0.4 m	0.3 m	0.2 m	o.5 $kg \cdot m^2/rad$	o.5 $kg \cdot m^2/rad$

- Joint 1

$$\begin{aligned} &\text{Kinetic energy: } K_1 = \frac{1}{2} m_1 \, l_{c1}^2 \, \dot{q}_1^2 + \frac{1}{2} I_1 \dot{q}_1^2 \\ &\text{Potential energy: } P_1 = m_1 \, g \, l_{c1} \, \sin q_1 \end{aligned} \tag{4.50}$$

- Joint 2

  Since the mass center of link 2, $c_2 = [c_{2x} \;\; c_{2z}]$ is given by

$$\begin{aligned} c_{2x} &= l_1 \cos q_1 + l_{c2} \cos \left(q_1 + q_2 \right), \\ c_{2z} &= l_1 \sin q_1 + l_{c2} \sin \left(q_1 + q_2 \right). \end{aligned} \tag{4.51}$$

  Thus, we have

$$\dot{c}_2^T \, \dot{c}_2 = l_1^2 \dot{q}_1^2 + l_{c2}^2 \left(\dot{q}_1 + \dot{q}_2 \right)^2 + 2 l_1 l_{c2} \cos q_2 \left(\dot{q}_1^2 + \dot{q}_1 \, \dot{q}_2 \right) \tag{4.52}$$

  Then,

$$\begin{aligned} &\text{Kinetic energy: } K_2 = \frac{1}{2} m_2 \, \dot{c}_2^T \, \dot{c}_2 + \frac{1}{2} I_2 \left(\dot{q}_1 + \dot{q}_2 \right)^2 \\ &\text{Potential energy: } P_2 = m_2 \, g \, \left(l_1 \sin q_1 + l_{c2} \sin \left(q_1 + q_2 \right) \right) \end{aligned} \tag{4.53}$$

Let us now define Lagrangian as $L = K_1 + K_2 - P_1 - P_2$. Then, according to Lagrange-Euler formulation we have

$$\frac{\mathrm{d}}{\mathrm{d}t} \left(\frac{\partial L}{\partial q_1} \right) - \frac{\partial L}{\partial q_1} = \tau_1 \tag{4.54}$$

thus,

$$\begin{aligned}\tau_1 = &\left[m_1\, l_{c1}^2 + I_1 + m_2\left(l_1^2 + l_{c2}^2 + 2l_1\, l_{c2}\cos q_2\right) + I_2\right]\ddot{q}_1 + \\ &\left[m_2\, l_1 l_{c2}\cos q_2 + m_2\, l_{c2}^2 + I_2\right]\ddot{q}_2 - m_2\, l_1\, l_{c2}\sin q_2\left(2\dot{q}_1\dot{q}_2 + \dot{q}_2^2\right) + \\ &m_1\, g\, l_{c1}\cos q_1 + m_2\, g\left(l_1\cos q_1 + l_{c2}\cos\left(q_1 + q_2\right)\right),\end{aligned} \tag{4.55}$$

and for joint 2 we have

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial q_2}\right) - \frac{\partial L}{\partial q_2} = \tau_2, \tag{4.56}$$

hence,

$$\begin{aligned}\tau_2 = &\left[m_2\, l_1\, l_{c2}\cos q_2 + m_2\, l_{c2}^2 + I_2\right]\ddot{q}_1 + \left[m_2\, l_{c2}^2 + I_2\right]\ddot{q}_2 + m_2 l_1 l_{c2}\sin q_2\dot{q}_1^2 + \\ &m_2\, g\, l_{c2}\cos\left(q_1 + q_2\right).\end{aligned} \tag{4.57}$$

The equations (4.55) and (4.57) can be rewritten as matrix form $M\left(q\right)\ddot{q} + N\left(q,\dot{q}\right) = \tau$:

$$\begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22}\end{bmatrix}\begin{bmatrix}\ddot{q}_1 \\ \ddot{q}_2\end{bmatrix} + \begin{bmatrix} -v\dot{q}_2 & -v\dot{q}_1 - v\dot{q}_2 \\ v\dot{q}_1 & 0\end{bmatrix}\begin{bmatrix}\dot{q}_1 \\ \dot{q}_2\end{bmatrix} + \begin{bmatrix} g_1 \\ g_2\end{bmatrix} = \begin{bmatrix}\tau_1 \\ \tau_2\end{bmatrix} \tag{4.58}$$

where

$$\begin{aligned} M_{11} &= m_1 l_{c1}^2 + I_1 + m_2\left(l_1^2 + l_{c2}^2 + 2l_1 l_{c2}\cos q_2\right) + I_2 \\ M_{22} &= m_2 l_{c2}^2 + I_2 \\ M_{12} &= M_{21} = m_2 l_1 l_{c2}\cos q_2 + m_2 l_{c2}^2 + I_2 \\ v &= m_2 l_1 l_{c2}\sin q_2 \\ g_1 &= m_1\, lc_1\, g\,\cos\left(q_2\right) + m_2\, g\,\left(lc_2\,\cos\left(q_1 + q_2\right) + l_1\,\cos\left(q_1\right)\right) \\ g_2 &= m_2\, lc_2\, g\,\cos\left(q_1 + q_2\right)\end{aligned} \tag{4.59}$$

and $N\left(q,\dot{q}\right) = \begin{bmatrix} N_1 \\ N_2\end{bmatrix} = \begin{bmatrix} -v\dot{q}_2 & -v\dot{q}_1 - v\dot{q}_2 \\ v\dot{q}_1 & 0\end{bmatrix}\begin{bmatrix}\dot{q}_1 \\ \dot{q}_2\end{bmatrix} + \begin{bmatrix} g_1 \\ g_2\end{bmatrix}$.

The state space representation of this robot is obtained by considering the following states

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} q_1 \\ \dot{q}_1 \\ q_2 \\ \dot{q}_2 \end{bmatrix}, \qquad \dot{x} = f(x) + g(x)\tau. \tag{4.60}$$

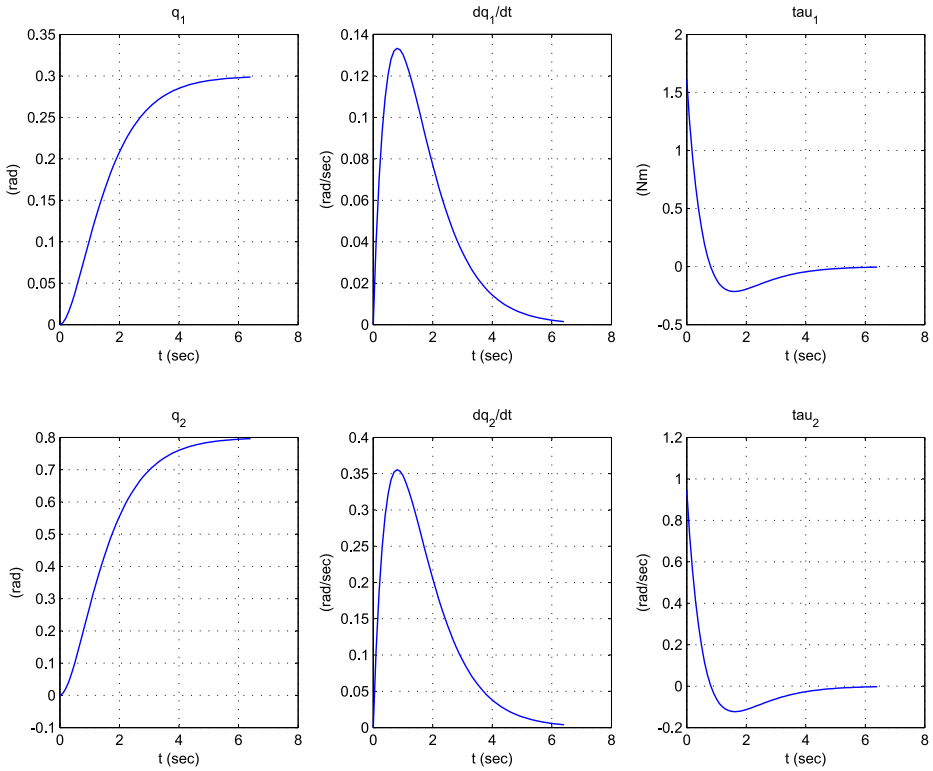


Figure 4.3: **Optimal profiles of vertical two links robot manipulator**

Let inverse of M is denoted by M_I whose ith row is represented as M_I^i, then

$$f(x) = \begin{bmatrix} x_2 \\ -M_I^1 N \\ x_4 \\ -M_I^2 N \end{bmatrix}, \quad g(x) = \begin{bmatrix} 0 & 0 \\ M_I^1 & \\ 0 & 0 \\ M_I^2 & \end{bmatrix} \tag{4.61}$$

The boundary conditions considered here are

$$\begin{aligned} &x_1(0) = 0, \quad x_2(0) = 0 \\ &x_3(0) = 0, \quad x_4(0) = 0 \\ &x_1(5) = 0.3\ rad, \quad x_2(5) = 0.8\ rad \\ &x_3(5) = 0, \quad x_4(5) = 0 \end{aligned} \tag{4.62}$$

According to equations (4.40) and (4.41), we consider matrices P and T as follows

$$P = \begin{bmatrix} -1 & 0 \\ 0 & -2 \end{bmatrix}, \quad T = \begin{bmatrix} 2 & 1 & 0 & 0 \\ 0 & 0 & 3 & 1 \end{bmatrix} \tag{4.63}$$

Then using equation (4.42) the following error terms are obtained

$$\begin{aligned} e_1 &= c_{11}e^{-t} + c_{12}e^{-2t} \\ e_2 &= -c_{11} - 2c_{12}e^{-2t} \\ e_3 &= c_{21}e^{-2t} + c_{22}e^{-3t} \\ e_4 &= -2c_{21}e^{-2t} - 3c_{22}e^{-3t} \end{aligned} \tag{4.64}$$

where constants c_{ki} are obtained according to initial error

$$e(0) = x_d(0) - x(0) = \begin{bmatrix} 0.3 & 0 & 0.8 & 0 \end{bmatrix}^T \tag{4.65}$$

which yields $c_{11} = 0.6$, $c_{12} = -0.3$, $c_{21} = 2.4$, $c_{22} = -1.6$. Then optimal controls can be obtained by steps given in subsection 4.5.3. Eventually, Figure 4.3 shows the optimal profiles contain joint dispositions, velocities and torques of this robot. In addition, the minimum value of (4.14) in the case of this robot is obtained 13.32 by considering The following matrix A

$$A = \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}$$

which is a positive definite matrix.

4.6.2 KUKA Robot

In chapter 2, we obtained the dynamic model of a KUKA industrial robot through an experimental identification and the resulted dynamic equations have been presented in Appendix A. In this subsection, we attempt to apply the above proposed method in this KUKA robot for two purposes: 1- Set-point regulating case 2- Trajectory tracking Case.

- Set-point regulating case

  In this case which is actually the point to point motion, the objective is to transfer the robot from an initial configuration into a final one so that the cost functional (4.14) is minimized. Let us consider the following boundary conditions

$$q_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \; q_f = \begin{bmatrix} 0.3 \\ 0.8 \\ 0.5 \end{bmatrix} \; (rad), \; \dot{q}_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \; \dot{q}_f = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{4.66}$$

  as well as the following matrices which are required in the proposed method

$$A = \begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix}, \; T = \begin{bmatrix} 2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 4 & 1 \end{bmatrix}$$

  where A is a positive definite matrix and matrix T is chosen according to (4.41). Then the optimal trajectories are obtained as shown in Figure 4.4. The minimum value of the cost functional (4.14) in this case study is 22.32.

- Trajectory tracking Case

  In this item the objective is to command the KUKA robot to move along the following reference (desired) trajectories

$$\begin{aligned} q_{d1} &= 0.3 + 0.1 \sin(\pi\, t) \\ q_{d2} &= 0.8 + 0.2 \sin(2\pi\, t) \\ q_{d3} &= 0.5 + 0.3 \sin(3\pi\, t) \end{aligned} \tag{4.67}$$

  so that the cost functional (4.14) is minimized.

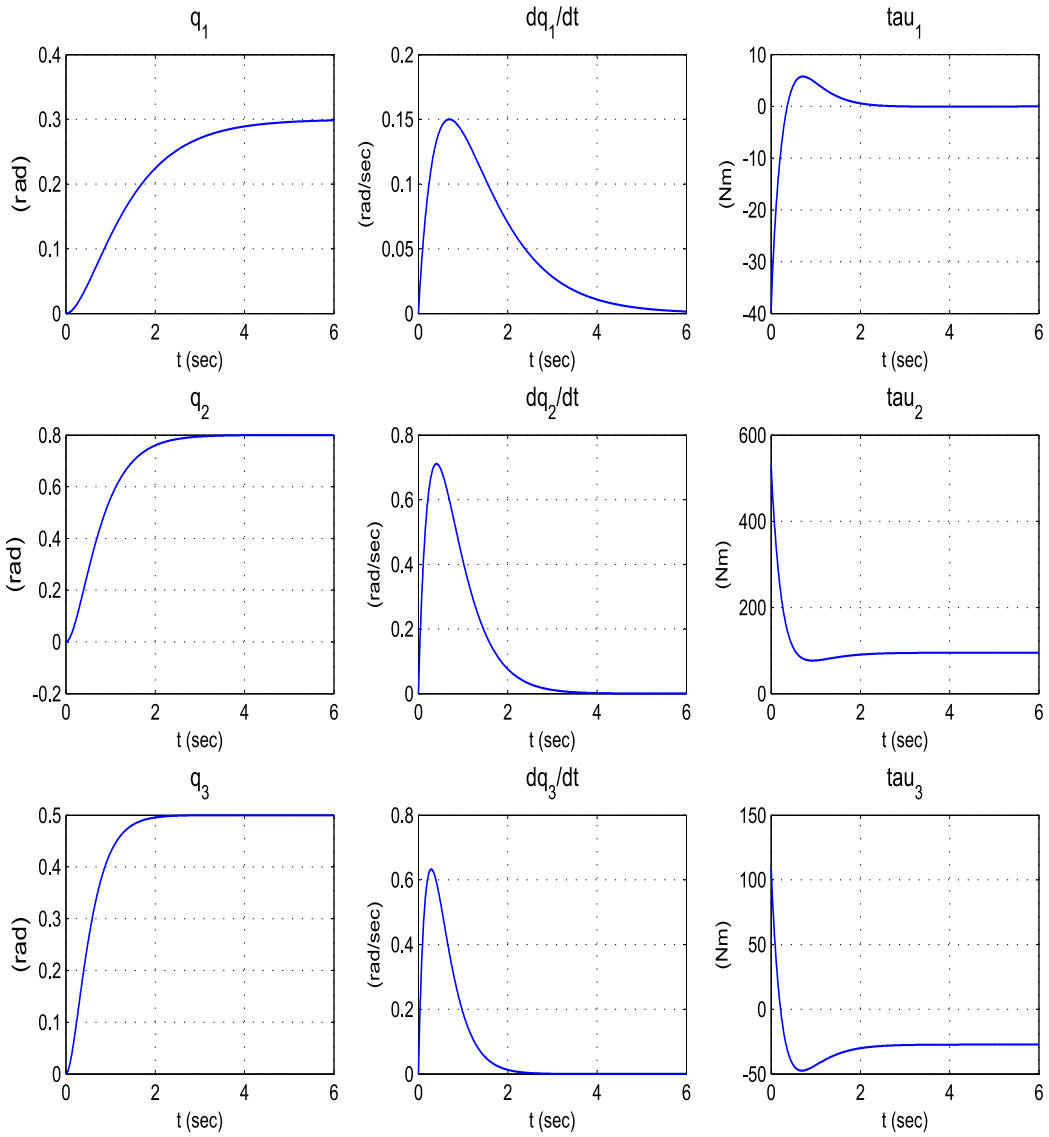


Figure 4.4: **Optimal trajectories of KUKA robot obtained by the proposed method in set-point regulating case**

Utilizing the matrices used in the previous item, the optimal trajectories are obtained as shown in Figure 4.5. Note that the robot is in its home position in $t = 0$. Of course, it can be in any other initial configuration. In this case the minimum value of the cost functional (4.14) is 50.138.

4.7 Adaptive Optimal Control of Robot Manipulator

In the previous sections of this chapter we developed an unconstrained global optimal controller for robot manipulators which actually is a model-based controller. However, very often, there are some uncertainties in the dynamic model of the robot manipulators. One possibility in controlling such systems whose exact models are not available is *adaptive control* technique [7]. In recent years, many adaptive control schemes have been developed for robot manipulators, such as [59; 97], which usually are in the framework of the adaptive control methods given in [86].

In this section we attempt to extend our proposed controller in the more general case in which an exact model of the considered robot does not exist. In fact, our objective is to design an *adaptive optimal controller* (AOC) whose central core is the optimal trajectory generator (OTG) proposed in the previous sections of this chapter.

As explained in the chapter 2, the dynamic model of an n-axes robot manipulator can be expressed as the regression form

$$Y\left(q, \dot{q}, \ddot{q}\right)\theta = \tau \tag{4.68}$$

where Y is an $n \times m$ matrix whose elements are nonlinear functions of $q, \dot{q}, \ddot{q}$ and $\theta \in \mathbb{R}^{m\times 1}$ is a vector whose entries are identifiable parameters of the considered robot. The elements of vector θ are functions of dynamic and friction parameters of the robot whose values usually are not provided by robot manufacturers and researchers have to measure these values themselves by robot identification experiments (as explained in section 2.3, in detail). In the sequel, we use the regression model in order to design the proposed AOC for robot manipulators.

Let us now consider the structure of the proposed AOC as illustrated in Figure 4.6

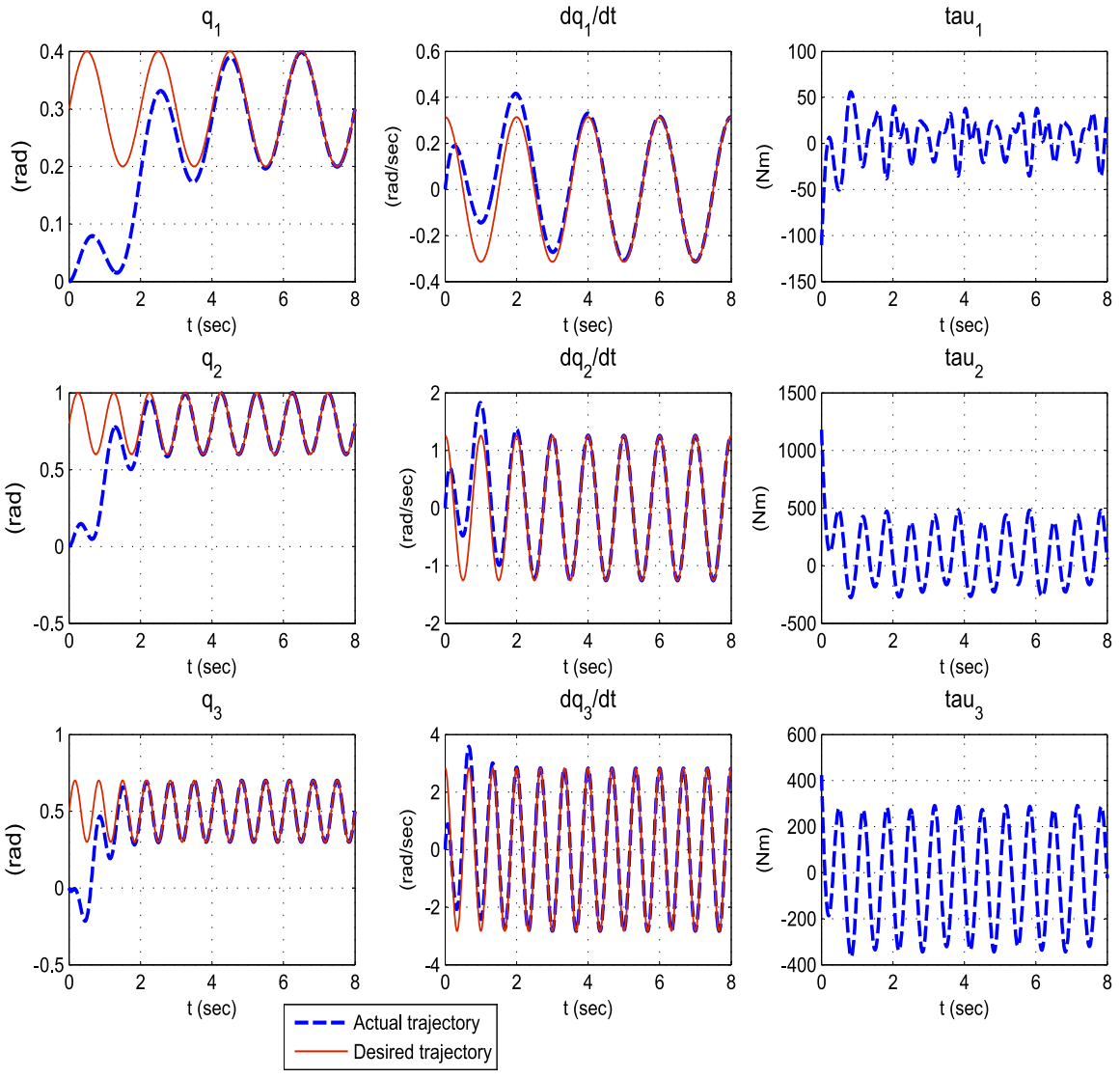


Figure 4.5: **Optimal trajectories of KUKA robot obtained by the proposed method in trajectory tracking case**

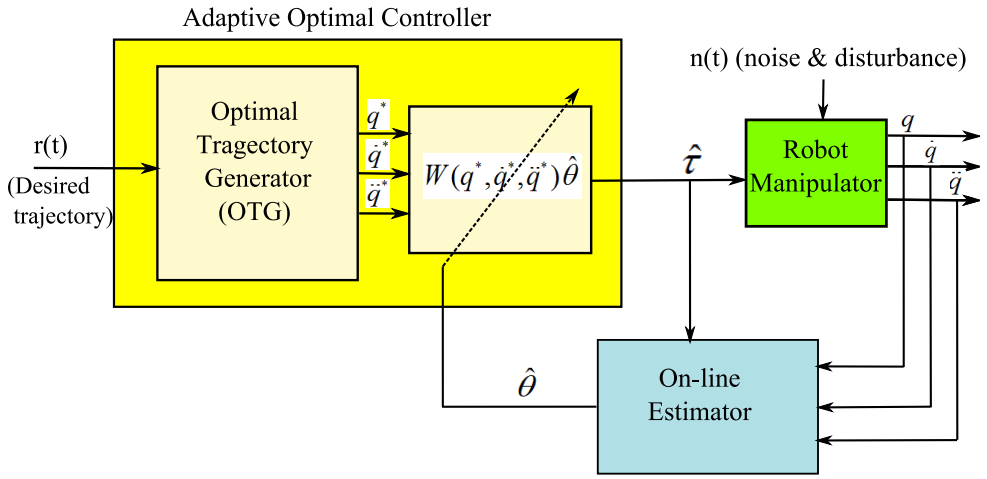


Figure 4.6: **General structure of adaptive optimal controller of robot manipulators**

which actually is an adaptive self-tuning controller. In this structure the values of the elements of vector θ are estimated in an on-line manner and then the estimated control input of the robot ($\hat{\tau}$) is calculated on the basis of these estimated parameters in each time instance. In fact, the on-lone estimator existed in this structure estimates the unknown parameters of the system based on the measurements of input and output signals of the system. Before explaining the internal structure of the on-line estimator, we require to introduce some variables in the proposed AOC:

- predicted torque defined as $\hat{\tau}(t) = Y(q(t), \dot{q}(t), \ddot{q}(t))\hat{\theta}(t)$
- exact value of unknown parameters, denoted θ
- parameter estimation error defined as $\tilde{\theta} = \hat{\theta} - \theta$
- prediction error defined as $\mathbf{e}(t) = Y\hat{\theta}(t) - Y\theta(t)$

The basic idea in designing the on-line estimator is that $\hat{\theta}$ should be updated so that $\mathbf{e}$ (prediction error) is reduced in each time instance. In doing so, we can use the procedure used in the standard least-square (LS) scenario. In fact, the estimation of the parameters can be obtained by minimizing the following total prediction error with respect to $\hat{\theta}$

$$J = \int_0^t \left\| \tau(s) - Y(q(s), \dot{q}(s), \ddot{q}(s))\hat{\theta}(s) \right\|^2 ds. \tag{4.69}$$

According to LS method, the solution of the above minimization problem is obtained with satisfying the following relation by $\hat{\theta}$

$$\hat{\theta}(t) = \left[\int_0^t Y^T Y ds\right]^{-1} \int_0^t Y^T \tau(s)\, ds. \tag{4.70}$$

However, in computation point of view, the equation (4.70) is not efficient and it can be converted into a more appropriate form with defining the following square matrix

$$\Phi(t) = \left[\int_0^t Y^T Y ds\right]^{-1}. \tag{4.71}$$

whose derivative is

$$\frac{\mathrm{d}\left[\Phi^{-1}(t)\right]}{\mathrm{d}t} = Y^T\left(q(t), \dot{q}(t), \ddot{q}(t)\right) Y\left(q(t), \dot{q}(t), \ddot{q}(t)\right). \tag{4.72}$$

Let us now consider the following identity

$$\frac{\mathrm{d}}{\mathrm{d}t}\left[\Phi\Phi^{-1}\right] = \dot{\Phi}\Phi^{-1} + \Phi\frac{\mathrm{d}}{\mathrm{d}t}\left[\Phi^{-1}\right] = 0, \tag{4.73}$$

and so

$$\boxed{\dot{\Phi} = -\Phi Y^T Y \Phi}. \tag{4.74}$$

Eventually, the unknown parameters can be updated by the following causal equation which is obtained by differentiating (4.70) and using (4.71)

$$\boxed{\dot{\hat{\theta}} = -\Phi(t)\, W^T \mathbf{e}}. \tag{4.75}$$

For investigating the convergence of the above on-line estimator, it is easy to obtain the following equation using equations (4.72) to (4.75)

$$\frac{\mathrm{d}}{\mathrm{d}t}\left[\Phi^{-1}(t)\,\tilde{\theta}(t)\right] = 0, \tag{4.76}$$

and hence

$$\tilde{\theta}(t) = \Phi(t)\,\Phi^{-1}(0)\,\tilde{\theta}(0). \tag{4.77}$$

Therefore, if smallest eigenvalue of the integral $\int_0^t Y^T Y\, ds$ (according to (4.71)) goes to infinity as $t \to \infty$, then in (4.77) $\Phi \to 0$ and so $\tilde{\theta} \to 0$ and each trajectory that

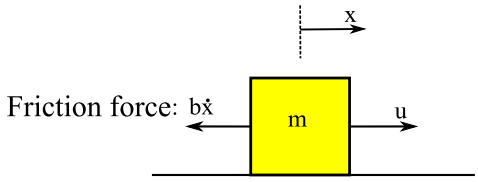


Figure 4.7: **A simple linear system**

satisfies this condition is called *persistent excitation trajectory*. It is worth to be noted also that, according to (4.77), if the initial value $\Phi(0)$ is large enough, then it results in smaller parameter error.

In the next section we shall apply the proposed AOC in the different case studies and more discussions shall be given for each cases.

4.8 Case Studies for Adaptive Optimal Controller

4.8.1 A Simple Linear System

In order to investigate the features of the proposed AOC, in this subsection we attempt to design an adaptive optimal controller for a simple linear system shown in Figure 4.7 which it is assumed that the value of the mass and friction parameter of the system are parameters which are estimated by the control system while the exact value of these parameters are $m = 10\,kg$ and $b = 0.2\,sec^{-1}$. The dynamic model of this system can be easily obtained by Newton's laws as follows:

$$m\ddot{x} + b\dot{x} = u$$

where $x(t)$ is the position of the mass m in time t and u is the exerted force as the input control. This dynamic model can be rewritten as the vector form

$$\underbrace{\begin{bmatrix}\ddot{x} & \dot{x}\end{bmatrix}}_{Y}\underbrace{\begin{bmatrix}m\\ b\end{bmatrix}}_{\theta} = u \;\Rightarrow Y\theta = u \tag{4.78}$$

Figure 4.8 shows the structure of the proposed AOC for this linear system. In this structure, $r(t)$ is the desired motion trajectory and also note that two unknown pa-

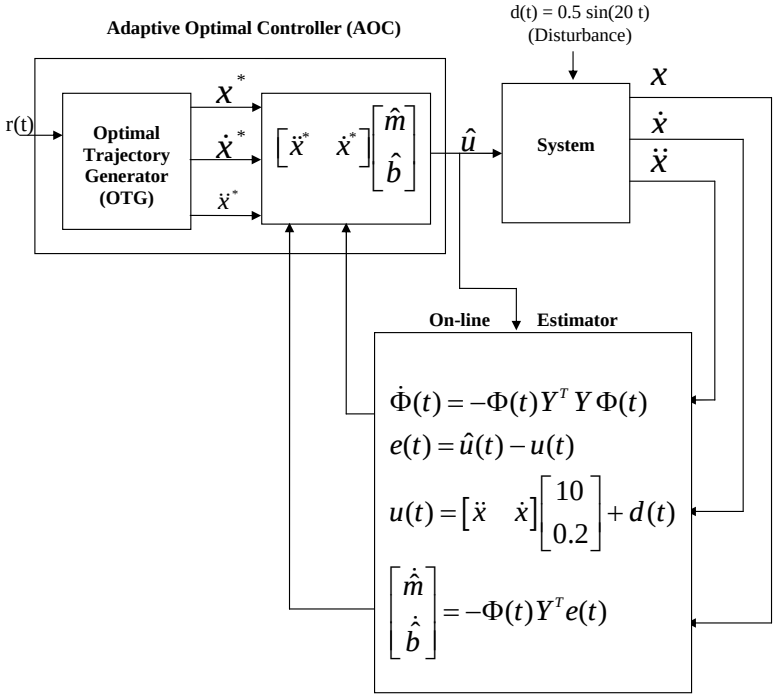


Figure 4.8: **Adaptive optimal control system for linear system**

rameters $\hat{m}$ and $\hat{b}$ are estimated by the equations given in on-line estimator of this control system.

Let us now apply the above control system into the following cases

- Motion of mass m from $x = 0$ to $x = 1.5$:

  Figure 4.9 shows the optimal disposition, velocity and force trajectories of the system in this case. In addition, Figure 4.10 illustrates the estimation parameters results for this case as well. In this case we consider the following initial condition for on-line estimator

$$\Phi(0) = \begin{bmatrix} 1000 & 1 \\ 2 & 1000 \end{bmatrix}, \hat{\theta}(0) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

- Motion of mass m along the desired trajectory $r(t) = 0.3 + 0.1 \sin(\pi t)$:

  Considering the same initial conditions as previous case, the optimal disposition, velocity and force trajectories are shown in Figure 4.11. The estimation results also are depicted in Figure 4.12.

