Output Feedback Adaptive Control for Robot Manipulators

by

DRAGAN JURKOVIC

B.Sc.

A thesis submitted to
the Faculty of Graduate Studies and Research
in partial fulfilment of
the requirements for the degree of
Master of Applied Science

Ottawa-Carleton Institute for Electrical and Computer Engineering

Department of Systems and Computer Engineering

Carleton University

Ottawa, Ontario

November, 2004

© Copyright

2004, Dragan Jurkovic

Library and
Archives Canada

Bibliothèque et
Archives Canada

Published Heritage
Branch

Direction du
Patrimoine de l'édition

395 Wellington Street
Ottawa ON K1A 0N4
Canada

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

ISBN: 0-494-00746-X

Our file Notre référence

ISBN: 0-494-00746-X

NOTICE:

The author has granted a non-exclusive license allowing Library and Archives Canada to reproduce, publish, archive, preserve, conserve, communicate to the public by telecommunication or on the Internet, loan, distribute and sell theses worldwide, for commercial or non-commercial purposes, in microform, paper, electronic and/or any other formats.

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

AVIS:

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque et Archives Canada de reproduire, publier, archiver, sauvegarder, conserver, transmettre au public par télécommunication ou par l'Internet, prêter, distribuer et vendre des thèses partout dans le monde, à des fins commerciales ou autres, sur support microforme, papier, électronique et/ou autres formats.

L'auteur conserve la propriété du droit d'auteur et des droits moraux qui protègent cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this thesis.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de cette thèse.

While these forms may be included in the document page count, their removal does not represent any loss of content from the thesis.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.

Canada

Abstract

The main focus of this thesis is comparison of the three adaptive control algorithms for robot manipulators. Algorithms #1 [1] and #2 [2] are model reference adaptive control algorithms while algorithm #3 [3] is an output feedback adaptive control algorithm with a high-gain observer. Algorithm #1 only requires measurement of the position signal to implement the adaptation and control laws. Algorithms #2 and #3 require measurement of the position signal and the estimate of the velocity signal. Algorithm #2 provides an estimate of the velocity signal by using a linear observer while algorithm #3 uses a high-gain observer. The algorithms' performance is tested through a number of simulations using different initial settings. The influence of the measurement and quantization noise on the algorithms' performance is investigated. For each of the three algorithms, we determined the algorithms design parameters that result in the best performance. Results have shown that algorithm #3 has outperformed algorithms #1 and #2 in simulation cases without noise. Algorithm #2 was shown to be the best choice in simulation cases with noise. We determined the maximum value of the high-gain observer design parameter ε for which algorithm #3 will asymptotically recover the performance achieved under state feedback. It is shown that the proof of stability for the algorithm with the high-gain observer does not have practical value.

Keywords

Adaptive Control, Model Reference Adaptive Control, Output Feedback Adaptive Control, High-Gain Observer, Observer, Separation Principle, Peaking Phenomenon, Robot Dynamic Model.

Acknowledgments

First of all, I would like to express my deepest gratitude to my supervisor, Professor Howard M. Schwartz, for his guidance, support and faith in me and my work throughout this research project. You have been my constant source of ideas and inspiration. Thank you for being very understanding when I needed some extra time to adapt myself to the new life role of being a father.

I would like to thank my fellow colleagues and friends for their support and patience during the last two years. In particular, thanks to John Daly and Ayman Ahmed for constructive conversations about the adaptive control of robot manipulators. I would like to thank Blazenka Powers for promptly solving all administrative problems that have occurred along the way.

I am grateful to my parents for their unconditional love and encouragement that helped me to overcome all kinds of obstacles. Thanks to my newborn son Noah for all the joyful moments that were so comforting during the long hours of work. And finally, thanks to my beloved wife and best friend Danijela for her unselfish love and unbounded support and certainly the happiness that she brought into my life.

Table of Contents

1.Introduction.....	1
1.1 Background and Motivation	1
1.2 Outline of the Thesis.....	5
2.Fundamentals of Adaptive Control.....	6
2.1 Introduction.....	6
2.2 Adaptive Control.....	7
2.2.1 Concept of Direct Adaptive Control	7
2.2.2 Concept of Indirect Adaptive Control.....	10
3.The Dynamic Model of Robot Manipulators.....	12
3.1 Introduction.....	12
3.2 Model of Robot Manipulator	13
3.2.1 Model of Manipulator - Form I.....	14
3.2.2 Model of Manipulator - Form II	17
3.2.3 Properties of Matrices M and C	18
3.2.4 Implementation of the Dynamic Model of the Robot.....	19
4.Description and Simulation of Algorithms	21
4.1 Introduction.....	21
4.2 Simulation Scenario	22
4.3 MRAC without Velocity Measurement - Algorithm #1	26
4.3.1 Description of Algorithm #1	26
4.3.2 Simulation of Algorithm #1	34
4.4 Nonlinear MRAC with Linear Observer - Algorithm #2.....	50
4.4.1 Description of Algorithm #2.....	50
4.4.2 Simulation of Algorithm #2.....	56
4.5 OFAC of Robot Manipulators using High-Gain Observer-Algorithm #3	76
4.5.1 Description of Algorithm #3	78

4.5.2	Selection of the Design Parameters for Algorithm #3	83
4.5.3	Observations about High-gain Observer Design Parameter $\mathcal{E}$	85
4.5.4	Observations about the Stability Proof of Algortihm #3	90
4.5.5	Simulation of Algorithm #3	99
5.Conclusions.....		129
5.1	Recommendations for Future work	133
References.....		134

List of Tables

4.1	Initial Settings for Different Simulation Cases.....	34
4.2	Numeric Values of Settling Times, Maximum Torques and Position Errors Obtained Through Simulations.....	39
4.3	Optimal Value of the Adaptation Gain γ	57
4.4	Initial Settings for Different Simulation Cases.....	58
4.5	Numeric Values of Settling Times, Maximum Torques and Position Errors Obtained Through Simulations.....	58
4.6	The Settling Times of Parameter estimation errors for Various Initial Parameter Values with Unsaturated and Saturated Torques.....	60
4.7	Initial Settings for Different Simulation Cases.....	100
4.8	Numeric Values of Settling Times, Maximum Torques and Position Errors Obtained Through Simulations.....	100

List of Figures

2.1	Concept of a Feedback Control.....	7
2.2	General Block Diagram of a Model Reference Adaptive Control.....	8
2.3	Block Diagram of a Self Tuning Regulator.....	11
3.1	Two DOF Robot Manipulator.....	16
3.2	Simulated Robot in Joint Space.....	16
4.1	Sequence of Unit Steps as a Reference Signal q_c (solid line), Desired Trajectory q_m (dash-dot line), $\dot{q}_m$ (dashed line), $\ddot{q}_m$ (dotted line).....	25
4.2	Unit Step as a Reference Signal q_c (solid line), Desired Trajectory q_m (dash-dot line), $\dot{q}_m$ (dashed line), $\ddot{q}_m$ (dotted line).....	25
4.3	General Controller Structure for MRAC approach.....	29
4.4	MRAC Controller Structure for each Joint of Robot Manipulator.....	30
4.5	Simulation Block Structure of Algorithm #1.....	33
4.6	Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$	40
4.7	Position Tracking Errors: e_1 (solid line), e_2 (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$	40
4.8	Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$	41
4.9	Torques [Nm] Applied to the Robot Joints, T_1 (solid line), T_2 (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$	41
4.10	Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=5$, $P_2(0)=10$	42
4.11	Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=5$, $P_2(0)=10$	42

4.12	Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) with Unit Step as a Reference Signal and $P_1(0)=1.5$, $P_2(0)=3$	43
4.13	Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) with Unit Step as a Reference Signal and $P_1(0)=1.5$, $P_2(0)=3$	43
4.14	Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$ and $e_i(0)=0.2$	44
4.15	Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ and $e_i(0)=0.2$	44
4.16	Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$ and White Noise $\sigma = 0.01$	45
4.17	Position Tracking Errors: e_1 (solid line), e_2 (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ and White Noise $\sigma = 0.01$	45
4.18	Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) and $P_1(0)=1.5$, $P_2(0)=3$ and White Noise $\sigma = 0.01$	46
4.19	Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$ and White Noise $\sigma = 0.003$	46
4.20	Position Tracking Errors: e_1 (solid line), e_2 (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ and White Noise $\sigma = 0.003$	47
4.21	Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ and White Noise $\sigma = 0.003$	47
4.22	Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$ and 10 Bits Quantization Noise.....	48
4.23	Position Tracking Errors: e_1 (solid line), e_2 (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ and 10 Bits Quantization Noise.....	48
4.24	Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ and 10 Bits Quantization Noise.....	49
4.25	Simulation Block Diagram for Algorithm #2.....	56

4.26	Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$	65
4.27	Position Tracking Errors: e_1 (solid line), e_2 (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$	65
4.28	Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$	66
4.29	Torques [Nm] Applied to the Robot Joints, T_1 (solid line), T_2 (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$	66
4.30	Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=5$, $P_2(0)=10$	67
4.31	Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=5$, $P_2(0)=10$	67
4.32	Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for Unit Step as Reference Signal and $P_1(0)=1.5$, $P_2(0)=3$	68
4.33	Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for Unit Step as Reference Signal and $P_1(0)=1.5$, $P_2(0)=3$	68
4.34	Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$ and $e_i(0)=0.2$	69
4.35	Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ and $e_i(0)=0.2$	69
4.36	Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$ and $e_i(0)=0.6$	70
4.37	Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ and $e_i(0)=0.6$	70
4.38	Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$ and White Noise $\sigma = 0.01$	71
4.39	Position Tracking Errors: e_1 (solid line), e_2 (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ and White Noise $\sigma = 0.01$	71

4.40	Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) and $P_1(0)=1.5$, $P_2(0)=3$ and White Noise $\sigma = 0.01$	72
4.41	Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$ and White Noise $\sigma = 0.003$	72
4.42	Position Tracking Errors: e_1 (solid line), e_2 (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ and White Noise $\sigma = 0.003$	73
4.43	Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ and White Noise $\sigma = 0.003$	73
4.44	Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$ and 10 Bits Quantization Noise.....	74
4.45	Position Tracking Errors: e_1 (solid line), e_2 (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ and 10 Bits Quantization Noise.....	74
4.46	Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ and 10 Bits Quantization Noise.....	75
4.47	Position Trajectory for the First Joint: Full State Feedback (solid line) and Output Feedback without Saturation for $\varepsilon = 0.0008$ (dotted line) and $\varepsilon = 0.01$ (dashed line).....	88
4.48	Position Trajectory for the Second Joint: Full State Feedback (solid line) and Output Feedback without Saturation for $\varepsilon = 0.0008$ (dotted line) and $\varepsilon = 0.01$ (dashed line).....	88
4.49	Position Trajectory for the First Joint: Full State Feedback (solid line) and Output Feedback with Saturation for $\varepsilon = 0.008$ (dotted line) and $\varepsilon = 0.05$ (dashed line).....	89
4.50	Position Trajectory for the Second Joint: Full State Feedback (solid line) and Output Feedback with Saturation for $\varepsilon = 0.008$ (dotted line) and $\varepsilon = 0.05$ (dashed line).....	89
4.51	Simulation Block Diagram for Algorithm #3.....	112
4.52	Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$	112

4.53	Position Tracking Errors: e_1 (solid line), e_2 (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$	113
4.54	Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$	113
4.55	Torques [Nm] Applied to the Robot Joints, T_1 (solid line), T_2 (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$	114
4.56	Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=5$, $P_2(0)=10$	114
4.57	Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=5$, $P_2(0)=10$	115
4.58	Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for Unit Step as Reference Signal and $P_1(0)=1.5$, $P_2(0)=3$	115
4.59	Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for Unit Step as Reference Signal and $P_1(0)=1.5$, $P_2(0)=3$	116
4.60	Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$ and $e_i(0)=0.2$	116
4.61	Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ and $e_i(0)=0.2$	117
4.62	Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$ and $e_i(0)=0.6$	117
4.63	Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ and $e_i(0)=0.6$	118
4.64	Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$, White Noise $\sigma = 0.01$ and $\varepsilon = 0.002$	118
4.65	Position Tracking Errors: e_1 (solid line), e_2 (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$, White Noise $\sigma = 0.01$ and $\varepsilon = 0.002$	119
4.66	Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) and $P_1(0)=1.5$, $P_2(0)=3$, White Noise $\sigma = 0.01$ and $\varepsilon = 0.002$	119

4.67	Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$, White Noise $\sigma = 0.01$ and $\varepsilon = 0.05$	120
4.68	Position Tracking Errors: e_1 (top), e_2 (bottom) for $P_1(0)=1.5$, $P_2(0)=3$, White Noise $\sigma = 0.01$ and $\varepsilon = 0.05$	120
4.69	Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) and $P_1(0)=1.5$, $P_2(0)=3$, White Noise $\sigma = 0.01$ and $\varepsilon = 0.05$	121
4.70	Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$, White Noise $\sigma = 0.003$ and $\varepsilon = 0.02$	121
4.71	Position Tracking Errors: e_1 (solid line), e_2 (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$, White Noise $\sigma = 0.003$ and $\varepsilon = 0.02$	122
4.72	Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$, White Noise $\sigma = 0.003$ and $\varepsilon = 0.02$	122
4.73	Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$ with Quantization Noise and $\varepsilon = 0.03$	123
4.74	Position Tracking Errors: e_1 (solid line), e_2 (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ with Quantization Noise and $\varepsilon = 0.03$	123
4.75	Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ and with Quantization Noise and $\varepsilon = 0.03$	124
4.76	Position Tracking Errors for the First Joint: Algorithm #1(solid line), Algorithm #2 (dotted line) and Algorithm #3 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$	124
4.77	Position Tracking Errors for the Second Joint: Algorithm #1(solid line), Algorithm #2 (dotted line) and Algorithm #3 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$	125
4.78	Parameter estimation error $\tilde{P}_1$ [kgm^2]: Algorithm #1 (solid line), Algorithm #2, (dotted line) and Algorithm #3 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$	125
4.79	Parameter estimation error $\tilde{P}_2$ [kgm^2]: Algorithm #1 (solid line), Algorithm #2, (dotted line) and Algorithm #3 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$	126

4.80	Position Tracking Errors for the First Joint with Quantization Noise: Algorithm #1(solid line), Algorithm #2 (dotted line) and Algorithm #3 (dashed line).....	126
4.81	Position Tracking Errors for the Second Joint with Quantization Noise: Algorithm #1(solid line), Algorithm #2 (dotted line) and Algorithm #3 (dashed line).	127
4.82	Parameter estimation error $\tilde{P}_1$ [kgm ²] with Quantization Noise: Algorithm #1 (solid line), Algorithm #2, (dotted line) and Algorithm #3 (dashed line).....	127
4.83	Parameter estimation error $\tilde{P}_2$ [kgm ²] with Quantization Noise: Algorithm #1 (solid line), Algorithm #2, (dotted line) and Algorithm #3 (dashed line).....	128

Chapter 1

Introduction

1.1 Background and Motivation

The focus of this thesis is on adaptive control of robot manipulators. The problem of controlling robot manipulators can be divided into regulation and trajectory following problems. In this thesis we will investigate the trajectory following problem. As a nonlinear system a robot manipulator is characterised with several issues which have to be overcome in order to achieve control with satisfactory performance.

The starting point for the successful design of the adaptive controller is an adequate dynamic model of the plant (in this case the robot manipulator). Obtaining an adequate dynamic model of the robot manipulator will provide the controller with the information necessary to construct an appropriate control signal. The control signal will compensate for the nonlinearities.

The dynamic model of a robot manipulator has time varying parameters. When the manipulator grasps a load, the parameters of the dynamic model are changed. The dynamic model, which is an adequate representation of the system before the load is seized, is not valid anymore. Adaptive control techniques have proven to be very successful in controlling systems with time varying parameters. Therefore, they are often used for control of robot manipulators. As a result, adaptive control has an ability to estimate the parameters' true values and consequently generate an appropriate control signal. On the other hand conventional controllers do not have ability to adapt their

control signal according to the changes in robot parameters. Accordingly, performance achieved with conventional controllers is degraded in comparison to adaptive controllers.

In order to increase the efficiency of the production lines, robot manipulators are mostly operated at high speed. During high speed robot operations issues of the nonlinear dynamics are even of more concern than in low speed operations. At high operational speeds, the centrifugal, Coriolis and inertial forces are increased. Therefore, these effects should not be omitted in the dynamic model of the manipulator. Moreover, elasticity of the manipulator links, coupling effects between links, joint frictions, backlashes and gravity forces are effects that should be considered during determination of the dynamic model of the manipulator.

The effects that are not included in the dynamic model are called unmodeled dynamics. The model accuracy and complexity increase with the number of the nonlinear effects included in the dynamic model. In order to avoid high complexity of the dynamic model some of the above mentioned effects are often omitted. That is a difficult task that requires years of engineering expertise. Due to the fact that unmodeled dynamics are always present, use of modern control theory like adaptive control is necessary for the control of the robot manipulators. That statement is particularly true in the case of high speed applications.

The literature on the control of robot manipulators can be divided into several strategies. At first robot manipulators were controlled by fairly simple control algorithms. The computers available were extremely slow and expensive when compared with modern computers. For that reason implementation of the correct dynamic model of the robot manipulator as a part of the control algorithm was impossible. Accordingly, performance achieved with these control algorithms is just acceptable for the low operational speeds of the robot manipulators [14]. This class of the control algorithms is based on linear control theory. The nonlinear robot manipulator dynamics are substituted with a linear approximation. In that approximation each joint of the robot manipulator is presented as a linear second order system. Moreover, coupling effects between the robot manipulator links is omitted. The next step is the design of an appropriate model reference adaptive controller (MRAC) or self tuning regulator (STR) for that linear system. The main characteristic of these design procedures is that they lacked proof of

stability. This is due to the fact that linear control theory used in the design of these controllers is valid just for the systems with time invariant parameters [23]. As mentioned earlier, the robot manipulator is a system with time varying parameters. Because the influence of unmodeled dynamics is particularly strong at high operational speeds, these algorithms are only used for low speed robot applications.

The newer group of control algorithms is based on the theory of feedback linearization [10, 25, 17]. This is a powerful tool for the design of nonlinear control systems. Feedback linearization transforms the nonlinear system into a linear system. One can then apply linear control theory to the transformed system. It is important to distinguish between feedback linearization and linearization by Taylor's series expansion. Feedback linearization is based on the principle of cancellation of nonlinear dynamics with a nonlinear control signal. The nonlinear control signal is actually output from the exact inverse of the robot manipulator dynamic model, while the input is the desired robot trajectory. One of the main representatives of this approach is an algorithm introduced by Craig [6] also known as the computed torque method. This algorithm is proven to be globally stable. Moreover, it is asymptotically stable if the desired trajectory is persistently exciting. This algorithm requires the measurement of the position, velocity and acceleration signals of robot manipulator joints. That is a disadvantage because on most industrial robot manipulators there are no sensors for the velocity and acceleration signal [16]. On the other hand, if they are available they are often very noisy. Nonetheless, this algorithm requires computation of the inverse of the inertial matrix in the adaptation law and this can lead to algorithm failure in a case of singular inertial matrix.

As a result, researchers have investigated a new class of adaptive control algorithms that do not require the acceleration signal. Slotine and Li [7] introduced an adaptive control algorithm for robot manipulators that uses a semi-positive definite Lyapunov function and requires persistent excitation to ensure parameter convergence. Otherwise, the algorithm could become unstable due to the parameter drift [8]. The Slotine and Li algorithm does not require the acceleration measurement or the inversion of the inertial matrix. This is a major contribution of this algorithm over Craig's [6] algorithm. On the other hand, the main drawbacks of Slotine and Li's algorithm [7] are high sensitivity to noise, and dependence of error dynamics on the estimated parameters

[8]. Schwartz [1] proposed an algorithm which goes one step further in terms of required signals. The Schwartz algorithm requires only the position signal to be measured, while the velocity signal is substituted by a velocity signal from the reference model which is always available and is not corrupted by noise. Stability of the algorithm is shown through computer simulations. The Schwartz [1] algorithm will be investigated more thoroughly in this thesis.

The new class of proposed algorithms requiring an observer is motivated by the fact that acceleration and velocity measurements are very noisy. As a result acceleration and velocity signals are estimated while the position signal is measured. These algorithms use the *separation principle* which allows the independent design of the observer and the state feedback controllers. The separation principle is a well known technique from linear systems theory and is applicable to a special class of nonlinear systems [12], while for other nonlinear systems stability is not guaranteed if the measured states are substituted with their estimates [16].

Accordingly, Schwartz [2] proposed the MRAC algorithm which uses a linear observer to estimate the joint velocity signal instead of a nonlinear observer. This simplifies the implementation. The theoretical proof of stability for this algorithm does not exist. Lee and Khalil [3] have presented an output feedback adaptive algorithm which uses a high gain observer to estimate velocity signals. The algorithm uses an adaptive law with a parameter projection feature. The authors state that the algorithm recovers performance achieved under full state feedback. The algorithm has two main drawbacks; there are difficulties in implementation of the high gain observer, and the algorithm requires estimation of several parameters before one can begin with the controller design.

Schwartz [2] and Lee and Khalil [3] algorithms along with Schwartz [1] algorithm are the main focus of this thesis. The thesis goal is to compare three adaptive algorithms for the control of robot manipulators. We use computer simulations to compare these algorithms based on both performance and implementation. Simulations are performed with two reference signals; unit step and a sequence of unit steps. The algorithms are tested with realistic level of a noise signal added to the system. Therefore, it is possible to compare algorithms in terms of stability, settling time, design complexity and

implementation issues. In particular we evaluate these algorithms based on complexity and effects of quantization and measurement noise.

1.2 Outline of the Thesis

In Chapter 2 we present some fundamental concepts of adaptive control for nonlinear systems. As a result, concepts of direct and indirect adaptive control are given. Moreover, motivational reasons for use of adaptive control for robot manipulators are given.

Chapter 3 presents the dynamics of a two degree of freedom robot manipulator. This is a dynamic model used by Schwartz [1, 2]. This model is reformulated to conform to model structure in Lee's [3] algorithm. The properties of the inertial matrix M and matrix C that represent the centrifugal and Coriolis forces are discussed. These properties are used for derivation of the adaptation law in Lee's [3] algorithm.

Chapter 4 presents in detail each of the three algorithms, and derivation of their adaptation laws. After that, each of the algorithms is simulated with a set of different reference signals and initial conditions, with and without noise added to the system.

In Chapter 5, simulation results obtained in the previous chapter are summarized and three algorithms are compared in terms settling times, design complexity and their susceptibility to measurement and quantization noise. Moreover recommendations for future research are given as well.

Chapter 2

Fundamentals of Adaptive Control

2.1 Introduction

The control of a system involves the use of all kinds of resources such as sensors, actuators and computers that run control algorithms. When the system is stable and it has desirable performance there is no need for control, but often that is not the case. If the linear system with known time invariant parameters has undesirable performance or it is unstable, it is possible just by employing the well-known concept of feedback control to change the behaviour of the system. According to Figure 2.1, an unstable system in combination with an adequate linear controller can be simply transformed into a stable system with performance tuned according to the desired specifications. The controller in this concept has constant parameters because the system has constant parameters as well. This simple control concept has one big problem, even though in theory it seems perfect, in the real world, most of the dynamic systems have constant or varying unknown parameters. Moreover, those dynamic systems are often nonlinear. As a result of that, we are not able to design the controller because we do not know the system parameters. Even if the system parameters are known, the use of the linear controller with constant parameters could result in an unstable closed-loop system because of controller's inability to adjust its output to a changing system. Accordingly, development of a control strategy that has the ability to successfully solve that problem was necessary. That control strategy is called adaptive control.

Figure 2.1: Concept of a Feedback Control

2.2 Adaptive Control

Adaptive controllers have parameters that are changing along with the change of the system parameters. When using adaptation or estimation techniques, the adaptive controller is capable of increasing accuracy of the controlled system when the parameters of the system are unknown. The mechanism that gives the relation between varying parameters of the plant and adjustable controller parameters is called the adaptation law. The purpose of the estimator is to estimate unknown or time varying plant parameters. The adaptive controller is nonlinear due to the fact that the adaptation law is always nonlinear. Generally, an adaptive controller has two loops: the normal feedback loop and the parameter adjustment loop. The normal feedback loop is actually an ordinary controller. The parameter adjustment loop is an adaptation mechanism whose purpose it is to adjust the controller parameters in order to decrease the error between the actual plant output and the desired output.

There are two main concepts of adaptive control: direct adaptive control and indirect adaptive control. In the concept of the direct adaptive control, the parameters of the adaptive controller are directly updated from the adaptation law. On the other hand, in indirect adaptive control, the plant parameters are estimated first and then the controller parameters are calculated by using the plant parameter estimates and algebraic design equations [17, 19].

2.2.1 Concept of Direct Adaptive Control

The main representative of the direct adaptive control concept is model reference adaptive control (MRAC). Generally, MRAC consists of four main parts such as the reference model, the controller, the adaptation law and the plant. Nevertheless, we have

added the linearizer as a fifth part of the MRAC block structure, due to the fact that this thesis is focused on nonlinear plants such as robot manipulators. The purpose of the linearizer is to compensate for the nonlinear dynamics of the system. The general block diagram of MRAC is given in Figure 2.2.

Figure 2.2: General Block Diagram of a Model Reference Adaptive Control.