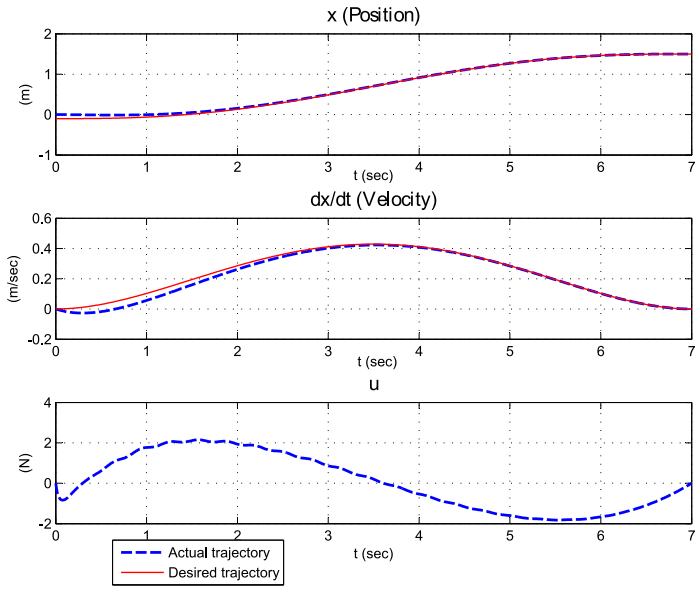


Figure 4.9: **Optimal point-to-point motion of linear system**

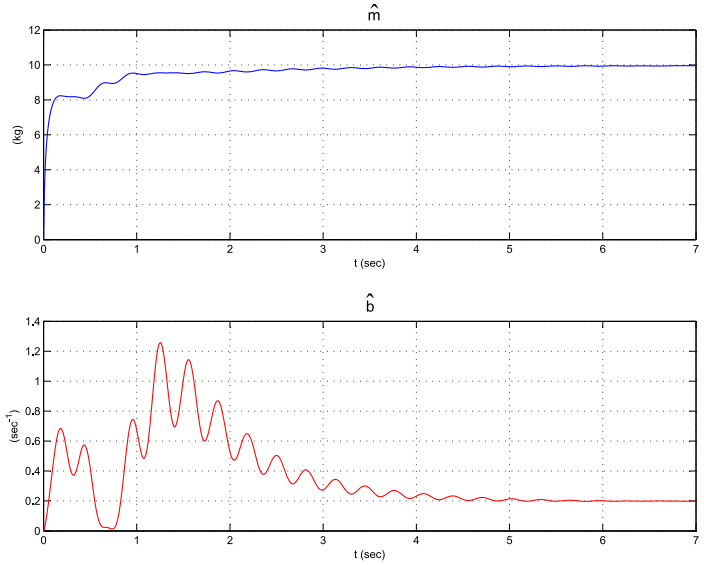


Figure 4.10: **Estimated parameters of linear system for point-to-point motion**

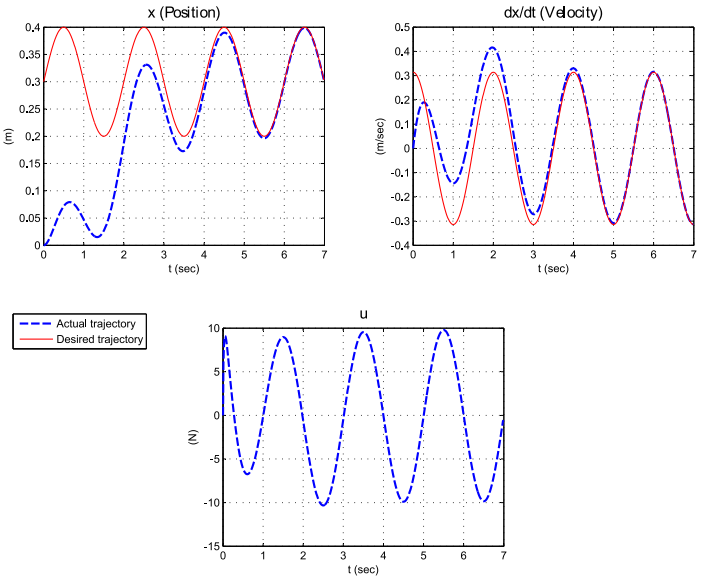


Figure 4.11: **Optimal motion of linear system in the case of motion tracking**

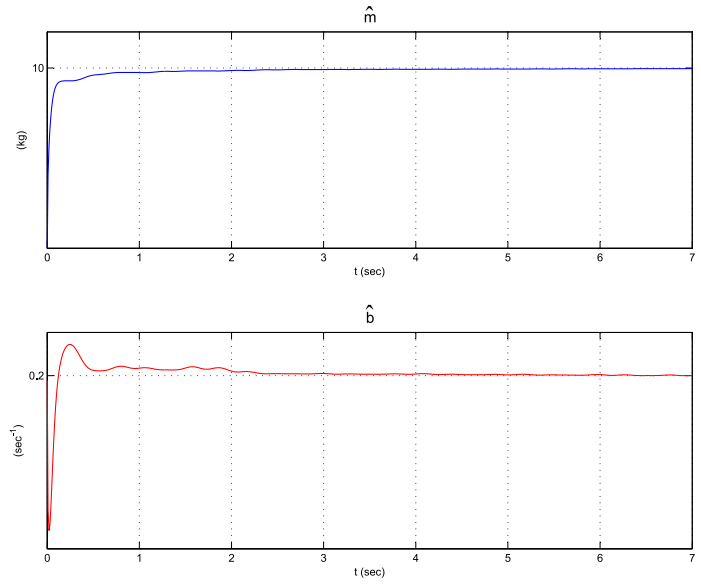


Figure 4.12: **Estimated parameters of linear system for motion tracking case**

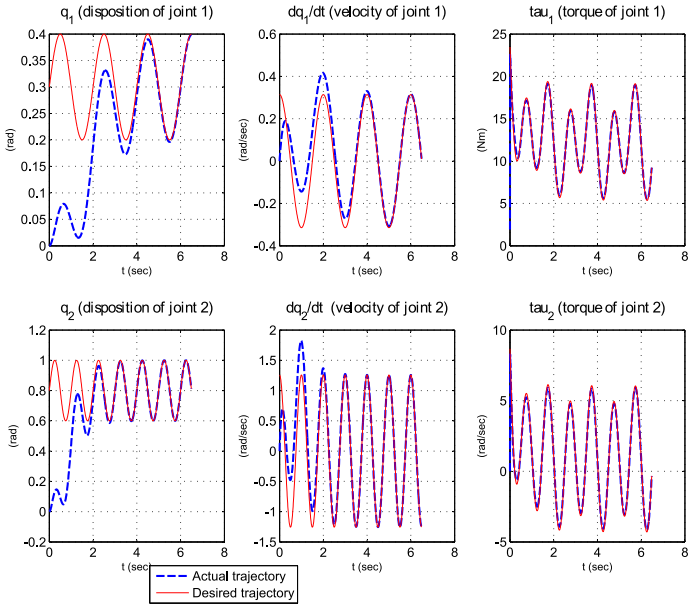


Figure 4.13: **Optimal trajectory of two links robot obtained by applying AOC**

As seen from the estimation results in Figure 4.10 and 4.12, the above control system has good robustness with respect to noise and disturbance. The other feature in the estimator of the proposed control system is that it converges fast initially, but slowly afterward.

4.8.2 Vertical Two Links Robot Manipulator

In the subsection 4.6.1, we examined this robot to design its optimal controller in the case that we have the exact dynamic model of this robot. In this subsection, it is attempted to apply the proposed adaptive optimal control for this robot. First of all, we require a regression model of this robot which is as follows:

$$\begin{bmatrix} Y_{11} & Y_{12} & Y_{13} & Y_{14} & Y_{15} \\ Y_{21} & Y_{22} & Y_{23} & Y_{24} & Y_{25} \end{bmatrix} \cdot \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \\ \theta_4 \\ \theta_5 \end{bmatrix} = \begin{bmatrix} \tau 1 \\ \tau 2 \end{bmatrix} \tag{4.79}$$

where

$$
\begin{aligned}
Y_{11} =& l_1^2 \ddot{q}_1 + g l_1 \cos(q_1), \\
Y_{12} =& 2 l_1 \cos(q_2) \ddot{q}_1 + l_1 \cos(q_2) \ddot{q}_2 + g \cos(q_1 + q_2) - l_1 \sin(q_2) \left(2 \dot{q}_1 \dot{q}_2 + \dot{q}_2^2\right), \\
Y_{13} =& \ddot{q}_1,\ Y_{14} = \ddot{q}_1 + \ddot{q}_2,\ Y_{15} = g \cos(q_1), \\
Y_{21} =& 0,\ Y_{22} = l_1 \cos(q_2) \ddot{q}_1 + l_1 \sin(q_2) \dot{q}_1^2 + g \cos(q_1 + q_2),\ Y_{23} = 0, \\
Y_{24} =& \ddot{q}_1 + \ddot{q}_2,\ Y_{25} = 0,
\end{aligned}
$$

and

$$
\theta_1 = m_2,\ \theta_2 = m_2 l_{c2},\ \theta_3 = I_1 + m_1 l_{c1}^2,\ \theta_4 = I_2 + m_2 l_{c2}^2,\ \theta_5 = m_1 l_{c1}.
$$

In this case study, the exact values of the dynamic parameters of the system are those given in Table 4.1. Therefore, the exact values of the parameters θ_i are $\theta_1 = 1$, $\theta_2 = 0.2$, $\theta_3 = 0.68$, $\theta_4 = 0.54$, $\theta_5 = 0.6$. The on-line estimator for this robotic system has the following form

$$
\begin{bmatrix}
\dot{\hat{\theta}}_1 \\
\dot{\hat{\theta}}_2 \\
\dot{\hat{\theta}}_3 \\
\dot{\hat{\theta}}_4 \\
\dot{\hat{\theta}}_5
\end{bmatrix}
= -\Phi(t)\ Y^T e,
$$

where $\Phi(t)$ is a 5×5 matrix whose elements are obtained from the following equation which is a matrix differential equation

$$
\dot{\Phi}(t) = -\Phi(t) Y^T Y \Phi(t).
$$

and $e = Y \hat{\theta} - Y \theta$. The objective is that the joints of the robot track the following desired trajectories:

$$
\begin{aligned}
q_{d1} &= 0.3 + 0.1 \sin(\pi t) \\
q_{d2} &= 0.8 + 0.2 \sin(2\pi t)
\end{aligned}
$$

Therefore, assuming a disturbance as $d(t) = o.5 \sin(50t)$ in the system, the optimal

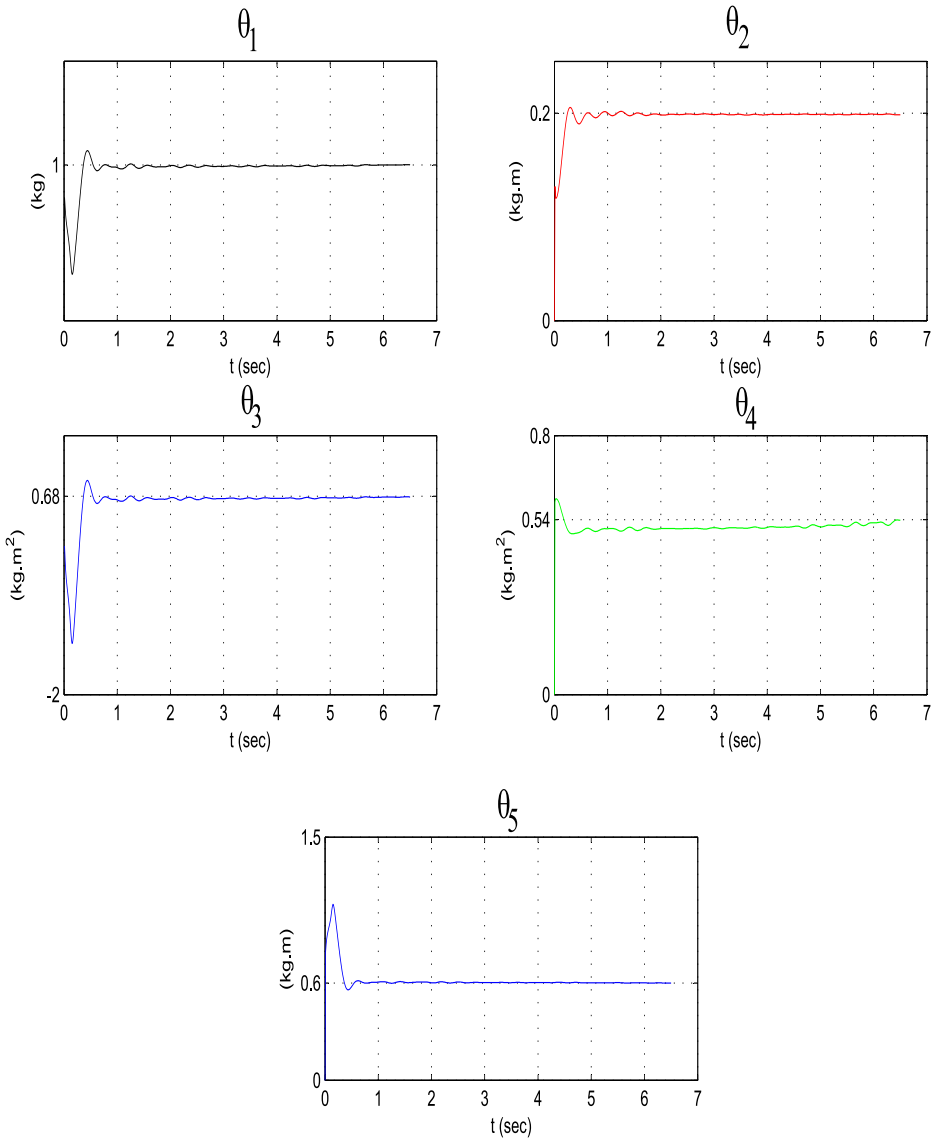


Figure 4.14: **Estimated parameters of two links robot**

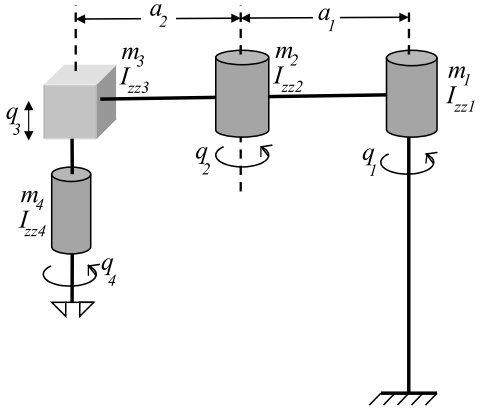


Figure 4.15: **SCARA robot**

trajectories are obtained as shown in Figure 4.13. In this figure the blue dashed trajectories are actual ones while the trajectories with continuous line are desired trajectories. In addition, Figure 4.14 depicts the estimation parameters $\hat{\theta}_1$ to $\hat{\theta}_5$ existed in dynamic model of the system. As these figures show, the tracking task is performed completely after $t = 5.65\,sec$.

4.8.3 SCARA Robot

In this subsection, we wish to apply the proposed AOC into a SCARA robot shown in Figure 4.15. This robot is a 4 degrees of freedom robot whose first, second and fourth joints are revolute while its third joint is prismatic. The regression model of this robot can be obtained as follows [29]:

$$\begin{bmatrix} Y_{11} & Y_{12} & Y_{13} & Y_{14} & Y_{15} & Y_{16} & Y_{17} & Y_{18} & Y_{19} \\ Y_{21} & Y_{22} & Y_{23} & Y_{24} & Y_{25} & Y_{26} & Y_{27} & Y_{28} & Y_{29} \\ Y_{31} & Y_{32} & Y_{33} & Y_{34} & Y_{35} & Y_{36} & Y_{37} & Y_{38} & Y_{39} \\ Y_{41} & Y_{42} & Y_{43} & Y_{44} & Y_{45} & Y_{46} & Y_{47} & Y_{48} & Y_{49} \end{bmatrix} \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \\ \theta_4 \\ \theta_5 \\ \theta_6 \\ \theta_7 \\ \theta_8 \\ \theta_9 \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \tau_2 \\ \tau_3 \\ \tau_4 \end{bmatrix} \tag{4.80}$$

where

$$
\begin{aligned}
Y_{11} =&\ddot{q}_1,\ Y_{12} = C2\,\ddot{q}_1 + 0.2C2\,\ddot{q}_2 - S2\,\dot{q}_2\left(\dot{q}_1 + 0.5\dot{q}_2\right),\ Y_{13} = \ddot{q}_2\\
Y_{14} =&0,\ Y_{15} = -\ddot{q}_4,\ Y_{16} = \dot{q}_1,\ Y_{17} = Y_{18} = Y_{19} = 0\\
Y_{21} =&0,\ Y_{22} = 0.5C2\,\ddot{q}_1 + 0.5S2\,\dot{q}_1^2,\ Y_{23} = \ddot{q}_1 + \ddot{q}_2,\ Y_{24} = 0,\\
Y_{25} =&-\ddot{q}_4,\ Y_{26} = 0,\ Y_{27} = \dot{q}_2,\ Y_{28} = Y_{29} = 0,\\
Y_{31} =&Y_{32} = Y_{33} = Y_{35} = Y_{36} = Y_{37} = Y_{39} = 0,\ Y_{34} = \ddot{q}_3 - g,\ Y_{38} = \dot{q}_3,\\
Y_{41} =&Y_{42} = Y_{43} = Y_{44} = Y_{46} = Y_{47} = Y_{48} = 0,\ Y_{45} = -\left(\ddot{q}_1 + \ddot{q}_2 - \ddot{q}_4\right),\ Y_{49} = \dot{q}_4,
\end{aligned}
$$

and

$$
\begin{aligned}
\theta_1 =&m_1x_1^2 + m_2\left(a_1^2 + x_2^2\right) + \left(m_3 + m_4\right)\left(a_1^2 + a_2^2\right) + \sum_{i=1}^{4} I_{zzi},\\
\theta_2 =&2\left[a_1x_2m_2 + a_1a_2\left(m_3 + m_4\right)\right],\ \theta_3 = m_2x_2^2 + a_2\left(m_3 + m_4\right) + \sum_{i=2}^{4} I_{zzi},\\
\theta_4 =&m_3 + m_4,\ \theta_5 = I_{zz4},\ \theta_6 = F_{v1},\ \theta_7 = F_{v2},\ \theta_8 = F_{v3},\ \theta_9 = F_{v4}
\end{aligned}
\tag{4.81}
$$

In the regression model (4.80), q_i, τ_i (for $i = 1, 2, 3, 4$) denote the disposition and torque variables of joints of the SCARA robot, respectively, as well as $C2, S2$ stand for $\cos(q_2), sin(q_2)$, respectively. In addition, the following variables have been used in equation (4.81):

- m_i: mass of link i,
- I_{zzi}: principal moments of inertia of joint i around the z axis of the principal frame of link i,
- x_1, x_2: center of mass of links 1 and 2,
- l_1, l_2: length of links 1 and 2,
- F_{vi}: viscose friction parameter of joint i.

for $i = 1, 2, 3, 4$.

In the case of presence of a disturbance as $d\left(t\right) = 0.5\sin\left(20t\right)$, the estimation results are shown in Figure 4.16. In addition, Figure 4.17 and Figure 4.18 show the desired and actual trajectories obtained by applying AOC scheme into this SCARA robot.

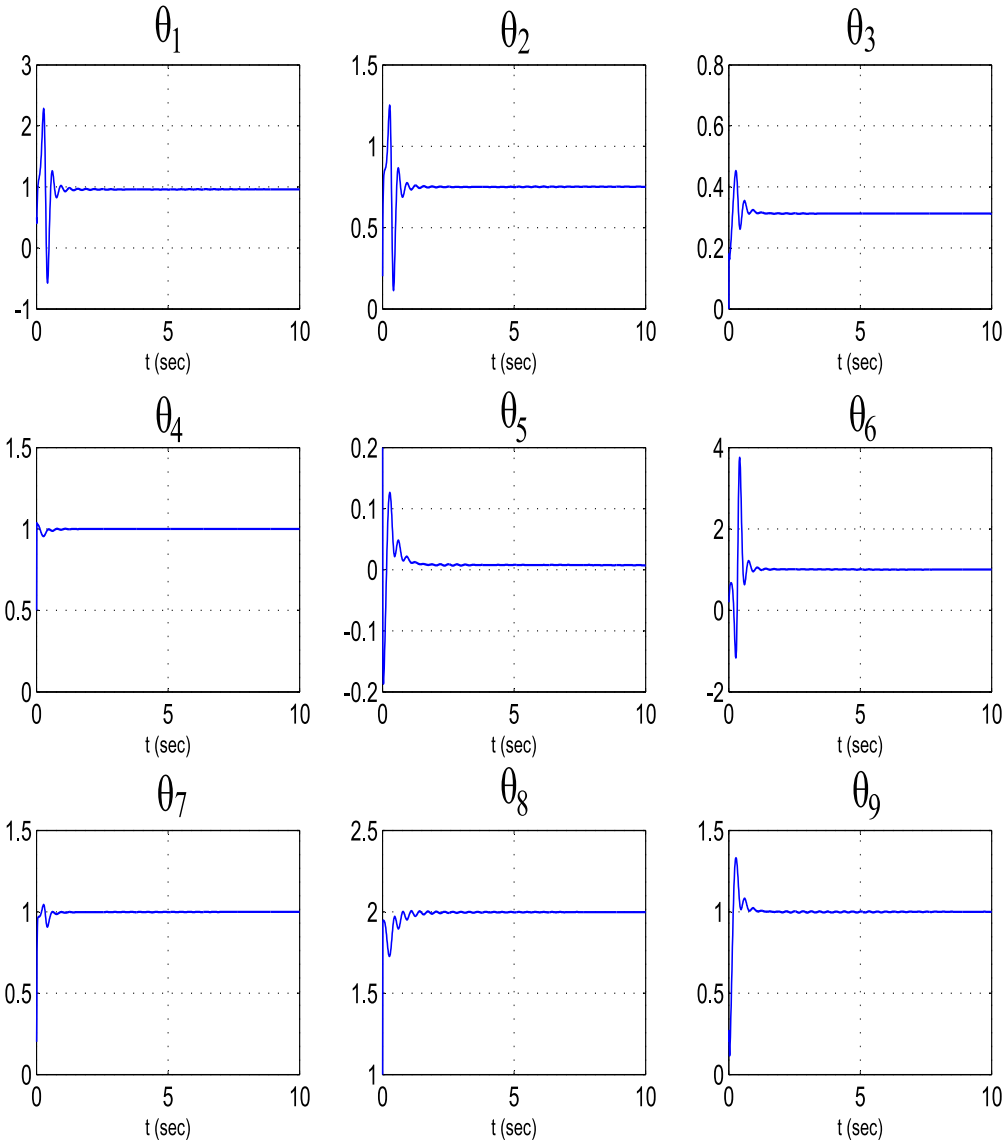


Figure 4.16: **Estimated parameters of SCARA robot**

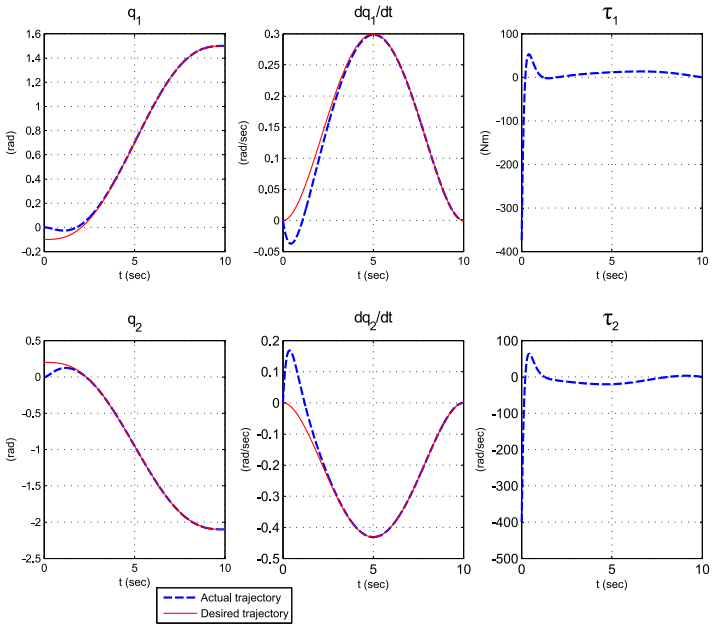


Figure 4.17: **Desired and optimal trajectories of joints 1 and 2 of SCARA robot**

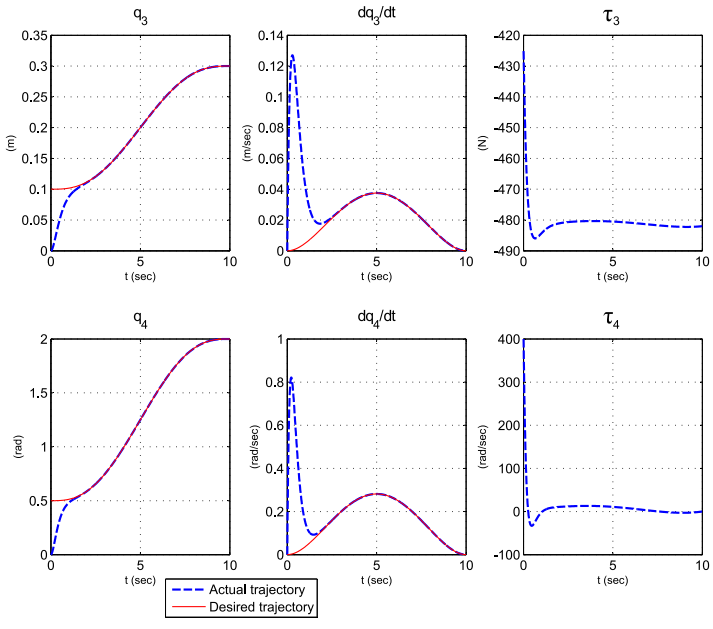


Figure 4.18: **Desired and optimal trajectories of joints 3 and 4 of SCARA robot**

Chapter 5

Second Proposed Method

5.1 Introduction

In this chapter the second proposed method is introduced by which the constrained OCP of robot manipulators is solved. In chapter 2 and 3 the necessary tools for this method were provided. Among these tools we can refer to the obtained exact dynamic model of the robot manipulator, the spline-based optimal control, iterative learning control and multiple shooting methods. The OCP dealt with in this chapter can be stated as follows:

let an industrial robot manipulator that should perform a desired task repeatedly in a given finite time, for instance pick and place parts in assembly lines. The OCP in this case is to design a controller to steer the robot to move from an initial configuration to final one in each repetition (trial) and at the same time some performance criterion (cost functional) is optimized (usually minimized).

This cost functional used in the OCP of robot manipulators can be the execution time [33; 92], actuator effort [87; 89], absolute value of the jerk (derivative of acceleration) [57; 73], or a combination of these variables. The goal of minimizing of the execution time is obvious which is used to increase the productivity in the industrial factories in which robot manipulators are employed. In the case of actuator effort, one attempts to minimize the total value of power consumption by the mechanism. The last case, i.e. minimizing the absolute value of the jerk, is an important subject to plan the trajectories for robot motions. Since high jerk values can cause the wearing in the

robot joints, and actually excite the resonance modes of the robot structure.

The basic idea in the proposed method in this chapter is to combine some methods as the building stone of a new optimal control scheme to solve the OCP of robot manipulators. In fact, the proposed method includes iterative linearization and ILC techniques as well as the spline-based optimal control method. Of course, the linearization used here is not the ordinary one in which Taylor expansion is used for approximation of nonlinear systems. The linearization employed here has an iteration nature and yields a global linearized version of the original nonlinear system. The only necessary condition to use this method is that a matrix associated with the nonlinear system must be locally Lipschitz. Actually, the optimal control inputs of the robot are obtained by solving a sequence of finite dimensional, linearized OCPs whose solutions are increasingly convergent to the solution of the original problem; hence these linearized OCPs are consistent approximation of the original problem. The solution of the linearized OCPs can be obtained using the standard programming techniques as explained in the subsection 3.5.

In the subsequent sections, the various parts of the proposed method will be presented and examined. Then, the proposed algorithm will be applied into all kinds of standard manipulator structures, i.e SCARA, spherical (Stanford), cylindrical and angular manipulators (such as Puma 560, ABB IRB 140, KUKA robots). It will be shown that the proposed method is very effective in the sense that it is easy to apply for any kind of complexity of robot dynamics and constraints and also its convergent is too fast. Also, it will be shown that the proposed method solves the OCP during a finite number of trials and so we can divide the time necessary to solve the optimization problem over these trials. Therefore, the optimal control obtained after converging the sequence of optimal solutions can be applied into the rest of repetitions without performing any computation.

5.2 Iterative Linearization

As stated earlier, one of the sub-methods in the proposed method is iterative linearization of nonlinear dynamic systems. In this section, this subject is examined in detail.

5.2.1 Some Preliminary Definitions

Definition 1. *Let $f : \mathbb{R}^n \to \mathbb{R}^n$ be a differentiable function, then the derivative Df of this function is given by the $n \times n$ Jacobian matrix*

$$Df = \left[\frac{\partial f_i}{\partial x_j}\right], \tag{5.1}$$

for $i, j = 1, 2, \cdots, n$.

Definition 2. *In the reminder of this text we use the following operator norm defined for operator $L : \mathbb{R}^n \to \mathbb{R}^n$:*

$$\|L\| = \sup_{x \in \mathbb{R}^n} |L(x)|, \tag{5.2}$$

where $|\cdot|$ is the Euclidean norm.

Definition 3. *Suppose E is an open subset of $\mathbb{R}^n$ and let $A(x)$ be a matrix-valued function. Then, it is said that $A(x)$ is locally Lipschitz on E if for each point $x_0 \in E$ there exist K and an open ball of radius ε centered at x_0 denoted by $N_\varepsilon(x_0) \subset E$, such that for all $x, y \in N_\varepsilon(x_0)$*

$$|A(x) - A(y)| \leq K |x - y|, \tag{5.3}$$

K can be chosen

$$K = \max_{|x - x_0| \leq \varepsilon/2} |DA(x)|. \tag{5.4}$$

Definition 4. *Logarithmic norm of a $n \times n$ matrix A is defined as follows*

$$\mu(A) = \lim_{h \to 0^+} \frac{\|I + hA\| - 1}{h}, \tag{5.5}$$

where I is the identity matrix. This norm can be computed in the following forms:

- $\mu_1(A) = \sup_j \left(Re(a_{jj}) + \sum_{i, i \neq j} |a_{ij}| \right)$;
- $\mu_2(A) = \alpha\left(A + A^T\right)/2$ *where* $\alpha\left(A + A^T\right)$ *is the maximal real part of the eigenvalues of* $A + A^T$;
- $\mu_\infty(A) = \sup_i \left(Re(a_{ii}) + \sum_{j, j \neq i} |a_{ij}| \right)$.