The plant of the adaptive control system usually has unknown and varying parameters. Also, the system can have known and constant parameters. In that case, the ordinary feedback control will do the job. Moreover, in the case of a plant with known and constant parameters, the use of adaptive control is not recommended due to its complexity and higher implementation costs. In order to be able to apply the adaptive control technique, the structure of the dynamic model of the plant has to be known. The reference model defines the ideal behaviour of the adaptive control system. The controller parameters will be updated as long as the difference between reference model output and the actual plant output is different from zero. The reference model is defined by the designer of the adaptive control system in a way that the response of the reference model y_m to the reference signal r meets the design specification. The controller of the adaptive control system has to be parameterized, otherwise, it would not be possible to update its parameters when they are estimated. The controller has to be capable of

achieving perfect tracking of the reference model output when the parameters of the plant are known, otherwise use of the adaptive control is pointless. The parameters of the controller will be updated until perfect tracking is achieved. At that moment the estimates of the plant parameters are equal to their true values, assuming that the reference signal is “rich” enough. The purpose of the adaptation law is to constantly vary the controller parameter until the tracking error is minimized to zero. In order to achieve that goal, the adaptation law needs the difference of the actual and desired output signal and the control signal u as input signals [17, 19].

In the case of conventional controllers, the stability of the closed loop system can be proven in a number of ways using classical stability analysis techniques such as root-locus, Nyquist criterion, Hurwitz criterion and others. In adaptive control systems the stability of the closed loop system is hard to prove particularly when the plant is nonlinear. In that case, the use of modern stability theory such as Lyapunov and hyperstability theory is necessary. The problem of proving the stability of the adaptive control system is complex because the closed loop system has to be stable even when the parameters of the plant are varying. Actually, the system has to be stable for the infinite set of plants with the same structure and different parameters values.

The purpose of the linearizer is to compensate the nonlinear dynamics of the nonlinear plant. The use of the linearizer block results in a transformation of the nonlinear state equations into linear form, by cancelling original plant nonlinearities. Using the principles of the feedback linearization, the robot manipulator as a nonlinear system with coupling effect between joints can be transformed into a linear system where each joint is treated as double integrator. That combination of linearizer and the real robot manipulator is exploited by two adaptive algorithms presented in this thesis. In an ideal case, the linearizer can be conceived of as an exact inverse of the dynamic model of the nonlinear plant. If the dynamic model is totally accurate, the representation of the real nonlinear plant in combination with the linearizer will result in perfect linearization. In the case of perfect linearization, the resulting system is linear and has an infinite bandwidth [15].

If some of the states that are necessary for implementation of the control and adaptation laws are unavailable or too noisy, the observer has to be added to the block

diagram of the model reference adaptive control given in Figure 2.2. In practical applications that is always the case. For instance, robot manipulators usually have available just the joint position signals while the velocity and acceleration signals are not used because they are corrupted by noise. The observer uses the plant output signal y and control signal u to estimate unknown states.

2.2.2 Concept of Indirect Adaptive Control

The main representative of the concept of indirect adaptive control is the self-tuning regulator. The block diagram representing the self-tuning regulator is illustrated in Figure 2.3. The concept of the indirect adaptive control has as well two loops like the concept of direct adaptive control. The first loop consists of the controller and the plant. Within the second loop, the plant parameters are estimated and then the controller parameters are calculated. The first loop is identical in both concepts of adaptive control. The controlled plant has unknown or varying parameters. At each sample time the estimator provides the controller design block with estimates of the plant parameters. By using estimates of the plant parameters as if they are true parameter values, the controller design block computes the controller parameters by solving an algebraic design equation. When obtained, the controller parameters are passed to the controller that is parameterized with adjustable parameters. Now, the controller will be able to generate control signals based on the reference and feedback signals. This process is repeated at every sample of time.

A purpose of an estimator is to provide the estimates of the plant parameters. The estimator uses some of the parameter estimation techniques such as recursive least square algorithm (RLS) and the plant input and output signal to produce the parameter estimates. If the input signal to the plant is “rich” enough, the convergence of the plant parameter estimates to their true values is faster in the case of indirect adaptive control than in the case of direct adaptive control. This is due to the fact that the direct adaptation techniques are gradient based while the indirect adaptation techniques are based on RLS algorithms [14].

The main disadvantage of indirect adaptive control is performance degradation that can lead to instability when the input signal to the plant is constant or equal to zero. The problem can be partially solved by addition of a perturbation signal to the input

signal. Moreover, separation of the controller and estimator design has resulted in lack of proof of stability for indirect adaptive control algorithms. Due to these disadvantages, the direct adaptive control algorithms are preferred in applications where stability is an issue. All three algorithms presented in this thesis are direct adaptive control algorithms.

Figure 2.3: Block Diagram of a Self Tuning Regulator.

Chapter 3

The Dynamic Model of Robot Manipulators

3.1 Introduction

One of the most complex steps in designing a control system is deriving a mathematical model of the plant. This is an even more complex task in the case of a nonlinear plant such as a robot manipulator. Nonlinear systems have much more complex dynamics than linear systems. Nonlinearities such as dead zones, backlashes and hysteresis can be minimized by precise construction of joints and gears. On the other hand, nonlinearities such as Coriolis and centripetal forces cannot be minimized by precise construction. Moreover, their influence increases with the square of operational speed. If they are not included in the dynamic model, accuracy of the robot control will degrade quickly with the increase of the robot's operational speed.

The quality of the robot control is highly dependant on the accuracy of the robot dynamic model. The control techniques presented in this document are based on feedback linearization theory. As such they are using a linearizer as a part of the controller. The linearizer is a pre-filter whose dynamics is the exact inverse of the robot dynamics. In that case, a combination of the linearizer and the robot will result in a linear system, even though both of them by themselves are nonlinear. In short, the purpose of the linearizer is to cancel nonlinear robot dynamics. If during the process of modeling some of the nonlinear dynamics are overlooked, the linearizer is not an exact inverse of the robot

dynamics. Accordingly, the linearizer will not be able to cancel the overlooked nonlinear robot dynamics. Hence, the composition of the linearizer and the real robot will not result in a linear system. Therefore, the control techniques based on feedback linearization may not solve the path tracking problem, because the hypothesis on which they are based is violated.

3.2 Model of Robot Manipulator

The dynamic model of a robot gives a relation between torques that cause motion on one side of the equation and manipulator joint positions, velocities and accelerations on the other side. In order to obtain it, one will need to apply one of the methods of analytical mechanics such as Newton-Euler or the more efficient Euler-Lagrange method.

The Euler-Lagrange method is the most common method used for derivation of the robot model. In order to use it one has to determine the Lagrangian of the plant. The Lagrangian is defined as the difference in kinetic energy of the system and its potential energy. Accordingly, the first step is to obtain equations for the velocity of each link. One can then calculate the kinetic energy of each link at any manipulator position. The potential energy can be calculated from the link position. Hence, the Lagrangian of the system can be determined. Applying Lagrange's equations to the robot manipulator will result in a general solution for robot motions given by equation (3.1)[24].

$$T = M(q)\ddot{q} + C(q, \dot{q}) + G(q) + F(\dot{q}) + T_d \quad (3.1)$$

The term $T \in R^{n \times 1}$ is the vector of torques applied on the manipulator, while n is number of joints, $q \in R^{n \times 1}$ is position of the robot joint, $M \in R^{n \times n}$ is the manipulator mass matrix, $C \in R^{n \times 1}$ is the vector of Coriolis and centrifugal forces, $G \in R^{n \times 1}$ is the vector of gravity forces, $F \in R^{n \times 1}$ is vector of friction forces, while $T_d \in R^{n \times 1}$ represents disturbance torques. The manipulator model given by equation (3.1) is considered to be a very detailed model because most of the robot nonlinear dynamics are addressed in it.

The complexity of the model will result in complex analysis and computational inefficiency. Therefore some model simplifications are welcomed. Inertial torques represented by matrix M vary with manipulator motions as well as with load changes, and cannot be omitted. Coriolis and centrifugal forces represented by vector C cannot be

omitted as well, because their influence increases drastically at the high operating speeds of the robot manipulator. High operating speeds are very common for modern manipulators. Moreover, if we consider that the robot manipulator operates in a horizontal plane, gravity forces can be neglected without a significant error in model representation [8]. In favour of that is the fact that the influence of gravity forces on manipulator links can be compensated by the feedforward term [16]. In that case analysis conducted with ignored gravity forces is still conceptually valid. Friction forces consist of two parts, viscous friction which is function of joint velocity and Coulomb friction which is constant but sign dependant on joint velocity. Friction can be significantly reduced with the careful design of manipulator joints and use of high performance lubricants. As a result of that friction forces can be neglected in the dynamic model of the manipulator. Friction is a process that cannot accurately be modeled with a deterministic model. According to the control literature [20, 24, 26], the term T_d in equation (3.1) can be omitted.

3.2.1 Model of Manipulator - Form I

According to the recommendation given in the previous paragraph, the equation of the dynamics for a two degree of freedom (DOF) rigid body robot, which is illustrated in Figure 3.1 and presented in joint space in Figure 3.2, is given with the following equation,

$$T = M(q)\ddot{q} + C(q, \dot{q}) \quad (3.2)$$

The gravity term G is neglected, because the manipulator is operating in a horizontal plane. The manipulator has two joints, one between ground and the first link and the second one between the links. The links have an equal length $l_1=l_2=l=1m$ and concentrated masses m_1 and m_2 at the end of each link as it is shown in Figure 3.2.

According to Schwartz [1] the inertial matrix M and vector C representing Coriolis and centrifugal forces are given as.

$$M(q) = \begin{bmatrix} m_1 l^2 + 2m_2 l^2 + 2m_2 l^2 \cos q_2 & m_2 l^2 + m_2 l^2 \cos q_2 \\ m_2 l^2 + m_2 l^2 \cos q_2 & m_2 l^2 \end{bmatrix} \quad (3.3)$$

$$C(q, \dot{q}) = \begin{bmatrix} -2m_2 l^2 \dot{q}_1 \dot{q}_2 \sin q_2 - m_2 l^2 \dot{q}_2^2 \sin q_2 \\ m_2 l^2 \dot{q}_1^2 \sin q_2 \end{bmatrix} \quad (3.4)$$

where q_1 and q_2 are joint positions, for the first and second link. The equation of robot dynamics (3.2) can be written in linear regression form which is often used for the derivation of the adaptation law.

$$T = Y(q, \dot{q}, \ddot{q})P \quad (3.5)$$

where, $Y \in R^{n \times m}$ is the regressor matrix while $P \in R^{m \times 1}$ is the vector of parameters.

$$Y(q, \dot{q}, \ddot{q}) = \begin{bmatrix} \ddot{q}_1 & 2\ddot{q}_1 + \ddot{q}_2 + 2\ddot{q}_1 \cos q_2 - 2\dot{q}_1 \dot{q}_2 \sin q_2 - \dot{q}_2^2 \sin q_2 + \ddot{q}_2 \cos q_2 \\ 0 & \ddot{q}_1 + \ddot{q}_1 \cos q_2 + \ddot{q}_2 + \dot{q}_1^2 \sin q_2 \end{bmatrix} \quad (3.6)$$

$$P = \begin{bmatrix} m_1 l^2 \\ m_2 l^2 \end{bmatrix} = \begin{bmatrix} P_1 \\ P_2 \end{bmatrix} \quad (3.7)$$

The joint position values for the first and second joint are given in radians as q_1 and q_2 , respectively. The concentrated masses are m_1 and m_2 . This manipulator model is developed with the assumption that the links are rigid bodies.

Figure 3.1: Two DOF Robot Manipulator.

Figure 3.2: Simulated Robot in Joint Space.

3.2.2 Model of Manipulator - Form II

Equation (3.2) of rigid robot dynamics can also be written in another form (3.8), which is going to be called *form II* throughout this thesis. The robot dynamics given by equation (3.8) is often used in the literature about the control of robot manipulators. It utilizes the fact that with the proper definition of C , the matrix $\dot{M} - 2C$ is a skew-symmetric matrix [7]. Moreover, matrix C has some additional properties which are going to be presented in the next section. It is important to mention that in *form II*, C is a matrix while in *form I*, it is a vector.

$$T = M(q)\ddot{q} + C(q, \dot{q})\dot{q} \quad (3.8)$$

It is important to note that these two forms represent identical systems, and that *form II* can be derived from *form I* by usage of Christoffel symbols [7]. The inertial matrix given by equation (3.3) can be written as:

$$M = \begin{bmatrix} P_1 + 2P_2(1 + \cos q_2) & P_2(1 + \cos q_2) \\ P_2(1 + \cos q_2) & P_2 \end{bmatrix} = \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix} \quad (3.9)$$

while the term representing Coriolis and centrifugal forces is now a $C \in R^{2 \times 2}$ matrix where the elements C_{ij} has to be derived.

$$C(q, \dot{q}) = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} \quad (3.10)$$

The elements of above matrix can be obtained by using the following equation and known elements of the inertial matrix M (3.9)[7].

$$C_{i,j} = \frac{1}{2} \sum_{k=1}^n \frac{\partial M_{i,j}}{\partial q_k} \dot{q}_k + \frac{1}{2} \sum_{k=1}^n \left(\frac{\partial M_{i,k}}{\partial q_j} - \frac{\partial M_{j,k}}{\partial q_i} \right) \dot{q}_k \quad (3.11)$$

In the case of a two DOF robot $n=2$. Partial derivatives of the inertial matrix M are,

$$\begin{aligned}
\frac{\partial M_{11}}{\partial q_1} &= \frac{\partial M_{12}}{\partial q_1} = \frac{\partial M_{21}}{\partial q_1} = \frac{\partial M_{22}}{\partial q_1} = 0 \\
\frac{\partial M_{11}}{\partial q_2} &= -2P_2 \sin q_2 \\
\frac{\partial M_{12}}{\partial q_2} &= \frac{\partial M_{21}}{\partial q_2} = -P_2 \sin q_2 \\
\frac{\partial M_{22}}{\partial q_2} &= 0
\end{aligned} \tag{3.12}$$

hence elements of matrix C are:

$$\begin{aligned}
C_{11} &= \frac{1}{2} \left(\frac{\partial M_{11}}{\partial q_1} \dot{q}_1 + \frac{\partial M_{11}}{\partial q_2} \dot{q}_2 \right) + \frac{1}{2} \left[\left(\frac{\partial M_{11}}{\partial q_1} - \frac{\partial M_{11}}{\partial q_1} \right) \dot{q}_1 + \left(\frac{\partial M_{12}}{\partial q_1} - \frac{\partial M_{12}}{\partial q_1} \right) \dot{q}_2 \right] \\
&= -P_2 \dot{q}_2 \sin q_2
\end{aligned} \tag{3.13}$$

$$\begin{aligned}
C_{12} &= \frac{1}{2} \left(\frac{\partial M_{12}}{\partial q_1} \dot{q}_1 + \frac{\partial M_{12}}{\partial q_2} \dot{q}_2 \right) + \frac{1}{2} \left[\left(\frac{\partial M_{11}}{\partial q_2} - \frac{\partial M_{21}}{\partial q_1} \right) \dot{q}_1 + \left(\frac{\partial M_{12}}{\partial q_2} - \frac{\partial M_{22}}{\partial q_1} \right) \dot{q}_2 \right] \\
&= -P_2 (\dot{q}_2 + \dot{q}_1) \sin q_2
\end{aligned} \tag{3.14}$$

$$\begin{aligned}
C_{21} &= \frac{1}{2} \left(\frac{\partial M_{21}}{\partial q_1} \dot{q}_1 + \frac{\partial M_{21}}{\partial q_2} \dot{q}_2 \right) + \frac{1}{2} \left[\left(\frac{\partial M_{21}}{\partial q_1} - \frac{\partial M_{11}}{\partial q_2} \right) \dot{q}_1 + \left(\frac{\partial M_{22}}{\partial q_1} - \frac{\partial M_{12}}{\partial q_2} \right) \dot{q}_2 \right] \\
&= P_2 \dot{q}_1 \sin q_2
\end{aligned} \tag{3.15}$$

$$\begin{aligned}
C_{22} &= \frac{1}{2} \left(\frac{\partial M_{22}}{\partial q_1} \dot{q}_1 + \frac{\partial M_{22}}{\partial q_2} \dot{q}_2 \right) + \frac{1}{2} \left[\left(\frac{\partial M_{21}}{\partial q_2} - \frac{\partial M_{21}}{\partial q_2} \right) \dot{q}_1 + \left(\frac{\partial M_{22}}{\partial q_2} - \frac{\partial M_{22}}{\partial q_2} \right) \dot{q}_2 \right] \\
&= 0
\end{aligned} \tag{3.16}$$

accordingly matrix C in *form II* is given with following equation,

$$C(q, \dot{q}) = \begin{bmatrix} -P_2 \dot{q}_2 \sin q_2 & P_2 (\dot{q}_1 + \dot{q}_2) \sin q_2 \\ P_2 \dot{q}_1 \sin q_2 & 0 \end{bmatrix} \tag{3.17}$$

while inertial matrix M is identical to the matrix M from *form I*. Nonetheless, P_2 is a robot parameter, and it is given by the same equation as in *form I*, where $P_2 = m_2 l^2$.

3.2.3 Properties of Matrices M and C

If the robot manipulator dynamic model is given by equation (3.8), then matrices M and C have specific properties which are exploited by a number of adaptive control algorithms as part of the proof of stability and during the derivation of the adaptation law.

These properties are used in an adaptive algorithm by Lee and Khalil [3], and therefore, it is considered important to present them here. The Lee and Khalil algorithm is one of the algorithms being compared in this thesis.

With a suitable definition of the matrix C according to the (3.11), matrices M and C are not independent, because C is derived from the matrix M . Consequently the following skew-symmetric property holds [7].

$$(\dot{M} - 2C)^T = -(\dot{M} - 2C) \quad (3.18)$$

Therefore the following equation holds for all the time.

$$\dot{g}^T \left(\frac{1}{2} \dot{M} - C \right) \dot{g} = 0 \quad (3.19)$$

The matrix C has a linear dynamic structure in $\dot{q}$, hence the following equations are valid [5].

$$C(q, g)h = C(q, h)g \quad (3.20)$$

or,

$$C(q, g + \beta h) = C(q, g) + \beta C(q, h) \quad (3.21)$$

where β is constant while q , x , and y are vectors.

3.2.4 Implementation of the Dynamic Model of the Robot

In order to successfully implement the dynamic robot model given by equation (3.2), it is necessary to express joint acceleration as a function of torques, joint velocities and positions and robot parameters P_1 and P_2 . Using equations (3.5), (3.6) and (3.7) of the robot dynamics written in linear regression form, one can obtain the following two equations.

$$T_1 = \ddot{q}_1 P_1 + (2\ddot{q}_1 + \ddot{q}_2 + 2\dot{q}_1 \cos q_2 - 2\dot{q}_1 \dot{q}_2 \sin q_2 - \dot{q}_2^2 \sin q_2 + \ddot{q}_2 \cos q_2) P_2 \quad (3.22)$$

$$T_2 = (\ddot{q}_1 + \dot{q}_1 \cos q_2 + \ddot{q}_2 + \dot{q}_1^2 \sin q_2) P_2 \quad (3.23)$$

Using equation (3.22) one can express acceleration of the first joint $\ddot{q}_1$ as a function of $\ddot{q}_2$,

$$\ddot{q}_1 = \frac{T_1 + (2\dot{q}_1 \dot{q}_2 + \dot{q}_2^2) P_2 \sin q_2}{P_1 + 2P_2 + 2P_2 \cos q_2} - \frac{P_2 (1 + \cos q_2)}{P_1 + 2P_2 + 2P_2 \cos q_2} \ddot{q}_2 \quad (3.24)$$

Similarly, using equation (3.23) acceleration of the second joint $\ddot{q}_2$ can be expressed as a function of $\ddot{q}_1$,

$$\ddot{q}_2 = \frac{T_2 - \dot{q}_1^2 \sin q_2 P_2}{P_2} - (1 + \cos q_2) \ddot{q}_1 \quad (3.25)$$

By substituting $\ddot{q}_2$ in equation (3.24) with right side of equation (3.25) one gets

$$\ddot{q}_1 = \frac{T_1 + P_2 (2\dot{q}_1 \dot{q}_2 + \dot{q}_2^2) \sin q_2 - (1 + \cos q_2) (T_2 - P_2 \dot{q}_1^2 \sin q_2)}{P_1 + 2P_2 (1 + \cos q_2) - P_2 (1 + \cos q_2)^2} \quad (3.26)$$

and similarly equation (3.25) under the action of equation (3.24) will become

$$\ddot{q}_2 = \frac{\left(\frac{T_2}{P_2} - \dot{q}_1^2 \sin q_2 \right) (P_1 + 2P_2 (1 + \cos q_2)) - (1 + \cos q_2) (T_1 + P_2 (2\dot{q}_1 \dot{q}_2 + \dot{q}_2^2) \sin q_2)}{P_1 + 2P_2 (1 + \cos q_2) - P_2 (1 + \cos q_2)^2} \quad (3.27)$$

Thereafter, the joint acceleration, as a highest order derivative, is expressed as a function of the joint velocity and position. Using equations (3.26) and (3.27), it is finally possible to implement the dynamic model of the robot given by equation (3.28).

$$T = M(q)\ddot{q} + C(q, \dot{q}) \quad (3.28)$$

Chapter 4

Description and Simulation of Algorithms

4.1 Introduction

The goal of this thesis is the comparison of three adaptive control algorithms for robot manipulators. In this chapter each of the algorithms will be described. Each adaptive control algorithm will be tested by simulations and the results are compared.

As it is mentioned by Khalil [11], comparison of the algorithms in terms of their performance is a very delicate process. Any algorithm can have performance ranging from extremely good to extremely poor, depending only on the algorithm's design parameters. If the design parameters are tuned to its optimal values the algorithm will show very good performance for the particular simulation scenario. On the other hand, if the algorithm's design parameters are chosen in a straight forward manner, performance will likely degrade. In order to avoid giving a biased conclusion about the algorithm performance, each of the algorithm design parameters are tuned to the values obtained through an iterative design procedure¹. Procedures for determining the optimal values of the design parameters for each of these adaptive control algorithms do not exist.

It is suggested by Khalil [11] that algorithms should be tested on number of different systems in order to achieve an unbiased comparison of the algorithms. This factor is neglected during the simulations. As a result, the purpose of this comparison is to determine the most suitable algorithm for the robot manipulator and not for a general

¹ Design parameters are determined through a number of simulations while also considering recommendations given in the literature.

nonlinear system. Moreover, the robot manipulator is strictly predefined to be a two degree of freedom (DOF) rigid body robot operating in the horizontal plane. Accordingly, simulations are performed on the dynamic model of the robot manipulator that is specified in detail in chapter 3.

4.2 Simulation Scenario

In order to test these three algorithms [1, 2, 3] the following simulation scenario has been developed. Testing is performed in five simulation cases. As a reference signal q_c , we have used a sequence of unit steps with period of eight seconds. That reference signal is used in every simulation case except in the second simulation case. In the second simulation case we have used a unit step function as a reference signal. The reference signal is passed through the reference model given with the following transfer function.

$$W_m = \frac{4}{s^2 + 4s + 4} \quad (4.1)$$

Using the transfer function in equation (4.1) we were able to obtain the desired model trajectory q_m as well as its first and second order derivatives $\dot{q}_m$ and $\ddot{q}_m$. The output of this filter for the sequence of unit steps as an input signal is illustrated in Figure 4.1, while in the case of a unit step as an input signal the output is given in Figure 4.2.

In the first simulation case, we were testing the stability of algorithms and the convergence rate of estimated parameter values to their true values. The testing is performed for two different values of the initial parameter error. Besides that, as a part of the first simulation case, we have tried to determine maximum initial values of parameters $P_1(0)$ and $P_2(0)$, for which the tracking error still converging to zero and parameters P_1 and P_2 converge to their true values, $P_1 = m_1 l^2 = 1kgm^2$ while $P_2 = m_2 l^2 = 2kgm^2$.

The second simulation case involved determining the stability of the algorithms and settling times for the parameter and the tracking errors, when the reference signal is a unit step. The initial value of the parameter estimation errors are set to $\tilde{P}_1(0) = 0.5kgm^2$ and $\tilde{P}_2(0) = 1kgm^2$.

The third simulation case involved testing of the algorithms behaviour when the initial condition of the position tracking error is different from zero. This testing step is included because the algorithms which are using high-gain observers could exhibit impulsive behaviour, which is known in the control literature as the peaking phenomenon [22]. Two out of three algorithms compared in this thesis use observers. The Lee algorithm [3] uses a high-gain observer while the Schwartz algorithm [2] uses an ordinary linear observer. For those reasons, it will be interesting to see how these adaptive control algorithms will behave when the initial position tracking error $e_i(0)$ is non-zero. Testing is performed for two different values of initial position tracking error $e_i(0)$ without quantization or measurement noise. The initial value of the parameter estimation errors are set to $\tilde{P}_1(0) = 0.5kgm^2$ and $\tilde{P}_2(0) = 1kgm^2$.

The fourth and fifth simulation cases have tested the algorithms' susceptibility to measurement and quantization noise respectively. In both cases the initial parameter values were set to $P_1 = 1.5kgm^2$ and $P_2 = 3kgm^2$ while initial position tracking error $e_i(0)$ is set to zero. Simulations with measurement noise are performed for two levels of Gaussian noise with a standard deviation of $0.003rad$ and $0.01rad$. The position signal was corrupted with noise.