Notice that unlike other norms, the logarithmic norm can have a negative value and so it is not norm in the classical meaning. For instance if $A = \lambda$ *where* λ *is a constant, then* $\|A\| = |\lambda|$ *while* $\mu(A) = Re(\lambda)$*. One of the important properties of this norm is that*

$$\left\|e^{tA}\right\| \leq e^{t\mu(A)}, \tag{5.6}$$

which is used in the theory of control.

Definition 5. *Let* $W \subset \mathbb{R}^n$ *be a complete space. Then, sequence* $\{x_n\} \subset W$ *is called* Cauchy*, if for any positive real number* ε*, there is a positive integer* k *so that for all integer numbers* i *and* j *greater than* k *the following relation is satisfied:*

$$\|x_i - x_j\| < \varepsilon. \tag{5.7}$$

In fact, a Cauchy sequence is a convergent sequence whose elements converge to some existing limit.

5.2.2 Successive Approximations of Nonlinear Dynamic Systems

The linearization methods of nonlinear dynamic systems can be categorized into two groups: local and global linearization methods. In the case of local linearizations in the phase space, the nonlinear system is linearized about its equilibrium points [72]. On the other hand, there exist some other methods which result in global linearizations such as Lie algebraic methods. The iterative linearziation method is also a type of linearizing methods through which one can obtain a global linearized version of the original nonlinear system. The procedure used in this method to linearize a nonlinear dynamic system can be justified via theory of connections in the differential geometry. This theory states how the tangent vectors to a compact differentiable manifold can be transported along a curve so that the properties of their parallelness and consistence are held from one point of a curve to another [9].

The basic idea in this kind of linearization is that the original nonlinear system can be approximated by a sequence of linear time varying (LTV) systems whose solutions eventually will converge to the solution of the original nonlinear system [10; 78; 79; 98].

In the next sections, we shall consider this method to use in the case of robot manipulators. Let us now consider an autonomous unforced nonlinear dynamical system given by

$$\dot{x}(t) = f(x(t)), \quad x(0) = x_0, \tag{5.8}$$

where $f : E \to \mathbb{R}^n$ is a continuously differentiable function on E which is an open subset of $\mathbb{R}^n$. Let us now approximate the equation (5.8) as the following form referred to as the state dependent coefficient (SDC) form

$$\dot{x} = A(x)\, x, \tag{5.9}$$

in which it is assumed that $x_e = 0$ is the equilibrium point as well as the matrix $A(x)$ is locally Lipschitz on E, for guaranteeing existence and uniqueness of the solution. It is worth to note that the replacement of (5.8) by (5.9) can be justified by the geometry concept of differential equations which is described by compact differentiable manifolds and it is proven that the dynamic system (5.9) is topologically equivalent with the original nonlinear system (5.8) in region E [72; 79]. Note also that the matrix A is not unique; however if two matrices A_1 and A_2 represents the same system, then they have to satisfy the following relation

$$(A_1(x) - A_2(x))\, x = 0, \tag{5.10}$$

which implies that the difference between A_1 and A_2 is a matrix $B = A_1 - A_2$ whose null space includes x for any $x \in \mathbb{R}^n$.

Next, based on this method, the SDC form given in (5.9) can be iterated by the following sequence of linear time varying (LTV) dynamic systems:

$$\begin{aligned}
&\dot{x}^{[1]}(t) = A\left(x^{[0]}(t)\right) x^{[1]}(t), \quad x^{[1]}(0) = x_0,\\
&\dot{x}^{[2]}(t) = A\left(x^{[1]}(t)\right) x^{[2]}(t), \quad x^{[2]}(0) = x_0,\\
&\vdots\\
&\dot{x}^{[i]}(t) = A\left(x^{[i-1]}(t)\right) x^{[i]}(t), \quad x^{[i]}(0) = x_0,
\end{aligned} \tag{5.11}$$

for $i > 1$; where the first approximation is a linear time invariant (LTI) system, since

we can choose $x^{[0]}(t) = x_0$ that is constant, while the rest approximations are linear time vary systems; since matrix A in each iteration is a function of solution of the previous approximation. However, the initial chosen solution $x^{[0]}(t)$ can be set to an appropriate function to achieve a faster convergence to the original solution.

Let us now have a convergence analysis in the sequence of the solutions obtained in (5.11). It can be shown that the sequence of these solutions is Cauchy. It means that the limit of the sequence $\left\{x^{[i]}(t)\right\}_{i\geq 1}$ equals $x(t)$ for t in any compact time interval, $[0, T]$. First of all, let a compact set as follows:

$$N_0 = \{x \in E \mid \ |x - x_0| \leq b\},$$

where $b = \varepsilon/2$ (see Definition. 3 in the previous subsection). Let

$$M = \sup_{x \in N_0} |A(x)\, x|.$$

Then, we can write

$$\sup_{[0,T]} \left|x^{[i-1]}(t) - x_0\right| \leq b. \tag{5.12}$$

Since the solution of $(i-1)$-th LTV in (5.11) is

$$x^{[i-1]}(t) = x_0 + \int_0^t A\left(x^{[i-2]}(\tau)\right) x^{[i-1]}(\tau)\, d\tau,$$

for $t \in [0, T]$. Then,

$$\left\|x^{[i-1]}(t) - x_0\right\| \leq \int_0^t \left\|A\left(x^{[i-2]}(\tau)\right) x^{[i-1]}(\tau)\right\| d\tau \leq M\, T.$$

Thus, choosing $0 < T \leq b/M$, it causes the establishment of (5.12).

Let us now consider i and $(i-1)$ th LTVs in (5.11)

$$\begin{aligned} \dot{x}^{[i]}(t) &= A\left(x^{[i-1]}(t)\right) x^{[i]}(t), \\ \dot{x}^{[i-1]}(t) &= A\left(x^{[i-2]}(t)\right) x^{[i-1]}(t). \end{aligned} \tag{5.13}$$

Subtracting these two equations results in

$$\begin{aligned}\dot{x}^{[i]}(t)-\dot{x}^{[i-1]}(t)&=A\left(x^{[i-1]}(t)\right)x^{[i]}(t)-A\left(x^{[i-2]}(t)\right)x^{[i-1]}(t),\\&+A\left(x^{[i-1]}(t)\right)x^{[i-1]}(t)-A\left(x^{[i-1]}(t)\right)x^{[i-1]}(t).\end{aligned}\tag{5.14}$$

Then we have

$$\dot{x}^{[i]}(t)-\dot{x}^{[i-1]}(t)=A\left(x^{[i-1]}(t)\right)\left[x^{[i]}(t)-x^{[i-1]}(t)\right]+\left[A\left(x^{[i-1]}(t)\right)-A\left(x^{[i-2]}(t)\right)\right]x^{[i-1]}(t).$$

Thus,

$$x^{[i]}-x^{[i-1]}=\int_0^t\Phi^{[i-1]}(t,\tau)\left[A\left(x^{[i-1]}(\tau)\right)-A\left(x^{[i-2]}(\tau)\right)\right]x^{[i-1]}(\tau)\,d\tau,\tag{5.15}$$

where $\Phi^{[i-1]}(t,\tau)$ is the state-transition matrix of the corresponding $(i-1)$-th LTV. Now, (5.15) can be rewritten as

$$\left\|x^{[i]}-x^{[i-1]}\right\|\leq\int_0^t\left\|\Phi^{[i-1]}(t,\tau)\right\|\left\|A\left(x^{[i-1]}(\tau)\right)-A\left(x^{[i-2]}(\tau)\right)\right\|\left\|x^{[i-1]}(\tau)\right\|d\tau.$$

So according to definition of $\mu_2(A)$ and property (5.6) in the previous subsection we have

$$\begin{aligned}\left\|x^{[i]}-x^{[i-1]}\right\|\leq\int_0^t e^{\mu(t-\tau)}K\left\|x^{[i-1]}(\tau)-x^{[i-2]}(\tau)\right\|b\,d\tau\leq\\\sup_{t\in[0,T]}\left\{e^{\mu(T-t)}\right\}b\,K\,T\left\|x^{[i-1]}(\tau)-x^{[i-2]}(\tau)\right\|.\end{aligned}\tag{5.16}$$

Let us now suppose $\alpha=\sup_{t\in[0,T]}\left\{e^{\mu(T-t)}\right\}b\,K\,T<1$, then we can claim that $x^{[i]}(t)\to x(t)$. For establishing the recent condition, it is necessary that the final time T is small enough. A such procedure is used in linearizing the nonlinear dynamics of robot manipulators in the next sections.

Note that the above discussion can be extended to the forced dynamical system

$$\dot{x}=f(x,u),\tag{5.17}$$

which according to iterative linearization method can be rewritten as the following SDC form:

$$\dot{x} = A(x,u)\,x + B(x,u)\,u, \tag{5.18}$$

so that its corresponding LTV system in the i-th iteration is

$$\dot{x}^{[i]}(t) = A\left(x^{[i-1]}(t), u^{[i-1]}(t)\right) x^{[i]}(t) + B\left(x^{[i-1]}(t), u^{[i-1]}(t)\right) u^{[i]}(t), \quad x^{[i]}(0) = x_0, \tag{5.19}$$

where we can take $u^{[1]}(t) = 0$ and $x^{[t]}(t) = x_0$ for initial chosen guesses. In the next section, it will be shown how this extension can be applied in the case of robot manipulator systems.

5.3 Main Proposed Method for Solving OCP of Robot Arms

In this section, the various parts of the proposed method are examined. First, we will consider the cubic spline interpolation (trajectory planning) which is consistent with our algorithm. Then, the OCP of linear time varying systems is taken into account. In the next step, the original problem, i.e., optimal control of robot manipulators will be presented.

5.3.1 Cubic Spline Interpolation Consistent with Proposed Method (Trajectory Planning)

One of the parts in designing any kind of robot control systems is *trajectory (path) planning*. A *path* actually is a sequence of points (Cartesian coordinates) in task space which describes the robot motion, namely motion route of the end-effector. On the other hand, a trajectory is a curve in the state space of the robot which describes the configuration of the robot in every moment. In fact, trajectory is a curve which associates with the dynamics of the system.

Ordinarily, the robot motions are categorized into two groups: 1- motion along a predetermined path such as those in spraying, gluing and welding tasks as well as

motions for obstacle avoidance tasks, 2- motion through a sequence of points, well known as *point to point motions*, e.g. those in spot welding or pick and place tasks. In this work we study the point to point optimal paths along which the robot moves to carry out the desired task and at the same time some factors (e.g. traversal time along the path or energy consumption during the motion) are minimized. Therefore, the motion planning can be performed either in task space or joint space. However, since the robot motion equations usually are represented in joint space it is better to transfer the sequence of points in the tasks space by an appropriate transformation (inverse kinematics and Jacobian transformations) into joint space to accomplish the motion planning in this space. It is worth to note that performing the motion planning in task space is too difficult since it is necessary to transfer the actuator force/torque bounds into corresponding bounds in the task space (Cartesian bounds) and due to highly nonlinear and coupled robot dynamics it results in a high burden of computations. There are a lot of works to solve the motion planning problem in the joint space [28; 60]. However, some works such as [34] solved path planning problem in the task level by deriving the smooth path in closed form using semi-algebraic sets and calculus of variations.

In this subsection, the trajectory generation consistent with our algorithm is presented. Among the mathematic curves usually spline functions are used to approximate a trajectory for robot joints via curve fitting method. A spline curve is a piecewise polynomial function which can be used as a mathematical device allowing us to easily design and control the shape of complex curves and surfaces. There are several kinds of spline functions which can be used in the mentioned application including straight-line splines, polynomial curves, cubic functions like Hermite cubic splines, Bezier curves, B-splines and so on [23; 80]. In the case of robot manipulators cubic polynomial splines can be used to plan a smooth trajectory with C^2 continuity. In other words, they guarantee the continuity of angular velocity and acceleration in the joint space which causes not exciting mechanical resonances and vibrations of the robot. In fact, non-smoothing trajectories can damage the actuators of the robot and also cause undesirable error in the path tracking tasks.

Let us now suppose two points $(0, X_0)$ and (T, X_T) through which we wish to interpolate a function by a cubic spline curve as shown in Figure 5.1 [12]. A spline function $S : [0, T] \to \mathbb{R}$ consists of polynomial pieces $s_j : [t_j, tj + 1) \to \mathbb{R}$ as follows:

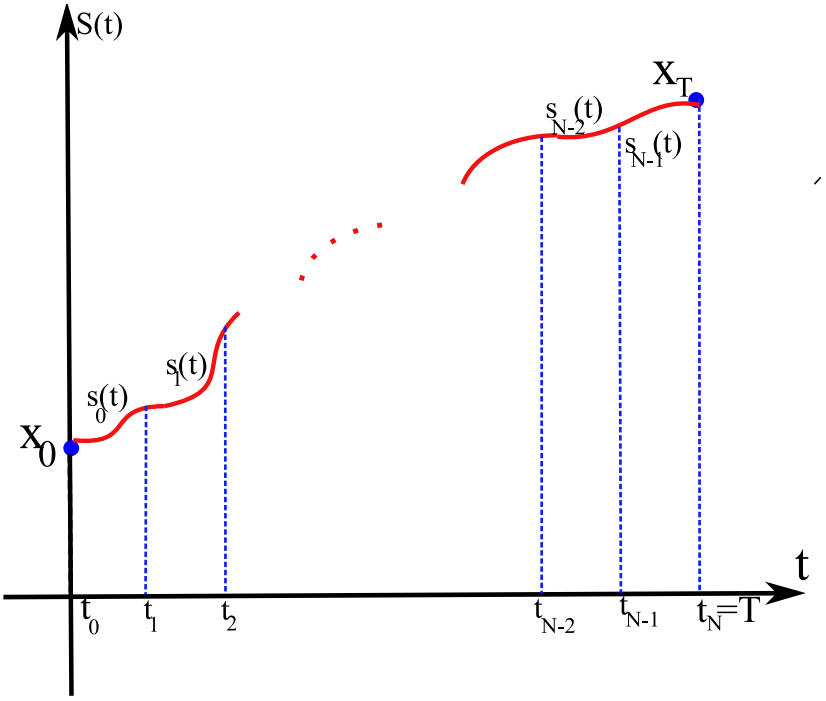


Figure 5.1: **Fitting a spline to data points**

$$S(t;P)=\begin{cases} s_0(t)+P_{04}t^4 & t_0\le t<t_1,\\ s_1(t) & t_1\le t<t_2,\\ \vdots & \\ s_j(t) & t_j\le t<t_{j+1},\\ \vdots & \\ s_{N-1}(t)+P_{(N-1),0}\left(t-t_{N-1}\right)^4 & t_{N-1}\le t\le t_N=T, \end{cases} \tag{5.20}$$

where N is the number of subpolynomials and s_j and its first and second derivatives have the following forms:

$$\begin{aligned} s_j(t) &= P_{j0}+P_{j1}(t-t_j)+P_{j2}(t-t_j)^2+P_{j3}(t-t_j)^3,\\ \dot{s}_j(t) &= P_{j1}+2P_{j2}(t-t_j)+3P_{j3}(t-t_j)^2,\\ \ddot{s}_j(t) &= 2P_{j2}+6P_{j3}(t-t_j). \end{aligned} \tag{5.21}$$

In addition, if t_j (which well known as knot, node or breakpoint) are equidistantly distributed in the interval $[0,T]$, the spline is called *uniform*, otherwise it is non-

uniform. Considering $h = h_k = t_k - t_{k-1},\ k = 1, 2, \ldots, N$ and knot vector as $\mathbf{t} = (0, h, 2h, \ldots, Nh)$, the following smoothing conditions should be met:

1. $S(0; P) = X_0,\ S(Nh; P) = X_T$;
2. $\dot{S}(0; P) = Xd_0,\ \dot{S}(Nh; P) = Xd_T$;
3. $\ddot{S}(0; P) = 0,\ \ddot{S}(Nh; P) = 0$;
4. $s_j((j+1)h) = s_{j+1}((j+1)h)$ for each $j = 0, 1, \ldots, N-1$;
5. $\dot{s}_j((j+1)h) = \dot{s}_{j+1}((j+1)h)$ for each $j = 0, 1, \ldots, N-1$;
6. $\ddot{s}_j((j+1)h) = \ddot{s}_{j+1}((j+1)h)$ for each $j = 0, 1, \ldots, N-1$;

where in the condition 2, we suppose that the derivative of spline curve in the initial time $t = 0$ and final time $t = Nh = T$ are known as Xd_0 and Xd_T, respectively. Satisfying above conditions yields a set of equations in terms of spline parameters $P_{ji},\ j = 0, 1, \ldots, N-1,\quad i = 1, 2, 3, 4$. Note that we have $4N + 2$ unknowns and $3N + 3$ equations. In addition, these unknowns are dependent so that the number of independent parameters denoted as α, is given by

$$\alpha = N + 3 - \beta, \tag{5.22}$$

where β is the number of boundary conditions [95] which in our case according to conditions 1 and 2, $\beta = 4$, then $\alpha = N - 1$. Therefore, in our algorithm we considered $P_{j3},\ j = 0, 1, \ldots, N-2$ as independent parameters. In other words, these $N-1$ parameters are those which are used as parametric variables in the nonlinear programming examined in the next subsection. As a result, with satisfying the conditions 1 to 5

mentioned above, the following matrix equation is obtained:

$$\begin{bmatrix} -h^2 I & Z_{(N-1)\times(N-1)} & -h^2 I & Z_{(N-1)\times 1} & U & Z_{(N-1)\times 1} \\ -2hI & Z_{(N+1)\times(N-1)} & U & Z_{(N+1)\times(N-1)} & & \\ U & Z_{(N+1)\times(2N+1)} & & & & \\ A & & & & & \\ B & & & & & \\ C & & & & & \\ D & & & & & \end{bmatrix} \begin{bmatrix} \mathbf{P}_{i0} \\ \mathbf{P}_{i1} \\ \mathbf{P}_{i2} \\ P_{N-1,3} \end{bmatrix} = \begin{bmatrix} h^3 \mathbf{P}_{k3} \\ 3h^2 \mathbf{P}_{k3} \\ 3h \mathbf{P}_{k3} \\ X_T \\ X d_T \\ X_0 \\ X d_0 \end{bmatrix}$$

$$\text{for } i = 0, 1, \ldots, N-1,\ k = 0, 1, \ldots, N-2; \tag{5.23}$$

where I is an $(N-1) \times (N-1)$ identity matrix, Z denotes a matrix with all zero elements and matrix U and row vectors A, B, C, D are given as

$$\begin{aligned} A &= [0, \ldots, 0, A(N), 0, \ldots, 0, A(2N), 0, \ldots, 0, A(3N), A(3N+1)], \\ B &= [0, \ldots, 0, B(N), 0, \ldots, 0, B(2N), 0, \ldots, 0, B(3N+1)], \\ C &= [0, \ldots, 0, C(2N+1), 0, \ldots, 0, \ldots, 0], \\ D &= [0, \ldots, 0, C(N+1), 0, \ldots, 0, \ldots, 0], \\ U &= \begin{bmatrix} -1 & 1 & 0 & 0 & \cdots & 0 \\ 0 & -1 & 1 & 0 & \cdots & 0 \\ 0 & 0 & -1 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & -1 & 1 & 0 \\ 0 & \cdots & 0 & 0 & -1 & 1 \end{bmatrix}, \end{aligned} \tag{5.24}$$

where $A(N) = h^2$, $A(2N) = h$, $A(3N) = 1$, $A(3N+1) = h^3$, and $B(N) = 2h$, $B(2N) = 1$, $B(3N+1) = 3h^2$, as well as $C(2N+1) = 1$ and $D(N+1) = 1$. Notice that matrix equation given in (5.23) has a unique solution, i.e., the coefficient matrix in this equation is nonsingular; since it can be shown that in this matrix for each row $|a_{ii}| > \sum_{i \neq j} |a_{ij}|$ where a_{ij} denotes the element in row i and column j.

5.3.2 Optimal Control Problem of LTV Systems

In the proposed method, we require to solve the OCP for a sequence of LTV systems; hence in this subsection it is considered to use in the next sections.

Consider an OCP, denoted by $\Sigma 1$, of the form

$$\Sigma 1: \qquad \min_{u \in \mathbf{U}} J(u), \tag{5.25}$$

where $\mathbf{U} \subset L^2([0,T], \mathbb{R}^{2n})$ is the feasible set of controls u and $J(u) \in \mathbb{R}$ is the objective function defined by

$$J(u) = \phi(x(T), T) + \int_0^T L(x, u, t)\, dt, \tag{5.26}$$

subject to

$$\text{System dynamics constraint:} \qquad \dot{x}(t) = A(t)\, x(t) + B(t)\, u(t), \tag{5.27a}$$

$$\text{Boundary conditions:} \qquad x(0) = x_0, x(T) = x_T, \tag{5.27b}$$

$$\text{State and control constraints:} \qquad G(x, u) \leq 0, \tag{5.27c}$$

where $x \in L^2([0,T], \mathbb{R}^{2n})$, $A \in \mathbb{R}^{2n \times 2n}$ and $B \in \mathbb{R}^{2n \times m}$ so that vectors x and u are state and control input of the system, respectively. Moreover, in this OCP, the final state is fixed, hence the final time T is assumed to be free. We also considered $\mathbb{R}^{2n}$ to match with the the number of robot degree of freedom which is n and so dimension of the state vector of the system which consists of joint positions and velocities is a $2n$-vector.

In the case of unconstrained form of the above problem and provided that the cost function (5.26) be in quadratic form (see section)

$$J = \|x(T)\|_H^2 + \int_0^T \left[\|x(t)\|_{Q(t)}^2 + \|u(t)\|_{R(t)}^2 \right] dt, \tag{5.28}$$

then, the optimal control can be obtained as

$$u^*(t) = -R^{-1}(t)\, B^T(t)\, P(t)\, x(t), \tag{5.29}$$

where $P(t)$ is the solution of the standard *Riccati* equation

$$\begin{aligned} \dot{P}(t) = -Q(t) - P(t)A(t) - A(t)P(t) + P(t)B(t)R^{-1}(t)B(t)P(t), \\ P(T) = H. \end{aligned} \tag{5.30}$$

However, in this method there are a series of limitations which makes it impossible to use in the case of robot manipulators. The most important problem is that the optimal solution (5.29) has been obtained for the unconstrained problems; while in the case of robot manipulators there are a few of control and state constraints. Therefore, for robot applications in order to loose the stated limitations one can solve the problem $\Sigma 1$ using parametric optimization method [31; 41; 43; 96].

Note that in this chapter, we usually consider the time-energy OCP. On the other hand, when the robot dynamics is linearized by system (5.27a) in each trial, the state variable x contains two general parts $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} q \\ \dot{q} \end{bmatrix}$, where q denotes the joint position vector of the robot. Therefore, the problem $\Sigma 1$ is converted into the following form:

$$\Sigma 2: \qquad \min_{u \in \mathbf{U}} J(u), \tag{5.31}$$

where the objective function defined by

$$J(u) = \frac{1}{2}\int_0^{T=\alpha t_f} \left(u^T(t)\, Ru(t)\right) dt, \tag{5.32}$$

subject to

$$\text{System dynamics constraint:} \quad \dot{x} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = A(t) \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + B(t)\, u(t), \tag{5.33a}$$

$$\text{Boundary conditions:} \quad x(0) = x_0, x(T) = x_T, \tag{5.33b}$$

$$\text{Position Constraint:} \quad \max_{t\in[0,T]} |x_1| \leq PC, \tag{5.33c}$$

$$\text{Velocity Constraint:} \quad \max_{t\in[0,T]} |x_2| \leq VC, \tag{5.33d}$$

$$\text{Acceleration Constraint:} \quad \max_{t\in[0,T]} |\dot{x}_2| \leq AC, \tag{5.33e}$$

$$\text{Jerk Constraint:} \quad \max_{t\in[0,T]} |\ddot{x}_2| \leq JC, \tag{5.33f}$$

$$\text{Control Constraint:} \quad \max_{t\in[0,T]} |u| \leq UC, \tag{5.33g}$$

where the dimension of x, A, B are the same as problem $\Sigma 1$ and PC, VC, AC, JC are n-vector of constraints on joint positions, velocities, accelerations and jerks, respectively, as well as UC is a m-vector of constraints on the force/torque of robot joints. Also since the final state is fixed, hence the final time T is free and actually we require to compute it as minimum time. Therefore, we can parametrize it as $T = \alpha t_f$, where t_f can be given for the different systems.

The OCP $\Sigma 1$ is a infinite optimization problem. It means that subspace $\mathbf{U}$ is a infinite dimensional of control functions. In order to convert the problem $\Sigma 1$ into a finite dimensional optimization problem, one can parameterize the state or control variables of the system by an appropriate mathematical tools. As explained in the subsection 5.3.1, this tool can be cubic spline functions which are characterized by $N-1$ parameters, where N is the number of subpolynomials in the spline function. Let us denote a spline function in the interval $[t_{i-1}, t_i]$ as $S_i(t; P)$. Now, if we construct the x_1 part of the system state in (5.33a) by spline function $S(t; P)$, then control u can be obtained in terms of cubic spline function $S(t; P)$ via the system dynamics (5.33a). Substituting the reconstructed state and control variables into objective function (5.28) and constraints (5.27b) and (5.27c), the OCP $\Sigma 1$ is converted into following optimization

problem, denoted by Σ_P:

$$\Sigma_P : \qquad \min_{\alpha,P} \sum_{i=1}^{N} J_i, \tag{5.34}$$

where

$$J_i = \int_{h_i=t_{i-1}-t_i} \left(u_i^T R u_i\right) dt, \tag{5.35}$$

such that $T = \sum_{i=1}^{N} h_i$ and subject to the following constraints:

$$\text{Boundary conditions:} \qquad S_1(0) = x_0, S_N(T) = x_T, \tag{5.36a}$$

$$\text{Position Constraint:} \qquad \max_{t\in[t_{i-1},t_i]} |S_i(t;P)| \leq PC, \tag{5.36b}$$

$$\text{Velocity Constraint:} \qquad \max_{t\in[t_{i-1},t_i]} \left|\dot{S}_i(t;P)\right| \leq VC, \tag{5.36c}$$

$$\text{Acceleration Constraint:} \qquad \max_{t\in[t_{i-1},t_i]} \left|\ddot{S}_i(t;P)\right| \leq AC, \tag{5.36d}$$

$$\text{Jerk Constraint:} \qquad \max_{t\in[t_{i-1},t_i]} |\dddot{S}_i(t;P)| \leq JC, \tag{5.36e}$$

$$\text{Control Constraint:} \qquad \max_{t\in[t_{i-1},t_i]} |u_i| \leq UC, \tag{5.36f}$$

for $i = 1, 2, \ldots, N$.

In fact, Σ_P is an optimization problem with N degrees of freedom contains $N-1$ parameters in vector P and α. The resulted finite dimensional optimization problem Σ_P can be solved through standard, nonlinear programming (NLP) techniques, as explained in section 3.5. Therefore, with solving Σ_P, N parameters are produced by which the state and control variables of the system are described and in such a way the constrained OCP $\Sigma 2$ is solved.

5.3.3 Proposed Algorithm

Let us now combine the explained techniques, i.e. iterative linearization, ILC and parametric optimization methods to solve the OCP of the robot manipulators.

First, consider a robot with following dynamics which performs a special repeated task

$$M(q)\ddot{q} + V(q,\dot{q}) + G(q) = \tau, \tag{5.37}$$

with the boundary conditions

$$\begin{aligned} q(0) = q_0, \dot{q}(0) = qd_0, \\ q(T) = q_f, \dot{q}(T) = qd_f. \end{aligned} \tag{5.38}$$

By considering the following states:

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} q \\ \dot{q} \end{bmatrix}, \tag{5.39}$$

the state space representation of the robot can be written as follows (as explained in section (3.4.1))

$$\begin{aligned} &\dot{x}(t) = f(x(t)) + g(x(t))\tau(t), \quad x(0) = x_0, \quad x(T) = x_T, \\ &f(x) = \begin{bmatrix} x_2 \\ -M^{-1}(x_1) N(x_1, x_2) \end{bmatrix}, g(x) = \begin{bmatrix} 0 \\ M^{-1}(x_1) \end{bmatrix}, \end{aligned} \tag{5.40}$$

which can be rewritten as the following SDC form:

$$\dot{x} = A(x) x + B(x) u, \tag{5.41}$$

where, without loss of generality, $x_e = 0$ is its equilibrium point and

$$A(x) = \nabla_x f(x), \; B(x) = g(x), \; u = \tau. \tag{5.42}$$

Then the matrices A and B are achieved as follows

$$A(x) = \begin{bmatrix} Z_{n\times n} & I_{n\times n} \\ -\bigtriangledown_x (M^{-1}(x_1) N(x_1, x_2)) \end{bmatrix}, \quad B = \begin{bmatrix} Z_{n\times n} \\ -M^{-1}(x_1) \end{bmatrix}, \tag{5.43}$$

where Z and I are zero and identity matrices, respectively, with the specified dimensions. Now we are going to obtain the optimal control of the considered robot which is performing the desired repeated task. Therefore, the following linear dynamics is considered as the model of the robot in the first trial

$$\dot{x}^{[1]}(t) = A\left(x^{[0]}(t)\right) x^{[1]}(t) + B\left(x^{[0]}(t)\right) u^{[1]}(t), \quad x^{[1]}(0) = x_0, \; x^{[1]}(T) = x_T, \tag{5.44}$$

as well as the cost functional considered in this trial is

$$J^{[1]} = \phi\left(x^{[1]}(T), T\right) + \int_0^T L\left(x^{[1]}(t), u^{[1]}(t), t\right) dt. \tag{5.45}$$

where can be as the form cost function (5.32) or any other form.

As explained in the previous subsection, the above OCP can be solved by parameterizing states of the system by spline functions $S(t;P)$. After solving the obtained parametric optimization problem, the optimal parameter matrix $P^{*[1]}$ is obtained for the first trial. Accordingly, the optimal control of the first trial is obtained as follows

$$u^{[1]}(t) = \left[Z_{n\times n} \quad M\left(x_1^{[0]}(t)\right)\right]\left(\begin{bmatrix}\dot{x}_1^{[1]}(t)\\ \dot{x}_2^{[1]}(t)\end{bmatrix} - \begin{bmatrix} Z_{n\times n} & I_{n\times n} \\ -\nabla_x\left(M^{-1}\left(x_1^{[0]}\right)N\left(x_1^{[0]}, x_2^{[0]}\right)\right) & \end{bmatrix}\begin{bmatrix}x_1^{[1]}\\ x_2^{[1]}\end{bmatrix}\right), \tag{5.46}$$

where $x_1^{[1]}(t) = S\left(t;P^{*[1]}\right)$ and $x_2^{[1]}(t) = \dot{x}_1^{[1]}(t) = \dot{S}\left(t;P^{*[1]}\right)$. In addition, let us define an error variable as

$$e^{[1]}(t) = u^{*[1]}(t) - u^{*[0]}(t), \tag{5.47}$$

where it is assumed $u^{*[0]}(t) = 0$. The optimal state $x^{*[1]}$ and control $u^{*[1]}$ together with $e^{[1]}$ are stored in memory of the system. Other variable stored in memory of the system from first trial is first order optimality, denoted $\delta^{[1]}$. It is a variable produced by nonlinear programming algorithm which actually shows the variation of the cost functional, i.e. $\delta J^{[1]}$. If $u^{*[1]}$ is optimal solution, then $\delta^{[1]}$ must vanish on $u^{*[1]}$.

As such in the first trial, the above procedure is performed in the subsequent trials so that in the trial i-th, the optimal state and control of trial $(i-1)$-th is used

$$\dot{x}^{[i]}(t) = A\left(x^{[i-1]}(t)\right)x^{[i]}(t) + B\left(x^{[i-1]}(t)\right)u^{[i]}(t), \quad x^{[i]}(0) = x_0,\ x^{[i]}(T) = x_T, \tag{5.48}$$

with considering the following cost functional in this trial:

$$J^{[i]} = \phi\left(x^{[i]}(T), T\right) + \int_0^T L\left(x^{[i]}(t), u^{[i]}(t), t\right) dt. \tag{5.49}$$

Note that in this iteration, the control input is

$$u^{[i]}(t) = \begin{bmatrix} Z_{n\times n} & M\left(x_1^{[i-1]}(t)\right) \end{bmatrix} \left(\begin{bmatrix} \dot{x}_1^{[i]}(t) \\ \dot{x}_2^{[i]}(t) \end{bmatrix} - \begin{bmatrix} Z_{n\times n} & I_{n\times n} \\ -\bigtriangledown_x \left(M^{-1}\left(x_1^{[i-1]}\right) N\left(x_1^{[i-1]}, x_2^{[i-1]}\right)\right) \end{bmatrix} \begin{bmatrix} x_1^{[i]} \\ x_2^{[i]} \end{bmatrix} \right), \tag{5.50}$$

and accordingly, the error variable is

$$e^{[i]}(t) = u^{*[i]}(t) - u^{*[i-1]}(t), \tag{5.51}$$

which together with the first order optimality of this iteration, i.e. $\delta^{[i]}$, are stored in memory of the system.

Let us now define two predetermined positive constants ε_1, ε_2 which are close to zero and are used as the stop criteria of the proposed algorithm.

Here, we present the **proposed algorithm** whose steps are listed as follows (see Figure 5.2):

1. Obtain the state space representation and then the SDC form of the considered robot manipulator system.
2. Get the initial and final configurations (q_0, q_T);
3. Guess an arbitrary state $x^{[0]}(t)$, $t \in [0, T]$, and store it in memory of the system. Let the iteration index i be one.
4. Using $x^{[i-1]}(t)$ and utilizing spline-based optimal control technique explained in the previous subsection, compute the optimal force/torque vector $u^{*[i]}(t)$ and optimal state vector $x^{*[i]}(t)$ of the LTV system in step i represented in (5.48) given a cost functional, physical constraints of the robot and boundary conditions. Also store $x^{*[i]}(t)$ and $u^{*[i]}(t)$ in memory of the system together with $\delta^{[i]}$.
5. Apply $u^{[i]}(t)$ to the ith trial.
6. If
$$\left\| e^{[i]}(t) \right\| \leq \varepsilon_1 \quad \text{and} \quad \delta^{[i]} \leq \varepsilon_2, \tag{5.52}$$

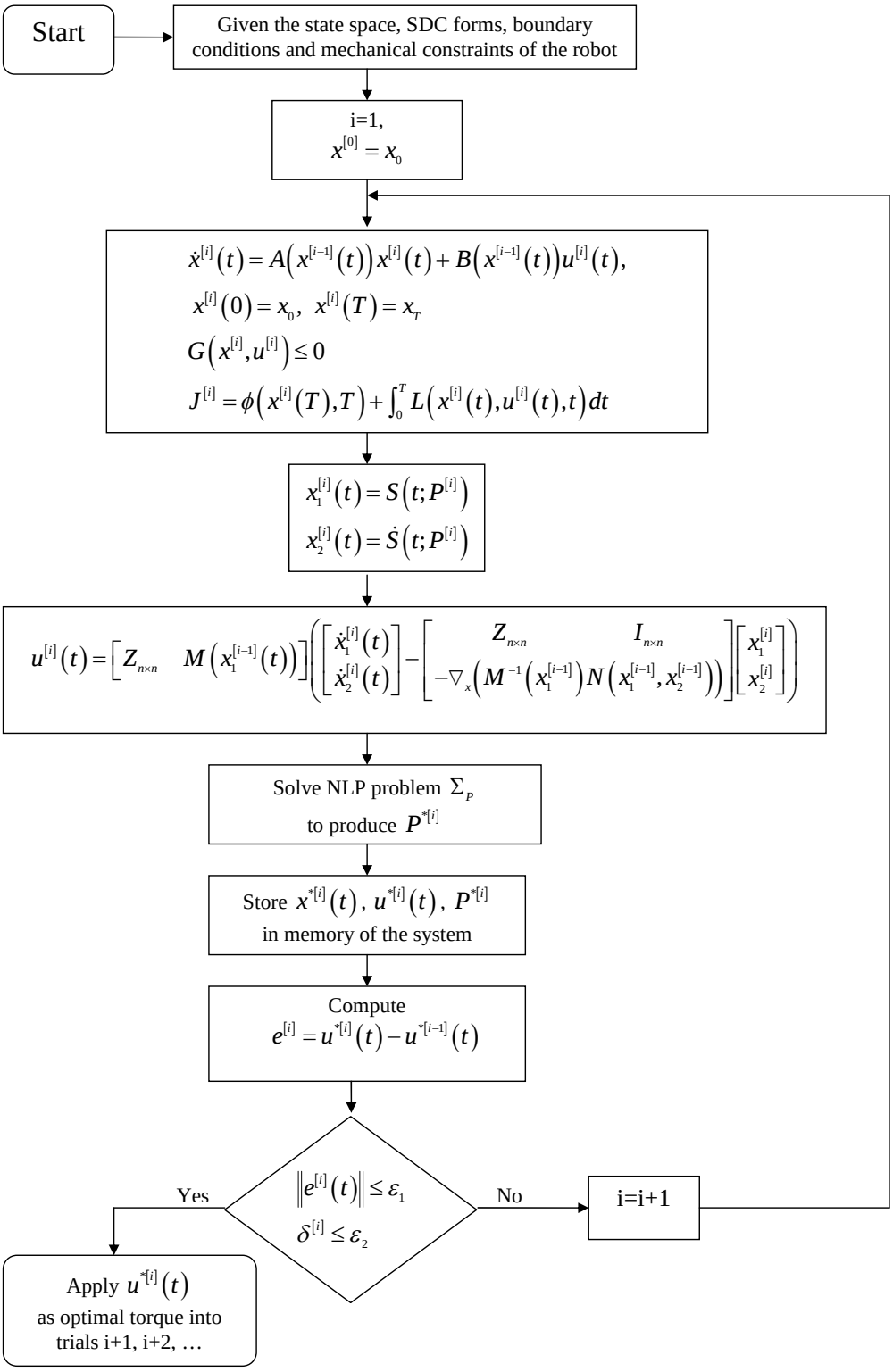


Figure 5.2: **Flowchart of the proposed method**

Table 5.1: Denavit-Hartenberg parameters of SCARA robot

Link i	θ_i	d_i	a_i	α_i
1	θ_1	0	L_1	0
2	θ_2	0	L_2	0
3	0	d_3	0	0

then terminate the computations and $u^{[i]}(t)$ can be used for the next trials. If stopping criteria given in (5.52) is not satisfied, then $i = i + 1$ and return to step 4.

In the next section, the proposed algorithm shall be applied into all standard kinds of manipulator structures and the different features of the proposed method like convergence property are discussed in detail.

5.4 Case studies

In this section, the proposed method is applied into various kinds of case studies. In order to compare the proposed method with some other methods, we make some comparisons between the proposed method with the spline-based optimal control and multiple shooting methods represented in sections 3.6 and 3.7, respectively, which are practical and usual methods in the case of robot manipulators.

5.4.1 SCARA Robot

Let us consider the SCARA robot manipulator depicted in Figure 5.3 which has three degrees of freedom. In subsection 2.2.1, the Euler-Lagrange formulation was given in detail to derive the dynamic equations of a robot manipulator. We now apply this formulation to derive the dynamics of this SCARA robot. We first require to obtain a kinematic model of this robot. Thus, according to the Denavit-Hartenberg notation we can assign an appropriate coordinate system for each link of the robot, as shown in Figure 5.3. According to these coordinate systems, the Denavit-Hartenberg parameters are obtained as given in Table 5.1; hence, the transformation matrices are obtained as follows

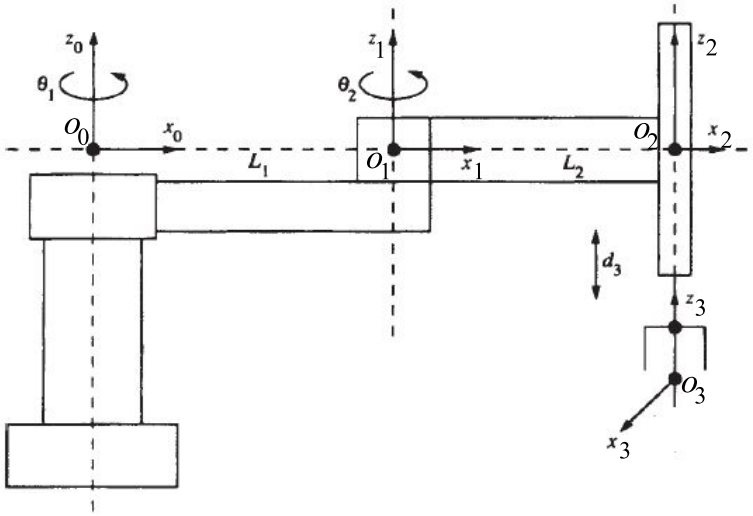


Figure 5.3: **SCARA robot**

$$ {}^0A_1 = \begin{bmatrix} C1 & -S1 & 0 & L_1C1 \\ S1 & C1 & 0 & L_1S1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \; {}^1A_2 = \begin{bmatrix} C2 & -S2 & 0 & L_2C2 \\ S2 & C1 & 0 & L_1S1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \; {}^2A_3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{5.53} $$

where Si, Ci (for $i = 1, 2$) stand for functions $\sin\theta_i$ and $\cos\theta_i$, respectively. Let us now consider the general coordinates $\begin{bmatrix} q_1 & q_2 & q_3 \end{bmatrix}^T = \begin{bmatrix} \theta_1 & \theta_2 & d_3 \end{bmatrix}^T$ and vector of corresponding generalized forces be $\tau = \begin{bmatrix} \tau_1 & \tau_2 & \tau_3 \end{bmatrix}^T$. We can now obtain the dynamic equations of this robot by using equations (2.13) to (2.19) in subsection 2.2.1 according to the above transformation matrices:

$$ \begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \end{bmatrix} \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \\ \ddot{q}_3 \end{bmatrix} + \begin{bmatrix} N_1 \\ N_2 \\ N_3 \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \tau_2 \\ \tau_3 \end{bmatrix}, \tag{5.54} $$

Table 5.2: Dynamic parameters of SCARA robot

Link i	$m\ (kg)$	$I_{xx},\ I_{yy},\ I_{zz}\ (kg\,m^2)$	$I_{xy},\ I_{xz},\ I_{yz}\ (kg\,m^2)$
1	1.98	0.00048 0.0179 0.017	0 0 0
2	0.9177	0.00047 0.00348 0.00314	0 0 0
3	0.703	0.0021 0.0021 0.00011	0 0 0

where

$$
\begin{aligned}
&M_{11} = (I_{zz1} + I_{zz2} + I_{zz3}) + \left(\frac{m_1}{4} + 2m_3\right) L_1^2 + \left(\frac{m_2}{2} + 3m_3\right) L_1 L_2 C2 + \left(\frac{m_2}{4} + m_3\right) L_2^2,\\
&M_{12} = M_{21} = I_{zz2} + I_{zz3} + \left(\frac{m_2}{4} + m_3\right) L_2^2 + 2m_3 L_1 L_2 C2,\\
&M_{22} = I_{zz2} + I_{zz3} + \left(\frac{m_2}{4} + m_3\right) L_2^2, \quad M_{33} = m_3,\\
&M_{13} = M_{23} = M_{31} = M_{32} = 0,\\
&N_1 = \left(\frac{m_2}{2} - m_3\right) L_1 L_2 \dot{q}_1^2 - 4m_3 L_1 L_2 \dot{q}_1 \dot{q}_2,\\
&N_2 = \left(\frac{m_2}{2} + m_3\right) L_1 L_2\, S2\, \dot{q}_1^2,\\
&N_3 = m_3\, g.
\end{aligned}
\tag{5.55}
$$

Note that in this case study we use the dynamic parameters given in [1] listed in Table 5.2. In addition, we consider $L_1 = 0.25\,m$, $L_2 = 0.15\,m$ and $d_3 = 0.075\,m$. Also notice that the equations (5.54) and (5.55) result in a highly and coupled nonlinear dynamic system for this robot with relatively simple structure. These equations will be complex to ever for robots with 6 degrees of freedom angular structures so that anyone who derived these equations can release this complexity.

Let us now obtain the state space representation of this robot by considering the

following states:

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix}, \qquad \dot{x} = f(x) + g(x)\tau, \tag{5.56}$$

where

$$f(x) = \begin{bmatrix} x_4 \\ x_5 \\ x_6 \\ -M^{-1}(x_1, x_2, x_3) N(x) \end{bmatrix}, \quad g(x) = \begin{bmatrix} Z_{3\times 3} \\ M^{-1}(x_1, x_2, x_3) \end{bmatrix}, \tag{5.57}$$

where $Z_{3\times 3}$ is a 3×3 zero matrix.

Moreover, the considered constraints and cost functional for this robot are as follows:

- constraints:

$$\begin{aligned} &|\tau_1| \leq 1\,(Nm),\ |\tau_2| \leq 1\,(Nm),\ |\tau_3| \leq 10\,(N), \\ &|x_4| \leq 2\,(rad/sec),\ |x_5| \leq 4\,(rad/sec),\ |x_6| \leq 1\,(m/sec), \end{aligned} \tag{5.58}$$

- boundary conditions:

$$\begin{aligned} &x_1(0) = 0,\ \ x_2(0) = 0,\ \ x_3(0) = 0, \\ &x_4(0) = 0,\ \ x_5(0) = 0,\ \ x_6(0) = 0, \\ &x_1(T) = \frac{\pi}{2}\, rad,\ \ x_2(T) = \frac{\pi}{4}\, rad,\ \ x_3(T) = 0.1\, m, \\ &x_4(T) = 0,\ \ x_5(T) = 0,\ \ x_6(T) = 0, \end{aligned} \tag{5.59}$$

- cost function:

$$J = \int_0^T \tau^T R \tau dt, \tag{5.60}$$

where

$$R = \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix}, \tag{5.61}$$

with $\lambda_1 = 1$, $\lambda_2 = 1$, $\lambda_3 = 0.001$.

As stated earlier, this OCP problem is solved by multiple shooting, spline-based optimal control and proposed methods:

- Multiple shooting method

  In order to solve the above OCP for this SCARA robot by multiple shooting method, one has to solve a nonlinear programming with many virtual constrains created by this method as explained in section 3.7. On the other hand, in order to realize a control signal in practice it must be continuous and it is a considerable drawback of the multiple shooting method which can not generate completely continuous controls. However, to obtain a semi-continuous optimal control we need to consider a large number for N which causes a too high computation time to solve its nonlinear programming. The Figure 5.4 shows the optimal profiles of the joint positions, velocities and torques of this robot obtained by this method. Moreover, the optimal data of this solution have been collected in Table 5.3. The first row of this table shows the number of SQP iterations needed to find an optimal solution in the used nonlinear programming algorithm (developed in optimization toolbox of MATLAB). The second row presents the number of math operations including performed summations, subtractions, multiplications and divisions to find the optimal solution. The third row presents the minimum of cost function and in the fourth row we have the minimum time during which the robot performs desired movement. The fifth row of this table (i.e. first order optimality) is a measure which shows how close the obtained solution is to optimal one. If it is less than a predefined small number for instance $\varepsilon = 0.0001$,

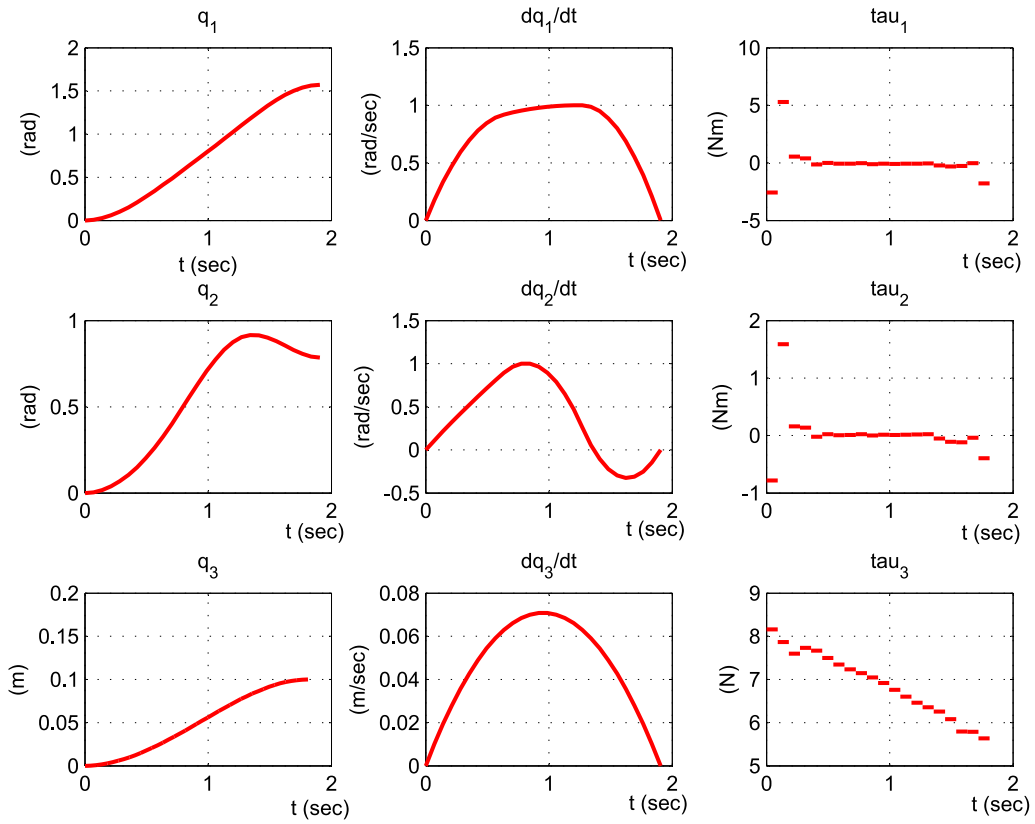


Figure 5.4: **Optimal profiles of SCARA robot by multiple shooting method**

the optimal finder algorithm is terminated. Eventually, the last row presents the time to compute the optimal solution by the computer. Note that all of the computations are carried out on a computer system of $2.2GHz$ CPU.

- Spline based optimal control method:

  As explained in the chapter 3 in section 3.6, this method solves the OCP of robots directly by parameterizing the joint dispositions of the robot by spline functions. Figure 5.5 shows the optimal profiles (joint dispositions, velocities and torques) obtained by this method for the above SCARA robot. Also the optimal data derived from this method are given in Table 5.4. These data have been computed for $N = 5$.

- Proposed method

Table 5.3: Optimal data of SCARA robot obtained from multiple shooting method

Number of SQP iterations	39
Number of math operations	5113
J_{min}	56.77
T_{min}	1.892
First order Optimality (δ)	1.84
Computation time (sec)	573

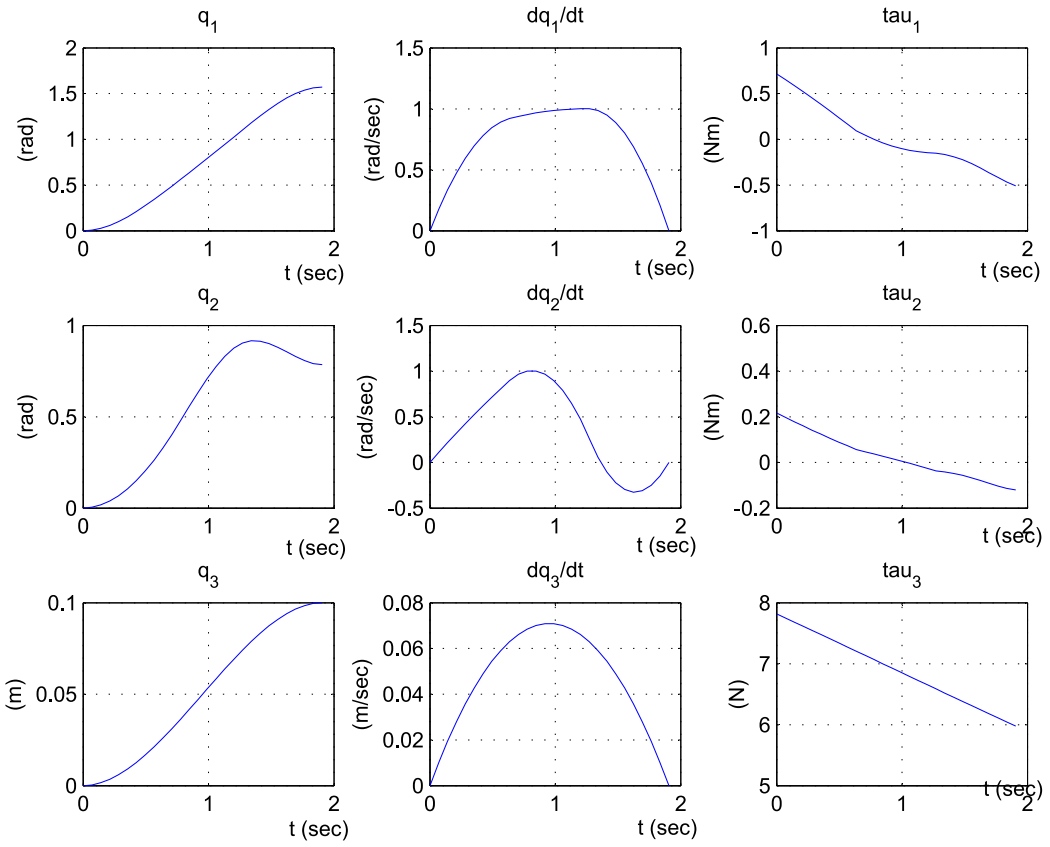


Figure 5.5: **Optimal profiles of SCARA robot by spline based optimal control method**

Table 5.4: Optimal data of SCARA robot obtained from spline based optimal control method

Number of SQP iterations	48
Number of math operations	382
J_{min}	55.37
T_{min}	1.892
First order optimality	2.64×10^{-5}
Computation time (sec)	3.86

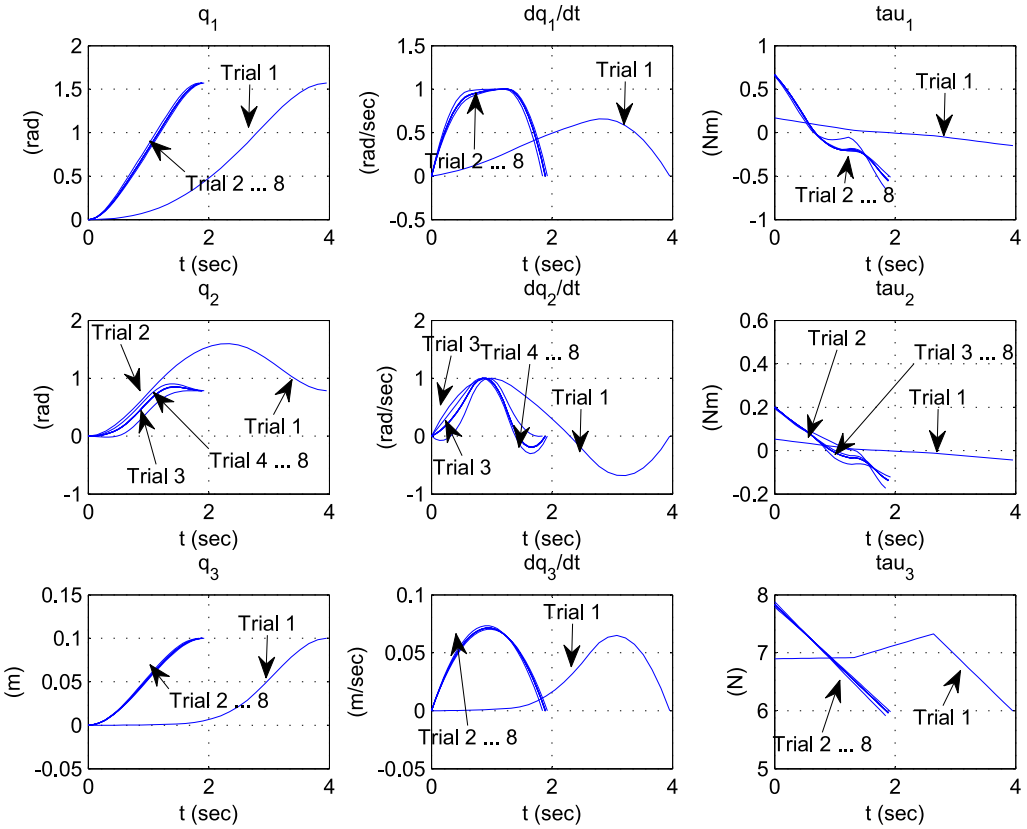


Figure 5.6: **Optimal profiles of SCARA robot by proposed method**

Here, the proposed method is applied into solve the OCP of SCARA robot. Figure 5.6 shows the optimal profiles obtained by this method including joint dispositions, velocities and torques of three joints of the robot. Also Table 5.5 presents some information about the optimal data obtained by this method. The first column of this table shows the number of trials during which the optimal solutions are obtained. We explained about the parameters in the other columns except the last three columns. These three columns present the norm of error signals in each trial defined as $\left\|e_j^{[i]}\right\| = \sup_{t\in[0,T]} \left|\tau_j^{[i]} - \tau_j^{[i-1]}\right|$ (for $j = 1, 2, 3$). For this robot the error signals are depicted in Figure 5.7 by which the data given in three last columns of the Table 5.5 are obtained. If we wish to have a convergence analysis on the sequences of these errors we can use the following theorem and remark derived from calculus of sequences

Theorem. Let the sequence $\left\{\zeta\,\mu^k\right\}_{k=0}^{\infty}$ where ζ is a constant. This sequence converges linearly to zero with rate μ if $|\mu| < 1$.

Remark. If $\{a_k\}_{k=0}^{\infty}$ be a sequence and $a_k \leq \zeta\,\mu^k$, then $\{a_k\}$ converges to zero with at most rate μ.

Thus, according to above theorem and remark, the sequence of $\left\|e_1^{[i]}\right\|$ converges linearly to zero with the rate $\mu_1 = \frac{1}{2}$ because the values in this row satisfy $\left\|e_1^{[i]}\right\| \leq \zeta_1\,(1/2)^i$. Also the sequence of $\left\|e_2^{[i]}\right\|$ converges linearly to zero with rate $\mu_2 = \frac{1}{6}$; since the elements of this sequence satisfies $\left\|e_2^{[i]}\right\| \leq \zeta_2\,\left(\frac{1}{6}\right)^i$. In the case of $\left\|e_3^{[i]}\right\|$, it converges to zero with rate $\mu_3 = \frac{1}{9}$.

Let us now make some comparisons between the proposed method and other two methods to solve the time-energy OCP of the above SCARA robot:

- Proposed method and multiple shooting method:

  As explained in section 3.7, multiple shooting algorithm results in an optimization programming problem with a large number of degrees of freedom (i.e. number of variables to be computed optimally) and also a large number of virtual constraints together with physical constraints of the robot which cause a high computation time needed to solve this optimization programming problem. On the other hand, we have to increase the number of divisions in the piecewise

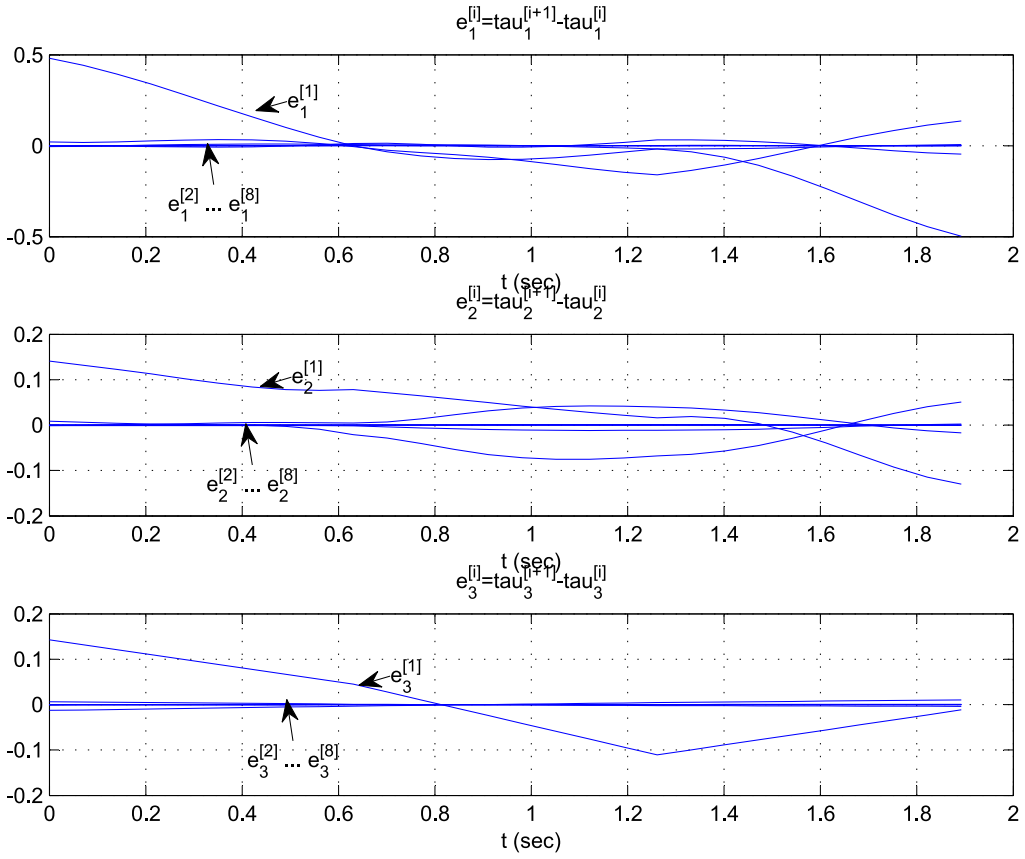


Figure 5.7: **Sequence of errors in the successive trials in proposed method for SCARA robot**

control to yield a semi-continuous control which, in order, causes increasing the number of parameters in optimization programming and also the number of constraints. For example, for $N = 20$ (number of discretization of control signal) we will have 175 parameters to find their optimality and 136 constraints so that according to Table 5.3 the computation time will be 573 sec required to perform 5113 math operations. While, in the proposed method in each trial, for $N = 5$ (number of sub-polynomials of used spline function) we have 13 parameters to find their optimal values and just physical constraints on the robot to satisfy. Thus, according to Table 5.5, in the proposed method we averagely have 83 math operations to perform which needs averagely 1.6 sec to compute the optimal solution. In addition, as explained earlier, in the proposed method the control signal

Table 5.5: Optimal data of SCARA robot obtained by applying proposed method

Trial	J_{min}	T_{min}	First order optimality (δ)	Number of math operations	time of computation	$\left\|\left\|e_1^{[i]}\right\|\right\|$	$\left\|\left\|e_2^{[i]}\right\|\right\|$	$\left\|\left\|e_3^{[i]}\right\|\right\|$
1	95.61	3.954	41.5	118	2.63	-	-	-
2	75.225	1.892	0.55	102	2.14	0.5	0.1	0.1
3	56.605	1.892	0.08	84	1.53	0.1	0.02	0.01
4	56.07	1.892	0.00301	82	1.46	0.04	0.003	5.17×10^{-3}
5	55.87	1.892	0.0014	78	1.33	0.01	3.7×10^{-4}	1.6×10^{-4}
6	55.65	1.892	3.2×10^{-4}	75	1.3	1.5×10^{-3}	3.2×10^{-4}	4.64×10^{-5}
7	55.39	1.892	5.14×10^{-5}	73	1.28	2.14×10^{-4}	4.15×10^{-6}	3.44×10^{-7}
8	55.3	1.892	1.63×10^{-7}	69	1.14	5.73×10^{-6}	2.64×10^{-9}	2.77×10^{-8}

is a function of spline functions and since spline functions are continuous then the control signals are continuous too. Of course, as explained before, in the proposed method since the optimization procedure is divided during successive trials then these good characterizations are achieved with respect to multiple shooting method.