Quantization noise has a uniform distribution therefore it is implemented using the SIMULINK[®] block called *uniform noise generator*. The output of this block is uniformly spread between specified upper and lower bounds. These bounds are determined for the precision of 10 bits according to the following equation:

$$\pm \frac{2\pi}{2^{10}} = \pm \frac{0.00613}{2} = \pm 0.003065rad \quad (4.2)$$

where, 2π is the range of joint positions in radians, 10 bits precision is equivalent to 2^{10} and everything is divided by 2 due to the nature of the quantization error. It is important to mention that the noise generation in SIMULINK[®] is dependant on the integration step size used in the simulation [11]. Therefore simulations with measurement or quantization noise for each of the three algorithms are performed with the same integration step size.

The value of standard deviation for the lower level of the measurement noise and the value of the bounds of uniformly distributed quantization noise are almost equal. In

the case of lower level of measurement noise, noise signal can be even greater than $0.003rad$, while in the case of quantization noise, noise signal is always within $\pm 0.003065rad$. Due to that difference between Gaussian distribution and uniform distribution simulation results obtained with lower level of measurement noise will be different from results obtained with quantization noise.

We will be determining settling time as a measure of performance for the particular algorithm during each simulation step. Settling time t_s will be determined for the position tracking error as well as for each parameter estimation error, and it is defined according to the 1% criteria. The parameter estimation error is defined as difference between parameter estimate and parameter true value as $\tilde{P}_i = \hat{P}_i - P_i$. Therefore, for the position tracking error and parameter estimation error the settling time will be defined as the time t_s when the error gets within the range of $\pm 0.01 rad$ and $\pm 0.01 kgm^2$, respectively, and stays within that range for every $t > t_s$. According to the Ogata [21], settling time is usually defined according to the 1%, 2% or 5% tolerance band in order to make comparison of system responses easier. Measuring the settling time with loose criteria such as 5% tolerance band will result in some simulation of algorithms achieving almost the same settling time. Therefore, it would not be possible to distinguish which algorithm can reduce the tracking or parameter error more quickly. Therefore, our choice was to perform the simulations with the 1% tolerance band. Only in the simulation cases where the algorithm was unable to bound the tracking or parameter errors within a range of $\pm 0.01rad$ and $\pm 0.01kgm^2$, we have determined the settling time with a 5% tolerance band, and that is noted for every such case.

Figure 4.1: Sequence of Unit Steps as a Reference Signal q_c (solid line), Desired Trajectory q_m (dash-dot line), $\dot{q}_m$ (dashed line), $\ddot{q}_m$ (dotted line).

Figure 4.2: Unit Step as a Reference Signal q_c (solid line), Desired Trajectory q_m (dash-dot line), $\dot{q}_m$ (dashed line), $\ddot{q}_m$ (dotted line).

4.3 MRAC without Velocity Measurement - Algorithm #1

The model reference adaptive controller for robot manipulators which is described in this section has been proposed by Schwartz [1]. It is the first out of three algorithms that will be compared in this thesis. We will refer to this control method as algorithm #1 throughout this document. This algorithm fits into a class of model reference based algorithms. The algorithm can be used with feedback linearizable systems, such as the robot manipulator described in Chapter 3. The main advantage of this algorithm over other methods for robot manipulator control is that this algorithm requires just joint position measurements. Only the measured position is used in both the control and adaptation law. That is a significant advantage because joint acceleration and velocity measurements are very noisy. In order to avoid the use of noisy joint acceleration and velocity measurements in the adaptation law and linearizer, the following substitutions are made. The measured joint velocity signal $\dot{q}$ is replaced with the first order derivative of model trajectory $\dot{q}_m$, while the measured joint acceleration signal $\ddot{q}$ is replaced with the control signal u . In order to be able to perform computer simulation of this algorithm, we have to figure out how to mathematically represent each of the five blocks: model, control law, adaptation law, linearizer and robot, as it is illustrated in Figure 4.5.

4.3.1 Description of Algorithm #1

The reference model is chosen as a second order critically¹ damped system with natural frequency $\omega_n = 2rad/sec$, hence model transfer function is

$$W_m = W_{m1} = W_{m2} = \frac{\omega_n^2}{(s + \omega_n)^2} = \frac{\omega_n^2}{s^2 + 2s\omega_n + \omega_n^2} = \frac{4}{s^2 + 4s + 4} = \frac{B_m}{A_m} \quad (4.3)$$

Passing a reference signal q_c through the reference model, we were able to obtain the desired trajectory q_m and its derivatives $\dot{q}_m$ and $\ddot{q}_m$ as illustrated in Figures 4.1 and 4.2.

¹ Damping factor $\xi = 1$

We have chosen the second order reference model for each joint. Therefore, assuming perfect feedback linearization, the combination of the feedback linearization law and the robot joint can be represented as a double integrator. The reference model dynamics can be chosen separately for each manipulator joint. In this case we will use the same reference model for each joint. Accordingly, the reference model dynamics for each manipulator joint is:

$$q_{mi} = W_m(s)q_{ci} \quad (4.4)$$

as illustrated in Figure 4.5.

We are using the following dynamic model of the manipulator:

$$T = M(q)\ddot{q} + C(q, \dot{q}) \quad (4.5)$$

Derivation of the control law for each joint of the robot manipulator could be considered complicated because the dynamics of each joint are nonlinear. In order to derive the control law, we will assume that perfect linearization has occurred. Hence each robot manipulator joint together with the linearizer will result in a double integrator. Accordingly, we derive the control law for the plant which has its transfer function given by equation (4.6).

$$W_{p1} = W_{p2} = \frac{B(s)}{A(s)} = \frac{1}{s^2} \quad (4.6)$$

where W_{p1} and W_{p2} are transfer functions for each joint of robot manipulator with the assumption that perfect linearization has occurred.

Using the pole placement design method [19] and model reference control theory, knowing the linearized plant transfer function (4.6) and reference model transfer function (4.3), we can write the reference model transfer function in the form,

$$W_{mi} = \frac{T(s)B(s)}{R(s)A(s) + B(s)S(s)} = \frac{T(s)B(s)}{A_c(s)} = \frac{B_m}{A_m} = \frac{4}{s^2 + 4s + s} \quad (4.7)$$

according to Figure 4.3. Polynomial A_c is the characteristic polynomial of the closed loop system, while polynomials $T(s)$, $R(s)$ and $S(s)$ are controller polynomials given by equation (4.12). In order to have equation (4.7) correct, we must be able to factor polynomials $T(s)B(s)$ and the polynomial $A_c(s)$ in order to have cancellation of their factors. Cancelling of any unstable factors is forbidden because this can result in an

unstable system with very small changes of the plant parameters. In the general case we factor polynomial B into B^+ and B^- polynomials as it is given by equation (4.8). The polynomial B^+ is a monic and Hurwitz polynomial and it can be cancelled, while the polynomial B^- is unstable or almost unstable and it cannot be cancelled. Hence, the polynomial B^- must be part of the polynomial B_m . Accordingly we can factor the polynomial B_m into two factors B^- and B'_m as it is given by equation (4.8). The factor B^+ of the polynomial B must be cancelled with factor of the polynomial A_c . Therefore the factor B^+ must be part of the polynomial A_c according to the equation (4.7). Due to the fact that the factor B^+ is part of the polynomials B and A_c and that $A_c = R(s)A(s) + B(s)S(s)$, the factor B^+ has to be a part of polynomial R as well. Thus, we factor the polynomial R into two factors B^+ and R_1 according to equation (4.8). Since in this case the polynomial $B=1$, we write that $B^+ = B^- = 1$. After factoring the numerator and the denominator polynomials of the plant, the controller and the reference model transfer functions we write,

$$\begin{aligned} B &= B^+ B^- = 1 \Rightarrow B^+ = B^- = 1 \\ B_m &= B^- B'_m \Rightarrow B^- = 1; B'_m = 4 \\ A_c &= B^+ A_0 A_m \\ R &= B^+ R_1 \end{aligned} \tag{4.8}$$

A_0 is arbitrary chosen polynomial with roots much faster than system dynamics. Using equations (4.7) and (4.8) we can write the *Bezout identity*¹ (4.9) [19] as follows,

$$R(s)A(s) + S(s)R(s) = B^+(s)A_0(s)A_m(s) \tag{4.9}$$

Thereafter, we can define the order of each polynomial and then find unknown coefficients of each of them. As a result the following equation holds,

$$\deg A_m + \deg B^+ < 2 \deg A - 1 \tag{4.10}$$

It is necessary to introduce polynomial A_0 in order to be able to solve the *Bezout identity* with,

$$\deg A_0 = \deg A - 1 = 1 \Rightarrow A_0(s) = s + a_0 \tag{4.11}$$

¹ Also known as Diophantine or Aryabhata equation.

while the order of other polynomials is as follows,

$$\begin{aligned} \deg T &= \deg B'_m + \deg A_0 = 1 \Rightarrow T(s) = t_1 s + t_0 \\ \deg R_1 &= \deg A_m + \deg A_0 - \deg A = 1 \Rightarrow R_1(s) = s + r_0 \\ \deg S &= \deg A - 1 = 1 \Rightarrow S(s) = s_1 s + s_0 \end{aligned} \quad (4.12)$$

where $R_1(s)$ is a monic polynomial. After the action of (4.8) the *Bezout identity* is simplified to the equation,

$$R_1(s)A(s) + B^-(s)S(s) = A_0 A_m \quad (4.13)$$

Now we can arbitrary choose the coefficient a_0 of polynomial $A_0(s)$ in equation (4.11) to be $a_0 = 10$. After that it is possible to solve the simplified *Bezout identity* (4.13) and obtain the coefficients of polynomials $R(s)$ and $S(s)$ as,

$$r_0 = 14; s_1 = 44, s_0 = 40 \quad (4.14)$$

The coefficients of the polynomial $T(s)$ are calculated from equation (4.15).

$$T(s) = A_0 B'_m \quad (4.15)$$

hence,

$$t_1 = 4, t_0 = 40 \quad (4.16)$$

Accordingly, the polynomials of the controller from Figure 4.3 are,

$$\begin{aligned} R(s) &= s + 14 \\ S(s) &= 44s + 40 \\ T(s) &= 4s + 40 \end{aligned} \quad (4.17)$$

Figure 4.3: General Controller Structure for MRAC approach.

Figure 4.4: MRAC Controller Structure for each Joint of Robot Manipulator.

Now we can go back to the problem of determining the control law in our specific robot manipulator case. According to [1, 18] we can determine the equation for the virtual control law (4.18) for each joint from the standard controller structure given by Figure 4.4. The virtual control law is,

$$u_i = Kq_{ci} + d_0q_i + D_1^T (sI - \lambda_0)^{-1} bu_i + D_2^T (sI - \lambda_0)^{-1} bq_i \quad (4.18)$$

If we compare the two block diagrams given in Figures 4.3 and 4.4, we will note that both of them have the controller part. The controllers illustrated in Figures 4.3 and 4.4 are identical if their polynomial coefficients are computed according to equations (4.19) [18]. From the system of equations (4.19), we can calculate polynomial coefficients of the controller in Figure 4.4 from parameter values of the controller that is shown in Figure 4.3. The polynomial coefficients of the controller illustrated in Figure 4.3 are already determined and given by equation (4.17).

$$\begin{aligned} T(s) &= K\lambda(s) = K(s + \lambda_0) = 4(s + 10) \Rightarrow K = 4, \lambda(s) = s + 10 \\ R(s) &= (s + \lambda_0) - D_1^T b = s + 14 \Rightarrow D_1^T b = -4 \\ S(s) &= -D_2^T b - d_0(s + \lambda_0) = 44s + 40 \Rightarrow d_0 = -44, D_2^T b = 400 \end{aligned} \quad (4.19)$$

After $K, \lambda(s)$ and d_0 are determined, we have to determine D_1^T, D_2^T and b from the system of equations (4.20).

$$\begin{aligned} D_1^T b &= -4 \\ D_2^T b &= 400 \end{aligned} \quad (4.20)$$

By arbitrarily choosing $b=1$, we can calculate D_1^T and D_2^T from equation (4.20) accordingly their values are given as,

$$\begin{aligned} b &= 1 \\ D_1^T &= -4 \\ D_2^T &= 400 \end{aligned} \quad (4.21)$$

while

$$\begin{aligned} \lambda_0 &= -10 \\ K &= 4 \\ d_0 &= -44 \end{aligned} \quad (4.22)$$

After determining the controller parameters, we can write the control law given by (4.18), in the form,

$$u_i = 4q_c - 44q_i - \frac{4}{s+10}u_i + \frac{400}{s+10}q_i \quad (4.23)$$

or as it is given in Figure 4.5.

$$u_i = \frac{(4s+40)q_c - (44s+40)q_i}{s+14} \quad (4.24)$$

The next step is to define torques which will be applied to the each joint of the robot manipulator. We know that the robot dynamic model is given by equation (4.5) and can be written in linear regression form as,

$$T = Y(q, \dot{q}, \ddot{q})P \quad (4.25)$$

In order to achieve perfect linearization of the robot nonlinear dynamic model (4.25), the torques applied to the each joint of the robot manipulator should be computed according to equation (4.26).

$$T_L = M(q)u + C(q, \dot{q}) \quad (4.26)$$

where u is a virtual control signal given by equation (4.23), while T_L is an output from the linearizer. Equation (4.26) can be written in linear regression form as,

$$T_L = Y(q, \dot{q}, u)P \quad (4.27)$$

If we use equation (4.27) to calculate torques applied to the each joint of the robot manipulator, we need to determine the values of the position q and velocity $\dot{q}$ of manipulator joints, the virtual control law u and the true parameter values P . Unfortunately, the velocity $\dot{q}$ of manipulator joints and the true parameter values P are unknown. The position q of manipulator joints is measured, while the virtual control law u can be calculated from equation (4.23). Therefore, the unknown true parameter values P will be substituted with their estimates $\hat{P}$. Moreover, the unknown velocity $\dot{q}$ of manipulator joints will be replaced with the velocity of the reference model $\dot{q}_m$. Accordingly, the linearization law of algorithm #1 [1] in linear regression form is given as,

$$T_L = Y(q, \dot{q}_m, u)\hat{P} \quad (4.28)$$

or in explicit form as,

$$T_L = \begin{bmatrix} u_1 & 2u_1 + u_2 + 2u_1 \cos q_2 - 2\dot{q}_{m1}\dot{q}_{m2} \sin q_2 - \dot{q}_{m2}^2 \sin q_2 + u_2 \cos q_2 \\ 0 & u_1 + u_1 \cos q_2 + u_2 + \dot{q}_{m1}^2 \sin q_2 \end{bmatrix} \begin{bmatrix} \hat{P}_1 \\ \hat{P}_2 \end{bmatrix} \quad (4.29)$$

The last part of the simulation block diagram illustrated in Figure 4.5 that has to be determined is the adaptation law. The adaptation law of algorithm #1 [1] for parameter P is given by equation (4.30).

$$\dot{\hat{P}} = \frac{Y^T(q, \dot{q}_m, u)\ddot{e}_v}{1 + \xi_1^T \xi_1} = \frac{1}{1 + \zeta^T \zeta} \begin{bmatrix} Y_{11} & Y_{21} \\ Y_{12} & Y_{22} \end{bmatrix} \begin{bmatrix} u_1 - \ddot{q}_{m1} \\ u_2 - \ddot{q}_{m2} \end{bmatrix} \quad (4.30)$$

where $(1 + \zeta^T \zeta)$ is a normalizing factor, $\ddot{e}_v$ is virtual error, $\hat{P}$ is an estimate of parameter P , and

$$\begin{aligned} Y_{11} &= u_1 \\ Y_{12} &= 2u_1 + u_2 + 2u_1 \cos q_2 - 2\dot{q}_{m1}\dot{q}_{m2} \sin q_2 - \dot{q}_{m2}^2 \sin q_2 + u_2 \cos q_2 \\ Y_{21} &= 0 \\ Y_{22} &= u_1 + u_1 \cos q_2 + u_2 + \dot{q}_{m1}^2 \sin q_2 \end{aligned} \quad (4.31)$$

The ζ is defined as

$$\zeta = W_m(s)IY^T(q, \dot{q}_m, u) \quad (4.32)$$

Any value of the virtual error e_{vi} different from zero will indicate that perfect linearization did not occur and that there is some parameter error. The adaptation law will act to reduce the parameter error. By reducing the parameter error the virtual error will decrease as well. If the normalizing factor is added to the adaptation law, it will result in a bigger region of convergence, and a slightly shorter settling time.

After analysing all the essential parts of this adaptive control algorithm, we have noticed that there are no design parameters, which have to be set. The algorithm performance is predefined with the selection of a reference model. That is another advantage of this algorithm over others in the adaptive control literature. Moreover, this algorithm only needs measurement of the joint positions. Both of these advantages have resulted in a simpler implementation procedure of this algorithm over others proposed in the literature.

Figure 4.5: Simulation Block Structure of Algorithm #1.

4.3.2 Simulation of Algorithm #1

We now proceed with computer simulations according to the simulation scenario given in section 4.2. The simulation scenario for this algorithm is summarised in Table 4.1. Each simulation case is given the initial settings and a figure number where the responses are illustrated.

Table 4.1: Initial Settings for Different Simulation Cases.

Simulation case	$P_{1,init} = 1$ $P_1(0)$	$P_{2,init} = 2$ $P_2(0)$	$e_i(0)$	White noise STD	Quantization noise	Reference signal	Figures
1a	1.5	3	0	0	NO	square	4.6-4.9
1b	5	10	0	0	NO	square	4.10-4.11
2	1.5	3	0	0	NO	unit step	4.12-4.13
3	1.5	3	0.2	0	NO	square	4.14-4.15
4a	1.5	3	0	0.01	NO	square	4.16-4.18
4b	1.5	3	0	0.003	NO	square	4.19-4.21
5	1.5	3	0	0	YES	square	4.22-4.24

Figures 4.6-4.9 illustrate results obtained for simulation case 1a when a reference signal is a sequence of unit steps, while the initial parameter values are 1.5 and 3 for P_1 and P_2 respectively. We will refer to this simulation setting throughout this section as a benchmark case. The algorithm has shown good performance for this scenario by achieving a settling time of the tracking error of 8.8sec for the first and second joints. The settling time for the parameter estimation errors are 13.09sec and 9.58sec for $\tilde{P}_1$ and $\tilde{P}_2$, respectively. Maximum values for torques applied on each manipulator joint are 90.6Nm and 43.8Nm for the first and second joint. After reducing the position tracking errors and parameter estimation errors to zero, the maximum value of applied torques decreased to 82.5Nm and 41Nm. The maximum value of the position error for the second joint is almost 2 times greater than for the first joint. The maximum value of the position error for the second joint is 0.106rad.

As part of simulation case 1b, we have determined how many times we can increase the initial value of the parameters P_1 and P_2 can be increased in comparison to

their true values $P_1 = 1kgm^2$ and $P_2 = 2kgm^2$, and still have that algorithm converge. After a number of simulations we have determined that the initial parameter values can be maximally increased up to five times their true values. The actual and the reference model trajectories as well as convergence of the parameter error to zero are shown in Figures 4.10 and 4.11. The settling time has increased approximately ten times while the maximal tracking error has increased about six times in comparison to the benchmark case 1a. The maximal tracking error for the second joint is still 2 times larger than for the first. The maximum values of torques have increased significantly during the transient period. They are higher by about 3 times than in the benchmark case 1a. After the transient period is finished, maximum torques in this case are identical to the maximum torque values obtained in the benchmark case. In short, an increase in the initial parameter estimation errors $\tilde{P}_1(0)$ and $\tilde{P}_2(0)$ has resulted in longer settling times, higher maximum tracking errors and the necessity for higher maximum torques.

Figures 4.12 and 4.13 present results for simulation case 2. The simulation settings are the same as in the benchmark case 1a except that here the reference signal is a unit step. The unit step is not as “rich” a signal as the sequence of unit steps. The adaptation law was unable to identify the true parameter values. Therefore, instead of obtaining true parameter values we have obtained $P_1(\infty) = 1.36kgm^2$ and $P_2(\infty) = 2.16kgm^2$. The parameter estimation errors are shown in Figure 4.13. Nonetheless, the algorithm has achieved a perfect following of the reference model trajectory in less than 4sec, with an overdamped response of the joint 1 trajectory and a slightly underdamped response of the joint 2 trajectory. These responses are illustrated in Figure 4.12. Moreover, the maximum position error is identical to the maximum position error in benchmark case 1a, while the maximum torques are about 10% smaller.

In simulation case 3 we were investigating the influence of the initial position error on the algorithm convergence. For that reason we have set the initial position error $e_1(0) = e_2(0) = 0.2$, while the initial parameter values are $1.5kgm^2$ and $3kgm^2$ for P_1 and P_2 as in benchmark case 1a. The obtained results are illustrated in Figures 4.14-4.15. Despite the initial position error, the algorithm has succeeded in achieving perfect tracking of the desired trajectory, and reducing the parameter estimation errors to zero.

The settling time for the parameter estimation errors and the position tracking errors has increased about 7 times over the benchmark case. Exact numerical values are given in Table 4.2. The maximum values of control torques are $397.4Nm$ and $200.9Nm$ during the transient period. The torques has increased 4 times over the case with zero initial position error $e_i(0) = 0$. Due to high torques, we have tried to saturate them, but that action has resulted in an unstable system. Such a high value of applied torques has resulted in a significant overshoot of the robot position trajectories over the reference model trajectory. Accordingly, the maximum position error for the second joint is $1.42rad$, while for the first joint is $0.51rad$. After the transient oscillations have decayed, the maximum torques values applied on the first and second joint are $82.5Nm$ and $41Nm$, respectively. Those are the same values as in the case with zero initial position error $e_i(0) = 0$. It is important to note that the algorithm has diverged for any initial position error greater than $0.2rad$.

Simulation case 4 is dedicated to testing of the algorithm performance when position signals q_1 and q_2 are corrupted with white Gaussian noise. The testing is performed with white noise with two values of standard deviation $\sigma = 0.01rad$ (case 4a) and $\sigma = 0.003rad$ (case 4b). Accordingly, the results are illustrated in Figures 4.16-4.18 ($\sigma = 0.01rad$) and Figures 4.19-4.21 ($\sigma = 0.003rad$). Noisy position signals q_1 and q_2 have resulted in an increase of the maximum values of torques $T_{i\max}$ and the maximum position errors $e_{i\max}$. Furthermore, the algorithm was unable to achieve a perfect following of the desired trajectory with any noise level. From Figure 4.20, where the position tracking errors are illustrated for $\sigma = 0.01rad$, it is clear that perfect following is not achieved even in the case of the lower noise level. The position errors are bounded for both noise levels. In the case of lower noise level, the position errors are bounded within a range of $\pm 0.02rad$ for the first joint and $\pm 0.03rad$ for the second joint for less than $9sec$. In the case of higher noise level, that range is about 10 times larger. A similar observation holds for the parameter estimation errors. As a result of the position and parameter estimation errors not being within the range of $\pm 0.01rad$ and $\pm 0.01kgm^2$ respectively, we were unable to calculate settling times according to the 1% criteria like we did in the previous simulation cases. Therefore, in Table 4.2 we gave upper and lower boundary values and the time necessary to get the position and parameter estimation

errors within those boundaries. Simulations with two different levels of white noise have shown that the algorithm is capable of staying within the region of stability, while lacking the ability to decrease the position and parameter estimation errors to zero. Besides the two noise levels, we have performed the simulation with white noise with standard deviation of $\sigma = 0.03rad$. In that case the system has become unstable.

Simulation case 5 has involved testing the influence of noise associated with a 10 bit A/D converter on algorithm performance. The level of the quantization noise is determined according to equation (4.2). Assuming that the noise has uniform distribution, we have obtained the results given in Figures 4.22-4.24. The simulation results show bounded parameter and position errors with significantly narrower range than in both simulation cases with white noise. The position and parameter estimation errors are bounded within a range of $\pm 0.01rad$ and $\pm 0.01kgm^2$ respectively, we were able to calculate settling times $t_{s\hat{p}_i}$ and t_{se_i} according to the 1% criteria. The maximum errors are smaller than in simulation cases with white noise. In short, algorithm #1 has shown better performance in simulation case with quantization noise than in simulation cases with measurement noise.

Figures 4.18, 4.21 represent parameter estimation errors in simulation cases with measurement noise ($\sigma = 0.01rad$, $\sigma = 0.003rad$) while Figure 4.24 represents parameter estimation errors with quantization noise. According to these figures, we can state that the parameter estimation errors are bounded. Furthermore, we can notice growth in the parameter estimation error when the first order derivative of the desired trajectory has positive sign. If we change the square wave reference signal into a step, algorithm #1 will exhibit the diverging parameter estimation error in simulation cases with noise. That observation does not hold in simulation cases without noise and with a step reference signal as it is visible from Figure 4.13 that represents the parameter estimation error when a step is the reference signal. The reason for diverging parameter estimation error in simulation cases with noise and a step as a reference signal is the adaptation law of algorithm #1. Using equation (4.30) that represents the adaptation law in general form for algorithm #1, one can obtain adaptation law for the parameter P_I .

$$\dot{\hat{P}}_1 = \frac{u_1(u_1 - \ddot{q}_m)}{1 + \xi_1 \xi_1^T} = \frac{u_1 u_1 - u_1 \ddot{q}_m}{1 + \xi_1 \xi_1^T} \quad (4.33)$$

According to equation (4.33) the following observations are true when a step is the reference signal:

- when $q_m = 1$ its first and second order derivatives are $\dot{q}_m = \ddot{q}_m = 0$
- $\xi_1 \xi_1^T$ is positive number or zero $\forall t$
- $u_1 u_1$ is positive number or zero $\forall t$

Accordingly $\dot{\hat{P}}_1$ is positive or zero when the reference signal is unit step function. Hence, $\hat{P}_1$ is positive number that increases with time. Similar observation holds for the $\hat{P}_2$. For that reason, the parameter estimate diverges when the reference signal is constant as it is visible in Figures 4.18, 4.21 and 4.24. Despite divergence of the parameter estimates, algorithm #1 did not exhibit diverging position tracking errors. Furthermore, in simulation cases with noise the position tracking errors are smaller with the step reference signal than with the square wave. A similar observation is reported by Schwartz [8] for the Craig [3] and Slotine [7] algorithm.