- Proposed method and spline based optimal control method:

  First of all, note that we considered $N = 5$ in these two methods and our computations showed that for $N > 8$ there no exists any optimal solution to the OCP of this robot.

  Also in the proposed method, the control is a linear function in $S(t; P)$, hence the resulted optimization programming will be a quadratic from (since we attempt to minimize (5.32)) and quadratic optimization is one of the well known forms of optimization problem which can be solved by existing algorithms. On the other hand, in the spline based optimal control method, the control u is a nonlinear function of spline functions (since $u = g^{-1}(S(t; P))\left(\dot{S}(t; P) - f(S(t; P))\right)$) and

also the cost function obtained from this control u is a nonlinear function of parameter vector P. Therefore, we can claim that in the proposed method in each trial we have to solve a simpler optimization problem (quadratic) in contrast to the spline based optimal control method which we require to obtain the optimal solution of a highly nonlinear function. This is one of the benefits of the proposed method which linearize the highly nonlinear robot dynamics. Thus, as Table (5.5) shows the number of math operations averagely is 83 in 8 trials, while the number of math operations for spline based optimal control method is 382, as shown in the Table 5.4 . Accordingly, the average computation time for proposed method in each trial is 1.6 sec, while it is 3.86 sec for the spline based optimal control method. As Figure 5.7 also shows , the optimal solution obtained by proposed method converges to the original one (shown in Figure 5.5) too quickly after 6 trials, since the norm of errors after forth trial is less that 1×10^{-3}.

Thus, as expected the proposed method solves the OCP of the considered robot after a finite number of trials and the load of computations are partitioned on the successive trials. But in the proposed method all computations must be accomplished in one step. Other advantages of the proposed method can be concluded from the above comparisons easily.

5.4.2 Spherical Robot (Stanford Arm)

Now let us consider a spherical robot shown in Figure 5.8 which has three degrees of freedom each of which is responsible for a motion so that location of the robot end-effector can be described by a spherical coordinate system with coordinates (θ_1, θ_2, d_3). In order to obtain the dynamic model of this robot, we again use the Euler-Lagrange formulation presented in subsection 2.2.1. According to Denavit-Hartenberg notation, we assign the desired coordinate systems to the robot links as shown in Figure 5.8 and accordingly the desired parameters are obtained as given in Table 5.6. Thus, the transformation matrices of this robot are

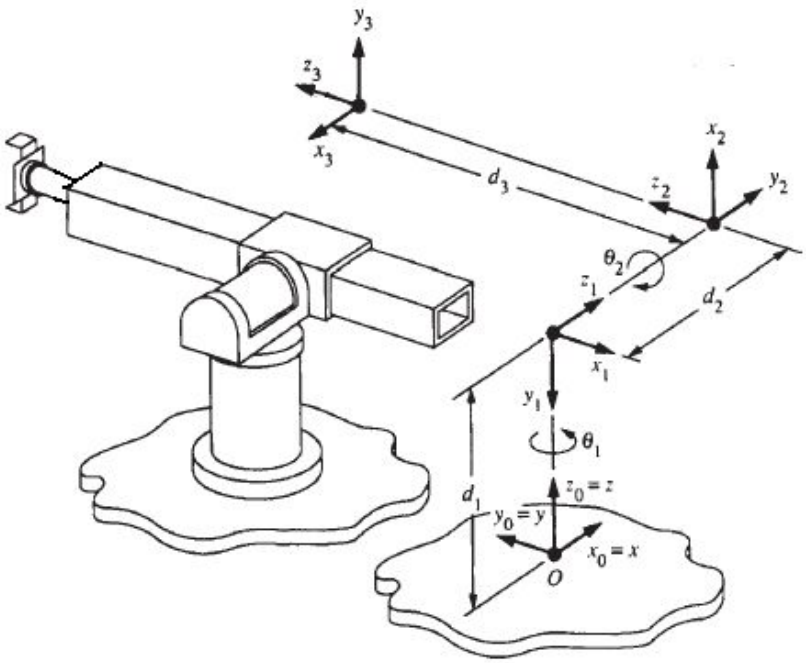


Figure 5.8: **Spherical (Stanford) robot**

Table 5.6: Denavit-Hartenberg parameters of spherical (Stanford) robot

Link i	θ_i	d_i	a_i	α_i
1	θ_1	d_1	0	$-\frac{\pi}{2}$
2	θ_2	d_2	0	$\frac{\pi}{2}$
3	$-\frac{\pi}{2}$	d_3	0	0

$$ {}^0A_1 = \begin{bmatrix} C1 & 0 & -S1 & 0 \\ S1 & 0 & C1 & 0 \\ 0 & -1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \; {}^1A_2 = \begin{bmatrix} C2 & 0 & S2 & 0 \\ S2 & 0 & -C2 & 0 \\ 0 & 1 & 0 & d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \; {}^2A_3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{5.62} $$

where Si, Ci (for $i = 1, 2$) stand for functions $\sin\theta_i$ and $\cos\theta_i$, respectively. Let the general coordinates be $\begin{bmatrix} q_1 & q_2 & q_3 \end{bmatrix}^T = \begin{bmatrix} \theta_1 & \theta_2 & d_3 \end{bmatrix}^T$ and vector of corresponding generalized forces be $\tau = \begin{bmatrix} \tau_1 & \tau_2 & \tau_3 \end{bmatrix}^T$.

We can now obtain the dynamic equations of this robot by using equations (2.13)

Table 5.7: Dynamic parameters of spherical (Stanford) robot

Link i	m (kg)	I_{xx}, I_{yy}, I_{zz} $(kg\,m^2)$	I_{xy}, I_{xz}, I_{yz} $(kg\,m^2)$
1	9.29	0.1724 0.1675 0.0874	0 0 0
2	5.01	0.1468 0.06 0.1413	0 0 0
3	4.25	0.7685 0.7685 0.0376	0 0 0

to (2.19) in subsection 2.2.1 as follows (and according to dynamic parameters (Table 5.7) given in [70]):

$$\begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \end{bmatrix} \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \\ \ddot{q}_3 \end{bmatrix} + \begin{bmatrix} N_1 \\ N_2 \\ N_3 \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \tau_2 \\ \tau_3 \end{bmatrix}, \tag{5.63}$$

where

$$\begin{aligned} &M_{11} = -5.25\,S2^2\,q_3 + 11.5\,d_2^2 + 1.32 + 2.5\,S2^2 + 6.5\,S2^2\,q_3^2 - 1.05\,d_2, \\ &M_{12} = M_{21} = -6.47\,C2\,d_2\,q_3,\ M_{13} = M_{31} = -6.47\,S2\,d_2, \\ &M_{22} = -5.25\,q_3 + 6.5\,q_3^2 + 5.6,\ M_{23} = M_{32} = 0,\ M_{33} = 7.65, \\ &N_1 = 0, \\ &N_2 = -6.5\,S2\,d_3 - 3.2\,S2, \\ &N_3 = 6.5\,g\,C2\,, \end{aligned} \tag{5.64}$$

Let us now obtain the state space representation of this robot by considering the

following states:

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix}, \qquad \dot{x} = f(x) + g(x)\,\tau, \tag{5.65}$$

where

$$f(x) = \begin{bmatrix} x_4 \\ x_5 \\ x_6 \\ -M^{-1}(x_1, x_2, x_3)\, N(x) \end{bmatrix}, \quad g(x) = \begin{bmatrix} Z_{3\times 3} \\ M^{-1}(x_1, x_2, x_3) \end{bmatrix}, \tag{5.66}$$

Moreover, the considered constraints and cost function for this robot are as follows:

- constraints:

$$\begin{aligned} &|\tau_1| \leq 10\,(Nm),\ |\tau_2| \leq 20\,(Nm),\ |\tau_3| \leq 50\,(N), \\ &|x_4| \leq 5\,(rad/sec),\ |x_5| \leq 2\,(rad/sec),\ |x_6| \leq 1\,(m/sec), \end{aligned} \tag{5.67}$$

- boundary conditions:

$$\begin{aligned} &x_1(0) = 0,\ \ x_2(0) = 0,\ \ x_3(0) = 0, \\ &x_4(0) = 0,\ \ x_5(0) = 0,\ \ x_6(0) = 0, \\ &x_1(T) = \frac{\pi}{2}\,rad,\ \ x_2(T) = \frac{\pi}{4}\,rad,\ \ x_3(T) = 0.05\ m, \\ &x_4(T) = 0,\ \ x_5(T) = 0,\ \ x_6(T) = 0; \end{aligned} \tag{5.68}$$

- cost function:

$$J = \int_0^T \left(\tau^T R\,\tau + X_d^T Q\, X_d\right) dt, \tag{5.69}$$

Table 5.8: Optimal data of spherical (Stanford) robot obtained from multiple shooting method

Number of SQP iterations	34
Number of math operations	3013
J_{min}	53.4
T_{min}	1.267
First order Optimality (δ)	2.17
Computation time (sec)	413

where $X_d = [x_4 \;\; x_5 \;\; x_6]^T$ and

$$R = \begin{bmatrix} \alpha_1 & 0 & 0 \\ 0 & \alpha_2 & 0 \\ 0 & 0 & \alpha_3 \end{bmatrix}, Q = \begin{bmatrix} \beta_1 & 0 & 0 \\ 0 & \beta_2 & 0 \\ 0 & 0 & \beta_3 \end{bmatrix}, \tag{5.70}$$

with $\alpha_1 = 1$, $\alpha_2 = 1$, $\alpha_3 = 0.001$, $\beta_1 = 10$, $\beta_2 = 10$, $\beta_3 = 1$.

As with SCARA robot, we solve the OCP of this spherical robot by three methods multiple shooting, spline based optimal control and proposed methods:

- Multiple shooting method:

  In this part we present the optimal results obtained by multiple shooting method for this spherical robot. The optimal joint disposition, velocity and force/torque of each spherical robot joint are demonstrated in Figure 5.9. Some information regarding this OCP are collected in Table 5.8. These date will be used in order to compare this method with other two methods.

- Spline based optimal control method:

  In this item we present the optimal results derived from spline based optimal control method applied into the above spherical robot. Figure 5.10 shows the optimal profiles (joint dispositions, velocities and torques) obtained by this method for this spherical robot with the specifications mentioned above. Also the optimal data derived from this method are given in Table 5.9. These data have been computed for $N = 5$.

- Proposed method:

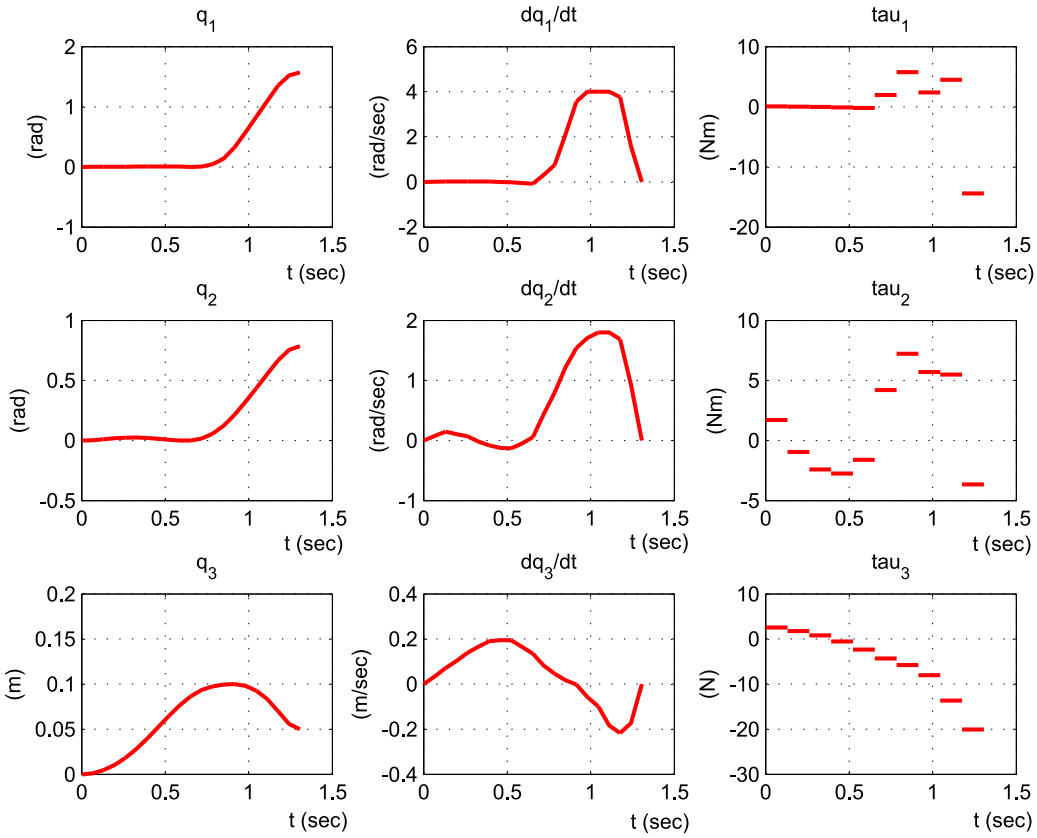


Figure 5.9: **Optimal profiles of spherical (Stanford) robot by multiple shooting method**

Table 5.9: Optimal data of spherical (Stanford) robot obtained from spline based optimal control method

Number of SQP iterations	41
Number of math operations	364
J_{min}	48.1
T_{min}	1.267
First order optimality	$9.33 \times 10_{-6}$
Computation time (sec)	4.74

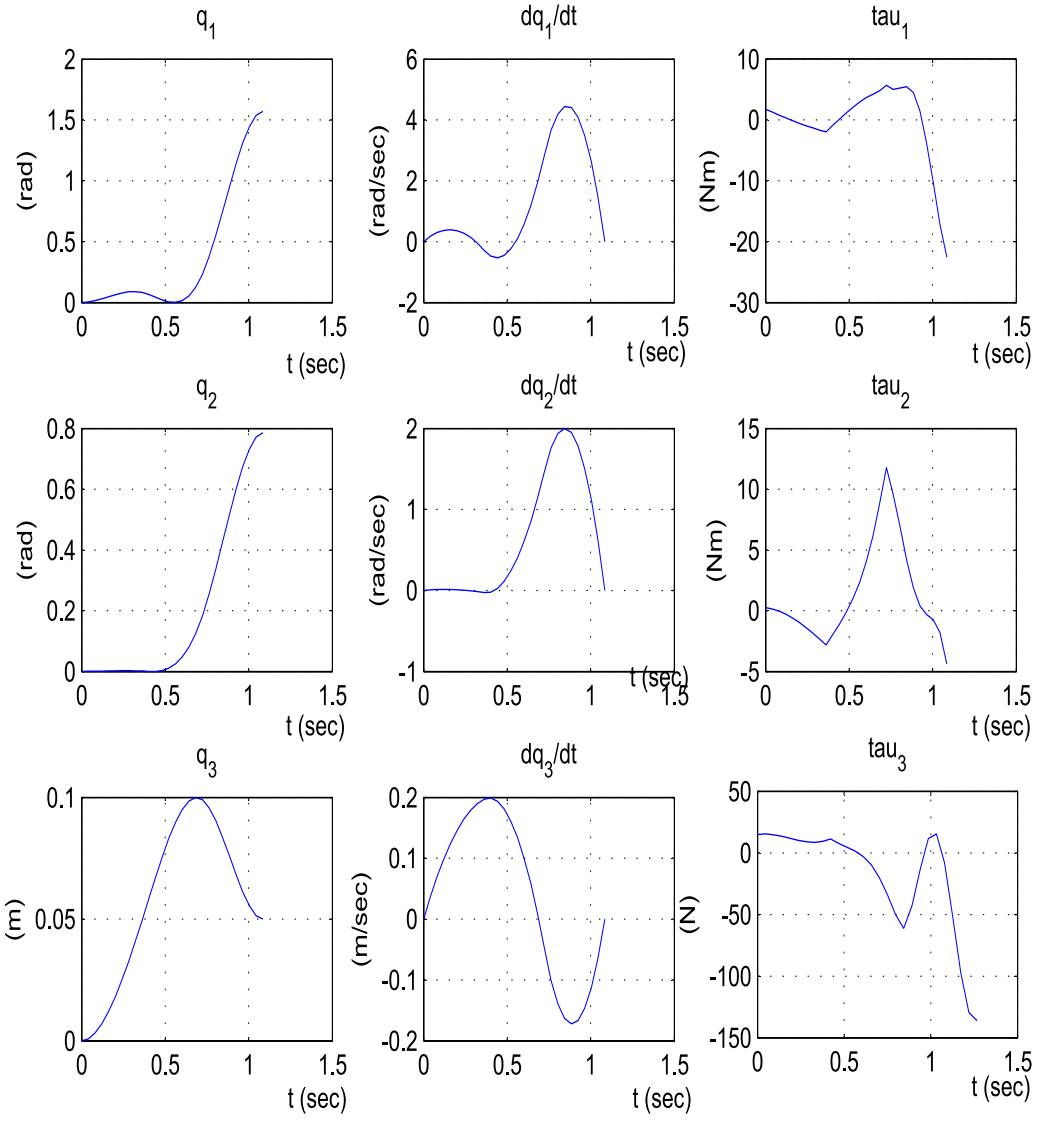


Figure 5.10: **Optimal profiles of spherical (Stanford) robot by spline based optimal control method**

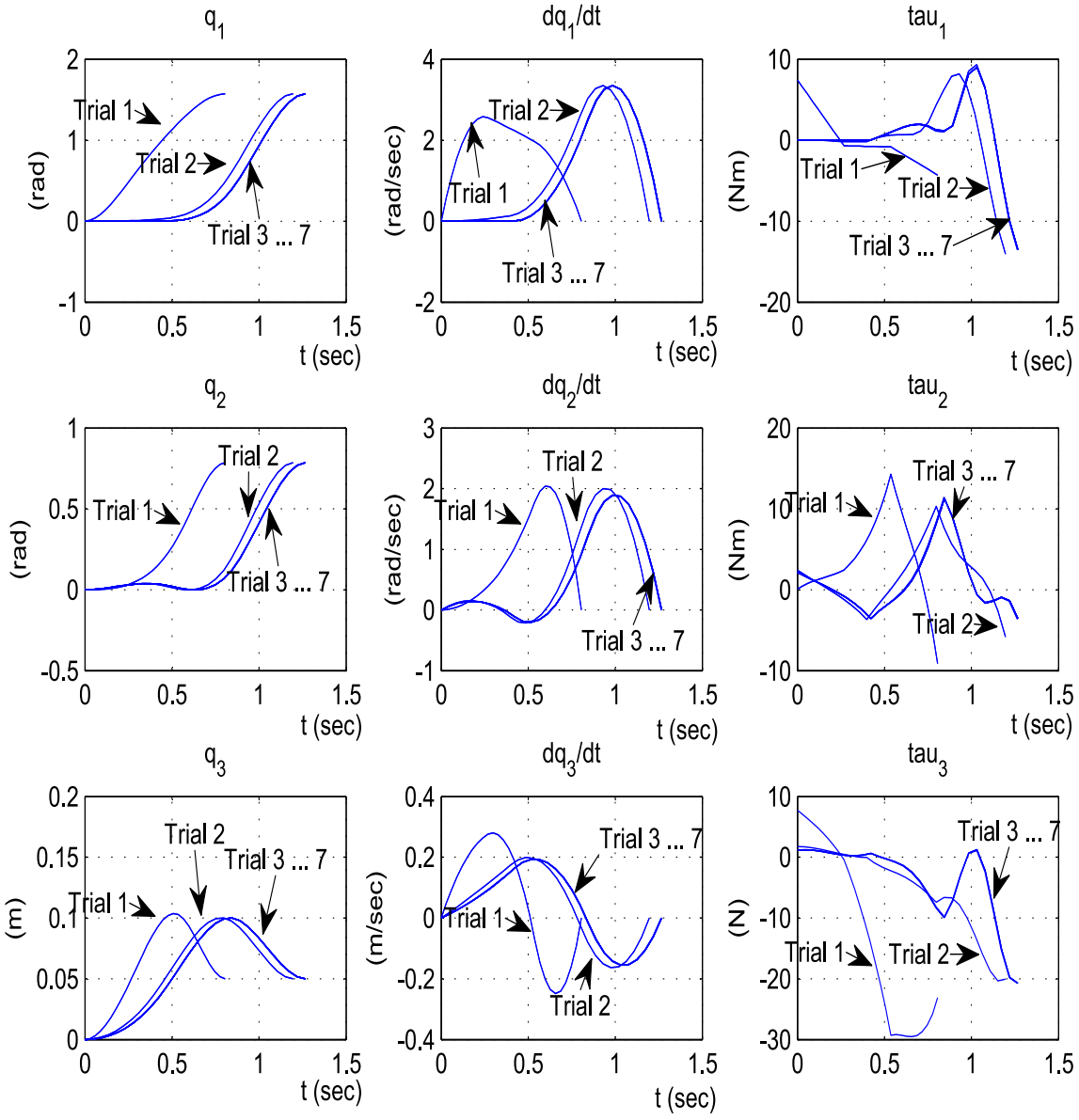


Figure 5.11: **Optimal profiles of spherical (Stanford) robot by proposed method**

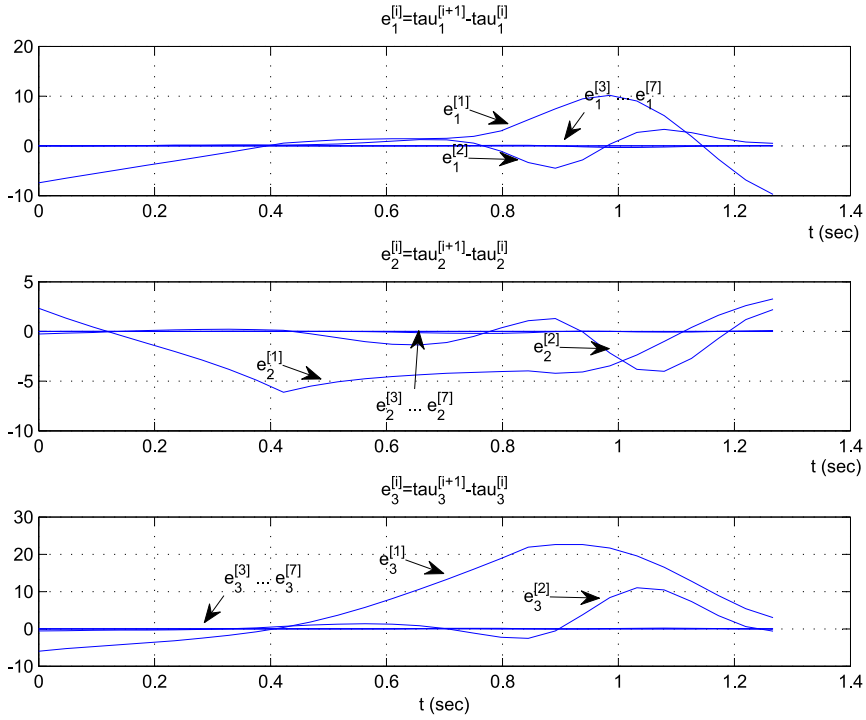


Figure 5.12: **Sequence of errors in the successive trials in proposed method for spherical (Stanford) robot**

For this spherical robot, Figure 5.11 shows the optimal profiles obtained by this method including joint dispositions, velocities and forces/torques of three joints of the robot. Also the values listed in Table 5.10 gives some information about the optimal data obtained by this method. Let us now use the same convergence analysis applied for SCARA robot in the case of this spherical robot. Figure 5.12 shows the successive errors, as defined in the previous case study, for joint force/torques of this spherical robot. According to the last three columns of the Table 5.10 we can conclude that all three sequences of error norms converge to zero with rate $\mu_1 = \mu_2 = \mu_3 = \frac{1}{2}$.

Let us now make some comparisons between the above three methods to solve the OCP of the considered spherical (Stanford) robot:

- Proposed method and multiple shooting method:

  In the case of multiple shooting method for this spherical robot, we set $N = 10$.

Table 5.10: Optimal data of spherical (Stanford) robot obtained by applying proposed method

Trial	J_{min}	T_{min}	First order optimality (δ)	Number of math operations	time of computation	$\left\|e_1^{[i]}\right\|$	$\left\|e_2^{[i]}\right\|$	$\left\|e_3^{[i]}\right\|$
1	162.29	0.806	45.1	147	3.14	-	-	-
2	60.55	1.196	0.694	114	2.59	9.6	5.02	15.7
3	48.94	1.267	0.0726	50	1.29	2.17	1.44	3.32
4	48.16	1.267	0.0014	48	1.069	0.25	0.31	0.15
5	48.1	1.267	3.5×10^{-4}	36	0.87	0.07	0.03	0.047
6	48.1	1.267	4.64×10^{-6}	32	0.83	0.01	4.17×10^{-3}	1.44×10^{-4}
7	48.1	1.267	7.1×10^{-8}	28	0.81	3.27×10^{-3}	1.8×10^{-4}	2.71×10^{-5}

Hence, the obtained optimization problem has 84 degrees of freedom and 73 constraints which should be satisfied. The total computation time is 413 sec for performing 3013 math operations. The final first order optimality is 2.17 to find the minimum value of the cost function which is 53.4 in this method.

The optimal data for this robot by proposed method, given in Table 5.10, shows that in each trial averagely 65 math operations are necessary to perform which take 1.51 sec averagely for $N = 4$. In addition, in the proposed method the total number of parameters whose optimal values must be obtained is 10 and total number of constraints which must be satisfied is 12. Also, the first order optimality in the last trial of the proposed method is 7.1×10^{-8}, as opposed the value 2.17 in multiple shooting method, which shows the obtained optimal solutions in proposed method are too closer to real optimal one with respect to those obtained in multiple shooting method. Thus, these comparisons show a better performance for proposed method with respect to multiple shooting method.

- Proposed method and spline based optimal control method:

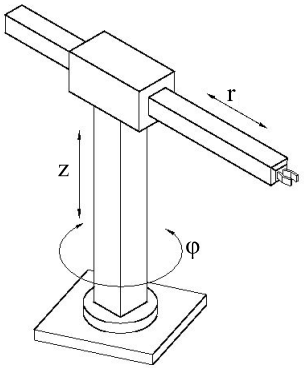


Figure 5.13: **Three DOF cylindrical robot**

In this case study we assumed $N = 4$ for both proposed and spline based optimal control methods. For making a comparison between these two methods we can use the data given in tables 5.9 and 5.10. The average number of math operations in each trial in the proposed method is 65, while it is 364 in the spline based optimal control method which accordingly the necessary computation time in each trial of proposed method is 1.51 sec averagely opposed to 4.74 sec in the spline based optimal control method. Also, as explained in the previous case study, the proposed method in each trial solves a quadratic optimization programming problem, while the optimization algorithm in the spline based optimal control method should solve a nonlinear optimization programming problem. Of course, both methods have the same number of constraints and parameters whose optimal values must be obtained.

5.4.3 Cylindrical Robot

The third class of standard robot arms are cylindrical robot manipulators. In this subsection we would like to solve the OCP of this kind of robots. Consider a three degrees of freedom cylindrical robot shown in Figure 5.13. This robot has three joints which the first translational joint (with generalized coordinate z) is for vertical motion and the rotary joint (with generalized coordinate φ) is responsible for angular motion and another translational joint (with generalized coordinate r) is for radial motion. Hence, the location of the end-effector of this robot can be described in a cylindrical coordinate system with coordinates (r, φ, z). The dynamic equations of this robot can

be obtained as follows [54]:

$$\begin{aligned}
&\left[I_R+\frac{m\,r_\ell^2}{3}+(m+m_\ell)\,r^2(t)-mr_\ell\,r(t)\right]\ddot{\varphi}(t)+2\left[(m+m_\ell)\,r(t)-\frac{m\,r_\ell}{2}\right]\dot{r}(t)\,\dot{\varphi}(t)=M_\varphi(t),\\
&(m+m_\ell)\,\ddot{r}(t)-\left[(m+m_\ell)\,r(t)-\frac{m\,r_\ell}{2}\right]\dot{\varphi}^2(t)=F_r(t),\\
&(m+m_\ell)\,[\ddot{z}(t)+g]=F_z(t),
\end{aligned} \tag{5.71}$$

where

$$\begin{cases} I_R=0.8\,kg\,m^2,\ m=20\,kg,\ m_\ell=15\,kg,\ r_\ell=2\,m,\ g=9.81\,m/sec^2,\\ \text{minimum radial length of robot}: A=1\,m. \end{cases} \tag{5.72}$$

Also the physical constraints of this robot are as follows:

$$\begin{cases} 0\le r\le 0.5,\ (m)\\ -\dfrac{3\pi}{2}\le\varphi\le\dfrac{3\pi}{2},\\ z_0\le z\le z_0+0.8, z_0=0.5\ (m), \end{cases},\ \begin{cases} |\dot{r}|\le 0.1\ (m/sec),\\ |\dot{\varphi}|\le 0.5\ (rad/sec),, \end{cases},\ \begin{cases} |M_\varphi|\le 5\ (Nm),\\ |F_r|\le 2\ (N),\\ |F_z|\le 2\ (N). \end{cases} \tag{5.73}$$

The third equation in (5.71) is an independent linear dynamics which is relative to vertical motion of the robot. Hence it can be rewritten as $\ddot{z}=\bar{F}_z=\frac{F_z}{m+m_\ell}-g$.