It is important to note that simulations with white (case 4) and quantization (case 5) noise are performed with an integration step of 0.0001sec. Noise generation models in SIMULINK[®] are dependant on the integration step size. Therefore, simulations of the two other algorithms with noise will be performed with the same integration step size used in this case.

In Table 4.2 we have given numerical values obtained for each simulation case, such as settling times for parameter $t_{s\hat{P}_i}$ and position t_{se_i} errors, maximum torque values T_{imax} , maximum torque values T_{iss} after initial oscillations are finished and maximum position errors e_{imax} . If parameter and position errors did not converge to zero, instead of giving the settling times $t_{s\hat{P}_i}$ and t_{se_i} , we have given the upper and the lower bounds and the time necessary for parameters and position errors to be bounded within those bounds. That is applicable for simulation cases 4a and 4b.

According to the numerical results presented in Table 4.2, we can state that the algorithm has shown good performance in the simulation case with quantization and low level white noise. In those two simulation cases, the algorithm was able to reduce the

parameter and position errors to acceptable levels. Required torques were within the range of the benchmark case (case 1). On the other hand it is surprising that the algorithm is very susceptible to the value of initial position errors (case 3). In that case, the torques necessary to reduce the parameter and position errors are very high, even higher than in the case with a high level white noise (case 4a). It is clear from numerical results given in Table 4.2 that an increase of the white noise level from $\sigma = 0.003rad$ to $\sigma = 0.01rad$ has resulted in significant performance degradation along with an increase in required maximum torques. Hence, we can state that the algorithm is able to handle white noise with standard deviation just slightly higher than $\sigma = 0.003rad$.

Table 4.2: Numeric Values of Settling Times, Maximum Torques and Position Errors Obtained Through Simulations.

Sim. case	t_{ϕ_1}	t_{ϕ_2}	t_{θ_1}	t_{θ_2}	T_{1max}	T_{2max}	T_{1ss}	T_{2ss}	e_{1max}	e_{2max}
1a	13.09	9.58	8.74	8.90	90.6	43.8	82.5	41	0.07	0.11
1b	84.70	81.60	76.83	80.74	287.5	124.6	82.5	41	0.39	0.62
2	2.57 $\tilde{P}_1(\infty)=0.36$	2.97 $\tilde{P}_1(\infty)=0.16$	3.76	3.96	78	36	0	0	0.07	0.11
3	76.46	73.46	68.80	72.72	397.4	201	82.5	41	0.51	1.42
4a	~ 100 $\tilde{P}_1 \in [0.5, 0.8]$	~ 100 $\tilde{P}_2 \in [0.15, 0.25]$	~ 20 $e_1 \in [\pm 0.15]$	~ 42 $e_2 \in [\pm 0.2]$	124.2	59.45	84	42	0.16	0.27
4b	20.08 $\tilde{P}_1 \in [\pm 0.05]$	4.28 $\tilde{P}_2 \in [\pm 0.03]$	8.75 $e_1 \in [\pm 0.02]$	8.54 $e_2 \in [\pm 0.03]$	93.2	44.9	84	42	0.07	0.11
5	87.14	28.86	12.56	28.6	91.6	44.12	84	42	0.07	0.09

Figure 4.6: Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$.

Figure 4.7: Position Tracking Errors: e_1 (solid line), e_2 (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$.

Figure 4.8: Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$.

Figure 4.9: Torques [Nm] Applied to the Robot Joints, T_1 (solid line), T_2 (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$.

Figure 4.10: Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=5$, $P_2(0)=10$.

Figure 4.11: Parameter estimation errors [kgm²]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=5$, $P_2(0)=10$.

Figure 4.12: Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) with Unit Step as a Reference Signal and $P_1(0)=1.5$, $P_2(0)=3$.

Figure 4.13: Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) with Unit Step as a Reference Signal and $P_1(0)=1.5$, $P_2(0)=3$.

Figure 4.14: Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$ and $e_i(0)=0.2$.

Figure 4.15: Parameter estimation errors $[\text{kgm}^2]$: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ and $e_i(0)=0.2$.

Figure 4.16: Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$ and White Noise $\sigma = 0.01$.

Figure 4.17: Position Tracking Errors: e_1 (solid line), e_2 (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ and White Noise $\sigma = 0.01$.

Figure 4.18: Parameter estimation errors [kgm²]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) and $P_1(0)=1.5$, $P_2(0)=3$ and White Noise $\sigma = 0.01$.

Figure 4.19: Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$ and White Noise $\sigma = 0.003$.

Figure 4.20: Position Tracking Errors: e_1 (solid line), e_2 (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ and White Noise $\sigma = 0.003$.

Figure 4.21: Parameter estimation errors $[\text{kgm}^2]$: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ and White Noise $\sigma = 0.003$.

Figure 4.22: Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$ and 10 Bits Quantization Noise.

Figure 4.23: Position Tracking Errors: e_1 (solid line), e_2 (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ and 10 Bits Quantization Noise.

Figure 4.24: Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ and 10 Bits Quantization Noise.

4.4 Nonlinear MRAC with Linear Observer - Algorithm #2

The second adaptive control algorithm that will be presented is proposed by Schwartz [2]. It is an example of nonlinear output feedback MRAC that is applicable to feedback linearizable systems that can be decoupled. Like the previously presented algorithm #1 [1], this adaptive control algorithm does not need measured joint velocities and accelerations for implementation of the linearization and adaptation laws. That is an important advantage of this algorithm, because the measured joint velocity and acceleration signals are often very noisy. The simple linear observer is used for the estimation of the joint velocity signals that are used in the linearization and adaptation laws. That is one of the differences between this algorithm and algorithm #1. In algorithm #1 the joint velocity $\dot{q}_i$ is substituted with an always available reference model velocity signal $\dot{q}_m$. On the other hand, both algorithms use the virtual control signal u as a replacement for an unavailable joint acceleration signal in their adaptation and linearization laws. Algorithm #2 uses the filter in the adaptation law in order to compensate phase shift between the parameter error and the virtual error. The use of the linear observer has resulted in a simpler implementation of this algorithm over the others that are using nonlinear observers or estimators.

4.4.1 Description of Algorithm #2

The reference model, used in this algorithm is identical to the one used in algorithm #1 and it is given by equation (4.1). We have chosen the same reference model due to the fact that we are doing comparisons of the algorithms. Furthermore, their performance are affected by a chosen reference model. Passing the reference signal q_c through the reference model we were able to obtain the desired robot trajectory q_m and its second order derivative $\ddot{q}_m$. The second order derivative of the robot desired trajectory $\ddot{q}_m$ is used for the calculation of the virtual error, as will be shown latter. The virtual error

is defined as the difference between the virtual control signal u and the second order derivative of robot desired trajectory $\ddot{q}_m$.

The algorithm is simulated using a two degree of freedom robot model which is explained in detail in chapter 3. The robot model in *form I* is given by equation (4.34),

$$T = M(q)\ddot{q} + C(q, \dot{q}) = Y(q, \dot{q}, \ddot{q})P \quad (4.34)$$

Similar to algorithm #1 equation (4.26), the torques applied to each joint of the robot manipulator are computed according to equation (4.35)

$$T_L = M(q)u + C(q, \dot{q}) \quad (4.35)$$

where u is a virtual control signal, while T_L is the output from the linearizer. There are two problems with equation (4.35). We do not want to use $\dot{q}$ because it is corrupted with noise and the true values of robot parameters are unknown. In order to solve the first problem we use a linear observer to estimate joint velocity $\dot{q}$. Accordingly, the unknown $\dot{q}$ will be replaced with its estimate $\hat{\dot{q}}$ in the linearization law. The linear observer for each manipulator joint is given by,

$$\begin{bmatrix} \hat{\dot{q}}_i \\ \hat{\ddot{q}}_i \end{bmatrix} = A \begin{bmatrix} \hat{q}_i \\ \hat{\dot{q}}_i \end{bmatrix} + Bu_i + L(q_i - \hat{q}_i) \quad (4.36)$$

where the matrix $A \in R^{2 \times 2}$, vectors $B \in R^{2 \times 1}$ and $L \in R^{2 \times 1}$ and u is the virtual control signal. The second order system matrix A and vector B are defined as,

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, B = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad (4.37)$$

The elements of the vector L are chosen according to the recommendation that the observer poles have to be at least 5 times faster than reference model poles. The reference model has two poles at -2. Because the dynamics of the observer are much faster than the reference model dynamics, the observer error will be small. Accordingly, the observer gains are chosen as,

$$L = \begin{bmatrix} L_1 \\ L_2 \end{bmatrix} = \begin{bmatrix} 20 \\ 100 \end{bmatrix} \quad (4.38)$$

The virtual control law u , can be obtained from Figure 4.4 in the following form,

$$u_i(s) = Kq_{ci} + d_0 q_i(s) + \frac{D_1^T b}{\lambda(s)} u_i(s) + \frac{D_2^T b}{\lambda(s)} q_i(s) \quad (4.39)$$

The virtual control law given here is identical to the control law used by algorithm #1. Derivation of the control law (4.39) and algebraic computation of its parameters $K, D_1^T, D_2^T, \lambda(s), d_0$ and b is already given by equations (4.18-4.24), therefore it will not be repeated again.

After defining the robot dynamic model (4.34), the reference model (4.1), control law (4.39) and observer (4.36) we can define the linearization law in the following form [2, 9, 23],

$$T_L = Y(q, \dot{q}, u) \hat{P} \quad (4.40)$$

where,

$$Y(q, \dot{q}, u) = \begin{bmatrix} u_1 & 2u_1 + u_2 + 2u_1 \cos q_2 - 2\dot{q}_1 \dot{q}_2 \sin q_2 - \dot{q}_2^2 \sin q_2 + u_2 \cos q_2 \\ 0 & u_1 + u_1 \cos q_2 + u_2 + \dot{q}_1^2 \sin q_2 \end{bmatrix} \quad (4.41)$$

where $\hat{P}$ is a vector of parameter estimates, because the true robot parameters are unknown. In the linearization law, we use the estimate of the joint velocity $\dot{q}$ instead of the measured velocity $\dot{q}$ that is noise corrupted. The joint acceleration $\ddot{q}$ is substituted with a virtual control signal u . That substitution is valid only if the parameter estimation errors and observer errors are equal to zero. Mostly, that is not true. Therefore, we have to define the adaptation law which will perform parameter adaptation as a function of the difference between the virtual control signal u_i and the reference model joint acceleration signal $\ddot{q}_{mi}$. That difference is called the virtual error signal.

In order to derive the adaptation law we will assume that the robot manipulator joints are decoupled as a result of perfect linearization. Therefore, this adaptive control algorithm is restricted to systems that can be decoupled. Accordingly any nonlinear system given by the following form,

$$\begin{aligned} \dot{x} &= f(x) + g(x)T \\ y &= [1, 0, \dots, 0]x \end{aligned} \quad (4.42)$$

can be written after a coordinate transformation as a set of linear differential equations,

$$\begin{aligned}
\dot{z}_1 &= z_2 \\
\dot{z}_2 &= z_3 \\
&\vdots \\
\dot{z}_{n-1} &= z_n \\
\dot{z}_n &= b(z) + a(z)T = u
\end{aligned} \tag{4.43}$$

Hence, the linearization law can be written as

$$T = \frac{1}{a(z)}(-b(z) + u) \tag{4.44}$$

where u is the virtual control signal defined by equation (4.39). The linearization law can be written in a linear regression form as,

$$T = Y(z, u)P \tag{4.45}$$

Due to the fact that true parameter values are unknown, we will write equation (4.45) as,

$$T = Y(z, u)\hat{P} = Y(z, u)P - Y(z, u)\tilde{P} \tag{4.46}$$

where $\hat{P} = P - \tilde{P}$ is the parameter error. After substitution of (4.46) into (4.43) we will get,

$$\dot{z}_n = b(z) + a(z)Y(z, u)P - a(z)Y(z, u)\tilde{P} \tag{4.47}$$

On the other hand, if we substitute the linearization law with known parameter values given by equation (4.45) into (4.43) we will get,

$$\dot{z}_n = b(z) + a(z)Y(z, u)P \tag{4.48}$$

Using the fact that in the case of perfect linearization the virtual control signal u is equal to the n^{th} order derivative of the system output,

$$\dot{z}_n = y^{(n)} = u \tag{4.49}$$

equation (4.47) under action of (4.48) and (4.49) can be written as,

$$\dot{z}_n = u - a(z)Y(z, u)\tilde{P} \tag{4.50}$$

Since the n^{th} order derivative of the output of a nonlinear plant y is equal to $\dot{z}_n$, we can express the output of a nonlinear plant y using equation (4.50) as,

$$y(s) = \frac{u}{s^n} - \frac{a(z)Y(z, u)\tilde{P}}{s^n} \tag{4.51}$$

If the robot manipulator is the nonlinear plant given by equation (4.42) then the output of the nonlinear plant y is the joint position q given by equation (4.51). After substituting equation (4.51) into equation (4.39) the equation that defines the virtual control law u is obtained as,

$$u(s) = Kq_c + d_0 \left(\frac{u}{s^n} - \frac{a(z)Y(z,u)\tilde{P}}{s^n} \right) + \frac{D_1^T b}{\lambda(s)} u(s) + \frac{D_2^T b}{\lambda(s)} \left(\frac{u}{s^n} - \frac{a(z)Y(z,u)\tilde{P}}{s^n} \right) \quad (4.52)$$

After rearranging equation (4.52) and solving for $u(s)$, one can obtain the Laplace transform of the virtual control signal u as,

$$u(s) = \frac{s^n q_c K \lambda(s) - (D_2^T b + d_0 \lambda(s))(a(z)Y(z,u)\tilde{P})}{s^n (\lambda(s) - D_1^T b) - D_2^T b - d_0 \lambda(s)} \quad (4.53)$$

Using the block diagrams given in Figures 4.4 and 4.3 one can show that the reference model given by equation (4.1) can also be written in the following form,

$$W_m = \frac{\lambda(s)K}{s^n (\lambda(s) - D_1^T b) - D_2^T b - d_0 \lambda(s)} \quad (4.54)$$

The values for the parameters $K, D_1^T, D_2^T, \lambda(s), d_0$ and b are already determined during derivation of the control law u . They are as follows,

$$b = 1, D_1^T = -4, D_2^T = 400, \lambda_0 = -10, K = 4, d_0 = -44 \quad (4.55)$$

Using the numeric values for the parameters $K, D_1^T, D_2^T, \lambda(s), d_0$ and b , one can show that equation (4.54) is really our reference model.

$$W_m = \frac{4(s+10)}{s^2(s+10+4) - 400 + 44(s+10)} = \frac{4(s+10)}{(s+10)(s^2+4s+4)} = \frac{4}{(s^2+4s+4)} \quad (4.56)$$

Accordingly, equation (4.53) can be written in the new form using equation (4.54),

$$u(s) = s^n W_m(s) q_c - \frac{D_2^T b + d_0 \lambda(s)}{s^n (\lambda(s) - D_1^T b) - D_2^T b - d_0 \lambda(s)} (a(z)Y(z,u)\tilde{P}) \quad (4.57)$$

where,

$$q_m^{(n)} = s^n W_m(s) q_c \quad (4.58)$$

while $q_m^{(n)}$ is n^{th} order derivative of the reference model position signal. After recalling the definition of the virtual error $e_v = u - q_m^{(n)}$ and using equation (4.57), one can also write the virtual error e_v in the form,

$$e_v = -\frac{D_2^T b + d_0 \lambda(s)}{s^n (\lambda(s) - D_1^T b) - D_2^T b - d_0 \lambda(s)} (a(z)Y(z, u)\tilde{P}) \quad (4.59)$$

Equation (4.59) gives the relationship between the virtual error e_v and the parameter error $\tilde{P}$. The fraction term in equation (4.59) represents the filter that will compensate the phase shift between the parameter error $\tilde{P}$ and the virtual error e_v . The filter is given by equation (4.60).

$$W_{\tilde{P}} = -\frac{D_2^T b + d_0 \lambda(s)}{s^n (\lambda(s) - D_1^T b) - D_2^T b - d_0 \lambda(s)} \quad (4.60)$$

After using numerical values for the parameters $K, D_1^T, D_2^T, \lambda(s), d_0$ and b equation (4.60) for $W_{\tilde{P}}$ can be written as,

$$W_{\tilde{P}} = \frac{44s + 40}{s^3 + 14s^2 + 44s + 40} \quad (4.61)$$

Therefore, one can propose the adaptation law in the form,

$$\dot{\tilde{P}} = \gamma \Sigma^T(t) e_v = \gamma \Sigma^T(t) \begin{bmatrix} u_1 - \ddot{q}_m \\ u_2 - \ddot{q}_m \end{bmatrix} \quad (4.62)$$

where γ is a gradient step size, while $\Sigma(t)$ is given by the following equation,

$$\Sigma(t) = \begin{bmatrix} W_{\tilde{P}} & 0 \\ 0 & W_{\tilde{P}} \end{bmatrix} Y(q, \dot{q}, u) \quad (4.63)$$

After defining the robot dynamic and reference models, the observer, the adaptation law, the linearization law and the control law for algorithm #2, we can proceed with the simulation. The simulations are performed according to the simulation block diagram illustrated in Figure 4.25. According to the given information one can give the following conclusion regarding the design procedure for algorithm #2. The linearization law is defined by the dynamic model of the robot. Because we wanted to avoid use of the noisy joint velocity, we have used its estimate. On the other hand the structure and the numerical values of variables inside the control law, the adaptation law and the observer are mostly predefined by a chosen reference model and its bandwidth. For instance, if we choose a faster reference model, we will need to make our observer faster as well. That change will result in a wider observer bandwidth making it more susceptible to noise. The only design parameter that is not predefined by the selection of the reference model

and the robot dynamic model, is the adaptation law gain γ . The adaptation law gain γ is actually the gradient step size and it has to be chosen by a designer. Unfortunately, a clear procedure of how to choose it does not exist. Hence, one can conclude that the implementation of this algorithm is free from obscurity, which is one of the strengths of this algorithm.

Figure 4.25: Simulation Block Diagram for Algorithm #2.

4.4.2 Simulation of Algorithm #2

Algorithm #2 [2] has the adaptation gain parameter γ which has to be chosen by a designer, before we can proceed with simulations. We tune algorithm's design parameters in order to optimize their performance for each simulation case. In the case of algorithm #2 we have to determine the value of the adaptation law gain γ . The value of the gain γ is determined through an iterative design procedure. Consequently, we have performed a number of simulations with various values of γ . The simulations are

performed with initial values of parameter estimation errors of $\tilde{P}_1(0) = 0.5 \text{kgm}^2$ and $\tilde{P}_2(0) = 1 \text{kgm}^2$. The reference signal is sequence of unit steps.

In order to be able to distinguish which value of γ will result in better algorithm performance, we have recorded parameter settling times and maximum torque values. These results are summarized in Table 4.3.

Table 4.3: Optimal Value of the Adaptation Gain γ

γ_i	$P_1(0)$	$P_2(0)$	$t_{\tilde{P}_1}$	$t_{\tilde{P}_2}$	$T_{1\max}$	$T_{2\max}$
0.1	1.5	3	80.48	56.97	100.2	49.0
0.2	1.5	3	44.36	36.79	90.7	43.6
0.3	1.5	3	28.50	24.90	82.5	40.8
0.4	1.5	3	17.27	16.77	82.3	40.8
0.5	1.5	3	8.66	9.18	82.3	40.8
0.6	1.5	3	25.19	29.31	82.4	40.9
0.7	1.5	3	17.35	21.09	82.2	40.7
0.8	1.5	3	21.30	22.20	82.1	40.7
0.9	1.5	3	~ 100 $\tilde{P}_1 \in [2.5, 3.6]$	~ 100 $\tilde{P}_1 \in [\pm 2]$	81.6	38.6

According to the results given in Table 4.3, it is clear that the best algorithm performance is achieved with an adaptation gain of $\gamma = 0.5$. Using the adaptation gain of $\gamma = 0.5$, we have achieved the shortest settling times for the parameter estimation errors. The value of the maximum torques has stayed the same as in other simulation cases. Therefore, all the following simulations of algorithm #2 are performed with an adaptation gain of $\gamma = 0.5$.

The simulations were performed using the parameters listed in Table 4.4. Table 4.4 lists the initial values of the parameter estimates $P_1(0)$ and $P_2(0)$, the initial value of tracking error $e_i(0)$, the type of the reference signal and the type of the noise and its standard deviation. In the last column of Table 4.4 are given figure numbers, where the results of simulations for each particular simulation case are illustrated. The simulations according to the settings for simulation case 3c are performed using saturated torques.

Table 4.4: Initial Settings for Different Simulation Cases.

Sim. case	$P_{1max} = 1$	$P_{2max} = 2$	$e_j(0)$	γ_i	Sat(Ti)	White noise σ	Quantization noise	Reference signal	Figures
	$P_1(0)$	$P_2(0)$							
1a	1.5	3	0	0.5	NO	0	NO	square	4.26-4.29
1b	5	10	0	0.5	NO	0	NO	square	4.30-4.31
2	1.5	3	0	0.5	NO	0	NO	unit step	4.32-4.33
3a	1.5	3	0.2	0.5	NO	0	NO	square	4.34-4.35
3b	1.5	3	0.6	0.5	NO	0	NO	square	4.36-4.37
3c	1.5	3	0.6	0.5	100	0	NO	square	N/A
4a	1.5	3	0	0.5	NO	0.01	NO	square	4.38-4.40
4b	1.5	3	0	0.5	NO	0.003	NO	square	4.41-4.43
5	1.5	3	0	0.5	NO	0	YES	square	4.44-4.46

Table 4.5 summarizes the numerical values of the parameter and tracking error settling times, the maximum torques and the position errors for the each simulation case. The maximum torques are given two values. The first one is the value of maximum torque during initial oscillations. The second value is the maximum torque when initial oscillations are finished and perfect tracking or almost perfect tracking of the desired trajectory is achieved.

Table 4.5: Numeric Values of Settling Times, Maximum Torques and Position Errors Obtained Through Simulations.

Sim. case	t_{σ_1}	t_{σ_2}	t_{σ_1}	t_{σ_2}	T_{1max}	T_{2max}	T_{1ss}	T_{2ss}	e_{1max}	e_{2max}
1a	8.66	9.18	6.90	8.95	82.5	41	82.5	41	0.07	0.11
1b	48.63	50.05	48.83	49.18	336.6	162.2	82.5	41	0.32	0.60
2	2.31 $\bar{P}_1(\infty) = 0.35$	2.54 $\bar{P}_1(\infty) = 0.13$	2.11	3.75	78	36	0	0	0.07	0.06
3a	25.01	25.24	21.08	21.21	131.8	64.4	82.5	41	0.23	0.62
3b	20.99	21.27	17.04	17.27	401.6	189.8	82.5	41	0.60	0.72
3c	48.50	48.96	45.00	46.91	100	100	82.5	41	1.11	1.25
4a	8.18 $\bar{P}_1 \in [\pm 0.5]$	5.60 $\bar{P}_2 \in [\pm 0.5]$	5.71 $e_1 \in [\pm 0.5]$	6.53 $e_2 \in [\pm 0.5]$	88.3	43	82.5	41	0.08	0.12
4b	8.69	20.58	21.45	22.16	82.7	41	82.5	41	0.07	0.11
5	8.63	9.9	8.74	8.99	83.1	41.2	84	42	0.07	0.12

As part of simulation case 1a, in Figure 4.26 we have shown desired and actual trajectories for each joint of the manipulator. It is noticeable that almost perfect tracking is achieved after the first period of the reference signal. That observation is confirmed with Figure 4.27 that shows quickly a converging tracking error for each joint (less than 9sec) with fairly small maximum values (around 0.1rad). Figure 4.28 shows that the parameter estimation errors are converging to zero very quickly, resulting in almost perfect tracking of the desired trajectory. The parameter error settling times are around 9sec, which is an extremely good result. Figure 4.29 illustrates the torques applied to each joint of the manipulator. The maximum values of torques during the initial oscillations are almost identical to the maximum torque values after the initial oscillations are finished. That behaviour can be explained with Figure 4.27 which illustrates the position tracking errors. The position tracking errors are fairly small when the system is undergoing the initial oscillations. Hence, the robot manipulator requires almost the same amount of torques applied to joints to achieve perfect tracking as well to preserve it. According to the obtained results, the algorithm has shown very good performance in the first and the least challenging simulation case 1a.

Using the initial settings according to Table 4.1 for simulation case 1b, we have obtained the robot manipulator actual trajectories and parameter estimation errors which are shown in Figures 4.30 and 4.31. From these figures it is noticeable that tracking and parameter estimation errors are reduced within the range of $\pm 0.01rad$ and $\pm 0.01kgm^2$ respectively in less than 50sec. That is a significantly better result than the one achieved with algorithm #1 for the same initial settings. The only drawback is the value of the maximum torques. The maximum torques are about 20% higher than in the same simulations with algorithm #1.

Another goal of simulation case 1b is to determine the maximum value of the initial parameter error for which the system will have converging responses. The initial settings of the parameter estimation errors $\tilde{P}_1(0) = 4kgm^2$ and $\tilde{P}_2(0) = 8kgm^2$ for simulation case 1b are the maximum acceptable initial parameter estimation errors for algorithm #1. For any higher initial parameter estimation errors algorithm #1 would become unstable. According to the responses illustrated in Figures 4.30 and 4.31, one can

notice that algorithm #2 is not unstable if the initial parameter estimation errors are $\tilde{P}_1(0) = 4kgm^2$ and $\tilde{P}_2(0) = 8kgm^2$. Therefore, we have conducted a number of simulations in which we have kept increasing the initial parameter estimation errors, expecting that the algorithm will become unstable. Even after having huge initial parameter estimation errors we have not been able to make the algorithm express unstable behaviour. In Table 4.6 are given: the initial parameter values, the maximum torque values along with parameter settling times for the unsaturated torque case and parameter settling times for two values of saturated torques.

Table 4.6: The Settling Times of Parameter estimation errors for Various Initial Parameter Values with Unsaturated and Saturated Torques.

$P_1(0)$	$P_2(0)$	Unsaturated torques			Saturated torques	
		T_{1max}	T_{2max}	t_s	t_s with $T_{1max} = T_{2max} = 200$	t_s with $T_{1max} = T_{2max} = 100$
5	10	336	162	~50	~50	~105
8	16	600	295	~90	~135	~175
20	40	1600	790	~290	~315	~400
30	60	2470	1200	~570	~400	~570
50	100	4260	2040	~1500	~590	~870

According to the results given in Table 4.6, one can conclude that even for the huge initial parameter estimation errors, parameters were converging to their true values. Therefore, we were able to determine parameter estimation errors settling times for each case. The converging responses are achieved with unsaturated torques and also saturated torques. In the simulation case with $P_1(0) = 5kgm^2$ and $P_2(0) = 10kgm^2$, we have achieved almost the same parameter error settling times for the unsaturated torques case and the case when torques are saturated to 200Nm. Hence, the drawback mentioned above that algorithm #2 has generated almost 20% higher maximum torques than algorithm #1 for the simulation case with $P_1(0) = 5kgm^2$ and $P_2(0) = 10kgm^2$ does not hold anymore.

Another interesting behaviour of algorithm #2 can be derived from results given in the last two rows of Table 4.6. When the initial parameter values are 30 or more times higher than its true values, algorithm #2 has achieved even shorter settling times with saturated torques than with unsaturated torques. That behaviour can be explained by the fact that the unsaturated torque case algorithm has generated extremely high torques¹ that have forced parameter estimates to diverge even more from their true values despite already very high initial parameter estimation error. As a result of that, parameter estimation errors have become twice as big than the initial parameter estimation errors in the first few cycles of the reference signal. After that, parameters have started to converge to their true values. Due to the fact that parameters have been pushed so far from their true values, the algorithm has needed much more time to return them to their true values in the unsaturated torque case. On the other hand in the saturated torque case, due to limited torque values, parameter estimates have not been pushed so far from their true values before they have started converging. Hence the algorithm has needed shorter time to return them to their true values.

This algorithm characteristic can be used in the case of a robot with unknown parameters. Then even if our guess of the robot parameters is very inaccurate, we can still estimate true robot parameters without having an unstable algorithm. We only need to saturate the torques to the values that can be generated by the real actuators.

Figures 4.32 and 4.33 illustrate results achieved with initial settings according to Table 4.4 for simulation case 2. In this case we have the same initial setting as in case 1a but the reference signal is changed to a unit step. From Figure 4.32 it is viewable that perfect tracking is achieved for both joints, while the tracking error settling time is less than 4sec. That is a slightly better result than with algorithm #1. In spite of achieving perfect tracking of the desired trajectory, algorithm #2 was unable to estimate the parameter values like algorithm #1. This is due to the fact that the desired trajectory is not “rich” enough to excite all the robot characteristic modes. After convergence the parameter estimation errors were 0.35kgm^2 and 0.13kgm^2 for P_1 and P_2 , respectively. The generated maximum torques are exactly the same as in the simulation with algorithm #1.

¹ It is impossible to generate such high torque values in practical applications.

Figure 4.34 shows desired and actual trajectories with the initial parameter values 1.5 times greater than their true values and with initial tracking error of $e_i(0) = 0.2rad$. Besides the presence of the initial tracking error, the algorithm has shown stable behaviour by achieving perfect tracking in less than 25sec. The maximum tracking error is still within the range achieved in simulation case 1b, where we had high initial parameter estimation errors. The parameter estimates have converged to their true values within 25sec. That is illustrated in Figure 4.35. If compared with results achieved with algorithm #1 for the same initial settings, algorithm #2 has achieved a 50sec shorter tracking error settling time with three times smaller maximum torques.

Very good results, with initial tracking error of $e_i(0) = 0.2rad$ (case 3a), have encouraged us to conduct a simulation with an even higher initial tracking error. Accordingly, we have set initial tracking error to $e_i(0) = 0.6rad$ (case 3b). Figures 4.36 and 4.37 illustrate the algorithm performance for that case. If we compare robot actual trajectories shown in Figures 4.34 ($e_i(0) = 0.2rad$) and 4.36 ($e_i(0) = 0.6rad$), one can notice that perfect tracking is achieved more quickly with a higher initial tracking error. That observation is proven with values of parameter and tracking error settling times, which are approximately shorter by 20% for the simulation case with higher initial tracking error. That is another interesting and unexpected behaviour of algorithm #2.

The shorter settling times have resulted in much higher maximum torque values. As a result of that, in simulation case 3c, we have used the same initial setting as in case 3b, but with torques saturated at 100Nm. Despite initial tracking error of $e_i(0) = 0.6rad$ and saturated torques the algorithm has shown stable behaviour, with twice as long settling times than in the case with unsaturated torques. Due to saturated torques maximum tracking error is almost two times higher than in the unsaturated torque case.

In order to determine algorithm susceptibility to measurement noise, we have performed testing with white noise with two different values of standard deviation $\sigma = 0.01rad$ (case 3a) and $\sigma = 0.003rad$ (case 3b). Figure 4.38 shows the model and actual trajectories of the manipulator, while Figures 4.39 and 4.40 show the position tracking errors and parameter estimation errors for $\sigma = 0.01rad$. According Figures 4.38-4.40 it is clear that the algorithm has exhibited converging responses. Due to a high noise

level, the algorithm was unable to reduce the tracking error to zero. Furthermore, the algorithm was unable to achieve zero parameter estimation errors. Therefore, we were unable to determine settling times according to the 1% criteria. We have determined settling times according to the 5% criteria. That is the time necessary for the position tracking and parameter estimation errors to get and remain within the range of $\pm 0.05rad$ and $\pm 0.05kgm^2$, respectively. The position tracking errors remained within the range of $\pm 0.05rad$ after $7sec$. The parameter estimation errors were bounded within the range of $\pm 0.05kgm^2$ in $8sec$ for the first parameter P_1 and $6sec$ for the second parameter P_2 . Both the position tracking and the parameter estimation errors have continued to decrease towards zero, but they have never reached it, due to the presence of the noise. The maximum tracking error is less than $0.12rad$ for both joints. That is a significant improvement over algorithm #1 where the maximum errors were twice as big for the same noise level.

The algorithm has exhibited even better performance with a lower noise level of $\sigma = 0.003rad$. That is noticeable from Figure 4.42 which illustrates the tracking errors and Figure 4.43 that shows the parameter estimation errors. Figure 4.41 shows the reference model and manipulator position trajectories. Due to the lower noise level the parameter estimation errors and the tracking errors converge towards zero more smoothly than in the case with $\sigma = 0.01rad$. Besides smoother convergence, parameter estimation and tracking errors are reduced within the range of $\pm 0.01kgm^2$ and $\pm 0.01rad$ respectively. Accordingly, we have been able to determine settling times according to the 1% criteria. The settling times for the parameter and tracking errors are around $20sec$. The maximum torque values are smaller than $90Nm$ for both noise levels. In short algorithm #2 has significantly outperformed algorithm #1 in simulation cases with white noise. The maximum tracking errors for both noise levels are the same as in benchmark case 1a.

Simulation case 5 tests the algorithm sensitivity to the quantization noise associated with 10 bit A/D converter. Again, the algorithm has achieved very good performance. Figure 4.44 shows that perfect tracking of the desired trajectory is established within less than $10sec$. Figure 4.45 shows fast convergence of the tracking

errors to zero. Moreover, the parameter estimates are quickly converging to their true values as it is shown in Figure 4.46. The maximum torque values are below $84Nm$ similar to torque values in the benchmark case 1a. The values of the settling times for the parameter estimation and tracking errors, maximum torques and maximum tracking errors are very close to the values achieved with the benchmark case 1a. In short, algorithm #2 has shown that it is not susceptible to the quantization noise associated with a 10 bit A/D converter.

After conducting a number of the simulations for different initial settings according to Table 4.4, the results have shown that algorithm #2 has outperformed algorithm #1 for all simulation scenarios.

Figure 4.26: Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$.

Figure 4.27: Position Tracking Errors: e_1 (solid line), e_2 (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$.

Figure 4.28: Parameter estimation errors [kgm²]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$.

Figure 4.29: Torques [Nm] Applied to the Robot Joints, T_1 (solid line), T_2 (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$.

Figure 4.30: Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=5$, $P_2(0)=10$.

Figure 4.31: Parameter estimation errors [kgm²]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=5$, $P_2(0)=10$.

Figure 4.32: Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for Unit Step as Reference Signal and $P_1(0)=1.5$, $P_2(0)=3$.

Figure 4.33: Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for Unit Step as Reference Signal and $P_1(0)=1.5$, $P_2(0)=3$.

Figure 4.34: Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$ and $e_i(0)=0.2$

Figure 4.35: Parameter estimation errors $[\text{kgm}^2]$: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ and $e_i(0)=0.2$.

Figure 4.36: Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$ and $e_i(0)=0.6$.

Figure 4.37: Parameter estimation errors [kgm²]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ and $e_i(0)=0.6$.

Figure 4.38: Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$ and White Noise $\sigma = 0.01$.

Figure 4.39: Position Tracking Errors: e_1 (solid line), e_2 (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ and White Noise $\sigma = 0.01$.

Figure 4.40: Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) and $P_1(0)=1.5$, $P_2(0)=3$ and White Noise $\sigma = 0.01$.

Figure 4.41: Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$ and White Noise $\sigma = 0.003$.

Figure 4.42: Position Tracking Errors: e_1 (solid line), e_2 (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ and White Noise $\sigma = 0.003$.

Figure 4.43: Parameter estimation errors [kgm²]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ and White Noise $\sigma = 0.003$.

Figure 4.44: Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$ and 10 Bits Quantization Noise.

Figure 4.45: Position Tracking Errors: e_1 (solid line), e_2 (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ and 10 Bits Quantization Noise.

Figure 4.46: Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ and 10 Bits Quantization Noise.

4.5 OFAC of Robot Manipulators using High-Gain Observer-Algorithm #3

The last adaptive control algorithm presented in this document is proposed by Lee and Khalil [3]. This algorithm will be referred to as algorithm #3 throughout this document. This adaptive algorithm belongs to the group of output feedback adaptive control algorithms. The first two algorithms presented here and proposed by Schwartz [1, 2] are based on model reference adaptive control theory. The Lee and Khalil algorithm can be applied to the class of feedback linearizable systems, where the states, can be expressed as a function of the input and output and their derivatives. Moreover, if the system is observable, and it is globally stable with the state feedback controller, semiglobal stability can be achieved for the same system with the output feedback controller and high-gain observer [4].

The development of Lee and Khalil's algorithm and previously presented algorithms [1, 2] is motivated by the fact that robot manipulators usually do not have joint velocity sensors. Therefore, the joint velocity is not measurable. Omitting the sensor for the joint velocities results in a decrease of the weight and implementation cost of the robot manipulator. Another good reason to avoid the use of the measured joint velocity signal is that a measured joint velocity signal is heavily corrupted with noise. Accordingly, algorithm #3 requires just measurement of the joint positions of the robot manipulator. That is the mutual characteristic of all three algorithms presented in this document. The joint velocity of the manipulator is estimated instead of being measured.

The *separation principle* that is well known and often used in the design of linear systems cannot be directly applied in the design of nonlinear systems. The *separation principle* in linear systems allows separate design of a state feedback controller and a state observer. That principle guarantees, if the state feedback controller stabilizes the system and meets design requirements, the output feedback controller using some of the estimated states will stabilize the system as well. Due to the nonlinear robot dynamics and unknown robot parameters the *separation principle* cannot be directly applied in the design of the output feedback controller in a robot manipulator case. It will result in

instability of the robot manipulator. Fortunately, the observer that is robust to uncertainties in robot nonlinear dynamics will allow application of the separation principle in a robot case. The class of observers known as high-gain observers have those capabilities. Accordingly, algorithm #3 uses the high-gain observer to estimate the joint velocity and uses the separation principle in a controller design.

Exploiting the properties of the *separation principle*, one can design a state feedback controller to meet the design requirements. An output feedback controller will be obtained from the state feedback controller by replacing unmeasurable states with the estimates provided by a high-gain observer. That design procedure is possible, if states that are estimated in the output feedback are globally bounded in the state feedback [22]. Moreover, the output feedback system will asymptotically recover the state feedback performance if the observer gain is sufficiently high.

Following the above procedure one can obtain the output feedback adaptive controller for the robot manipulator, called here algorithm #3. The globally stable state feedback controller for the robot manipulator is designed according to Berghuis [5], while the adaptation law is given by Khalil [4]. The adaptation law with the parameter projection feature proposed by Khalil [4] is chosen because it guarantees that all states with the state feedback controller are bounded. That is the property that makes the application of the *separation principle* possible. Then, the unmeasurable joint velocity is substituted with the estimate provided by the high-gain observer. Hence, the obtained output feedback controller [3] based on the high-gain observer will asymptotically recover performance achieved with the state feedback controller [5] that uses the adaptation law according to Khalil [4].

The high-gain observer will exhibit the *peaking phenomenon* if there is an initial estimation error. That is one of the most important drawbacks of using the high-gain observer in nonlinear control algorithms. The *peaking phenomenon* and ways to overcome it, have been investigated by a number of authors [12, 13, 16]. The *peaking phenomenon* is an impulsive-like behaviour that could be transmitted through the control signal to the plant. If transmitted to the plant, it will result in the peaking of the plant states that could be taken out of the region of the attraction. Then, the controlled system will become unstable. An increase of the observer gain will result in peaking with shorter

periods and higher amplitude. The increase of the observer gain will result in a faster observer that is capable of bounding the estimation error to zero in a shorter period of time. Nevertheless, chances of destabilizing the system will increase as a result of the *peaking phenomenon*. The solution for this problem is saturation of the control signal outside the region of the interest. Saturating the control signal, we will be able to increase the observer gains and make the observer faster without destabilizing the system. Furthermore, the decrease of the high-gain observer parameter ε , that is inversely proportional to the values of the observer gains, will result in a region of attraction of the output feedback controller with a high-gain observer that will approach the region of attraction of the full state feedback controller. In the special case, when $\varepsilon = 0$ the observer gains are ∞ , and the region of attraction of the output feedback controller with the high-gain observer is identical to the region of attraction of the full state feedback controller [22].

4.5.1 Description of Algorithm #3

After giving the fundamental facts about the separation principle and characteristics of the high-gain observers we can proceed with the presentation of the main parts of algorithm #3 [3]. Those parts are: the generator of the desired trajectory, the linearization law, the adaptation law, the high-gain observer and the robot dynamic model. The generation of the desired trajectory is structurally identical to the reference model used in the previous two MRAC algorithms [1, 2]. Here, we call it the generator of the desired trajectory instead of the reference model. As a result, algorithm #3 is the output feedback adaptive control algorithm whose goal is to achieve perfect following of the desired trajectory q_d with the robot manipulator arm. The generator of the desired trajectory W_d generates the desired trajectory q_d and its first $\dot{q}_d$ and second $\ddot{q}_d$ order derivatives from the reference signal using the same transfer function as the reference model W_m . Accordingly, the transfer function of the generator of the desired trajectory W_d is,

$$W_d = \frac{4}{s^2 + 4s + s} = W_m \quad (4.64)$$

The main goal of this algorithm is to generate the torques T_L applied to the each joint of the manipulator that will result in perfect tracking of the desired trajectory q_d using only the joint position feedback.

The dynamic model of the robot manipulator is given by *form II* that is explained in detail in Chapter 3. In *form II* matrices M and C have specific properties that are exploited by the Lee and Khalil algorithm during the derivation of the adaptation law. In that case the matrix C has a linear dynamic structure in $\dot{q}$. The dynamic model of the two DOF robot manipulator operating in the horizontal plane for algorithm #3 is given as,

$$T = M(q)\ddot{q} + C(q, \dot{q})\dot{q} \quad (4.65)$$

The matrices M and C are previously defined according to equations (3.9) and (3.17).

The robot dynamic model used by Lee and Khalil [3] has six unknown parameters while the robot used by Schwartz [1, 2] has two unknown parameters. The elements of the matrices M and C in these two dynamic models are not the same because they are predefined by the number of the unknown parameters and the way the inertial masses, centrifugal and Coriolis forces are modeled. The goal of this thesis is to compare the two MRAC based algorithms [1, 2] with the output feedback adaptive algorithm proposed by Lee and Khalil [3]. For that reason, we will use the robot dynamic model with two unknown parameters proposed by Schwartz [1] during the simulations of algorithm #3. Accordingly, we will be able to compare the results obtained through the simulations of algorithm #3 [3], algorithm #2 [2] and algorithm #1 [1]. As a result of the robot model change in algorithm #3, we had to rewrite and adapt equations for the linearizer, the high-gain observer and the adaptation law according to the robot model given by Schwartz [1].

The general equation of the linearization law is determined according to the theory of the feedback linearization. According to Berghuis [5] the linearization law is given as,

$$T_L = \hat{M}(q)\ddot{q}_d + \hat{C}(q, \dot{q}_r)\dot{q}_d - K_d\dot{e} - K_p e \quad (4.66)$$

where $T_L \in R^{n \times 1}$ is the vector of the torques applied to each joint of the robot manipulator, $\hat{M}(q) \in R^{n \times n}$ is the estimate of the inertial matrix, $\hat{C}(q, \dot{q}_r) \in R^{n \times n}$ is the estimate of the matrix of centrifugal and Coriolis forces, $e \in R^{n \times 1}$ is vector of position tracking error while $K_d, K_p \in R^{n \times n}$ are positive definite constant diagonal matrices. The robot

manipulator has two joints, therefore $n=2$. In the linearization law according to equation (4.66), we use estimates of the matrices M and C . This is due to the fact that true values of the robot parameters P_1 , P_2 and the true value of the joint velocities that are used for the calculation matrices M and C are unknown. The vector $\dot{q}_r \in R^{n \times 1}$ in equation (4.46) is defined as,

$$\dot{q}_r = \dot{q} - \alpha e = \dot{q} - \alpha(q - q_d) \quad (4.67)$$

while,

$$\alpha = \frac{\alpha_0}{1 + \|e\|} = \frac{\alpha_0}{1 + \sqrt{e_1^2 + e_2^2}} \quad (4.68)$$

where α_0 is positive number that is arbitrarily chosen.

After presenting the linearization law, the next step is to determine the adaptation law. Equation (4.65) under the action of the linearization law (4.66) becomes,

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} = \hat{M}(q)\ddot{q}_d + \hat{C}(q, \dot{q}_r)\dot{q}_d - K_d\dot{e} - K_p e \quad (4.69)$$

Subtracting the term $M(q)\ddot{q}_d + C(q, \dot{q})\dot{q}_d$ from both sides of equation (4.69) one will get [8],

$$M(q)\ddot{e} + C(q, \dot{q})\dot{e} = \hat{M}(q)\ddot{q}_d - M(q)\ddot{q}_d + \hat{C}(q, \dot{q}_r)\dot{q}_d - C(q, \dot{q})\dot{q}_d - K_d\dot{e} - K_p e \quad (4.70)$$

Using the linear property of the matrix C , the term $\hat{C}(q, \dot{q}_r)$ can be written as,

$$\hat{C}(q, \dot{q}_r) = \hat{C}(q, \dot{q} - \alpha e) = \hat{C}(q, \dot{q}) - \alpha \hat{C}(q, e) \quad (4.71)$$

After defining the error of matrix M and matrix C as,

$$\tilde{M}(q) = \hat{M}(q) - M(q), \tilde{C}(q, \dot{q}) = \hat{C}(q, \dot{q}) - C(q, \dot{q}) \quad (4.72)$$

one can write equation (4.70) as,

$$M(q)\ddot{e} + C(q, \dot{q})\dot{e} + K_d\dot{e} + K_p e = \tilde{M}(q)\ddot{q}_d + \tilde{C}(q, \dot{q})\dot{q}_d - \alpha \hat{C}(q, e)\dot{q}_d \quad (4.73)$$

After writing the term $\alpha \hat{C}(q, e)\dot{q}_d$ as $\alpha \tilde{C}(q, e)\dot{q}_d + \alpha C(q, e)\dot{q}_d$ and rearranging equation (4.73) one will get,

$$M(q)\ddot{e} + C(q, \dot{q})\dot{e} + \alpha C(q, e)\dot{q}_d + K_d\dot{e} + K_p e = \tilde{M}(q)\ddot{q}_d + \tilde{C}(q, \dot{q}_r)\dot{q}_d \quad (4.74)$$

or in linear regression form equation (4.75) is,

$$\tilde{M}(q)\ddot{q}_d + \tilde{C}(q, \dot{q}_r)\dot{q}_d = Y_r(q, \dot{q}_r, \dot{q}_d, \ddot{q}_d)\tilde{P} \quad (4.75)$$

Equation (4.75) will later be used for determination of the adaptation law. According to Berghuis [5], one can consider the following function,

$$V = \frac{1}{2} \left(s^T M(q) s + e^T K_p e + \tilde{P}^T \Gamma^{-1} \tilde{P} \right) \quad (4.76)$$

as a positive definite Lyapunov function candidate. After differentiating equation (4.76) and introducing several substitutions and bounds, as described by Berghuis [5], one can obtain the first order derivative of the Lyapunov function

$$\dot{V} \leq -\kappa_1 \|s_1\| - \kappa_2 \left\| \frac{\alpha}{2} e \right\| + \tilde{P}^T \Gamma^{-1} (\dot{\tilde{P}} + \Gamma Y_r^T s) \quad (4.77)$$

where,

$$\begin{aligned} \kappa_1 &= K_d - 3\alpha_0 M_{\max} - 2\alpha_0 C_{\max} > 0 \\ \kappa_2 &= \frac{4}{\alpha_0} K_p - K_d - 2\alpha_0 (M_{\max} + C_{\max}) > 0 \\ s &= \dot{e} + \alpha e \\ s_1 &= \dot{e} + \frac{\alpha}{2} e = s - \frac{\alpha}{2} e \end{aligned} \quad (4.78)$$

According to equation (4.78) one can state that the first two terms in equation (4.77) are always negative. In order to ensure that the state feedback system is globally stable, the third term in equation (4.77) has to be

$$\tilde{P}^T \Gamma^{-1} (\dot{\tilde{P}} + \Gamma Y_r^T s) \leq 0 \quad (4.79)$$

Hence, one can choose the adaptation law as $\dot{\tilde{P}} = -\Gamma Y_r(q, \dot{q}_r, \dot{q}_d, \ddot{q}_d) s$, but in order to achieve boundedness of all trajectories in the system Lee and Khalil [3] proposed the adaptation law with a parameter projection feature that is given by equation (4.80).

$$\dot{\hat{P}} = \begin{cases} -\Gamma Y_r(q, \dot{q}_r, \dot{q}_d, \ddot{q}_d) s & \text{if } \hat{P}_i \geq b_i \text{ \& } Y_r s \leq 0 \text{ or } \hat{P}_i \leq a_i \text{ \& } Y_r s \geq 0 \\ -\Gamma \left(1 + \frac{a_i - \hat{P}_i}{\delta} \right) Y_r(q, \dot{q}_r, \dot{q}_d, \ddot{q}_d) s & \text{if } \hat{P}_i \geq b_i \text{ \& } Y_r s \geq 0 \\ -\Gamma \left(1 + \frac{\hat{P}_i - a_i}{\delta} \right) Y_r(q, \dot{q}_r, \dot{q}_d, \ddot{q}_d) s & \text{if } \hat{P}_i \leq a_i \text{ \& } Y_r s \leq 0 \end{cases} \quad (4.80)$$

where $\Gamma \in R^{n \times n}$ is positive diagonal matrix. The term $Y_r(q, \dot{q}_r, \dot{q}_d, \ddot{q}_d)$ can be determined from equation (4.75) as,

$$Y_r(q, \dot{q}_r, \dot{q}_d, \ddot{q}_d) = \begin{bmatrix} \ddot{q}_d & 3\ddot{q}_d(1 + \cos q_2) - \dot{q}_d(\dot{q}_{r1} + 2\dot{q}_{r2}) \sin q_2 \\ 0 & \ddot{q}_d(2 + \cos q_2) + \dot{q}_d \dot{q}_{r1} \sin q_2 \end{bmatrix} \quad (4.81)$$

The high-gain observer is given with the following equation,

$$\begin{bmatrix} \dot{\hat{e}}_1 \\ \dot{\hat{e}}_2 \\ \ddot{\hat{e}}_1 \\ \ddot{\hat{e}}_2 \end{bmatrix} = A \begin{bmatrix} \hat{e}_1 \\ \hat{e}_2 \\ \dot{\hat{e}}_1 \\ \dot{\hat{e}}_2 \end{bmatrix} + B \left[-\ddot{q}_d - M^{-1}(q)C(q, \dot{q})\dot{q} + M^{-1}(q)T_L \right] + LD(\varepsilon) \begin{bmatrix} e_1 - \hat{e}_1 \\ e_2 - \hat{e}_2 \\ \dot{e}_1 - \dot{\hat{e}}_1 \\ \dot{e}_2 - \dot{\hat{e}}_2 \end{bmatrix} \quad (4.82)$$

where,

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}, L = \begin{bmatrix} L_1 & 0 & 0 & 0 \\ 0 & L_1 & 0 & 0 \\ 0 & 0 & L_2 & 0 \\ 0 & 0 & 0 & L_2 \end{bmatrix}, D(\varepsilon) = \begin{bmatrix} \frac{1}{\varepsilon} & 0 & 0 & 0 \\ 0 & \frac{1}{\varepsilon} & 0 & 0 \\ \frac{1}{\varepsilon^2} & 0 & 0 & 0 \\ 0 & \frac{1}{\varepsilon^2} & 0 & 0 \end{bmatrix} \quad (4.83)$$

while $e_i = q_i - q_d$ and $\hat{e}_i = \hat{q}_i - q_d$. The parameter ε is the high-gain observer design parameter. If the parameter ε is decreased, the observer will become faster and estimation error will decay more quickly towards zero, the peaking period will decrease, peaking amplitude will increase and the region of attraction under output feedback will approach the region of attraction under full state feedback. Due to the peaking caused by the decrease in parameter ε , the system with the unsaturated control signal could become unstable [16]. A clear recommendation of how to determine the value of the parameter ε does not exist. That is the main disadvantage of the Lee and Khalil's [3] algorithm from the theoretical point of view. The high-gain observer is stable if the following matrix given by equation (4.84) has negative eigenvalues.

$$A - LD(\varepsilon) = \begin{bmatrix} -\frac{L_1}{\varepsilon} & 0 & 1 & 0 \\ 0 & -\frac{L_1}{\varepsilon} & 0 & 1 \\ -\frac{L_2}{\varepsilon^2} & 0 & 0 & 0 \\ 0 & -\frac{L_2}{\varepsilon^2} & 0 & 0 \end{bmatrix} \quad (4.84)$$

The matrix in this form has four identical eigenvalues that are equal to $-\sqrt{\frac{L_2}{\varepsilon^2}}$. Choosing a very small ε close to zero, we will obtain a very fast observer with a high disturbance rejection feature. Moreover, the response of the output state feedback with a saturated control signal will approach the response of the state feedback system. A very small value of the parameter ε will result in a very high value of the eigenvalues. The bandwidth of the observer will dramatically increase, making it very susceptible to measurement noise.