Let $q_1=\varphi$, $q_2=r, q_3=z$, thus the the motion equations (5.71) can be given in matrix form

$$\mathbf{M}(\mathbf{q})\,\ddot{\mathbf{q}}+\mathbf{N}(\mathbf{q},\dot{\mathbf{q}})=\boldsymbol{\tau}, \tag{5.74}$$

where

$$\mathbf{q}=\begin{bmatrix} q_1\\ q_2\\ q_3 \end{bmatrix},\ \mathbf{M}=\begin{bmatrix} I_R+\frac{m\,r_\ell^2}{3}+(m+m_\ell)\,r^2(t)-mr_\ell\,r(t) & 0 & 0\\ 0 & m+m_\ell & 0\\ 0 & 0 & 1 \end{bmatrix}, \tag{5.75}$$

and

$$\mathbf{N} = \begin{bmatrix} 2\left[(m+m_\ell)\, r\,(t) - \frac{m\, r_\ell}{2}\right] \dot{r}\,(t)\, \dot{\varphi}\,(t) \\ -\left[(m+m_\ell)\, r\,(t) - \frac{m\, r_\ell}{2}\right] \dot{\varphi}^2\,(t) \\ 0 \end{bmatrix},\ \boldsymbol{\tau} = \begin{bmatrix} \tau_1 \\ \tau_2 \\ \tau_3 \end{bmatrix} = \begin{bmatrix} M_\varphi \\ F_r \\ \bar{F}_z \end{bmatrix} \tag{5.76}$$

and also let the initial and final configurations

$$\mathbf{q}\,(\mathbf{0}) = \begin{bmatrix} 0.2 \\ 0 \\ 0.5 \end{bmatrix},\ \mathbf{q}\,(\mathbf{T}) = \begin{bmatrix} 1 \\ 0.02 \\ 0.7 \end{bmatrix}. \tag{5.77}$$

Let us now obtain the state space representation of this robot considering the following state vector:

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix}. \tag{5.78}$$

As explained earlier, the state space representation of robot manipulators can be written as

$$\dot{x} = f\,(x) + g\,(x)\,\tau, \tag{5.79}$$

where in the case of this cylindrical robot we will have

$$f\,(x) = \begin{bmatrix} x_4 \\ x_5 \\ x_6 \\ -\frac{15x_4x_5(70x_2+30)}{525x_2^2+450x_2+337} \\ \frac{1}{7}x_4^2\,(7x_2+3) \\ 0 \end{bmatrix},\ g\,(\mathbf{x}) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \frac{15}{525q_2^2+450q_2+337} & 0 & \\ 0 & \frac{1}{35} & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{5.80}$$

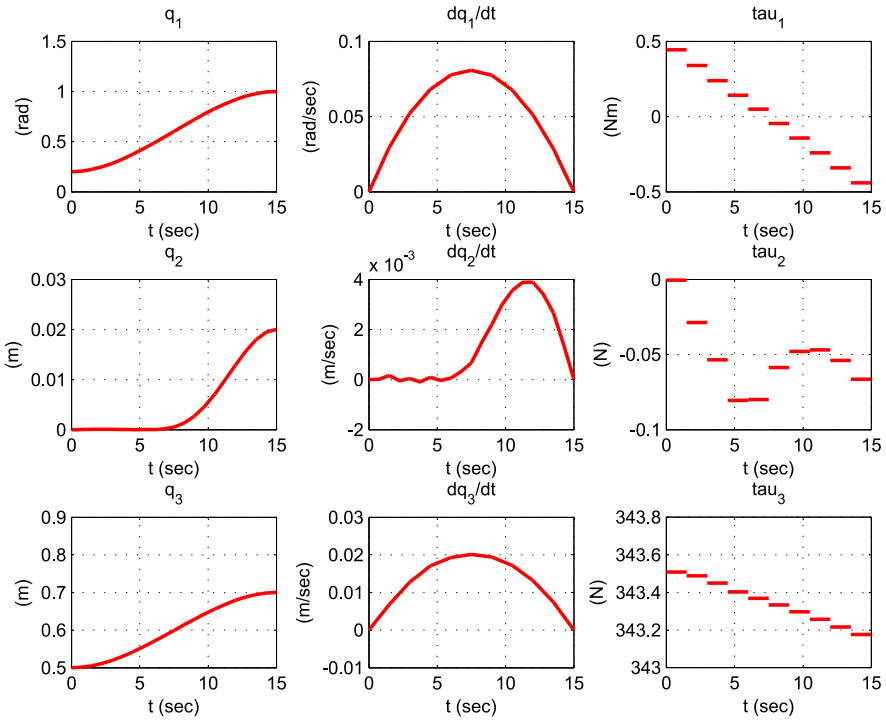


Figure 5.14: **Optimal profiles of cylindrical robot obtained by multiple shooting method**

In addition, the cost functional considered in this case study is as follows

$$\min J_i = \int_0^T \left(\tau^T R \tau + X_d^T W X_d\right) dt, \tag{5.81}$$

where

$$R = \begin{bmatrix} \alpha_1 & 0 & 0 \\ 0 & \alpha_2 & 0 \\ 0 & 0 & \alpha_3 \end{bmatrix}, \; W = \begin{bmatrix} \beta_1 & 0 & 0 \\ 0 & \beta_2 & 0 \\ 0 & 0 & \beta_3 \end{bmatrix}, \; X_d = \begin{bmatrix} x_4 \\ x_5 \\ x_6 \end{bmatrix}, \tag{5.82}$$

with $\alpha_1 = 1$, $\alpha_2 = 0.1$, $\alpha_3 = 0.001$, $\beta_1 = 10$, $\beta_2 = 1$, $\beta_3 = 1$.

As with previous case studies, we solve the OCP of this robot using three methods: multiple shooting, spline based optimal control and proposed method:

- Multiple shooting method: In this case, we consider discretization number $N = 10$. Figure 5.14 shows the optimal profiles including joint dispositions, velocities

Table 5.11: Optimal data of cylindrical robot obtained from multiple shooting method

Number of SQP iterations	25
Number of math operations	2236
J_{min}	0.609
T_{min}	15
First order optimality	4.32
Computation time (sec)	537

Table 5.12: Optimal data of cylindrical robot obtained from spline based optimal control method

Number of SQP iterations	32
Number of math operations	311
J_{min}	0.5448
T_{min}	13.52
First order optimality	0.00015
Computation time (sec)	4.17

and torques trajectories of this cylindrical robot. As well as, the optimal data for this robot resulted from this method are collected in the Table 5.11. As stated earlier, in this method we have to consider a large number of parameters and virtual constraints. The optimal controls of this robot are obtained after performing 2236 math operations and taking 537 sec computation time.

- Spline based optimal control method:

  For this case study, we considered $N = 4$, then the optimal profiles of this robot containing joint disposition, velocity and force/torque trajectories are shown in Figure 5.15. Also the optimal data obtained from this method are given in Table 5.12.

- Proposed method

  Here, we attempt to obtain the optimal solution of the cylindrical robot by proposed method. Figure 5.16 represents the optimal trajectories of this robot obtained by this method. As this figure shows, the sequence of optimal profiles converge after 6 trials. More optimal information can be achieved from Table 5.13. Also, Figure 5.17 shows the sequence of successive errors defined as with in the previous case studies. According to the last three columns of Table 5.13, the

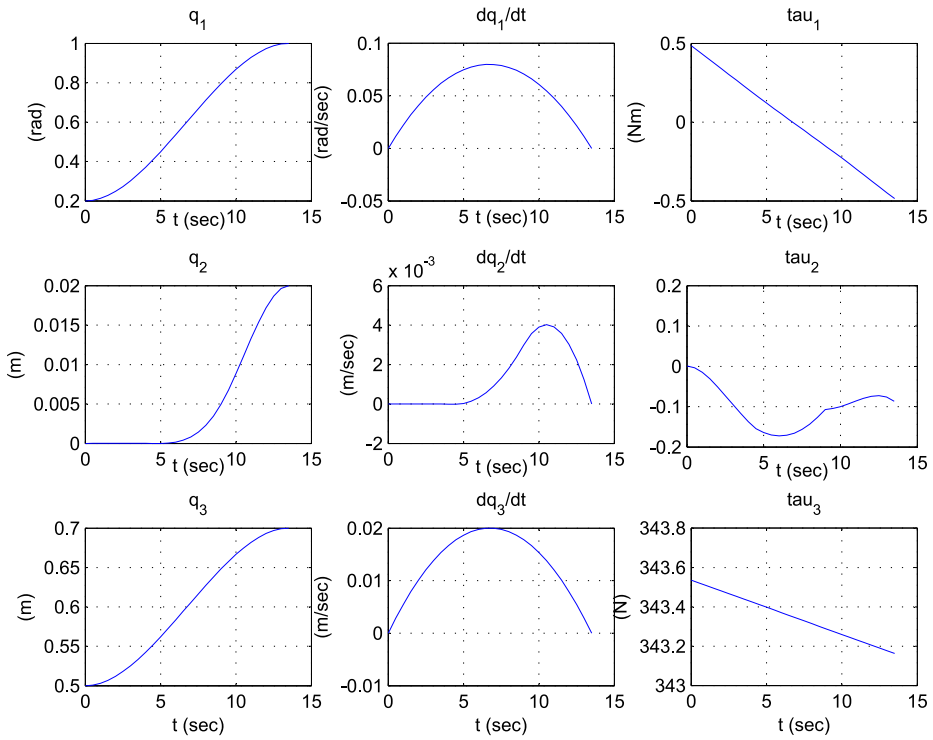


Figure 5.15: **Optimal profiles of cylindrical robot obtained by spline based optimal control method**

rate of convergence of the three joints of the robot are $\mu_1 = \frac{1}{11}$, $\mu_2 = \frac{1}{6}$, $\mu_3 = \frac{1}{3}$, respectively.

Now we wish to make a comparison between proposed method and other two methods from the obtained results presented in the above tables and figures for the cylindrical robot:

- Proposed method and multiple shooting method:

  As mentioned above, for this case study in the multiple shooting method we consider $N = 10$. Thus, the number of parameters in the optimization programming is 84, while this number in the proposed method for $N = 4$ is only 12 for three joints of the robot. Also the number of constraints which should be met in the multiple shooting and proposed method are 73 and 10, respectively and hence, as shown in Table 5.11 , it causes a considerable number of math operations

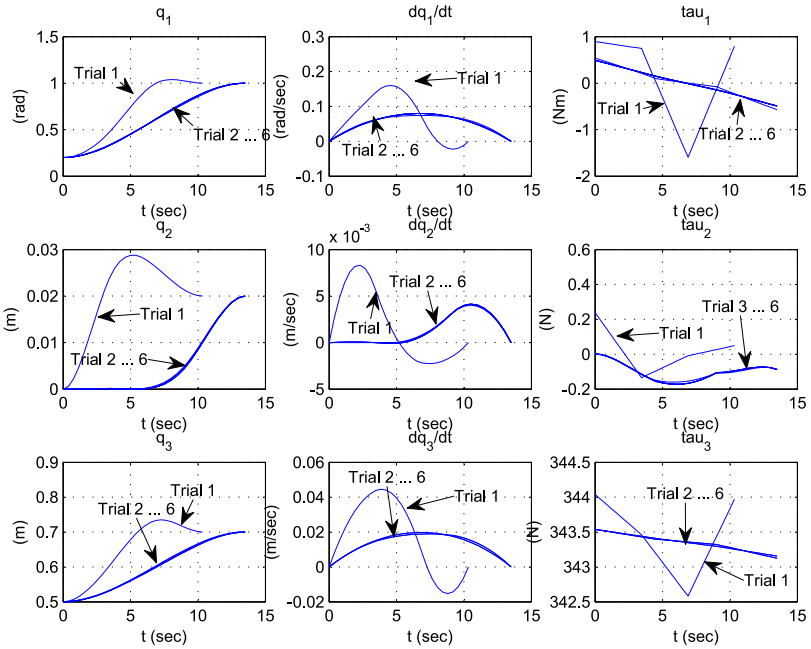


Figure 5.16: **Optimal profiles of cylindrical robot obtained by proposed method**

Table 5.13: Optimal data of cylindrical robot obtained by applying the proposed method

Trial	J_{min}	T_{min}	First order optimality (δ)	Number of operations	Time of computation	$\left\|e_1^{[i]}\right\|$	$\left\|e_2^{[i]}\right\|$	$\left\|e_3^{[i]}\right\|$
1	3.42	10.34	1426	109	2.42	$\times$	$\times$	$\times$
2	0.839	13.52	49.6	117	2.59	0.9	0.2	0.05
3	0.61	13.52	10.6	73	1.42	0.05	0.014	0.02
4	0.608	13.52	5.46	65	1.25	2.3×10^{-3}	2.1×10^{-4}	0.008
5	0.605	13.52	0.129	52	0.96	5.14×10^{-4}	3.46×10^{-6}	2.17×10^{-4}
6	0.601	13.52	0.00093	33	0.64	1.5×10^{-6}	7×10^{-8}	4.32×10^{-5}

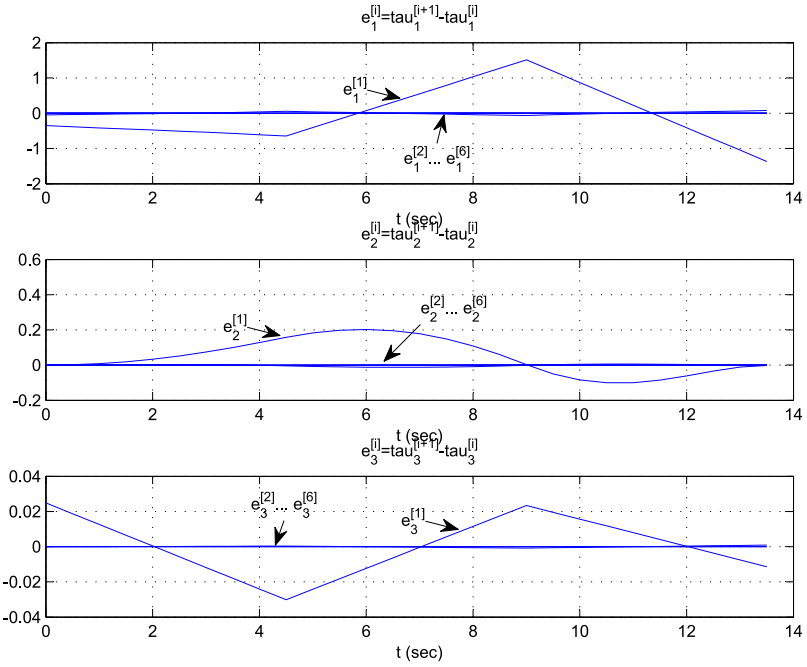


Figure 5.17: **Error profiles obtained for cylindrical robot in various trials**

and high computation time for multiple shooting method. The average computation time in the proposed method is 1.55 sec while it is 537 sec in the multiple shooting method. In addition, if we increase the number of discretization (N) in the multiple shooting to obtain a more continuous control, it causes the higher computation time and even there no exists any optimal solution for $N > 60$. For example, in this case study for $N = 10$ in multiple shooting method we found a near optimal solution with first order optimality equals 4.32. Thus, according to these data one can judge between these two methods easily.

- Proposed method and spline based optimal control method

  As first order optimality in the Table 5.13 shows, the OCP for this robot is solved during 6 trials so that the average computation time in each trial is 1.55 sec, while this value in the spline based optimal control method is 4.17 sec. In addition, the average number of math operations in the proposed method in each trial is 75 and this number in the spline based optimal control method is 311. Hence, these numbers show that the proposed method solves the OCP gradually. Note also

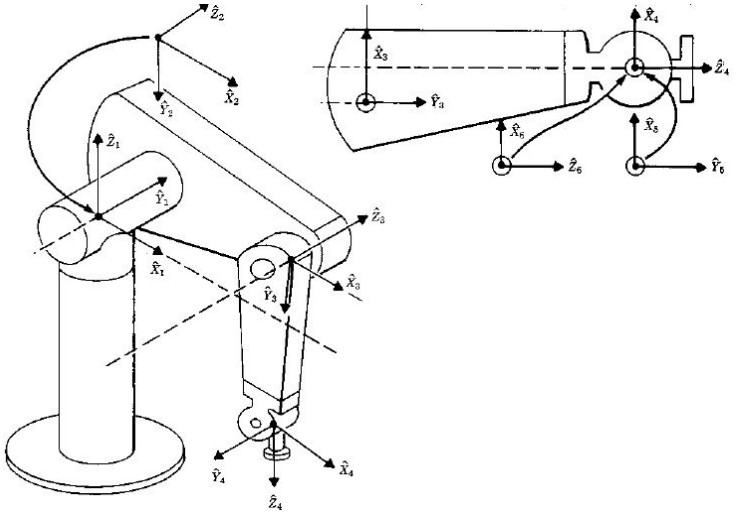


Figure 5.18: **Puma 560 robot manipulator**

that as last two columns in Table 5.13 show, the sequence of optimal solutions converge quickly and after 5 trials.

5.4.4 Puma 560 Robot Manipulator

In the previous case studies, we solved the OCP of the SCARA, spherical and cylindrical robots by three methods and using the optimal data obtained by each method we made some comparisons for the performance of these methods. The rest case studies are some well known angular robot manipulators in which we shall only apply the proposed method; since the necessary comparisons were made in the previous subsections.

In this subsection, we are going to apply the proposed method to the well known Puma 560 robot manipulator shown in Figure 5.18. As seen in this figure, it is a 6 degrees of freedom robot manipulator which we only consider the first three joints of this robot while the last three joints are fixed in their home positions. According to the MDH frames in this figure the MDH parameters are obtained as given in Table 5.14. The dynamic model of this robot can be obtained using Euler-Lagrange Formulation presented in subsection 2.2.1. The inertia matrix, centripetal and Coriolis as well as the gravity terms of the Puma 560 robot are obtained by the dynamic parameters of this robot given in Table 5.15 [6]. By this information, the dynamic equations of the first three joints of Puma 560 can be derived (given in Appendix B).

Table 5.14: DH parameters of Puma 560 robot manipulator

i	α_i	d_i	q_i	r_i
1	$-\pi/2$	0	q_1	0
2	0	0.432	q_2	0.15
3	$\pi/2$	0.02	q_3	0
4	$-\pi/2$	0	$q_4 = 0$	0.433
5	$\pi/2$	0	$q_5 = 0$	0
6	0	0	$q_6 = 0$	0.056

Table 5.15: DH parameters of puma 560 robot manipulator

Link i	1	2	3
I_{xxi}	0	0.13	0,066
I_{yyi}	0.35	0.524	0.086
I_{zzi}	0	0.539	0.0125
I_{xyi}	0	0	0
I_{xzi}	0	0	0
I_{yzi}	0	0	0
$\bar{x}_i$	0	-0.3638	-0.0203
$\bar{y}_i$	0	0.006	-0.014
$\bar{z}_i$	0	0.2275	0.07
m_i	0	17.4	6.05

For Computing the optimal profiles of this robot by proposed method, we use the following cost functional and constraints given in Table 5.16, as well as the boundary conditions presented in Table 5.17,

$$J = 0.5 \int_0^T \left(\tau_1^2 + \tau_2^2 + \tau_3^2 \right) dt, \tag{5.83}$$

where τ_1, τ_2 and τ_3 are the torques of the first three joints of the robot.

Applying the proposed method into this robot, the optimal profiles are obtained as shown in Figure 5.19. In addition, the Table 5.18 presents the optimal data obtained from optimization algorithm in the case of this robot. According to these data, the minimum value of the traversal time and cost functional are obtained after 5 trials. As seen from the fifth and sixth columns of Table 5.18 the direction of variation of the T_{min} and J_{min} are opposite each other. Thus, the minimum value of these two parameters are obtained after some trade-off between these two values and satisfying

Table 5.16: Joint position and velocity constraints of Puma 560 robot

positions	$q_1\,(^\circ)$	$q_2\,(^\circ)$	$q_3\,(^\circ)$
min	-160	-225	-45
max	160	45	225
velocities	$\dot{q}_1\,(^\circ/sec)$	$\dot{q}_2\,(^\circ/sec)$	$\dot{q}_3\,(^\circ/sec)$
min	-200	-200	-260
max	200	200	260
torques	$\tau_1\,(N.m)$	$\tau_2\,(N.m)$	$\tau_3\,(N.m)$
min	-150	-150	-100
max	150	150	100

Table 5.17: Initial and final conditions of Puma 560 robot

positions	$q_1\,(rad)$	$q_2\,(rad)$	$q_3\,(rad)$
Initial	0	0	0
Final	$50 \times (\pi/180)$	$25 \times (\pi/180)$	$30 \times (\pi/180)$
velocities	$\dot{q}_1\,(rad/s)$	$\dot{q}_2\,(rad/s)$	$\dot{q}_3\,(rad/s)$
Initial	0	0	0
Final	0	0	0

the constraints so that in trial 5 these parameters take 664.14 and 5.297, respectively. For having some information regarding the rate of convergence in this case study, we use the sequence of errors (as defined in the previous case studies) which are shown in Figure 5.20. For this robot the sequence of error norms of robot joints converges to zero with rates $\mu_1 = \frac{1}{3}, \mu_2 = \frac{1}{3}, \mu_3 = \frac{1}{5}$, respectively.

Table 5.18: Optimal data of optimal control of Puma 560 robot

Trial	Number of SQP iterations	Number of math operations	Time of computation	J_{min}	T_{min}	First order Optimality (δ)
1	22	252	3.74	629	6.479	9.09
2	20	235	3.21	733	5.351	4.52
3	18	214	2.87	666.83	5.297	0.014
4	16	178	1.96	664.15	5.297	0.00037
5	13	132	1.53	664.14	5.297	0.000013

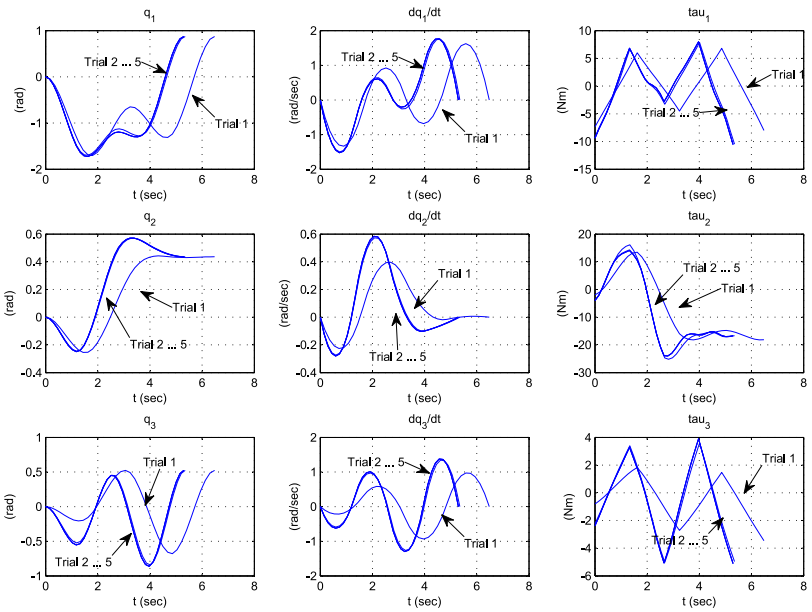


Figure 5.19: **Optimal profiles of Puma 560 robot manipulator**

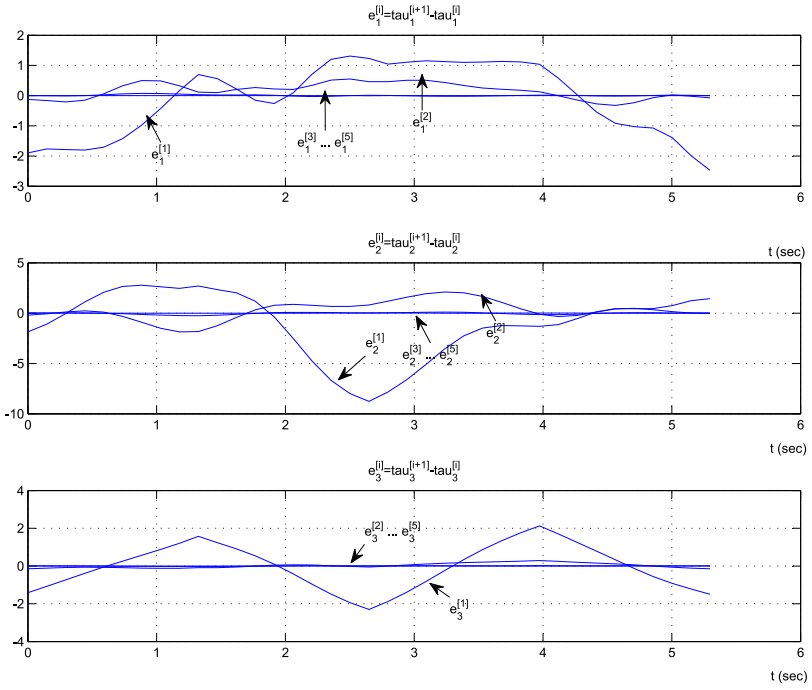


Figure 5.20: **Sequence of errors in successive trials obtained for Puma 560 robot**

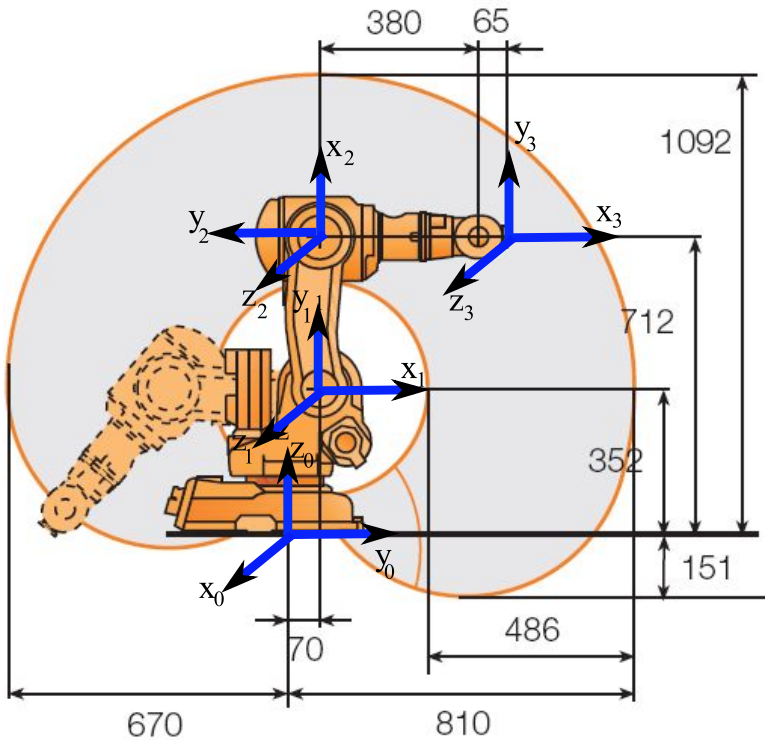


Figure 5.21: **ABB IRB140 robot manipulator**

5.4.5 ABB IRB140 Robot Manipulator

Let us now apply the proposed method into an ABB IRB140 robot arm, as shown in Figure 5.21, which is a six degrees of freedom (DOF) industrial manipulator. The first three joints of this robot are used to position the end-effector, while the last three ones are employed to set the orientation of the end-effector. In the optimal control of robot manipulators usually the positioning part of the robot, i.e. three first joints, are considered. Accordingly, we can consider the ABB IRB140 robot as a three degrees of freedom arm so that the orientation joints of the robot are fixed in their home positions. So as to obtain the dynamic model of this robot, we require the transformation matrices of this robot according to Denavit-Hartenberg notation. The Denavit-Hartenberg parameters of this robot can be obtained, as given in Table 5.19, by considering the desired frame of each robot link illustrated in Figure 5.21.