4.5.2 Selection of the Design Parameters for Algorithm #3

There are five design parameters K_p, K_d, Γ, L and α_0 that have to be determined prior to the algorithm implementation. The high-gain observer design parameter ε has to be determined as well. This is one of the disadvantages of algorithm #3 over algorithm #2 [2] and algorithm #1 [1]. These unknown design parameters are part of the equations of the linearization law, the adaptation law and the high-gain observer for algorithm #3. Depending on the chosen values of these design parameters, algorithm performance could range from extremely good to extremely poor. A procedure for determining their optimal values does not exist.

In the proof of stability of the state feedback system we have several conditions that have to be satisfied. Using these conditions, one can derive the equations that will determine the range of the values of the design parameters K_p, K_d and α_0 , but not their optimal values [5]. The values of the observer gain in matrix L and the gain in matrix Γ in the adaptation law have to be determined using an iterative design procedure because there are no recommendations or procedures for their determination. In order to achieve

this, the first order derivative of the Lyapunov function candidate (4.77) must be negative, and constants κ_1 and κ_2 given by equation (4.78) have to be positive. From equation (4.78) one can determine the range for the gain matrices K_d and K_p . Before that, we have to determine the minimum eigenvalues $M_{\min}$ and maximum eigenvalues $M_{\max}$ of the inertial matrix M , and maximum eigenvalue $C_{\max}$ of the matrix C according to the equations,

$$0 < M_{\min} \leq \|M(q)\| \leq M_{\max} \quad (4.85)$$

$$C_{\max} = \frac{1}{2} \sum_{k=1}^n \|C_k(q)\|, n = 2 \quad (4.86)$$

where $\|M(q)\| = [\lambda_{\max}(M^T M)]^{\frac{1}{2}}$, $\lambda_{\max}$ is maximum eigenvalue, while the elements of the matrix C_k are calculated according to equation (4.87).

$$C_{k,ij} = \left(\frac{\partial M_{ij}}{\partial q_k} + \frac{\partial M_{ik}}{\partial q_j} - \frac{\partial M_{jk}}{\partial q_i} \right), i, j, k = 1, \dots, n; \quad (4.87)$$

Using equation (4.87), we have obtained,

$$C_1 = \begin{bmatrix} 0 & -2P_2 \sin q_2 \\ 2P_2 \sin q_2 & 0 \end{bmatrix}; C_2 = \begin{bmatrix} -2P_2 \sin q_2 & -2P_2 \sin q_2 \\ -P_2 \sin q_2 & 0 \end{bmatrix} \quad (4.88)$$

Using equation (4.85) and (4.86) and knowing that the true parameter values are $P_1 = 1$ and $P_2 = 2$ we are able to determine $M_{\min}$, $M_{\max}$ and $C_{\max}$ as,

$$M_{\max} = 11; M_{\min} = 0.18; C_{\max} = 4 \quad (4.89)$$

After arbitrarily choosing that $\kappa_1 = 5$, $\kappa_2 = 20$ in order to ensure that the first derivative of the Lyapunov function (4.77) is negative along the desired trajectory and that $\alpha_o = 1$, we have,

$$K_d \geq 3\alpha_o M_{\max} + 2\alpha_o C_{\max} + \kappa_1 = 46 \quad (4.90)$$

So we choose that $K_d = 50$, and from the following equations [5] that have to be satisfied,

$$K_p \geq \frac{K_d + 2\alpha_o (M_{\max} + C_{\max}) + \kappa_2}{4} = 25 \quad (4.91)$$

$$K_p > \frac{\alpha_o^2 M_{\max}^2}{4M_{\min}} = 168.05 \quad (4.92)$$

we can choose $K_p = 170$. After that, we have to check the validity of the arbitrarily chosen value of $\alpha_o = 1$ using the condition given by equation (4.93). If the condition given by equation (4.93) is not satisfied we have to go through the described procedure again.

$$\alpha_o < \min \left\{ \frac{K_d}{3M_{\max} + 2C_{\max}}, \frac{2K_p}{K_d} \right\} \quad (4.93)$$

$$\alpha_o < \min \{1.22, 6.8\}$$

Hence, the arbitrarily chosen value of $\alpha_o = 1$ is correct. Choosing even higher values of the gains K_d and K_p will result in faster error dynamics which are a positive feature. On the other hand, the higher values of the gains K_d and K_p will result in an increase of the torque values that are necessary to achieve that error dynamics. This is not desirable.

4.5.3 Observations about High-gain Observer Design

Parameter ε

Choosing the right value of the high-gain observer parameter ε is a very challenging operation that will have a significant influence on the performance of algorithm #3. The high-gain observer gains are inversely proportional to the value of the parameter ε . According to observer theory, it is desirable to have a higher value in the observer gain in order to reduce the estimation error towards zero in a shorter period of time. Hence, one could conclude that it is desirable to set the value of the parameter ε as close as possible to zero. With a decrease of the value of the parameter ε , the response of the output feedback system should become closer and closer to the response of the state feedback system. In order to test that statement, we have performed the following simulation.

We have performed a computer simulation with an initial position error for two different values of the high-gain observer parameter $\varepsilon = 0.0008$ and $\varepsilon = 0.01$. We have used the output feedback adaptive controller with high-gain observer and unsaturated torques. We have chosen an adaptation law without the parameter projection feature in order to obtain the results that will better show the influence of the value of the high-gain observer parameter ε and the *peaking phenomenon* on the system response. If we use the adaptation law with the parameter projection feature like the one that is used in algorithm

#3 proposed by Lee and Khalil [3] the influence of the value of the high-gain observer parameter ε would not be a significant. This is due to the fact that the adaptation law with the parameter projection feature guarantees boundedness of all trajectories in the system. In fact, the results shown in Figures 4.47-4.50 are not obtained by use of the original Lee and Khalil algorithm [3] or algorithm #3 as it is called throughout this thesis.

The robot dynamic model used in simulations is proposed by Schwartz [1]. The simulations are performed with an initial position error in order to provoke peaking of the observer states that will enable the investigation of the *peaking phenomenon*. Next, using the same initial settings we have performed the simulation with the full state feedback system. The output feedback adaptive controller is based on this full state feedback controller. Before performing the simulation, we expected that the simulation case with a smaller value of the parameter ε will have a response closer to the response of the full state feedback. According to Figures 4.47 and 4.48 that illustrate the position trajectories for the first and second joint, that statement was not confirmed. From Figures 4.47 and 4.48 it is visible that a response with the output feedback controller with unsaturated torques is closer to the full state feedback response when parameter ε has a higher value.

The reason for such a surprising result is the *peaking phenomenon*. Due to the initial position error and the high observer gains, peaking of the observer states was generated and transmitted through the control signal to the robot manipulator states. The smaller value of the parameter ε has resulted in higher observer gains and a higher peaking amplitude. Due to the higher peaking amplitude, the chances of taking the robot manipulator states out of the region of attraction have increased. Therefore, in the case of any control algorithm with the high-gain observer, unsaturated torques and initial tracking error, closer following of the state feedback response will be achieved with an increase in the value of the parameter ε , until the system becomes unstable due to a high estimation error caused by large value of the parameter ε .

The *peaking phenomenon* and its negative effects on the system stability can be overcome with saturation of the control signal. Accordingly we have performed the computer simulation for two different values of the parameter ε but now with saturated torques. The simulation results are given in Figures 4.49 and 4.50. After saturation of the torques, peaking of the states has been put under control. As a result of the peaking of the

manipulator states being overcome, the response with output feedback and a smaller value of the parameter ε is closer to the state feedback response. In short, after the saturation of the torques we can decrease the value of the parameter ε and obtain a faster observer without causing the peaking of the plant states that could result in an unstable system. The only limit on how much we can decrease the value of the parameter ε is the amount of the measurement noise that is introduced to the system. Decreasing the value of the parameter ε will increase the value of the observer gains, observer eigenvalues, observer bandwidth and, unfortunately, observer susceptibility to noise.

According to the performed computer simulations it is clear that a uniform recommendation for the value of the parameter ε does not exist. If the torques are not saturated, better performance will be achieved with an increase of the value of the parameter ε because the peaking of the states will decrease. But there is the limitation on how much we can increase the value of the parameter ε . An increase of the value of the parameter ε will result in a slower observer with higher estimation error that could lead to an unstable system [16]. On the other hand, if the torques applied to the each joint of the robot manipulator are saturated, better performance will be achieved with a decrease in the value of the parameter ε . In this case, the limitation on how much we can decrease the value of the parameter ε is determined by the amount of the measurement noise that is introduced to the system. The amount of the measurement noise that is introduced to the system is time varying due to a number of factors that are present in a robot manipulator work environment. Therefore, the limit on how far we can go with decreasing the value of the parameter ε is time varying as well. In both saturated and unsaturated torques cases, there is a risk of obtaining an unstable system if the value of the parameter ε is not chosen properly. That is a significant problem of all algorithms that use the high-gain observer [16, 22].

Figure 4.47: Position Trajectory for the First Joint: Full State Feedback (solid line) and Output Feedback without Saturation for $\varepsilon = 0.0008$ (dotted line) and $\varepsilon = 0.01$ (dashed line)

Figure 4.48: Position Trajectory for the Second Joint: Full State Feedback (solid line) and Output Feedback without Saturation for $\varepsilon = 0.0008$ (dotted line) and $\varepsilon = 0.01$ (dashed line)

Figure 4.49: Position Trajectory for the First Joint: Full State Feedback (solid line) and Output Feedback with Saturation for $\varepsilon = 0.008$ (dotted line) and $\varepsilon = 0.05$ (dashed line)

Figure 4.50: Position Trajectory for the Second Joint: Full State Feedback (solid line) and Output Feedback with Saturation for $\varepsilon = 0.008$ (dotted line) and $\varepsilon = 0.05$ (dashed line)

4.5.4 Observations about the Stability Proof of Algorithm #3

The output feedback adaptive algorithm, explained here and originally proposed by Lee and Khalil [3], is based on the state feedback adaptive algorithm suggested by Berghuis [5]. The state feedback adaptive algorithm by Berghuis is proven to be asymptotically stable. Moreover, all internal signals are bounded. The boundedness of all states of the state feedback adaptive controller is the first condition that has to be satisfied in order to allow the use of the *separation principle* during the design of the output feedback adaptive controller for a nonlinear system such as a robot manipulator. The *separation principle* is a design technique widely exploited in the world of linear systems and mostly is not applicable to nonlinear systems because it could result in an unstable system. The *separation principle* is applicable to nonlinear systems if the states that are estimated in the output feedback algorithm are bounded in the state feedback algorithm [22]. Moreover, the unavailable states in the output feedback algorithm have to be estimated by an observer such as a high-gain observer.

The observer will provide estimates that asymptotically converge to their true values. The disturbance term coming from the uncertainties in modeling of the nonlinear functions will be annulled when the observer parameter ε heads toward zero. The disturbance term is the difference between the real robot manipulator and its dynamic model. That difference is particularly significant when the parameter estimates are far away from their true values. That is always the case during the first few seconds of robot operation because the initial parameter values are unknown and they are often arbitrarily chosen. Due to its high gains, the high-gain observer is able to reject the effect of the disturbance term and to provide the estimates with very small estimation error.

Lee and Khalil [3] use the properties of the *separation principle* and high-gain observers to obtain the adaptive output feedback algorithm based on the state feedback algorithm by Berghuis [5]. The properties of the *separation principle* and high-gain observers are used in the proof of stability. Using the asymptotically stable state feedback algorithm with adaptation law with the parameter projection feature, Lee and Khalil [3] were able to obtain an output feedback adaptive algorithm by replacing joint velocity with the estimate provided by the high-gain observer. In this design procedure, the use of the *separation principle* was possible as a result of the joint velocity signal in the state

feedback being bounded. Boundedness of states in the state feedback is achieved with use of an adaptation law with the parameter projection feature. The estimate of the joint velocity is provided by the observer.

According to the theory behind the *separation principle* and high-gain observers, the obtained output feedback algorithm will asymptotically recover performance of the state feedback algorithm for sufficiently small value of the parameter ε [3]. Furthermore, the equations representing the output feedback algorithm will become identical to the equations representing the state feedback algorithm when the parameter $\varepsilon = 0$. That property is particularly noticeable if the output feedback algorithm is given in a standard singularly perturbed form.

The dynamical system given in the singularly perturbed form is actually the state feedback system model where some of the states are multiplied by a small positive parameter. In the case of algorithm #3, that small positive number is the design parameter of the high-gain observer ε . Algorithm #3 can be presented in singularly perturbed form as follows [3],

$$\dot{x} = Ax + B \left[-\ddot{q}_d - M^{-1}(x_1, q_d)C(x, q_d, \dot{q}_d)(x_2 + \dot{q}_d) + M^{-1}(q)T_L(\hat{x}, \hat{P}) \right] \quad (4.94)$$

$$\begin{aligned} \varepsilon \dot{\xi} = \bar{A}\xi + \varepsilon B \left[\left\{ M^{-1}(\hat{x}_1, q_d)C(\hat{x}, q_d, \dot{q}_d)(\hat{x}_2 + \dot{q}_d) - \right. \right. \\ \left. \left. M^{-1}(x_1, q_d)C(x, q_d, \dot{q}_d)(x_2 + \dot{q}_d) \right\} + \left\{ M^{-1}(x_1, q_d) - M^{-1}(\hat{x}_1, q_d) \right\} \right] T_L(\hat{x}, \hat{P}) \end{aligned} \quad (4.95)$$

where,

$$\bar{A} = \begin{bmatrix} -L_1 & 0 & 1 & 0 \\ 0 & -L_1 & 0 & 1 \\ -L_2 & 0 & 0 & 0 \\ 0 & -L_2 & 0 & 0 \end{bmatrix} \quad (4.96)$$

and

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} e \\ \dot{e} \end{bmatrix} = \begin{bmatrix} e_1 \\ e_2 \\ \dot{e}_1 \\ \dot{e}_2 \end{bmatrix} \quad (4.97)$$

ξ is scaled estimation error and it is defined as,

$$\xi_i = \frac{1}{\varepsilon^{2-i}}(x_i - \hat{x}_i), i = 1, 2 \quad (4.98)$$

$$\xi = \begin{bmatrix} \xi_1^T & \xi_2^T \end{bmatrix}^T \quad (4.99)$$

The estimates of x in equations (4.94) and (4.95) can be written as

$$\hat{x} = \begin{bmatrix} \hat{x}_1 \\ \hat{x}_2 \end{bmatrix} = \begin{bmatrix} \hat{q}_1 - q_d \\ \hat{q}_2 - q_d \\ \dot{\hat{q}}_1 - \dot{q}_d \\ \dot{\hat{q}}_2 - \dot{q}_d \end{bmatrix} = x - \bar{D}(\varepsilon)\xi \quad (4.100)$$

in order to achieve the exact singularly perturbed form. The matrix $\bar{D}(\varepsilon)$ in equation (4.100) is defined as

$$\bar{D}(\varepsilon) = \begin{bmatrix} \varepsilon & 0 & 0 & 0 \\ 0 & \varepsilon & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.101)$$

The proof of stability of algorithm #3 is based on the stability analysis of singularly perturbed systems. That is made possible by representing algorithm #3 in singularly perturbed form by equations (4.94) and (4.95). According to the theory of singularly perturbed systems, the closed loop output feedback systems given by equations (4.94) and (4.95) can be considered for the purpose of stability analysis as an interconnection of the two subsystems. The first subsystem is obtained by setting the high-gain observer design parameter ε to zero. The obtained system is called the reduced system or slow model, and it is actually the closed loop system under state feedback. It is represented by equation (4.94). The second subsystem is called in the control literature the fast model or boundary layer system. The second subsystem is obtained by introducing the time change t/ε and then setting the parameter ε to zero [22]. As a result of that action the following term in equation (4.95)

$$\varepsilon B \left[\left\{ M^{-1}(\hat{x}_1, q_d) C(\hat{x}, q_d, \dot{q}_d) (\hat{x}_2 + \dot{q}_d) - \right. \right. \quad (4.102)$$

$$\left. \left. - M^{-1}(x_1, q_d) C(x, q_d, \dot{q}_d) (x_2 + \dot{q}_d) \right\} + \left\{ M^{-1}(x_1, q_d) - M^{-1}(\hat{x}_1, q_d) \right\} \right]$$

is zero. Accordingly, the fast model is then represented with following equation

$$\varepsilon \dot{\xi} = \bar{A} \xi \quad (4.103)$$

The names fast and slow model come from the fact that the variable x is considered to be slower in comparison with the variable ξ that is usually called the fast variable. In order to prove the stability of the output feedback algorithm, we need to define the Lyapunov function. According to the theory of singularly perturbed systems, the Lyapunov function of the output feedback algorithm is actually the linear combination of the Lyapunov functions for the slow model (in this case that is the state feedback system) and the fast model given by equation (4.103). The Lyapunov function of the singularly perturbed system is called a composite Lyapunov function.

According to Lyapunov stability theory, after determining that the Lyapunov function candidate is a positive definite function, we also have to determine that its first order derivative is negative definite in order to achieve asymptotic stability. The composite Lyapunov function is positive definite if the Lyapunov functions of the slow and fast model are positive definite. The Lyapunov functions for the slow and fast model are also chosen to have their first order derivatives negative definite. This does not guarantee that the first order derivative of the composite Lyapunov function is a negative definite. The first order derivative of the composite Lyapunov function has four terms. Two of them are first order derivatives of the Lyapunov functions for the slow and fast model, while the other two come from the effect of the interconnection between the fast and slow model. The two interconnection terms are not always negative definite. As a result of that, the first order derivative of the composite Lyapunov function might not be negative definite. Fortunately, if the Lyapunov functions for the slow and fast model are carefully chosen, it is possible to achieve that the two interconnection terms are negative definite for sufficiently small parameter ε . There is no clear recommendation on how to choose the Lyapunov functions for the slow and fast model, except that they have to be quadratic type Lyapunov functions [22, 3]. Accordingly, if the Lyapunov functions for the slow and fast model are properly chosen, then the stability analysis of the singularly perturbed system is actually the stability analysis of the slow and fast model. Accordingly, stability analysis of the output feedback algorithm with a high-gain observer is translated into the stability analysis of the state feedback algorithm [5] and the boundary-layer system of the output feedback algorithm that is given by equation (4.103). A significant

simplification of the stability analysis is made possible by representation of the output feedback algorithm in the singularly perturbed form (4.94, 4.95).

The positive definite Lyapunov function proposed by Lee and Khalil [3] for state feedback system is:

$$V = \frac{1}{2} \left(s^T M(q) s + e^T K_p e + \tilde{P}^T \Gamma^{-1} \tilde{P} \right) \quad (4.104)$$

The first order derivative of the Lyapunov function V along the trajectories of the state feedback system is given by equation (4.105).

$$\dot{V} \leq -\kappa_1 \|s_1\|^2 - \kappa_2 \left\| \frac{\alpha}{2} e \right\|^2 + \tilde{P}^T \Gamma^{-1} (\dot{\tilde{P}} + \Gamma Y_r^T s) < -\beta_1 \quad (4.105)$$

where β_1 is some positive number. The Lyapunov function for the fast model is

$$v = \xi^T P_0 \xi \quad (4.106)$$

where P_0 is a positive definite matrix that is solution of the following Lyapunov equation

$$P_0 \bar{A} + \bar{A}^T P_0 = -I \quad (4.107)$$

The matrix $\bar{A}$ is given by equation (4.96), while I is identity matrix.

After defining the Lyapunov function for the slow and fast model, we can define the sets Ω, Θ_δ and Σ as,

$$\Omega = \{x | V \leq c\} \quad (4.108)$$

$$\Theta_\delta = \{P | a_i - \delta \leq P_i \leq b_i + \delta\}, i = 1, 2 \quad (4.109)$$

$$\Sigma = \{\xi | v \leq \rho \varepsilon^2\} \quad (4.110)$$

where δ, ρ and c are positive numbers. The positive number c is defined as

$$c > c_1 + c_2 + c_3 \quad (4.111)$$

$$c_1 = \max \left(\frac{1}{2} e^T K_p e \right); c_2 = \max \left(\frac{1}{2} s^T M(q) s \right); c_3 = \max \left(\frac{1}{2} \tilde{P}^T \Gamma^{-1} \tilde{P} \right)$$

Defining the set Θ_δ according to equation (4.109) acknowledges that the parameter estimates are bounded. Definition of that set comes from the adaptation law with the parameter projection feature. The set Ω is interior to the region of the attraction $\mathfrak{R}$ of the state feedback controlled system. According to Lee and Khalil [3], for sufficiently large ρ there exists an ε_1 dependant on ρ such that for every ε chosen within the range

$0 < \varepsilon \leq \varepsilon_1$ the origin of the output feedback adaptive controller with high-gain observer will recover performance of the state feedback controller. Actually, for every ε chosen within the range $0 < \varepsilon \leq \varepsilon_1$ we have that

$$\dot{V} \leq 0, \forall (x, \xi) \in \{V = c\} \cap \Theta_\delta \times \Sigma \quad (4.112)$$

$$\dot{v} \leq -\frac{v}{2\varepsilon\lambda_{\max}(P_0)}, \forall (x, \xi) \in \Omega \cap \Theta_\delta \times \{v = \rho\varepsilon^2\} \quad (4.113)$$

Hence the conclusion is that the set,

$$\Xi = \Omega \times \Theta_\delta \times \Sigma \quad (4.114)$$

is a positively invariant subset of the region of attraction $\mathfrak{R}$ of the state feedback system. As a result, the set Ξ is positively invariant, once the trajectory enters the set it stays inside the set thereafter. Inside the set Ξ the scaled estimation error ξ is of order $O(\varepsilon)$ while outside of the set Ξ the estimation error is of order $O(1/\varepsilon)$.