Now, the dynamic model of this robot can be obtained according to Euler-Lagrange Formulation presented in subsection 2.2.1. Notice that we used the dynamic parameters of ABB IRB140 robot given in [49]. Using these data, the dynamic equations of this

Table 5.19: Denavit-Hartenberg parameters of ABB IRB140 robot

Link (i)	$q_i\,(rad)$	$d_i\,(m)$	$a_i\,(m)$	$\alpha_i\,(rad)$
1	$\pi/2$	0.352	0.07	$\pi/2$
2	$\pi/2$	0	0.36	0
3	$-\pi/2$	0	0.445	0

robot can be derived as follows:

$$M\left(q\right)\ddot{q}+N\left(q,\dot{q}\right)=\tau, \tag{5.84}$$

where

- Inertial Matrix :

$$M=\begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \end{bmatrix}, \tag{5.85}$$

  where

$$\begin{aligned} &M_{11}=-2.05SC-1.705CD+0.8CC-1.94S2+2.05S3+2.1097,\\ &M_{12}=M_{21}=0.34S2-0.058SA+0.924CC,\\ &M_{13}=M_{31}=0.34S2-0.058SA+0.924CC,\ M_{22}=4.11S3+4.35,\\ &M_{23}=M_{32}=0.58+0.674S3-0.042CC,\ M_{3,3}=-0.0694 \end{aligned} \tag{5.86}$$

- Centripetal, Coriolis and gravity term $(N\left(q,\dot{q}\right))$:

  In robot dynamics (3.12), $N\left(q,\dot{q}\right)$ is the summation of centripetal, Coriolis $(h\left(q,\dot{q}\right))$ and gravity $(G\left(q\right))$ terms each of which are represented by the following 3-vectors:

$$h=\begin{bmatrix} h_1 & h_2 & h_3 \end{bmatrix}^T, \tag{5.87}$$

where

$$\begin{aligned} h_1 &= (0.0667\,SA - 4.1\,CB + 3.142\,SC - 0.8CC - 1.94\,C2)\,\dot{q}_1\dot{q}_2 + \\ &\quad (0.069\,SA - 2.055\,CB + 2.055\,C3)\,\dot{q}_1\dot{q}_3, \\ h_2 &= (2.05\,CB - 1.707\,SD + 0.4\,C3 + 0.97C2)\,\dot{q}_1^2 + \\ &\quad (0.0054\,CA + 0.0085\,C3)\,\dot{q}_2^2 + 2.0558\,C3\,\dot{q}_2\dot{q}_3, \\ h_3 &= (-0.033\,SA + 1.028\,CB + 0.4SC - 1.027\,C3)\,\dot{q}_1^2 - \\ &\quad 2.054\,C3\,\dot{q}_2^2 - 2.054\,C3\,\dot{q}_3^2 - 2.055\,C3\,\dot{q}_2\dot{q}_3 \end{aligned} \tag{5.88}$$

and gravity vector

$$G = \begin{bmatrix} g_1 \\ g_2 \\ g_3 \end{bmatrix}, \qquad \begin{array}{c} g_1 = 0, \\ g_2 = 55.98\,CC - 136\,S2, \\ g_3 = 55.98\,CC. \end{array} \tag{5.89}$$

where

$$\begin{aligned} &SA = \sin(2q_2 + 2q_3),\ CB = \cos(2q_2 + q_3),\ SC = \sin(q_2 + q_3), \\ &SB = \sin(2q_2 + q_3),\ CC = \cos(q_2 + q_3),\ SD = \sin(2q_2), \\ &CD = \cos(2q_2),\ S2 = \sin(q_2),\ C2 = \cos(q_2), \\ &S3 = \sin(q_3),\ C3 = \cos(q_3). \end{aligned} \tag{5.90}$$

Let us now solve the OCP of this robot by proposed method according to the following cost functional and boundary conditions given in Table 5.21:

$$J = \int_0^T \left(\tau_1^2 + \tau_2^2 + \tau_3^2 + .001\,(q_1 - 0.4)^2 + .001\,(q_2 - 0.5)^2 + .001\,(q_3 - 0.2)^2\right) dt, \tag{5.91}$$

where τ_1, τ_2 and τ_3 are the torques of the first three joints (major joints) of the robot. Also the position and velocity constraints of this robot have been listed in Table 5.20 [32]. By these data, the optimal joint position, velocity and torque trajectories are obtained using the proposed method as shown in Figure 5.22.

As seen in this figure, the proposed optimal control algorithm has been performed

Table 5.20: Joint position and velocity constraints of ABB IRB140 robot

positions	$q_1\,(^\circ)$	$q_2\,(^\circ)$	$q_3\,(^\circ)$
min	-180	-90	-230
max	180	110	50
velocities	$\dot{q}_1\,(^\circ/sec)$	$\dot{q}_2\,(^\circ/sec)$	$\dot{q}_3\,(^\circ/sec)$
min	-200	-200	-260
max	200	200	260
torques	$\tau_1\,(N.m)$	$\tau_2\,(N.m)$	$\tau_3\,(N.m)$
min	0	0	0
max	150	150	100

Table 5.21: Initial and final conditions of ABB IRB140 robot

positions	$q_1\,(rad)$	$q_2\,(rad)$	$q_3\,(rad)$
Initial	0	0	0
Final	0.4	0.5	0.2
velocities	$\dot{q}_1\,(rad/s)$	$\dot{q}_2\,(rad/s)$	$\dot{q}_3\,(rad/s)$
Initial	0	0	0
Final	0	0	0

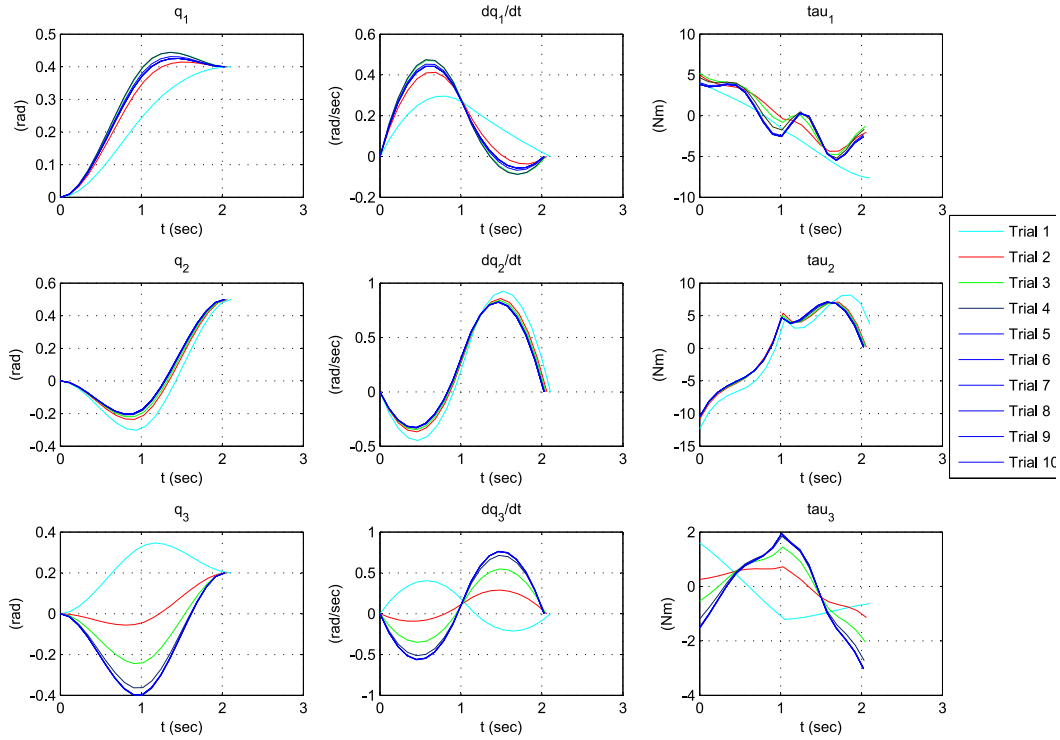


Figure 5.22: **Optimal joint position, velocity and torques of ABB IRB140 robot**

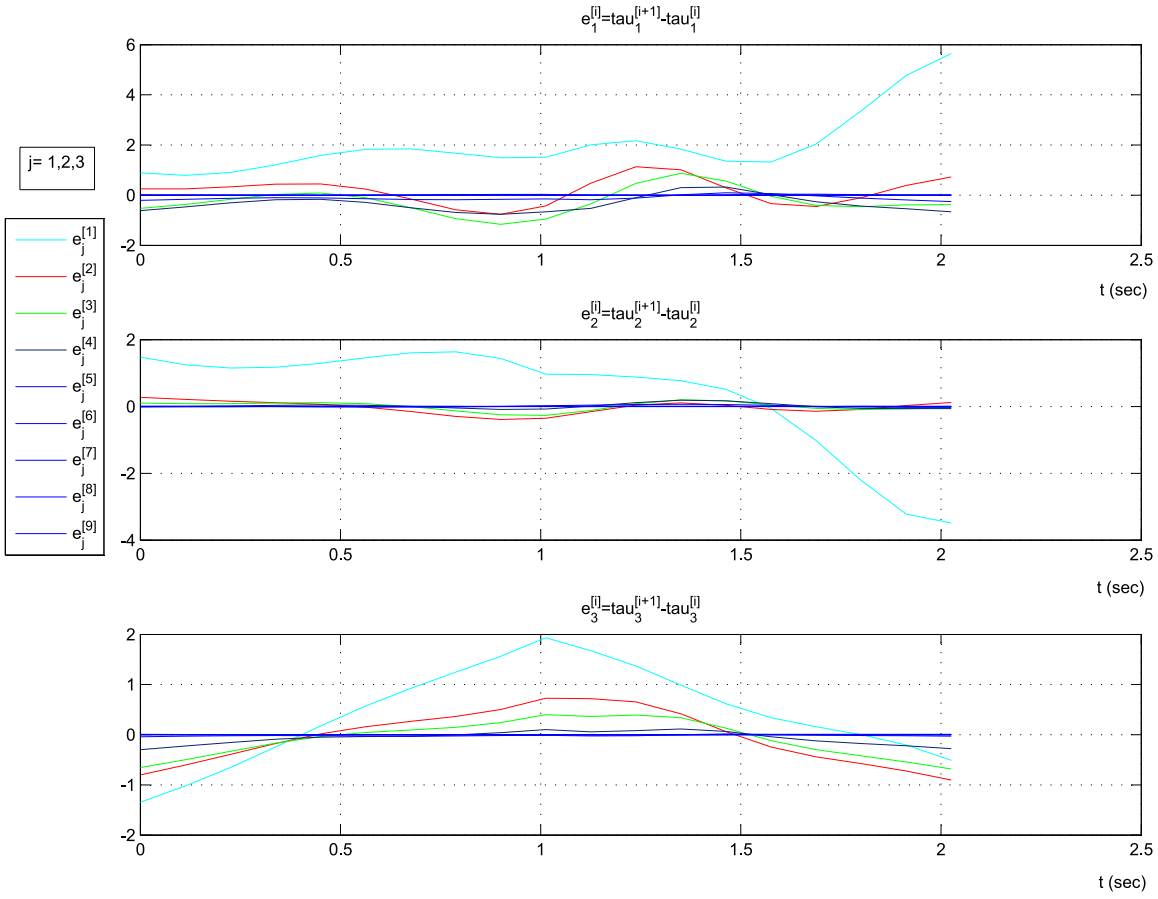


Figure 5.23: **Successive errors obtained for ABB IRB140 robot joints**

for 10 trials. In addition, the optimal data derived from optimization algorithm are collected in Table 5.22. These data shows the system performs the desired task in 2.106 sec with $J_{min} = 61.37$ in the first trial. These parameters will find their real minimum values so that in the tenth trial the first order optimality shows that the minimum value of these parameters are achieved. Also for having some information about the rate of convergence in the case of this robot, we can use Figure 5.23. The rate of convergence for the sequence of error norm of robot joints are $\mu_1 = \frac{1}{3}$, $\mu_2 = \frac{1}{6}$, $\mu_3 = \frac{1}{4}$, respectively.

5.4.6 KUKA Robot

In chapter 2, we obtained the dynamic model of the KUKA robot through an experimental identification. In this subsection we are going to obtain the optimal trajec-

Table 5.22: Optimal data of optimal control of ABB IRB140 robot

Trial	Number of SQP iterations	Number of math operations	Time of computation	J_{min}	T_{min}	First order Optimality (δ)
1	9	57	0.96	61.37	2.106	32.4
2	6	52	0.84	41.57	2.061	1.19
3	7	57	0.93	41.54	2.049	0.116
4	6	53	0.82	42.3	2.036	0.032
5	6	53	0.82	41.898	2.026	0.0126
6	6	57	0.88	41.983	2.026	0.00708
7	6	57	0.86	41.924	2.026	0.0009
8	5	61	0.93	41.91	2.026	0.00037
9	6	61	0.9	41.91	2.026	1.5×10^{-5}
10	6	61	0.92	41.91	2.026	2.97×10^{-6}

Table 5.23: Joint position, velocity and acceleration constraints of KUKA robot

Joint	$q_i\,[deg]$	$\lvert\dot{q}_i\rvert\,[deg/s]$	$\lvert\ddot{q}_i\rvert\,[deg/s^2]$	$\tau_i[N.m]$
1	± 155	151	450	550
2	100 to -55	151	450	550
3	70 to -220	151	450	550

tories of this robot according to the joint position, velocity, acceleration and torque constraints of this robot listed in Table 5.23 [47]. For this problem, we consider the following cost function:

$$J_c = \frac{1}{2}\int_0^T \left(\boldsymbol{\tau}^T R \boldsymbol{\tau} + \dot{\mathbf{q}}^T Q \dot{\mathbf{q}}\right) dt, \tag{5.92}$$

where R and Q are symmetric positive definite weighting matrices. In this case study we use the following diagonal matrices R, Q:

$$R = \begin{bmatrix} 100 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 1 \end{bmatrix}, Q = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{5.93}$$

The Optimal results are shown in Figure 5.24, considering the following boundary

Table 5.24: Optimal data obtained for KUKA robot

Trial	Number of SQP iterations	Number of math operations	Time of computation	J_{min}	T_{min}	First order Optimality (δ)
1	22	302	2.85	148.78	4.643	151
2	17	271	2.68	122.63	5.38	38.7
3	15	258	2.61	124.87	5.42	23.9
4	15	259	2.6	126.83	5.647	2.88
5	15	262	2.62	126.8	5.647	0.43
6	14	257	2.5	125.792	5.647	0.0636
7	12	245	2.38	125.78	5.647	0.0022
8	11	235	2.34	125.627	5.647	00.000127

conditions:

$$\begin{aligned} &q_1(0) = q_2(0) = q_3(0) = 0, \\ &\dot{q}_1(0) = \dot{q}_2(0) = \dot{q}_3(0) = 0, \\ &q_1(T) = 50\,(deg)\,,\ q_2(T) = 25\,(deg)\,,\ q_3(T) = 30\,(deg)\,, \\ &\dot{q}_1(T) = \dot{q}_2(T) = \dot{q}_3(T) = 0. \end{aligned} \tag{5.94}$$

This figure shows, the optimal controls of KUKA robot converge after 8 trials. The Table 5.24 represents some information regarding different trails of the optimization procedure for the KUKA robot. As this table shows, in the first trial the minimum traversal time is 4.63 sec which results in a minimum cost function equals 148.78. However, in the subsequent trials a trade-off is made between these two values so that from trial forth the value of T_{min} is fixed. Moreover, we can obtain the necessary information about the rate of convergence in this case study by referring to Figure 5.25. As explained in the previous case studies, we can use the sequence of error norms to obtain the rate of convergence. According to the value of elements of these three sequences (i.e. $\left\|e_j^{[i]}\right\|$ for $j = 1, 2, 3$), their convergence rates are $\mu_1 = \frac{1}{3}, \mu_2 = \frac{1}{6}, \mu_3 = \frac{1}{4}$, respectively.

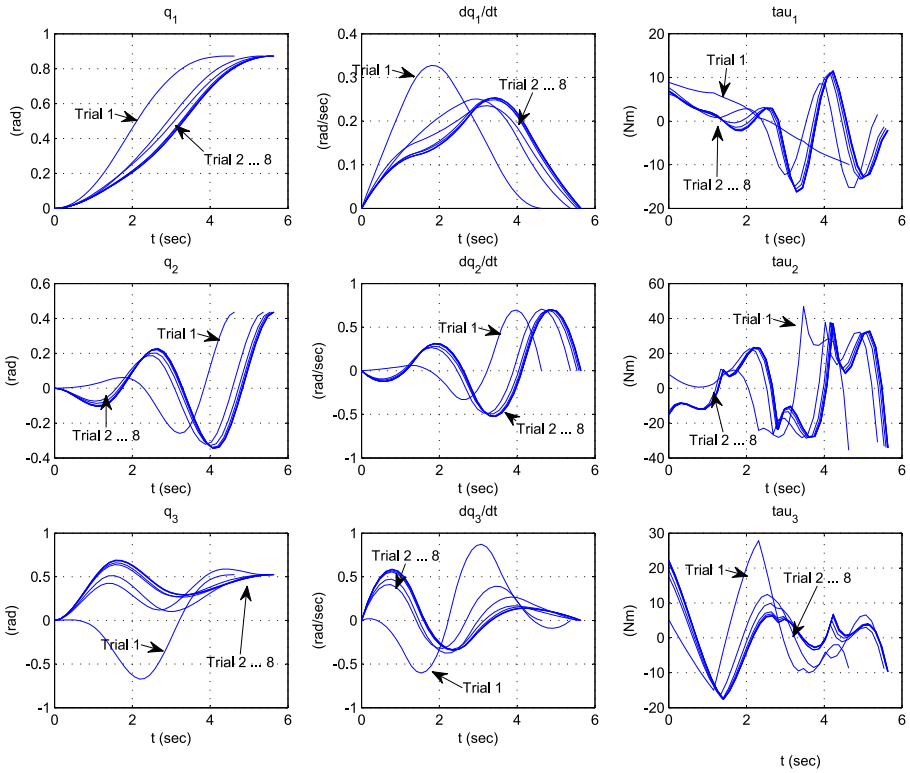


Figure 5.24: **Optimal profiles of KUKA robot**

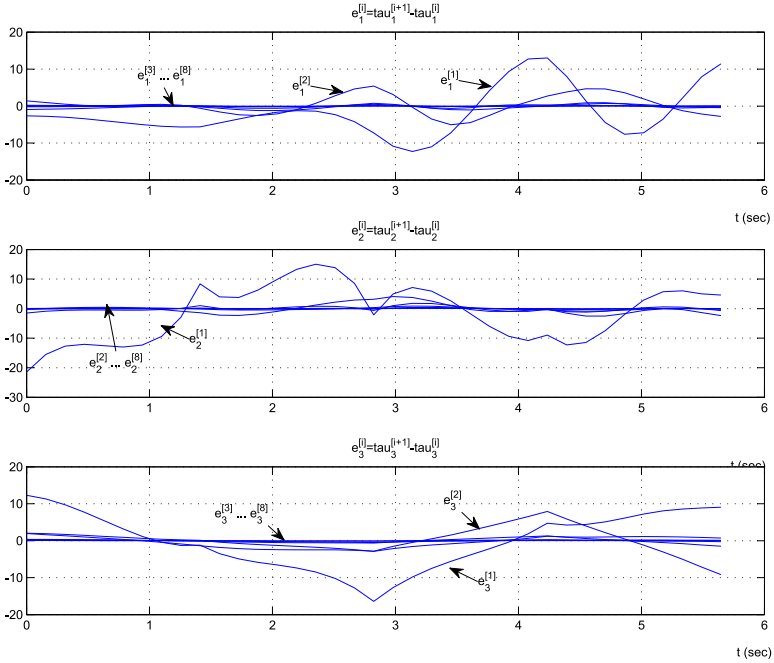


Figure 5.25: **Successive errors of KUKA robot joints**

5.5 A Particular Comparison

In this section, it is attempted to make some comparisons between the first and second proposed methods presented in chapter 4 and 5, respectively, as well as multiple shooting method. In doing so, these methods are applied into a vertical two links robot manipulator used as one of the case studies in chapter 4. However, so as to make these comparisons it is necessary to consider an unconstrained OCP for second proposed method and multiple shooting methods. In chapter 4, Figure 4.3 showed the optimal trajectory for this robot given the boundary conditions (4.62). As such, the second proposed method yields the optimal profiles for this robot as shown in Figure 5.26. As well as the optimal solution of this unconstrained OCP are obtained by multiple shooting method as represented in Figure 5.27. The minimum value of the cost functional (4.14) calculated by first proposed method, i.e. $J_{min} = \xi^T(0)\, C\, \xi(0)$, is 1.12 as well as the computation time to find the optimal trajectories by this method is 0.01 sec. Table 5.25 presents some information regarding the optimal data obtained from second proposed and multiple shooting methods. As this table and Figure 5.26 show the second proposed method converges after forth trial which the minimum of cost functional (4.14) is obtained similar to first proposed method and the first order optimality confirms this similarity since its value is too close to zero. However, the computation time to find the optimal trajectories in trial 4 is 1.3 sec while, as mentioned above, this value in the first proposed method is only 0.01 sec. In addition, the multiple shooting method presents a near optimal solution after 4375 math operations which yields the first order optimality equals 0.94.

Thus, these data show the first proposed method demonstrates the best performance for solving the unconstrained OCP of robot manipulators with respect to other mentioned methods.

Table 5.25: Optimal data of vertical two links robot obtained from second proposed and multiple shooting methods

	Proposed method (last trial)	Multiple shooting method
Number of SQP iterations	13	16
Number of math operations	68	4375
J_{min}	1.12	1.47
First order optimality	2.64×10^{-5}	0.94
Computation time (sec)	1.3	613

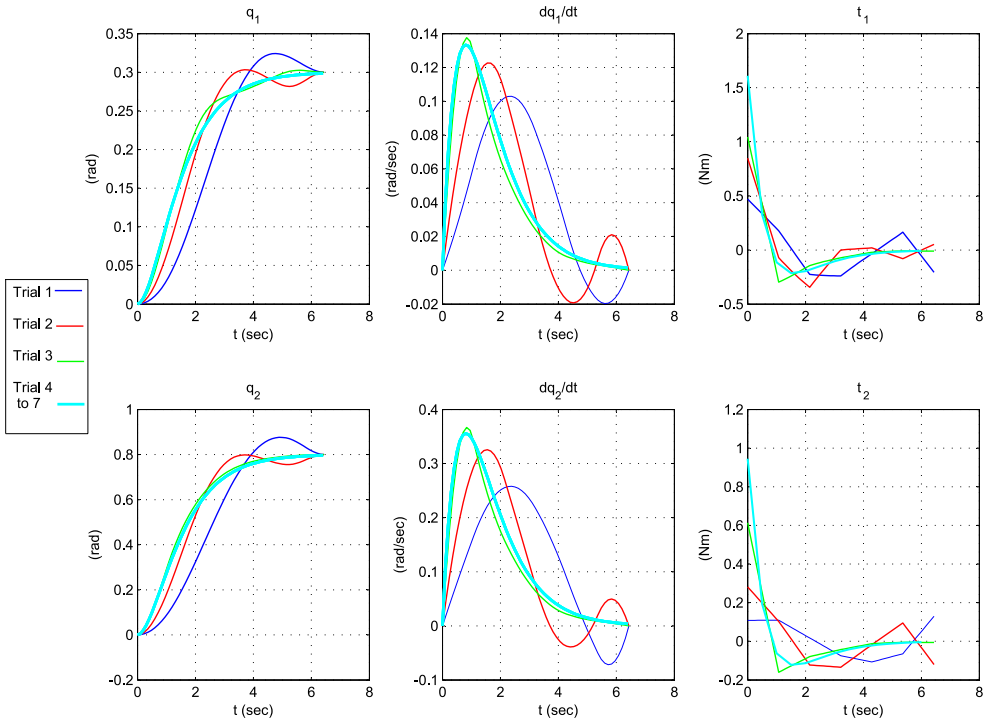


Figure 5.26: **Optimal profiles for unconstrained OCP of vertical two links robot obtained by second proposed method**

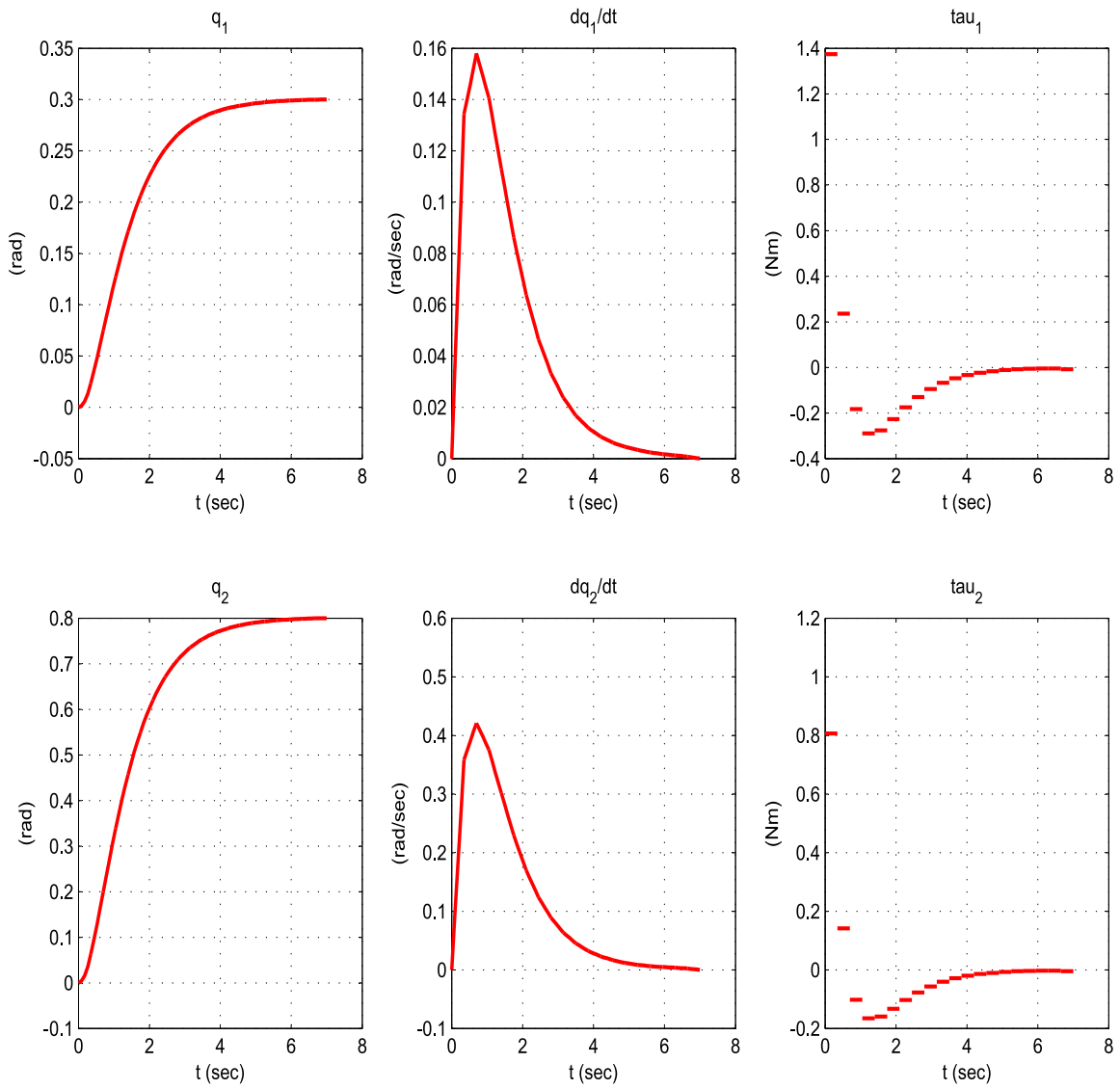


Figure 5.27: **Optimal profiles for unconstrained OCP of vertical two links robot obtained by multiple shooting method**

Chapter 6

Conclusion

Motivation of this study was to propose the new method(s) to solve the optimal control problem of serial robot manipulators. In the following the fulfillment of the considered objectives are given:

Objective 1: The first objective is to obtain the kinematic and dynamic models of our main case study, i.e., KUKA IR 364/10 robot manipulator existed in robotic laboratory of Mechatronic faculty of TUL. The kinematic model of this robot was derived employing modified Denavit-Hartenberg (MDH) notation. In the context of dynamic model of this KUKA robot, we first develop an algorithm using recursive Newton-Euler formulation. This algorithm was used as the main core of a GUI by which user can derive the dynamic model of either 3 or 6 degrees of freedom robot manipulators by entering just robot MDH parameters. A writing task was used to verify the validation of obtained kinematic and dynamic models of the KUKA robot in comparison with these models produced by Robotic Toolbox of MATLAB (RTM).

Objective 2: The second objective is KUKA robot identification. In doing so, in the first stage a new (regression) model of the robot dynamics which is linear in terms of a new set of parameters, so-called base parameter set (BPS), which are compound of dynamic and friction parameters of the robot is derived. In the second stage an excitation trajectory is calculated. This trajectory which is calculated from an optimization problem has a considerable affect on the identification result and hence this stage must be carried out with high attention. Eventually the elements of BPS for

this KUKA robot which contains 21 parameters are estimated and a validation stage is accomplished to verify the obtained model.

Objective 3: The third objective considered in this thesis is to present a completely innovative and new approach to solve the unconstrained optimal control of robot manipulators in the case of point to point motion and trajectory tracking tasks. Unlike the existing methods which yield a local optimal solution, the proposed method solves the considered optimal control problem with obtaining a global optimal solution so that the computation time to find this solution is less than 0.01 sec. noting that the robot dynamics is highly nonlinear and coupled. However, this method can not support any physical constraints on a robot arm. The proposed method which is a model-based controller was extended into a more general case in which an exact model of the robot is not available, namely designing an *adaptive optimal control* scheme for robot manipulators.

Objective 4: The fourth objective is to propose a new method to solve the constrained time-energy optimal control problem of serial robot manipulators. we propose a combined method which contains Iterative Linearization (IL), Iterative Learning Control (ILC) and Parametric Optimization (PO). In this method it is assumed that the robot is performing a repeated task which is usual for robot arms in their applications. In accordance with this method, in each repetition (trial) a linear time varying (LTV) version of robot dynamics is derived by IL with the original considered cost functional. Then PO is used to solve the optimal control problem in this trial and its solution is stored in memory of the system to use in the next trial (ILC). The above procedure is repeated in the next trials so that after a finite number of trials the sequence of optimal solutions converge to the optimal solution of the original nonlinear system (robot dynamics). Then, the limit of the sequence is used to control of the next trials. The corresponding developed algorithm was applied into all standard types of robot arm structures, i.e. SCARA, spherical, cylindrical and angular robots (such as Puma 560, ABB IRB140 and KUKA IR 364/10 manipulators) for the different case of cost functionals. For having a better insight regarding the proposed method, the optimal solution of the considered optimal control problem for SCARA, spherical and cylindrical robots are obtained by direct multiple shooting and spline-based optimal control

methods as well. Then, a series of comparisons are made between the proposed method and the other two methods. According to these comparisons, the following results were obtained for the proposed method:

- In each trial a linear version of highly nonlinear robot dynamics is dealt with.
- Optimization problem is solved gradually during the the successive trials. In other words, as shown by the optimal data given in tables of different case studies in chapter 5, the number of math operations and computation time to find the optimal solution are divided on successive trials.
- The convergence rate of the sequence of optimal solutions is too fast, as shown in various case studies.
- It supports any type of cost functions (quadratic, non-quadratic, linear, nonlinear and so on) and any kind of constraints.
- It generates the smooth trajectory for robot motions causing reduction the stresses to the actuators and to the manipulator structure.
- The possibility to set the initial and final joint accelerations and jerks a priori by the user.
- Unlike the multiple shooting method which produces a constant piecewise control, the proposed method provides a continuous optimal control which can be implemented in practice.
- The structure of the proposed optimal control system is almost simple and it can be implemented easily.