Hence the question, is there any guarantee that the trajectory will get into the set Ξ ? According to Lee and Khalil [3], for any bounded $\hat{x}(0)$ and $\left\{ \left\{ x(0), \hat{P}(0) \right\} \in \Omega_0 \times \Theta_\delta \right\} \in \Omega \times \Theta_\delta$ there exist ε_1 such that for every ε chosen within the range $0 < \varepsilon \leq \varepsilon_1$, the trajectory will enter the set Ξ in the finite period of time T_ε and it will stay inside the set Ξ for every $t \geq T_\varepsilon$. In the special case when $\varepsilon = 0$ the region of attraction of the output feedback controller Ξ will become identical to the region of attraction $\mathfrak{R}$ of the state feedback controller. Moreover, the time T_ε tends to zero when the parameter ε tends to zero, due to the fact that the set Ξ approaches to the region of attraction $\mathfrak{R}$ of the state feedback controller with decrease of the parameter ε [3, 22].

According to the given facts, it is clear that performance and stability of the output feedback algorithm are in a significant correlation with the value of the parameter ε . For a sufficiently small value of the parameter ε the output feedback adaptive controller will asymptotically recover performance achieved with the state feedback adaptive controller. As parameter ε tends to zero, the trajectory and the region of attraction of the output feedback controller will approach the trajectory and the region of attraction of the state feedback controller. For that reason it would be convenient to know

what is the numerical value of the upper bound of the parameter ε . We are certain that the adaptive output feedback algorithm will recover performance achieved under the state feedback if the value of the parameter ε is smaller than its upper bound ε_1 . One can obtain the upper bound ε_1 of the parameter ε using the stability analysis of the output feedback algorithm.

According to Khalil [4], we do not choose ε according to stability analysis because it could result in the very conservative value of ε_1 and then a very small value of the parameter ε . A very small value of the parameter ε is not desirable because the bandwidth of the high-gain observer is inversely proportional to the value of the parameter ε . On the other hand, a large value of the parameter ε could result in an unstable algorithm due to large observer error [16, 22]. Therefore, we are looking for the largest value of the parameter ε that will recover performance of the state feedback and make the algorithm robust to measurement noise [4]. That value of the parameter ε is mostly larger than the upper bound ε_1 obtained through stability analysis.

According to Khalil [4], there are two ways to determine the value of the parameter ε that will be used in the algorithm. In the first approach we choose a large value of the parameter as a starting point and then we decrease it until satisfactory performance is reached. In the second approach we determine the value of the upper bound ε_1 of the parameter ε through stability analysis. Thereafter, we start simulations with $\varepsilon = \varepsilon_1$, and then keep increasing the value of the parameter ε until we reach the value of the parameter ε beyond which the algorithm has poor performance. Here, we will present the second approach for the adaptive output feedback controller of a two DOF robot manipulator operating in the horizontal plane.

We will determine the value of the upper bound ε_1 using inequalities from the stability analysis of the output feedback adaptive algorithm given by Lee and Khalil [3]. For every $(x, \hat{P}, \xi) \in \{V = c\} \cap \Theta_\delta \times \Sigma$ the inequality (4.115) is valid for the second order derivative of the Lyapunov function given by equation (4.104).

$$\dot{V} \leq -\beta_1 + \|s\| \|T_L(\hat{x}, \hat{P}) - T_L(x, \hat{P})\| \leq -\beta_1 + k_1 \|s\| \|\xi\| \leq -\beta_1 + k_2 \varepsilon \quad (4.115)$$

where β_1 , k_1 and k_2 are positive numbers. According to the condition $\forall (x, \hat{P}, \xi) \in \{V = c\} \cap \Theta_\delta \times \Sigma$ for which the inequalities (4.115) are valid, we can say that the trajectory is on the border of the positively invariant set Ξ . We know from the previous analysis that the scaled estimation error ξ is order of $O(\varepsilon)$ in the set Ξ .

According to Lee and Khalil [3], for the upper bound of the parameter ε , the following inequality is valid, $\forall (x, \hat{P}, \xi) \in \{V = c\} \cap \Theta_\delta \times \Sigma$.

$$\varepsilon_1 \leq \frac{\beta_1}{k_2} \quad (4.116)$$

In order to find the numerical value of the upper bound ε_1 , we will run the simulation using the following initial settings and design parameters.

$$\begin{aligned} K_d &= 50I_{2 \times 2}, K_p = 170I_{2 \times 2}, \Gamma = 100I_{2 \times 2}, P_1(0) = 1.5, P_2(0) = 3, \\ q_1(0) &= q_2(0) = 0.1, \alpha_0 = 1, \varepsilon = 0.01, \end{aligned} \quad (4.117)$$

Previously, we have determined that

$$\begin{aligned} C_{\max} &= \frac{1}{2} \sum_{k=1}^{n=2} \|C_k(q)\| \Rightarrow C_{\max} = 4 \\ M_{\max} &\geq \|M(q)\| \Rightarrow M_{\max} = 11 \end{aligned} \quad (4.118)$$

Using the sequence of the unit steps as a reference signal we have that

$$|q_d| \leq 1; |\dot{q}_d| \leq 1.5; |\ddot{q}_d| \leq 8.1 \quad (4.119)$$

The maximum torques applied to each joint of the robot manipulator can be calculated from the following equations.

$$\begin{aligned} |T_1|_{\max} &= |\hat{M}_{11}| |\ddot{q}_d|_{\max} + |\hat{M}_{12}| |\ddot{q}_d|_{\max} + |C_{11}| |\dot{q}_d|_{\max} + |C_{12}| |\dot{q}_d|_{\max} + \|K_d\| \|\dot{e}\| + \|K_p\| \|e\| \\ |T_2|_{\max} &= |\hat{M}_{21}| |\ddot{q}_d|_{\max} + |\hat{M}_{22}| |\ddot{q}_d|_{\max} + |C_{21}| |\dot{q}_d|_{\max} + |C_{22}| |\dot{q}_d|_{\max} + \|K_d\| \|\dot{e}\| + \|K_p\| \|e\| \end{aligned} \quad (4.120)$$

Also, we defined that the initial position and velocity tracking errors are bounded as $|e| \leq 1 \text{ rad}$ and $|\dot{e}| \leq 1 \text{ rad/sec}$. We can choose the smaller or larger error bounds, but once we choose them we cannot go outside of them. Using equation (4.120) and the numerical values given by (4.117), (4.118) and (4.119) one can determine the torque saturation levels.

$$\begin{aligned} \text{sat}\{T_1\} &= 11 \cdot 8.1 + 11 \cdot 8.1 + 4 \cdot 1.5 + 4 \cdot 1.5 + 170 \cdot \sqrt{2} + 50 \cdot \sqrt{2} = 501.32 \approx 502 \\ \text{sat}\{T_1\} &= 11 \cdot 8.1 + 11 \cdot 8.1 + 4 \cdot 1.5 + 0 \cdot 1.5 + 170 \cdot \sqrt{2} + 50 \cdot \sqrt{2} = 495.3 \approx 496 \end{aligned} \quad (4.121)$$

The values of the saturation torques are extremely high for practical applications, but they are acceptable for the purpose of computer simulation. The next step is to find the value of the positive number c that is given by equation (4.111). Accordingly, we have

$$\begin{aligned} c_1 &= \max \left(\frac{1}{2} e^T K_p e \right) \leq 170 \\ c_2 &= \max \left(\frac{1}{2} s^T M(q) s \right) \leq 44 \\ c_3 &= \max \left(\frac{1}{2} \tilde{P}^T \Gamma^{-1} \tilde{P} \right) \leq 0.1 \\ c &> c_1 + c_2 + c_3 = 214.1 \\ c &= 215 \end{aligned} \quad (4.122)$$

Thereafter, we can run the computer simulation in which we are looking for the period of time t_0 where the Lyapunov function given by equation (4.104) is equal to c . We have

$$V(t_0) = c = 215 \text{ at } t_0 = 0.84 \text{ sec} \quad (4.123)$$

and we can determine that

$$\beta_1 \approx -\kappa_1 \|s_1\|^2 - \kappa_2 \left\| \frac{\alpha}{2} e \right\|^2 + \tilde{P}^T \Gamma^{-1} (\dot{\tilde{P}} + \Gamma Y_r^T s) = 98.587 \text{ at } t_0 = 0.84 \text{ sec} \quad (4.124)$$

$$\|s\| = 3.056 \text{ at } t_0 = 0.84 \text{ sec} \quad (4.125)$$

$$\|T_L(\hat{x}, \hat{P}) - T_L(x, \hat{P})\| = 157.136 \text{ at } t_0 = 0.84 \text{ sec} \quad (4.126)$$

From equation (4.116) we can determine the upper bound ε_1

$$\varepsilon_1 \leq \frac{\beta_1}{k_2} \approx \frac{\beta_1 \varepsilon}{\|s\| \|T_L(\hat{x}, \hat{P}) - T_L(x, \hat{P})\|} = 0.002053 \approx 0.002 \quad (4.127)$$

The value of the upper bound ε_1 is even smaller than the value of the parameter $\varepsilon = 0.01$ that is used in the computer simulation. According to Khalil [4] that is the expected result because the value of the upper bound ε_1 obtained through the stability analysis is very conservative. According to the recommendation by Khalil [4] the value of the upper bound ε_1 obtained in this procedure can be used as a starting value of the parameter ε .

Then, we can increase the value of the parameter ε until we reach the value of the parameter ε beyond which the algorithm has poor performance. Therefore, we have run a number of simulations with different values of the parameter ε using initial settings and design parameters given by (4.117). We have determined the poor performance of algorithm #3 has become evident for the value of parameter $\varepsilon = 0.07$ while unstable behaviour is observed for $\varepsilon = 0.1$.

4.5.5 Simulation of Algorithm #3

A computer simulation of algorithm #3 is performed according to the simulation scenario given in section 4.2. The simulation scenario for this algorithm is summarized in Table 4.7. Table 4.7 lists the initial values of the parameter estimates, the initial values of the tracking errors $e_i(0)$, the type of reference signal and the type of noise and its standard deviation. The last column of Table 4.7 lists figures which show the results for each simulation case. The simulations are performed according to the simulation block diagram illustrated in Figure 4.51.

Numeric values of the parameter and tracking error settling times, maximum torques, and position errors for each simulation case are given in Table 4.8. The values of the parameter and tracking error settling times are calculated according to the 1% criterion. In simulation case 4a, the parameter and tracking error settling times are larger than $\pm 0.01 \text{ kgm}^2$ and $\pm 0.01 \text{ rad}$, respectively. In this case, the values of the parameter and tracking error settling times are determined according to the 5% criterion. There are two values given for maximum torques. The first one, T_{max} is the value of maximum torque during initial oscillations. The second value, T_{ss} is the maximum torque at the end of the initial oscillations when perfect tracking, or almost perfect tracking, of a desired trajectory is achieved.

Table 4.7: Initial Settings for Different Simulation Cases.

Sim case	$P_{ver}=1$ $P_1(0)$	$P_{ver}=2$ $P_2(0)$	$e_i(0)$	ε	Sat(Ti)	White noise σ	Quantization noise	Reference signal	Figures
1a	1.5	3	0	0.002	NO	0	NO	square	4.52-4.55
1b	5	10	0	0.002	NO	0	NO	square	4.56-4.57
2	1.5	3	0	0.002	NO	0	NO	unit step	4.58-4.59
3a	1.5	3	0.2	0.002	131.8/ 64.4	0	NO	square	4.60-4.61
3b	1.5	3	0.6	0.002	401.6/ 189.8	0	NO	square	4.62-4.63
3c	1.5	3	0.6	0.002	100	0	NO	square	N/A
4	1.5	3	0	0.002	500	0.01	NO	square	4.64-4.66
4a	1.5	3	0	0.05	500	0.01	NO	square	4.67-4.69
4b	1.5	3	0	0.02	500	0.003	NO	square	4.70-4.72
5	1.5	3	0	0.03	500	0	YES	square	4.73-4.74

Table 4.8: Numeric Values of Settling Times, Maximum Torques and Position Errors Obtained Through Simulations.

Sim case	t_{st1}	t_{st2}	t_{ss1}	t_{ss2}	T_{1max}	T_{2max}	T_{1ss}	T_{2ss}	e_{1max}	e_{2max}
1a	8.05	5.13	0	0	82.5	41	82.5	41	0.002	0.001
1b	8.09	5.82	0.1	0	260	120	82.5	41	0.011	0.001
2	0.98 $\tilde{P}_1(\infty) = -0.19$	2.33 $\tilde{P}_2(\infty) = 0.1$	0	0	78	36	0	0	0.002	0.001
3a	8.99	8.04	1.55	0.99	131.8	64.4	82.5	41	0.2	0.2
3b	12.01	8.05	1.38	0.99	401.6	189.8	82.5	41	0.6	0.6
3c	12.01	8.05	1.38	0.99	100	100	82.5	41	0.6	0.6
4	diverging	diverging	$e_1 \in [\pm 1.6]$	$e_1 \in [\pm 0.59]$	500	500	N/A	N/A	1.61	0.59
4a	~ 20 $\tilde{P}_1 \approx -0.33$	~ 20 $\tilde{P}_2 \approx 0.06$	4.65 $e_1 \in [\pm 0.05]$	0 $e_2 \in [\pm 0.05]$	114.8	63.3	90	46	0.066	0.048
4b	~ 15 $\tilde{P}_1 \approx -0.33$	~ 15 $\tilde{P}_2 \approx 0.05$	18.65	18.64	107.6	57.6	85	44	0.05	0.03
5	~ 15 $\tilde{P}_1 \approx -0.33$	~ 15 $\tilde{P}_2 \approx 0.05$	8.71	4.83	111.1	51.5	82.5	41	0.05	0.02

The simulations of algorithm #3 are performed using the feedback controller gains, $K_p=170$ and $K_d=50$. We had to choose high values for these feedback controller gains in order to satisfy the stability condition for algorithm #3. The procedure for their selection is detailed in section 4.5.2. The high values of feedback controller gains resulted in a closed loop bandwidth of $\omega_n = 13.03 \text{ rad/s}$. The two previously presented MRAC algorithms [1, 2] have a closed loop bandwidth of $\omega_n = 2 \text{ rad/s}$ that is predefined by the critically damped second order reference model given by equation (4.7). The high K_p and K_d values have also resulted in fast error dynamics for algorithm #3. The fast error dynamics result in short settling times of the position tracking errors. The high values of K_p and K_d will increase the algorithm's sensitivity to noise as a result of the bandwidth increase.

The adaptation gains for each of the three algorithms are set to the value that resulted in best performance. Algorithm #1 is simulated with adaptation gain 1 while algorithm #2 is simulated with adaptation gain 0.5. In the case of algorithm #1, simulations diverged for adaptation gains greater than 2. In the case of algorithm #2, parameter estimates did not converge to their true values for adaptation gain greater than 1. In the case of algorithm #3, the parameter estimation errors did not converge to zero if the adaptation gain is less than 1. For the values where the adaptation gain is greater than 1, the parameter estimates started converging to their true values. For instance, the parameter estimates converged to their true values in approximately 500sec with the adaptation gain set to 5 and the initial parameter estimates set to 50% of their true value. Accordingly, we kept increasing the adaptation gain in algorithm #3 until we reached a settling time for the parameter estimation error similar to the value achieved with algorithm #2. This procedure resulted in an adaptation gain of 100 for algorithm #3. This value is 200 times greater than the adaptation gain used in algorithm #2. Nevertheless, both algorithms achieved similar values for the settling times of the parameter estimation errors, when the initial parameter estimates were set to 50% of their true value. An increase of the adaptation gain in algorithm #3 from 0.5 to 100 resulted in a decrease of the maximum position tracking error by 10 times.

Accordingly, it is expected that algorithm #3 will achieve shorter settling times of position tracking errors and a smaller value for the maximum position tracking errors in

simulation cases without noise than algorithms #1 and #2. This is due to faster error dynamics and a greater closed loop bandwidth due to higher feedback gains. Furthermore, algorithm #3 uses a higher value of adaptation gain than the other two algorithms. As such, this comparison is not of value. On the other hand, significant performance degradation could occur in simulation cases with measurement and quantization noise due to high values of K_p and K_d and the use of a high-gain observer.

The responses for simulation case 1a are illustrated in Figures 4.52-4.55. The simulation is performed with a value of high-gain observer design parameter $\varepsilon = 0.002$. This value of parameter ε is equal to its upper bound ε_1 determined in section 4.5.3. This value of parameter ε guarantees the asymptotic stability of algorithm #3. According to Figure 4.52, which illustrates desired and actual trajectories, it is noticeable that perfect tracking is achieved almost immediately due to the fast error dynamics associated with high feedback gains. Figure 4.53 illustrates position tracking errors. It is visible that tracking errors quickly converged towards zero. Furthermore, the maximum value of tracking errors for the first and the second joint are $0.002rad$ and $0.001rad$, respectively. As a result of the tracking errors being smaller than $0.01rad$ for any point in time, the settling times of the tracking errors, according to the 1% criteria, are zero for both joints.

Figure 4.54 illustrates quickly converging parameter estimation errors. The settling times of the parameter estimation errors achieved with algorithm #3 are shorter than those of algorithm #1 and similar to the values achieved with algorithm #2. The maximum torques, during and after initial oscillations, are identical. The maximum torques are $82.5Nm$ for the first joint and $41Nm$ for the second. These values for the maximum torques are identical to maximum torques generated in algorithm #2 for the same simulation case and slightly smaller than those generated in algorithm #1.

In simulation case 1b the initial values of the parameter estimates are set to $P_1(0) = 5kgm^2$ and $P_2(0) = 5kgm^2$. This initial setting resulted in the initial parameter estimation errors of $\tilde{P}_1(0) = 4kgm^2$ and $\tilde{P}_2(0) = 8kgm^2$. These values of the initial parameter estimation errors are the maximum acceptable initial parameter estimation errors for algorithm #1. For any higher values of initial parameter estimation errors, algorithm #1 diverges. Figure 4.56 illustrates desired and actual trajectories for the first

and second joint of the robot manipulator. According to Figure 4.56, perfect tracking is quickly achieved due to the high values of feedback controller gains. The parameter estimation errors are shown in Figure 4.56. The parameter estimation error settling times are almost the same as the values obtained in simulation case 1a. The settling times of the tracking errors are 0.1sec for the first joint and 0sec for the second joint.

As part of simulation case 1b, we also tried to determine the maximum value of initial parameter estimation errors for which the system will exhibit unstable behaviour. For algorithm #1, the maximum values of initial parameter estimation errors are $\tilde{P}_1(0) = 4$ and $\tilde{P}_2(0) = 8$. On the other hand, algorithm #2 had converging responses with $\tilde{P}_1(0) = 50$ and $\tilde{P}_2(0) = 100$ and torques saturated at 100Nm. Accordingly, we ran simulations with algorithm #3 using the initial parameter estimation errors of $\tilde{P}_1(0) = 50$ and $\tilde{P}_2(0) = 100$ and torques saturated at 100Nm. Algorithm #3 has achieved short tracking error settling times of $\sim 2\text{sec}$ despite high initial parameter estimation errors. The parameter estimates converged to their true values in less than 10sec. Algorithm #3 is robust to high initial parameter estimation errors. The high-gain observer is capable of producing accurate state estimates despite huge initial parameter estimation errors. Accurate state estimates resulted in the ability of algorithm #3 to produce an appropriate control signal that will result in quick convergence of the parameter estimation errors and the position tracking errors.

Figure 4.58 illustrates the desired and actual trajectories when the reference signal is a unit step. The initial settings are given according to simulation case 2. Perfect trajectory following is quickly achieved. The maximum position tracking errors are 0.002rad and 0.001rad for the first and second joint. The tracking error settling times, according to the 1% criteria, are zero. Due to the “poor” identifiability property of the unit step as a reference signal, the parameter estimates did not converge to their true values. The parameter estimation error was $\tilde{P}_1(\infty) = -0.19\text{kgm}^2$ and $\tilde{P}_2(\infty) = 0.1\text{kgm}^2$. This is a slightly smaller parameter estimation error than in the case of algorithm #1 and algorithm #2. The maximum torque values are identical in all three algorithms.

Figure 4.60 shows the desired and actual trajectories when the initial position errors are $e_i(0) = 0.2\text{rad}$ and initial parameter estimation values are 1.5 times greater than

their true values. The torques applied to the first and the second joint of the robot manipulator are saturated at $131.8Nm$ and $64.4Nm$. We chose these saturation levels because the maximum torque values of algorithm #2 for this simulation case are $T_{1max} = 131.8Nm$ and $T_{2max} = 64.4Nm$. Algorithm #2 has significantly outperformed algorithm #1 for this simulation case. According to Figure 4.60 it is clear that the position tracking errors quickly converge towards zero. The tracking error settling times are less than $1.55sec$. This very short tracking error settling time is achieved due to the fast error dynamics of algorithm #3 and the high adaptation gain. Figure 4.61 illustrates that the parameter estimates quickly converge to their true values, achieving settling times of around $9sec$. The maximum value of the position tracking error for both joints is $0.2rad$, which is actually the initial position error. It is significant that position error was not increased during initial oscillations.

Figures 4.62 and 4.63 illustrate the results obtained using the initial position error $e_i(0) = 0.6rad$ and initial parameter estimation values 1.5 times greater than their true values. The torques applied to first and second joint are saturated at $401.6Nm$ and $189.8Nm$. These two saturation levels are equal to the maximum torque values in algorithm #2 for this simulation case. It is visible from Figure 4.62 that perfect tracking of the desired trajectory is quickly achieved. The tracking error settling times are around $1sec$. Algorithm #1 showed unstable behaviour in this simulation case. Figure 4.63 shows the parameter estimation errors. It is visible from Figure 4.63 that both parameter estimation errors saturate for a short period of time, due to the adaptation law, with the parameter projection feature and a high adaptation gain. The settling times of the parameter estimation errors are around $9sec$. The maximum position tracking error is equal to the initial position error like in the simulation case with $e_i(0) = 0.2rad$.

Simulation case 3c has the same initial setting as simulation case 3b, except that the torque saturation levels are different. In simulation case 3b torques are saturated at $401.6Nm$ and $189.8Nm$, while in simulation case 3c both torques are saturated at $100Nm$. Despite the saturation of torques at a much lower level, obtained values of settling times for parameter estimation errors and position tracking errors are also identical in both 3b

and 3c simulation cases. Furthermore, the maximum tracking errors are identical in both simulation cases. Algorithm #3 is robust to high initial position errors.

This surprising result is possible due to the fact that the torque saturation levels are greater than the maximum torque values T_{iss} after the initial oscillations are finished. When the initial oscillations are finished, maximum torque values are $82.5Nm$ for the first joint and $41Nm$ for the second. Therefore, by setting the torque saturation level at $100Nm$, we did not saturate torques after the initial oscillations were gone. During the initial oscillations we also did not saturate the torques except when a high-gain observer peaking was exhibited. Due to the fact that the value of the high-gain observer design parameter $\varepsilon = 0.002$ is very small, the observer gains are large. The peaking period is short but the peaking amplitude is large when ε is small. The peaking phenomenon is exhibited only when the initial position error is different from zero. In this simulation case, the initial position error is $e_i(0) = 0.6rad$. Due to the small value of ε and the large initial position error, the system will exhibit peaking for a very short period of time with a very large amplitude. The peaking of the system states will result in the controller generating¹ the torques on the order of 10^5Nm for a very short period of time. This large torque is not generated because it is necessary to reduce the tracking error, but because of the property of the high-gain observer to exhibit peaking when $e_i(0) \neq 0$. The saturation of the torques at $100Nm$ did not have much influence on the system responses and settling times because we saturated the torques generated by the peaking phenomenon. In the case of algorithm #2, saturation of the torques resulted in an increase of the parameter estimation error and position tracking error settling times and the value of the maximum tracking error by two times.

In simulation case 4, we corrupted the measurements of the position signal with Gaussian noise with a standard deviation of $\sigma = 0.01rad$. The initial parameter estimates are 1.5 times greater than their true values, while the initial position errors are zero. Algorithm #3 design parameters are exactly the same as in the previous simulation cases. The value of the high-gain observer design parameter is $\varepsilon = 0.002$. This small value of ε resulted in a very large observer bandwidth. Therefore, it was expected that the

¹ only in computer simulation

algorithm's performance will degrade when the measurements are corrupted with Gaussian noise. This performance degradation is worse than expected. Figure 4.64 shows tracking of the desired trajectory is not achieved. The joints somehow follow the desired trajectory but with extremely high tracking error. Figure 4.65 presents large position tracking errors. The tracking error is bouncing between $\pm 1.6rad$ for the first joint and $\pm 0.59rad$ for the second joint. The parameter estimation error is diverging from zero, as is illustrated in Figure 4.66. Therefore, we did not determine the settling times for parameter estimation errors and position tracking errors. The torques applied to each joint of the manipulator were bouncing between the upper and lower saturation level that was set at $\pm 500Nm$. The torques were saturated all the time during the simulation. In this simulation case algorithm #3 was outperformed by algorithm #2 in every respect.