Appendix A: Regression Model of KUKA ROBOT

$$
\begin{aligned}
&Y_{1,1} = \ddot{q}_1,\\
&Y_{1,2} = \ddot{q}_1\left(0.5CD + 0.5\right) + \dot{q}_1\dot{q}_2 SD,\\
&Y_{1,3} = 2\dot{q}_1\dot{q}_2 CD - \ddot{q}_1 SD,\\
&Y_{1,4} = -\dot{q}_2^2 S2 - \ddot{q}_2 C2,\\
&Y_{1,5} = \dot{q}_2^2 C2 - \ddot{q}_2 S2,\\
&Y_{1,6} = 0,\\
&Y_{1,7} = -d_2 S2\dot{q}_2^2 - 2\dot{q}_1\dot{q}_2\left(r_2 SD - r_1 C2\right) - \ddot{q}_1\left(r_2 CD - r_2 + 2r_1 S2\right) - d_2\ddot{q}_2 C2,\\
&Y_{1,8} = 2\dot{q}_1\dot{q}_2\left(r_2 CD + r_1 S2\right) - \ddot{q}_1\left(r_2 SD - 2r_1 C2\right) - d_2\ddot{q}_2 S_2 - d_2\ddot{q}_3 C_2\\
&Y_{1,9} = 2d_2\ddot{q}_1 + r_2\ddot{q}_3 S2 - r_2\ddot{q}_2 C2,\\
&Y_{1,10} = \ddot{q}_1\left(r_1^2 + 0.5r_2^2\left(1 - CD\right) + d_2^2 - 2r_1 r_2 S2\right) - 2\dot{q}_1\dot{q}_2\left(0.5r_2^2 - r_1 r_2 C2\right)\\
&\quad - r_2 d_2\left(S2\dot{q}_2^2 + C2\ddot{q}_2\right),\\
&Y_{1,11} = SA\ddot{q}_1 - 2CA\dot{q}_1\left(\dot{q}_2 + \dot{q}_3\right),\\
&Y_{1,12} = CB\left(\dot{q}_2 + \dot{q}_3\right)^2 - SB\left(\ddot{q}_2 + \ddot{q}_3\right),\\
&Y_{1,13} = CB\left(\ddot{q}_2 + \ddot{q}_3\right) + SB\left(\dot{q}_2 + \dot{q}_3\right)^2,\\
&Y_{1,14} = \left(d_2 + d_3\right)\left(CB\left(\dot{q}_2 + \dot{q}_3\right)^2 - SB\left(\ddot{q}_2 + \ddot{q}_3\right)\right) +\\
&\ddot{q}_1\left(r_3\left(1 + CA\right) - r_2\left(S3 + SC\right) + 2r_1 CB\right) + 2\dot{q}_1\dot{q}_3\left(0.5r_2\left(CC - C3\right) + r_3 SA + r_1 SB\right)\\
&\quad + 2\dot{q}_1\dot{q}_2\left(r_1 SB + r_2 CC + r_3 SA\right),\\
&Y_{1,15} = \left(d_2 + d_3\right)\left(SB\left(\dot{q}_2 + \dot{q}_3\right)^2 + CB\left(\ddot{q}_2 + \ddot{q}_3\right)\right) +\\
&\ddot{q}_1\left(2r_1 SB + r_2\left(CC - C3\right) + r_3 SA\right)\\
&\quad - 2\dot{q}_1\dot{q}_3\left(r_1 CB - 0.5r_2\left(SC + S3\right) + r_3 CA\right) - 2\dot{q}_1\dot{q}_2\left(r_1 CB - r_2 SC + r_3 CA\right),
\end{aligned}
\tag{1}
$$

$$
\begin{aligned}
&Y_{2,1} = 0,\ Y_{2,2} = -0.5SD\dot{q}_1^2,\ Y_{2,3} = -CD\dot{q}_1^2,\\
&Y_{2,4} = C2\ddot{q}_1,\ Y_{2,5} = -S2\ddot{q}_1,\ Y_{2,6} = \ddot{q}_2,\\
&Y_{2,7} = \dot{q}_1^2\left(r_2SD - r_1C2\right) + 2r_2\ddot{q}_2 + gS2 - d_2C2\ddot{q}_1,\\
&Y_{2,8} = -\dot{q}_1^2\left(r_2CD + r_1S2\right) - gC2 - d_2S2\ddot{q}_1\\
&Y_{2,9} = -r_2C2\ddot{q}_1,\\
&Y_{2,10} = \dot{q}_1^2\left(0.5r_2^2SD - r_1r_2C2\right) + r_2^2\ddot{q}_2 + r_2gS2 - r_2d_2C2\ddot{q}_1\\
&Y_{2,11} = CA\dot{q}_1^2\\
&Y_{2,12} = -SB\ddot{q}_1,\\
&Y_{2,13} = CB\ddot{q}_1\\
&Y_{2,14} = 2\ddot{q}_2\left(r_3 - r_2S3\right) - \dot{q}_1^2\left(r_1SB + r_2CC + r_3SA\right) + \ddot{q}_3\left(2r_3 - r_2S3\right)\\
&- gCB - r_2C3\left(\dot{q}_3^2 + 2\dot{q}_2\dot{q}_3\right) - \left(d_2 + d_3\right)SB\ddot{q}_1\\
&Y_{2,15} = \dot{q}_1^2\left(r_1CB - r_2SC + r_3CA\right) - gSB + \left(d_2 + d_3\right)CB\ddot{q}_1\\
&- 2r_2C3\ddot{q}_2 - r_2C3\ddot{q}_3 + r_2S3\dot{q}_3\left(2\dot{q}_2 + \dot{q}_3\right)\\
&Y_{3,1} = Y_{3,2} = Y_{3,3} = Y_{3,4} = Y_{3,5} = Y_{3,6} = Y_{3,7} = Y_{3,8} = Y_{3,9} = 0,\ Y_{3,10} = 0\\
&Y_{3,11} = CA\dot{q}_1^2,\ Y_{3,12} = -SB\ddot{q}_1,\ Y_{3,13} = CB\ddot{q}_1\\
&Y_{3,14} = 2r_3\ddot{q}_3 - \dot{q}_1^2\left(r_1SB + 0.5r_2\left(CC - C3\right) + r_3SA\right) +\\
&\ddot{q}_2\left(2r_3 - r_2S3\right) - gCB + r_2C3\dot{q}_2^2 + \left(d_2 + d_3\right)SB\ddot{q}_1\\
&Y_{3,15} = \dot{q}_1^2\left(r_1CB - 0.5r_2\left(SC + S3\right) + r_3CA\right) - r_2S3\dot{q}_2^2 +\\
&\left(d_2 + d_3\right)CB\ddot{q}_1 - r_2C3\ddot{q}_2 - gSB
\end{aligned}
$$

where

$$
\begin{aligned}
&SA = \sin\left(2q_2 + 2q_3\right),\ CB = \cos\left(2q_2 + q_3\right),\ SC = \sin\left(q_2 + q_3\right),\\
&SB = \sin\left(2q_2 + q_3\right),\ CC = \cos\left(q_2 + q_3\right),\ SD = \sin\left(2q_2\right),\\
&CD = \cos\left(2q_2\right),\ S2 = \sin\left(q_2\right),\ C2 = \cos\left(q_2\right),\\
&S3 = \sin\left(q_3\right),\ C3 = \cos\left(q_3\right).
\end{aligned}
\tag{2}
$$

$$
\begin{aligned}
&\theta_{B1} = I_{zz1} + I_{a1} + I_{yy2} + I_{yy3} + \left(d3^2 + r4^2\right) \\
&\left(m_2 + m_3 + m_4 + m_5 + m_6\right), \\
&\theta_{B2} = I_{xx2} - I_{yy2} - d_3^2\left(m_3 + m_4 + m_5 + m_6\right), \\
&\theta_{B3} = I_{xy2}, \\
&\theta_{B4} = I_{xz2} - d_3 m_3 z_3, \\
&\theta_{B5} = I_{yz2}, \\
&\theta_{B6} = I_{zz2} + I_{a2} + d_3^2\left(m_3 + m_4 + m_5 + m_6\right), \\
&\theta_{B7} = m_2 x_2 + d_3\left(m_3 + m_4 + m_5 + m_6\right), \\
&\theta_{B8} = m_2 y_2, \\
&\theta_{B9} = I_{xx3} + I_{yy4} + 2r_4 m_4 z_4 + \left(r_4^2 - d_4^2\right)\left(m_4 + m_5 + m_6\right) \\
&- I_{yy3}, \\
&\theta_{B10} = I_{xy3} - d_4 m_4 z_4 - d_4 r_4\left(m_4 + m_5 + m_6\right), \\
&\theta_{B11} = I_{xz3}, \\
&\theta_{B12} = I_{yz3}, \\
&\theta_{B13} = I_{zz3} + I_{yy4} + 2r_4 m_4 z_4 + \left(d_4^2 + r_4^2\right)\left(m_4 + m_5 + m_6\right), \\
&\theta_{B14} = m_3 x_3 + d_4\left(m_4 + m_5 + m_6\right), \\
&\theta_{B15} = m_3 y_3 + m_4 z_4 + r_4\left(m_4 + m_5 + m_6\right),
\end{aligned}
\tag{3}
$$

where r_i and d_i , for $i = 1, 2, \ldots, 6$, are MDH parameters of the KUKA robot.

Appendix B: PUMA 560 ROBOT Dynamics

$$M(\mathbf{q})\,\ddot{\mathbf{q}} + N(\mathbf{q}, \dot{\mathbf{q}}) = \boldsymbol{\tau}, \tag{4}$$

where

- Inertial Matrix :

$$M = \begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \end{bmatrix}, \tag{5}$$

  where

$$\begin{aligned} &M_{11} = -0.2CB + 0.76CB - 0.03C3 - 0.11CA - 0.2S3,\ M_{12} = 0.138S2, \\ &M_{13} = -0.006CC,\ M_{21} = M_{12},\ M_{22} = 1.91 - 0.063C3 - 0.41S3, \\ &M_{23} = 0.25 - 0.031C3 - 0.2,\ M_{31} = M_{13},\ M_{32} = M_{23},\ M_{33} = 0.252 \end{aligned} \tag{6}$$

- Coriolis, centripetal and gravity terms

$$N = \begin{bmatrix} N_1 \\ N_2 \\ N_3 \end{bmatrix}, \tag{7}$$

where

$$
\begin{aligned}
N_1 =& 0.14\dot{q}^2 C2 - 0.022\dot{q}_1\dot{q}_2 CA - 0.022\dot{q}_1\dot{q}_3 CA + 0.03\dot{q}_1\dot{q}_3 S3 + \\
& 0.216\dot{q}_1\dot{q}_2 SA + 0.216\dot{q}_1\dot{q}_3 SA - 0.411\dot{q}_1\dot{q}_2 CB - 0.205\dot{q}_1\dot{q}_3 CB \\
& + 0.0624\dot{q}_1\dot{q}_2 SB + 0.0312\dot{q}_1\dot{q}_3 SB - 1.52\dot{q}_1\dot{q}_2 SD + 0.0135\dot{q}_2\dot{q}_3 SC \\
& - 0.205\dot{q}_1\dot{q}_3 C3, \\
N_2 =& 0.71CC + 4.67SC + 36.5C2 + 1.02S2 + 0.0109\dot{q}_1^2 CA + \\
& 0.0312\dot{q}_3^2 S3 - 0.11\dot{q}_1^2 SA + 0.205\dot{q}_1^2 CB - 0.0312\dot{q}_1^2 SB \\
& + 0.759\dot{q}_1^2 SD - 0.205\dot{q}_3^2 C3 + 0.0624\dot{q}_2\dot{q}_3 S3 - 0.411\dot{q}_2\dot{q}_3 C3, \\
N_3 =& + 0.7CC + 4.67SC - 0.016\dot{q}_1^2 S3 - 0.0312\dot{q}_2^2 S3 - 0.11\dot{q}_1^2 SA + \\
& 0.103\dot{q}_1^2 CB - 0.0156\dot{q}_1^2 SB + 0.103\dot{q}_1^2 C3 + 0.205\dot{q}_2^2 C3,
\end{aligned}
\tag{8}
$$

with

$$
\begin{aligned}
& SA = \sin\left(2q_2 + 2q_3\right), \; CA = \cos\left(2q_2 + 2q_3\right), \\
& SB = \sin\left(2q_2 + q_3\right), \; CB = \cos\left(2q_2 + q_3\right), \\
& SC = \sin\left(q_2 + q_3\right), \; CC = \cos\left(q_2 + q_3\right), \\
& SD = \sin\left(2q_2\right), \; CD = \cos\left(2q_2\right), \\
& S2 = \sin\left(q_2\right), \; C2 = \cos\left(q_2\right), \\
& S3 = \sin\left(q_3\right), \; C3 = \cos\left(q_3\right).
\end{aligned}
\tag{9}
$$

References

[1] M. S. Alshamasin, F. Ionescu, and R. T. Al-Kasasbeh. Kinematic modeling and simulation of a scara robot by using solid dynamics and verification by matlab/simulink. *European Journal of Scientific Research*, **37**:388–405, 2009. 123

[2] N. Amann, D.H. Owens, and E. Rogers. Iterative learning control using optimal feedback and feedforward actions. *International Journal of Control*, **65**:277–293, 1996. 61

[3] S. Arimoto. *Control Theory of Nonlinear Mechanical Systems: A Passivity-Based and Circuit-Theoretic Approach*. Oxford Univ. Press, Oxford, U.K., 1996. 59

[4] S. Arimoto, S. Kawamura, and F. Miyazaki. Bettering operation of dynamic systems by learning: A new control theory for servomechanism or mechatronics systems. In *Proceedings of 23rd Conference on Decision and Control*, Las Vegas, 1984. 59, 60

[5] B. Armstrong. On finding exciting trajectories for identification experiments involving systems with non-linear dynamics. *Proceeding IEEE International Conference on Robotics and Automation*, pages 1131–1139, 1987. 33

[6] B. Armstrong, O. Khatib, and J. Burdick. The explicit dynamic model and inertial parameters of the puma 560 arm. *Proceeding IEEE International Conference of Robotics and Automation*, **1**:510–518, 1986. 28, 150

[7] A. Astolfi, D. Karagiannis, and R. Ortega. *Nonlinear and Adaptive Control with Applications*. Springer, London, 2008. 86

[8] T. Balkan. A dynamic programming approach to optimal control of robotic manipulators. *Mechanics research Communication*, **25**:225–230, 1998. 6

[9] S.P. Banks. *Mathematical Theories of Nonlinear Systems*. Prentice-Hall, London, 1988. 104

[10] S.P. Banks and K. Dinesh. Approximate optimal control and stability of nonlinear finite- and infinite-dimensional systems. *Annals of Operations Research*, **98**:1944, 2000. 104

[11] F. Barbosa and F. Reyes. Optimal position control for robot manipulators. *Proc. of the IEEE Midwesr Symposium on Circuits and Systems*, pages 1043–1046, 2003. 50

[12] R. H. Bartels, J. C. Beatty, and B. A. Barsky. *An Introduction to Splines for Use in Computer Graphics and Geometric Modeling*. Morgan Kaufmann, California, 1987. 109

[13] M. Bartholomew-Biggs. *Nonlinear Optimization*. Springer, Berlin, 2008. 51

[14] V.M. Becerra. Solving optimal control problems with state constraints using nonlinear programming and simulation tools. *IEEE Trans. on Education*, **47**:377–384, 2004. 50

[15] R. Bellman. *Dynamic Programming*. Princeton University Press, Princeton, New Jersey, 1957. 5

[16] D.P. Bertsekas. *Dynamic Programming and Optimal Control I, II*. Athena Scientific, Belmont, Massachusetts, 1995. 46

[17] J.T. Betts. *Practical Methods for Optimal Control Using Nonlinear Programming*. Society for Industrial and Applied Mathematics, Philadelphia, 2001. 5

[18] M. B. Biggs. *Nonlinear optimization with engineering applications*. Springer, Hartfield, 2008. 51

[19] J. E. Bobrow. Optimal robot path planning using the minimum time criterion. *IEEE Journal of Robotics and Automation*, **4**:443–450, 1988. 6

[20] J. E. Bobrow, S. Dubowsky, and J. S. Gibson. Time optimal control of robot manipulators along specified paths. *The International Journal of Robotic research*, **4(3)**:3–17, 1985. 6

[21] J.E. Bobrow. *Optimal Control of Manipulators*. PhD thesis, University of California, Los Angeles, 1982. 5

[22] H.G. Bock, , and K.J. Plitt. A multiple shooting algorithm for direct solution of optimal control problems. In *Proceeding of 9th IFAC World Congress*, pages 243–247, Budapest, 1984. 7, 48

[23] C. D. Boor. *A Practical Guide to Splines*. Springer, New York, 2001. 109

[24] A.E. Bryson and Y.C. Ho. *Applied optimal control, Optimization, Estimation and Control*. Taylor & Francis, New york, 1975. 57

[25] B. Bukkems, D. Kostic, and M. Steinbuch. Learning-based identification and iterative learning control of direct-drive robots. *IEEE Transaction On Control Technology*, **13**:537–549, 2005. 4, 29

[26] G. Calafiore, M. Indri, and B. Bona. Robot dynamic calibration: Optimal excitation trajectories and experimental parameter estimation. *Journal of Robotic Systems*, **18**[2]:55–68, 2001. 33

[27] B. Cao, G. I. Dodds, and G. W. Irwin. Constrained time-efficient and smooth cubic spline trajectory generation for industrial robots. *IEEE Proceedings of Control Theory and Applications*, **144**[5]:467–475, 1997. 53

[28] B. Cao, G.I. Dodds, and G.W. Irwin. Constrained time-efficient and smooth cubic spline trajectory generation for industrial robots. *IEE Proceedings of Control Theory and Applications*, **144**:467–475, 2002. 109

[29] S. P. Chan and H. P. Chen. An efficient algorithm for identification of scara robot parameters including drive characteristics. *25th Annual Conference of the IEEE Industrial Electronics Society*, pages 1014–1019, 1999. 97

[30] Y. C. Chen. Solving robot trajectory planning problems with uniform cubic b-splines. *Optimal Control Appl Methods*, **12**[4]:247–261, 1991. 53

[31] T. Chettibi, H.E. Lehtihet, M. Haddada, and S. Hanchi. Minimum cost trajectory planning for industrial robots. *European Journal of Mechanics A/Solids*, **23**:703715, 2004. 114

[32] Abb Company. *Products Manual of IRB 140.*, 2001. 156

[33] D. Constantinescu and E. A. Croft. Smooth and time-optimal trajectory planning for industrial manipulators along specified paths. *Journal of Robotic Systems*, **17**[5]:233–249, 1986. 101

[34] P.K. De, T.J., and A.K. Bejczy. Synthesis of trajectory planning using semi-algebraic sets and control of manipulators. *IEEE International Conference on Robotics and Automation*, **4**:2742 – 2748, 1994. 109

[35] J. Denavit and R. S. Hartenberg. A kinematic notation for lower-pair mechanisms based on matrices. *Trans ASME Journal of Applied Mechanics*, **23**:215–221, 1955. 3

[36] M. Diehl, H.G. Bock, H. Diedam, and P.B Wieber. Fast direct multiple shooting algorithms for optimal robot control. *NCIS, Fast Motions in Biomechanics and Robotics*, **340**:65–93, 2006. 5, 7, 48, 57

[37] C.R. Dohrmann and R.D. Robinett. Robot trajectory planning via dynamic programming. In *Proceedings of the Fifth International Symposium on Robotics and Manufacturing, and Applications*, pages 75–81, 1994. 6

[38] E. Dombre and W. Khalil. *Modeling, Performance Analysis and Control of Robot Manipulators*. London, 2007. 4

[39] S. Dubowsky, N. A. Norris, and Z. Shiller. Time optimal trajectory planning for robotic manipulators with obstacle avoidance: A cad approach. *IEEE International Conference on Robotics and Automation*, pages 1906–1912, 1986. 6

[40] P. Dutkiewicz, K. R. Kozlowski, and W. S. Wroblewski. Experimental identification of robot and load dynamic parameters. *Second IEEE Conference on Control Applications*, **2**:767–776, 1993. 29, 35

[41] M. Egerstedt and C. F. Martin. Optimal trajectory planning and smoothing splines. *Automatica*, **37**:1057–1064, 2001. 114

[42] A. Gasparetto and V. Zanott. A technique for time-jerk optimal planning of robot trajectories. *Robotics and Computer-Integrated Manufacturing*, **24**:415426, 2008. 6, 7

[43] A. Gasparetto and V. Zanotto. Optimal trajectory planning for industrial robots. *Advances in Engineering Software*, **41**:548556, 2010. 114

[44] M. Gautier. Dynamic identification of robots with power model. *International Conference on Robotics and Automation*, **3**:1922 – 1927, 1997. 29, 35

[45] M. Gautier and P. Poigent. Extended kalman filtering and weighted least squares dynamic identification of robot. *Control Engineering Practice*, **9**:1361–1372, 2001. 4, 29

[46] H.P. Geering. *Optimal Control with Engineering Applications*. Springer, Berlin, 2007. 43, 47, 48

[47] KUKA Schweissaniagen-Roboter GmbH. *KUKA IR 364/10.0-VK 10 Technical Data*, 1995. 159

[48] Y. Guan, K. Yokoi, O. Stasse, and A. Kheddar. On robotic trajectory planning using polynomial interpolations. *2005 IEEE International Conference on Robotics and Biomimetics*, pages 111–116, 2006. 6, 7

[49] F. J. L. Guerrero and A. R. Gomez. isa.umh.es/asignaturas/crss/irb140cr.ppt. 2011. 154

[50] K. Hamamoto and T. Sugie. Iterative learning control for robot manipulators using the finite dimensional input subspace. *IEEE Transactions on Robotics and Automation*, **18**:632–635, 2002. 29

[51] F.B. Hildebrand. *Introduction to Numerical Analysis*. Dover Publications Inc., NEW YORK, second edition edition, 1987. 5

[52] D.G. Hull. *Optimal Control Theory for Applications*. Springer, Berlin, 2003. 45, 48

[53] G. T. Huntington and A. V. Rao. Optimal configuration of spacecraft formations via a gauss pseudospectral method. *Charles Stark Draper Laboratory*, pages 1–18, 2005. 43

[54] A. Jochheim, M. Gerke, and A. Bischoff. *Modeling and simulation of robotic systems*. Hagen, Germany, 2000. 143

[55] W. Khalil and J.F. Kleinfinger. A new geometric notation for open and closed loop robots. *IEEE International Conference on Robotics and Automation.*, pages 1174–1179, 1986. 3, 13

[56] D. Kirk. *Optimal Control Theory, an Introduction*. Dover publication Inc., 1988. 45, 46, 48

[57] K.J. Kyriakopoulos and G.N. Saridis. Minimum jerk path generation. In *IEEE 1988 International Conference on Robotics and Automation*, Philadelphia, USA, 1988. 50, 101

[58] F.L. Lewis, D.M. Dawson, and C.T. Abdallah. *Robot Manipulator Control Theory and Practice*. Marcel Dekker Inc., 2004. 4, 49

[59] K. Y. Lim and M. Eslami. Robust adaptive controller designs for robot manipulator systems. *IEEE Journal of Robotics and Automation*, **RA-3**[1]:54–66, 1987. 86

[60] C. Lin, P. Chang, and J. Luh. Formulation and optimization of cubic polynomial joint trajectories for industrial robots. *IEEE transaction on Automatic Control*, **AC-28**:467–475, 1983. 109

[61] L. Ljung. *System Identification, Theory for the User*. Prentice-Hall, New Jersey, second edition edition, 1999. 29

[62] J. M. Maciejowski. *Predictive Control with Constraints*. Prentice Hall, New York, 2000. 5

[63] MathWorks. Optimization toolbox 4 users guide. Technical report, 2009. 53

[64] D. S. Naidu. *Optimal Control Systems*. CRC Press, Washington DC, USA, 2003. 47

[65] M. R. Nasiri. An optimal learning control for continuous-time systems. In *IEEE Industrial Electronics, IECON 2006 - 32nd Annual Conference*, pages 114–119, 2006. 61

[66] S. B. Niku. *Introduction to Robotics, Analysis, Control, Applications*. John Willey & Sons, USA, second edition edition, 2011. 3, 20

[67] J. Nocedal and S.J. Wright. *Numerical Optimization*. Springer, Berlin, second edition edition, 1999. 5

[68] M.M. Olsen and H.G. Petersen. A new method for estimating parameters of a dynamic robot model. *IEEE Transaction On Robotic and Automation*, **17**:95 – 100, 2001. 29

[69] H. Ozaki and C. j. Lin. Optimal b-spline joint trajectory generation for collision free movements of a manipulator under dynamic constraints. pages 3592–3597, Minneapolis, Minnesota, 1996. 53

[70] R. P. Paul. *Robot Manipulators: Mathematics, Programming, and Control*. MIT Press, Massachusetts, 1981. 4, 134

[71] P. Pedel and Y. Bestaoui. Actuator constraints in optimal motion planning of manipulators. In *IEEE International Conference on Robotics and Automation*, pages 2427–2432, Nagoya, Japan, 1995. 6

[72] L. Perko. *Differential Equations and Dynamical Systems*. Springer, Springer-Verlag, New York. Inc., 2001. 104, 105

[73] A. Piazzi and A. Visioli. Global minimum-jerk trajectory planning of robot manipulators. *IEEE transactions on industrial electronics,*, **47**[1]:140 – 149, 2002. 101

[74] L. S. Pontryagin, V. G. Boltyanskii, R. V. Gamkrelidze, and E. F. Mishchenko. *The Mathematical Theory of Optimal Processes*. John Wiley & Sons, New York, 1962. 5, 43

[75] R. Pytlak. *Numerical Methods for Optimal Control Problems with State Constraints*. Springer, Berlin, 1999. 48

[76] M. H. Ramezani and N. Sadati. Hierarchical optimal control of large-scale nonlinear chemical processes. *ISA Transactions*, **48**:38–47, 2009. 43

[77] R.D. Robinett, D.G. Wilson, G.R. Eisler, and J.E. Hurtado. *Applied Dynamic Programming for Optimization of Dynamical Systems*. Society for Industrial and Applied Mathematics, Philadelphia, 2005. 5

[78] M. T. Rodriguez and S. P. Banks. Linear approximations to nonlinear dynamical systems with applications to stability and spectral theory. *IMA Journal of Mathematical Control and Information*, **20**:89–103, 2003. 104

[79] M.T. Rodriguez and S.P. Banks. *Linear Time varying Approximations to Nonlinear Dynamical Systems with Applications in Control and Optimization*. Springer, Berlin Heidelberg, 2010. 104, 105

[80] L. L. Schumaker. *Spline Functions: Basic Theory*. Cambridge University Press, London, 2007. 109

[81] Z. Shiller. On singular points and arcs in path constrained time optimal motions. *ASME Dynamic System Control*, **42**:141–147, 1992. 6

[82] Z. Shiller. Time-energy optimal control of articulated systems with geometric path constraints. In *IEEE International Conference on Robotics and Automation*, San Diego, USA, 1994. 6

[83] Z. Shiller and S. Dubowsky. On the optimal control of robot manipulators with actuator and end-effector constraints. *IEEE International Conference in Robotic and Automation*, pages 614–620, 1985. 6

[84] Z. Shiller and H. Lu. Computation of path constrained time optimal motions with dynamic singularities. *Journal of Dynamic Systems*, **114**:34–40, 1992. 6

[85] K.G. Shin and N.D. Mckay. A dynamic programming approach to trajectory planning of robotic manipulators. *IEEE Trans. On Automatic Control*, **6**:491–500, 1986. 6

[86] J. J. E. Slotine and W. Li. *Applied Nonlinear Control*. Prentice Hall, New Jersey, 1991. 86

[87] S. F. P. Sraramago and V. Steffen. Optimization of the trajectory planning of robot manipulators taking into account the dynamic of the system. *Mechanism and Machine Theory*, **33**[7]:141–147, 1998. 6, 101

[88] S. F. P. Sraramago and V. Steffen. Optimal trajectory planning of robot manipulators in the presence of moving obstacles. *Mechanism and Machine Theory*, **35**[8]:1079–1094, 2000. 6

[89] O.V. Stryk and M. schlemmer. Optimal control of the industrial robot manutec r3. *Computational Optimal Control, International Series of Numerical Mathematics*, pages 367–382, 1994. 50, 101

[90] J. Swevers, C. Ganseman, D.B. Tukel, J.D. Schutter, and H.V. Brussel. Optimal robot excitation and identification. *IEEE Transaction On Robotic and Automation*, **13(5)**:730–740, 1997. 29, 33

[91] A. Tayebi. Adaptive iterative learning control for robot manipulators. *Proceedings of the American Control Conference*, pages 4518–4523, 2003. 61

[92] S. E. Thompson and R. V. Patel. Formulation of joint trajectories for industrial robots using bsplines. *IEEE Transactions on Industrial Electronics*, **IE-34**[2]:192–199, 2007. 53, 101

[93] J. Xu, S. K. Panda, and T. H. Lee. *Real time Iterative Learning Control*. Springer-Verlag London Limited, 2009. 59

[94] J. Xu and Y. Tan. *Linear and Nonlinear Iterative Learning Control*. Springer, Verlag Berlin Heidelberg, 2003. 59

[95] V. Záda. Optimal control of robots. In *9th International Conf. on Computer, Communication and Control Technology CCCT'03*, pages 330–335, 2003. 6, 7, 55, 111

[96] X. F. Zha. Optimal pose trajectory planning for robot manipulators. *Mechanism and Machine Theory*, **37**:10631086, 2002. 114

[97] F. Zhang, D. M. Dawson, M. S. d. Queiroz, , and W. E. Dixon. Global adaptive output feedback tracking control of robot manipulators. *IEEE Transactions on Automatic Control*, **45**[6]:1203–1208, 2000. 86

[98] J. Zhenga, S. P. Banksb, and H. Alleyneb. Optimal attitude control for three-axis stabilized flexible spacecraft. *Acta Astronautica*, **56**:519528, 2005. 104

Author's Publications

- Conference:

  [1]- A. Chatraei, Vision Based Control of a Robot Manipulator to Swing-up and Stabilizing Control of an Inverted Pendulum, *International Conference on Automation, Robotics and Control Systems*, Orlando, USA, pp. 97-102, 2009.

  [2]- V. Záda, A. Chatraei, D. Linder, Learning Control for Robot Manipulators, *Engineering Mechanics Conference*, Svratka, Czech Republic, pp. 1489-1494, 2009.

  [3]- V. Záda, A. Chatraei, D. Linder, Analytical Solution of Inverse task With Sixth Rotational Joints and Its Concrete Utilization, *Engineering Mechanics Conference, Svratka*, Czech Republic, pp. 1495-1502, 2009.

  [4]- A. Chatraei, M. A. Zanjani, M. Montazeri, M. Sanatizadeh, Prototype Design and Fabrication Of An Intelligent Surveillance And Guard Robot, *Second National Electrical Conference (NEEC2010)*, Iran, Najafabad , February, 2010.

  [5]- A. Chatraei, Solving Position Kinematics Of ABB IRB 140 Industrial Robot and its Motion Planning, *Second National Electrical Conference (NEEC2010)*, Iran, Najafabad , February, 2010.

- Journal:

  [6]- A. Chatraei, V. Záda, A Combined Optimal Control Technique for Robot Manipulators, *International Scientific Journal Acta Technica*, reviewing (will be published in Apr. 2011).

[7]- A. Chatraei, V. Záda, D. Lindr, Modeling and Identification of Positioning part of the KUKA robot, *International Scientific Journal Acta Technica*, reviewing.

[8]- A. Chatraei, V. Záda, Global Optimal Control of Robots, International Asian Journal of Control, (Submitted recently).