The design parameters of algorithm #3, in the simulation case with Gaussian noise, were identical to the design parameters in the previous simulation cases where the algorithm showed excellent performance. Nevertheless, the algorithm demonstrated performance degradation. The main reason for the algorithm's performance degradation is a large observer bandwidth due to the small value of parameter ε . The value of parameter ε is inversely proportional to the value of the observer gains. We, therefore, decided to increase the value of the high-gain observer design parameter ε over its upper bound of $\varepsilon_1 = 0.002$, in order to decrease the high-gain observer bandwidth and the algorithm's susceptibility to noise. The upper bound ε_1 is analytically determined through stability analysis of algorithm #3. The proof of stability for algorithm #3, given by Lee and Khalil [3], is not valid any more if the value of parameter ε is greater than ε_1 . An increase of the value of ε was necessary in order to achieve acceptable performance of algorithm #3 with measurement noise. An increase of ε results in a decrease of the high-gain observer bandwidth. Accordingly, the algorithm is less sensitive to the noise. Therefore, determination of a value of parameter ε that will do the job is a challenging design problem. There is no clear procedure to determine the value of parameter ε , so we had to use an iterative design procedure that is time consuming. We ran a number of simulations with different values of ε and decided that the best performance of algorithm #3 with Gaussian noise $\sigma = 0.01rad$ is achieved when $\varepsilon = 0.05$. Furthermore, we had to

decrease the value of the adaptation gain from 100 to 0.3 because the simulation with measurement noise diverges when the adaptation gain is 100. Accordingly, the convergence rate of the parameter estimates will be significantly decreased in comparison to simulation cases without noise.

A computer simulation of algorithm #3 for simulation case 4a is performed with the same initial settings as in simulation case 4 but the value of parameter ε is changed from 0.002 to 0.05 and adaptation gain is changed from 100 to 0.3. Figure 4.67 shows the desired and actual trajectories. Figure 4.68 shows the position tracking error for both joints. The maximum values of the position tracking error are $0.066rad$ for the first joint and $0.048rad$ for the second. The algorithm bounded the tracking error for the first joint within a range of $\pm 0.05rad$ in $5sec$, while for the second joint the tracking error remained within the range of $\pm 0.03rad$. This result is very similar to the results achieved with algorithm #2. Figure 4.69 illustrates the parameter estimation errors. In Figure 4.69, the small oscillations associated with parameter estimation error of the second parameter are noticeable. These oscillations are exhibited only when the first order derivative of the desired trajectory changes sign. The reason for the existence of these small oscillations is a large value of the parameter ε . Similar oscillations of parameter estimates are noted in simulation without measurement noise when ε was greater than 0.01. The adaptation law of algorithm #3 is not capable of identifying the true value of the robot manipulator parameters. This is a significant disadvantage of algorithm #3 over algorithm #2. The final value of the identified parameters results in an error of $0.33kgm^2$ for the first parameter and $0.06kgm^2$ for the second parameter. In terms of parameter estimation errors, algorithm #2 showed better behaviour. The maximum torque values are about $35Nm$ greater than in the case of algorithm #2. The performance of algorithm #3 with Gaussian noise $\sigma = 0.01rad$ improved significantly after increasing the value of ε from 0.002 to 0.05. Nevertheless, algorithm #2 is still a better choice, over algorithm #3, if the measurements are corrupted with Gaussian noise with standard deviation of $\sigma = 0.01rad$.

As a part of the simulation step 4b, we tested the performance of algorithm #3 when position measurements were corrupted with Gaussian noise with a standard deviation of $\sigma = 0.003$. Since the standard deviation of Gaussian noise in this simulation case is smaller than in simulation case 4a, we expected to be able to decrease the value of

parameter ε without degrading the algorithm's performance. The value of ε for simulation case 4a was 0.05. After performing a number of simulations with different values of parameter ε we discovered that the best performance is achieved when parameter $\varepsilon = 0.02$. Figure 4.70 illustrates the desired and actual trajectories. Figure 4.71 illustrates a significant decrease in position tracking errors within the first 10sec. The position tracking errors remain within a range of $\pm 0.01rad$ after 18.6sec. This is slightly shorter than in the case of algorithm #2 and algorithm #1. The maximum values of tracking errors are also slightly smaller than in the case of algorithms #2 and #1.

Figure 4.72 illustrates the parameter estimation errors. Like in simulation case 4a, the adaptation law was unable to identify the true parameter values. As a result, we have parameter estimation errors of approximately $0.33kgm^2$ for the first parameter estimate and $0.05kgm^2$ for the second parameter estimate. In this simulation case the parameter estimates reached their final values 5sec earlier than in simulation case 4a where the noise level was higher. In terms of the parameter estimation errors, algorithm #2 outperformed algorithm #3 and algorithm #1 because it was able to bound the parameter estimation errors within a range of $\pm 0.01kgm^2$. The maximum torque values are about 25Nm greater than in the case of algorithm #2. Even with a lower level of Gaussian noise, algorithm #2 is still a better choice over algorithm #3.

Simulation case 5 tests the influence of the 10 bit A/D converter on the performance of algorithm #3. The simulation results obtained in this simulation case are very similar to the results obtained in simulation case 4b. Similar to the simulation cases with Gaussian noise, we ran a number of simulations with different values of ε . We discovered that the best algorithm performance is achieved when the parameter $\varepsilon = 0.03$. Figure 4.73 illustrates the desired and actual trajectories. Figure 4.74 shows the converging position tracking errors. The settling times of the tracking errors are 8.7sec for the first joint and 4.8sec for the second joint according to the 1% criteria. The settling time of the tracking error for the first joint is equal to the value achieved with algorithm #2, while for the second joint it is 4sec shorter. The maximum values of tracking error are smaller than in the cases of algorithms #2 and #1. The maximum torque values are about 28Nm greater than in the case of algorithm #2. As in the previous two simulation cases with Gaussian noise, the adaptation law was unable to estimate true parameter values. On

the other hand, algorithm #1 and algorithm #2 had converging parameter estimates for this simulation case. The parameter estimation error for the first parameter estimate is 0.33kgm^2 while for the second parameter estimate is 0.05kgm^2 .

The simulation results of algorithm #3 presented in Figures 4.52-4.75 are obtained using the feedback controller gains $K_p=170$ and $K_d=50$. These high values of feedback controller gains resulted in faster error dynamics of algorithm #3 than algorithms #1 and #2. As a result of this, algorithm #3 has outperformed the two previously tested algorithms in terms of settling times and maximum position errors for every simulation scenario without measurement or quantization noise. The maximum torques were within the same range like in the simulation cases of the other two algorithms. For this reason, it would be interesting to compare the parameter estimation errors and the position tracking errors of the three algorithms when the feedback controller gains of algorithm #3 are set to $K_p=4$ and $K_d=4$. In this case all three algorithms would have the same closed loop bandwidth. These values of feedback controller gains are not permitted from the theoretical point of view because they violate the proof of stability of algorithm #3 as is shown in section 4.5.2. Besides changing the values of the controller gains, we also changed the value of the adaptation gain in algorithm #3 from 100 to 15 in order to achieve the similar values of the settling times and maximum tracking errors of algorithm #3 and algorithms #1 and #2. Figures 4.76 and 4.77 illustrate the position tracking errors for all three algorithms for the first and the second joint respectively when the initial parameter estimates are set to 50% of their actual value.

Figures 4.78 and 4.79 illustrate the parameter estimation errors for the first and the second parameter respectively. The algorithms showed similar responses. Algorithm #2 has the smallest value of maximum tracking error. If we did not decrease the value of the adaptation gain for algorithm #3 from 100 to 15, algorithm #3 would have the smallest value of the maximum tracking error. If the adaptation gain of algorithm #3 is set to 100, the maximum tracking error would be 10 times smaller than is shown in Figures 4.76 and 4.77. Furthermore, the settling times of the position tracking errors and the parameter estimation errors of algorithm #3 would be shorter by approximately 5sec than it is shown in Figures 4.76-4.79. Increasing the value of the adaptation gain from 15 to 100 did not result in a significant increase of the maximum torque values. Hence,

algorithm #3 is the better choice over algorithms #2 and #1 in simulation cases without measurement and quantization noise. This is due to the fact that, by increasing the value of the adaptation gain in algorithm #3, we can decrease the settling times and maximum tracking errors without increasing the value of maximum torques. This observation is only valid for the simulation cases without measurement or quantization noise. In the case of algorithm #1 maximum acceptable adaptation gain is 2 while in case of algorithm #2 is 1.

Figures 4.80 and 4.81 illustrate position tracking errors for the first and the second joint for all three algorithms with quantization noise associated with a 10 bit A/D converter. Figures 4.82 and 4.83 illustrate the parameter estimation errors for the first and second parameters respectively. The feedback controller gains of algorithm #3 are set to $K_p=4$ and $K_d=4$. The position tracking errors shown in Figures 4.80 and 4.81 prove a sensitivity of algorithm #3 to quantization noise. Algorithm #3 is outperformed by algorithms #2 and #1 in terms of settling times and the maximum position errors. Furthermore, algorithm #3 was unable to estimate the true parameter values and to decrease the tracking errors towards zero. Algorithm #3 showed much better behaviour in the simulation cases with quantization noise when the feedback controller gains were $K_p=170$ and $K_d=50$.

After testing the performance of algorithm #3 using different initial settings, the following conclusion can be made for the simulation cases without noise. Algorithm #3 with feedback controller gains set to $K_p=170$ and $K_d=50$ has outperformed the two previously tested algorithms in terms of settling times and maximum position errors due to the faster errors dynamics and greater closed loop bandwidth than in algorithms #1 and #2. That was an expected result and as such that comparison is not of value. When the feedback controller gains were decreased to $K_p=4$ and $K_d=4$ algorithm #3 achieved shorter settling times of the parameter estimation errors and position tracking errors with smaller values of the maximum position errors than the other two algorithms. This is due to the fact that by increasing the adaptation gain of algorithm #3 the maximum values of the tracking errors and the tracking error and parameter estimation error settling times will decrease as well. Increase of the value of the adaptation gain did not result in increase of the maximum torque values, but it was associated with bursting of the

parameter estimates. In the case of algorithms #1 and #2 we were unable to increase the adaptation gain over 2 and 1, respectively.

In simulation cases with noise, Algorithm #3, with the design parameters ($K_p=170$, $K_d=50$, $\Gamma=100$, $\varepsilon=0.002$) that result in the best performance without noise, exhibited almost unstable behaviour when noise was introduced to the systems. In this case, we increased the value of parameter ε over its upper bound ε_1 determined through stability analysis in order to decrease the bandwidth of the high-gain observer. Accordingly, the proof of the stability of algorithm #3 was violated by the fact that $\varepsilon > \varepsilon_1$. The simulation results, when position measurement is corrupted with noise obtained with parameter $\varepsilon > \varepsilon_1$, were much better than in the case when $\varepsilon \leq \varepsilon_1$. Nevertheless, algorithm #3 did not outperform algorithm #2 in the simulation cases with measurement and quantization noise. As a result of increasing the value of parameter ε , the accuracy of the high-gain observer decreased, resulting in parameter estimation errors much larger than in the case of algorithm #2. There is no clear procedure to determine the value of parameter ε for different noise levels. Therefore, the designer does not have any other choice than to go through a time consuming iterative design approach to determine the value of parameter ε for every type and level of noise. The performance of algorithm #3 significantly degraded in the simulation case with quantization noise when the feedback controller gains were decreased from $K_p=170$ and $K_d=50$ to $K_p=4$ and $K_d=4$ resulting in almost diverging responses. For these lower values of the feedback controller gains algorithm #3 was outperformed by algorithms #1 and #2 in simulation case with quantization noise.

Figure 4.51: Simulation Block Diagram for Algorithm #3.

Figure 4.52: Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$.

Figure 4.53: Position Tracking Errors: e_1 (solid line), e_2 (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$.

Figure 4.54: Parameter estimation errors [kgm²]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$.

Figure 4.55: Torques [Nm] Applied to the Robot Joints, T_1 (solid line), T_2 (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$.

Figure 4.56: Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=5$, $P_2(0)=10$.

Figure 4.57: Parameter estimation errors [kgm²]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=5$, $P_2(0)=10$.

Figure 4.58: Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for Unit Step as Reference Signal and $P_1(0)=1.5$, $P_2(0)=3$.

Figure 4.59: Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for Unit Step as Reference Signal and $P_1(0)=1.5$, $P_2(0)=3$.

Figure 4.60: Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$ and $e_i(0)=0.2$

Figure 4.61: Parameter estimation errors [kgm²]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ and $e_i(0)=0.2$.

Figure 4.62: Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$ and $e_i(0)=0.6$.

Figure 4.63: Parameter estimation errors [kgm²]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ and $e_i(0)=0.6$.

Figure 4.64: Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$, White Noise $\sigma = 0.01$ and $\varepsilon = 0.002$.

Figure 4.65: Position Tracking Errors: e_1 (solid line), e_2 (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$, White Noise $\sigma = 0.01$ and $\varepsilon = 0.002$.

Figure 4.66: Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) and $P_1(0)=1.5$, $P_2(0)=3$, White Noise $\sigma = 0.01$ and $\varepsilon = 0.002$.

Figure 4.67: Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$, White Noise $\sigma = 0.01$ and $\varepsilon = 0.05$.

Figure 4.68: Position Tracking Errors: e_1 (top), e_2 (bottom) for $P_1(0)=1.5$, $P_2(0)=3$, White Noise $\sigma = 0.01$ and $\varepsilon = 0.05$.

Figure 4.69: Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) and $P_1(0)=1.5$, $P_2(0)=3$, White Noise $\sigma = 0.01$ and $\varepsilon = 0.05$.

Figure 4.70: Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$, White Noise $\sigma = 0.003$ and $\varepsilon = 0.02$.

Figure 4.71: Position Tracking Errors: e_1 (solid line), e_2 (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$, White Noise $\sigma = 0.003$ and $\varepsilon = 0.02$.

Figure 4.72: Parameter estimation errors [kgm^2]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$, White Noise $\sigma = 0.003$ and $\varepsilon = 0.02$.

Figure 4.73: Model and Robot Trajectories: Model (solid line), q_1 (dotted line), q_2 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$ with Quantization Noise and $\varepsilon = 0.03$.

Figure 4.74: Position Tracking Errors: e_1 (solid line), e_2 (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ with Quantization Noise and $\varepsilon = 0.03$.

Figure 4.75: Parameter estimation errors [kgm²]: $\tilde{P}_1$ (solid line), $\tilde{P}_2$ (dotted line) for $P_1(0)=1.5$, $P_2(0)=3$ and with Quantization Noise and $\varepsilon = 0.03$

Figure 4.76: Position Tracking Errors for the First Joint: Algorithm #1(solid line), Algorithm #2 (dotted line) and Algorithm #3 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$.

Figure 4.77: Position Tracking Errors for the Second Joint: Algorithm #1(solid line), Algorithm #2 (dotted line) and Algorithm #3 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$.

Figure 4.78: Parameter estimation error $\tilde{P}_1$ [kgm^2]: Algorithm #1 (solid line), Algorithm #2, (dotted line) and Algorithm #3 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$.

Figure 4.79: Parameter estimation error $\tilde{P}_2$ [kgm^2]: Algorithm #1 (solid line), Algorithm #2, (dotted line) and Algorithm #3 (dashed line) for $P_1(0)=1.5$, $P_2(0)=3$.

Figure 4.80: Position Tracking Errors for the First Joint with Quantization Noise: Algorithm #1(solid line), Algorithm #2 (dotted line) and Algorithm #3 (dashed line).

Figure 4.81: Position Tracking Errors for the Second Joint with Quantization Noise: Algorithm #1(solid line), Algorithm #2 (dotted line) and Algorithm #3 (dashed line).

Figure 4.82: Parameter estimation error $\tilde{P}_1$ [kgm²] with Quantization Noise: Algorithm #1 (solid line), Algorithm #2, (dotted line) and Algorithm #3 (dashed line).

Figure 4.83: Parameter estimation error $\tilde{P}_2$ [kgm²] with Quantization Noise: Algorithm #1 (solid line), Algorithm #2, (dotted line) and Algorithm #3 (dashed line).

Chapter 5

Conclusions

A robot manipulator, as a nonlinear system, is characterized with several issues which have to be overcome in order to be able to successfully solve the trajectory following problem. The complexity of the robot manipulator control problem is even higher if the robot manipulator operates at high operational speed. This is the case with most modern robot manipulators. The influence of nonlinear dynamics such as Coriolis and centripetal forces increases with the square of operational speed. Moreover, the parameters of the robot manipulator are changed every time the load is grasped or released. The number of control techniques that can be used to solve the robot manipulator trajectory following problem is limited by the fact that the joint velocity signal is mostly unavailable or corrupted with noise. The application of full state feedback control would successfully solve the trajectory following problem. Due to the fact that the only available signal is a joint position signal, the application of the state feedback control is limited to simulations and theoretical analysis. Thus far, a number of control algorithms have been developed to solve the robot manipulator control problem. In this thesis, we have focused on three algorithms.

Algorithm #1 [1] is the model reference control algorithm that uses just the joint position signal in the adaptation and control laws. Moreover, it does not use any kind of observers because it does not use the velocity or acceleration signal or their estimates. Algorithm #2 [2] is the model reference control algorithm that uses the position signal but also requires the estimate of the velocity signal in order to implement the adaptation and control laws. The linear observer is used to estimate the velocity signal. The third

algorithm presented here is proposed by Lee and Khalil [3]. Algorithm #3 is the output feedback control algorithm that requires the joint position signal and the estimate of the joint velocity signal. The high-gain observer is used to estimate the unavailable velocity signal. Out of these three algorithms, only algorithm #3 has the proof of stability. According to a proof of the stability, for every ε chosen within the range $0 < \varepsilon \leq \varepsilon_1$, the origin of the output feedback adaptive controller with the high-gain observer will recover the performance of the state feedback controller. One of the focuses of this thesis was to determine the value of a positive number ε_1 for the robot manipulator with two degree of freedom operating in a horizontal plane. Using the stability analysis of algorithm #3 we have determined that the value of the parameter ε_1 is 0.002.

Before we compared the algorithms, we had tuned their design parameters in order to optimize their performance for each simulation case. Accordingly, we determined the value of the adaptation gain for algorithm #2 which resulted in the best performance for each simulation case. Moreover, for algorithm #3 we had to determine the value of several gains and parameters that have an influence on algorithm performance. That was a challenging task since a clear guideline for their selection does not exist. The number of parameters and gains that have to be determined is considered a significant disadvantage of algorithm #3 over algorithm #2 and, particularly, over algorithm #1. Algorithm #1 does not have any parameters that have to be determined prior to simulation or practical implementation.

In order to be able to adequately compare these three algorithms in terms of parameter estimation and tracking error settling times, maximum torques and maximum tracking errors and their sensitivity to measurement and quantization noise, we designed five simulation cases. During the first two simulation cases we tested the algorithms for different values of the initial parameter estimation errors and two types of reference signals. Algorithm #3, with value of the parameter $\varepsilon = 0.002$ and a larger closed loop bandwidth, outperformed the other two algorithms by having much shorter settling times and much smaller maximum tracking errors with slightly smaller values of maximum torques. The reason for this result is faster error dynamics due to much higher feedback gains of algorithm #3 than in other two algorithms. Algorithm #3 also had a higher value

of adaptation gain which results in faster convergence of the parameter estimates to their true values and very fast dynamics of the high-gain observer. We have also performed a simulation of algorithm #3 with a closed loop bandwidth identical to that of the other two algorithms. In this simulation case, the performance of algorithm #3 was very similar to the performance of the other two algorithms. However, by increasing the value of algorithm #3 adaptation gains, we were able to achieve better performance than with the other two algorithms. Furthermore, we did not notice any significant increase in torque maximum values when the adaptation gain was significantly increased. In the case of algorithm #3, it is possible to set the value of the adaptation gains over 100 without destabilizing the algorithm. On the other hand, algorithm #1 simulations diverged for adaptation gains greater than 2, while algorithm #2 parameter estimates did not converge to their true values for adaptation gains greater than 1. All three algorithms showed stable behaviour and the ability to estimate the true values of the parameters. However, algorithms #2 and #3 showed stable behaviour despite large¹ initial parameter estimation errors.

In the third simulation case, we tested the influence of different values of initial tracking errors on the algorithm performance. Again, algorithm #3, with a value of parameter $\varepsilon = 0.002$, significantly outperformed the other two algorithms in every observed aspect. The reason for this is faster error dynamics of algorithm #3 than in the other two algorithms, a higher value of adaptation gain and very fast dynamics of the high-observer. Moreover, algorithm #3 and algorithm #2 showed very slow performance degradation with an increase of the initial tracking errors. The only drawback was a significant increase in the value of their torques. In the case of algorithm #2, after saturating the torques, the parameter estimation and tracking error settling times were significantly increased, along with the maximum tracking errors,. On the other hand, despite torque saturation, algorithm #3 barely showed any increase in settling times and maximum tracking errors.

In the last two simulation cases, we tested the influence of the realistic levels of measurement noise and quantization noise associated with a 10 bit A/D converter on

¹ the initial parameter estimates were fifty times greater than their true parameter values.

algorithm's performance. Surprisingly, algorithm #3, with design parameters that enabled good performance in the simulation cases without noise, exhibited poor performance when noise was introduced to the systems. The parameter estimates diverged, the torques were bouncing between the saturation levels. The maximum tracking error during the simulation was an unacceptably high $1.6rad$. After increasing the value of parameter ε over its upper bound ε_1 the performance of algorithm #3 improved significantly. Increase of the value of ε results in a decrease of the high-gain observer bandwidth and a decrease in algorithm #3's sensitivity to noise. Nevertheless, algorithm #3 did not outperform algorithm #2 in the simulation case with measurement and quantization noise, but it did achieve better results than algorithm #1 in terms of maximum tracking error and tracking error settling time. Algorithm #3 and algorithm #1 showed an inability to converge to the true parameters within a range of $\pm 0.01kgm^2$ in simulation cases with Gaussian noise, while only algorithm #3 had the same problem with quantization noise. The maximum torque values of algorithm #3 were about 30% higher than the maximum torque values of algorithms #2 and #1. In terms of maximum tracking errors, algorithm #3 has showed slightly better results than the other two algorithms. However, in the simulation case with quantization noise where algorithm #3 closed loop bandwidth was set to be identical to that of the other two algorithms, algorithm #3 showed poor responses. In this simulation case, we were unable to tune algorithm #3 performance by increasing the adaptation gain as was possible in the simulation case without noise.

In simulation cases with measurement and quantization noise, we had to increase the value of parameter ε to be over its upper bound ε_1 in order to achieve acceptable performance of algorithm #3. In doing so, we violated the main postulate of the stability proof for algorithm #3. The presence of some level of noise in measured position signal could not be avoided, no matter how carefully we performed the measurement. Therefore, in practical applications of algorithm #3, we will be forced to use the value of ε that is greater than its upper bound ε_1 determined through stability analysis. Accordingly, we can state that the stability proof for algorithm #3 does not have a practical value. In short, algorithm #3 is an excellent adaptive control algorithm from the theoretical point of view however, it has questionable practical implementation issues.

5.1 Recommendations for Future work

After completing detailed comparisons of the three adaptive control algorithms for robot manipulators using computer simulations, we would recommend implementation of the observed algorithms on the real robot manipulator such as the Carleton direct drive robot manipulator. After implementation on the real robot manipulator, it would be interesting to compare their performance, and particularly parameter estimation abilities, when the desired trajectories are persistently exciting and persistently not exciting. Moreover, it would be significant to determine the universal set of values for the unknown gains and parameters such as parameter ε that would result in satisfactory performance of algorithm #3 for different levels of the quantization noise. Furthermore, it would be significant to theoretically prove the stability of algorithm #2 and algorithm #1.

References

- [1] H.M. Schwartz, 'Model Reference Adaptive Control for Robotic Manipulators without Velocity Measurements', International Journal of Adaptive Control and signal Processing, Vol. 8, 1994.
- [2] H.M. Schwartz, 'An MRAC Output Feedback controller for Robot Manipulators', Proceedings of the 7th Mediterranean Conference on Control and Automation, Haifa, June 1999.
- [3] K.W. Lee, H.K. Khalil, 'Adaptive Output Feedback Control of Robot Manipulators using High-Gain Observers', International Journal of Control, Vol. 67, No.6, 1997.
- [4] H.K. Khalil, 'Adaptive Output Feedback Control of Nonlinear Systems Represented by Input-Output Models', IEEE Transaction on Automatic Control, Vol. 41, No.2, February 1996.
- [5] H. Berghuis, R. Ortega, H. Nijmeijer, 'A Robust Adaptive Robot Control', IEEE Transaction on Robotics and Automation, Vol. 9, No.6, December 1993.
- [6] J.J. Craig, P. Hsu, S.S. Sastry, 'Adaptive Control of Mechanical Manipulators', The International Journal of Robotics Research, Vol. 6, No.2, Summer 1987.
- [7] J.E. Slotine, W. Li, 'On the Adaptive Control of Robot Manipulators', The International Journal of Robotics Research, Vol. 6, No.3, Fall 1987.
- [8] H.M. Schwartz, G. Warshaw, T. Janabi, 'Issues in Robot Adaptive Control', Proceedings of the American Control Conference, 1990.
- [9] H. Hajjir, H.M. Schwartz, 'An Adaptive Nonlinear Output Feedback Controller for Robot Manipulators', Proceedings of the American Control Conference, San Diego, CA, June 1999.
- [10] S.S. Sastry, A. Isidori, 'Adaptive Control of Linarizable Systems', IEEE Transaction on Automatic Control, Vol. 34, No.11, February 1989.

- [11] H.K. Khalil, 'Comparison of Different Techniques for Nonlinear Output Feedback Adaptive Control', Proceedings of the 38th Conference on Decision and Control, Phoenix, AZ, December 1999.
- [12] A.N. Atassi, H.K. Khalil, 'A Separation Principle for the Stabilization of a Class of Nonlinear Systems', IEEE Transaction on Automatic Control, Vol. 44, No.9, September 1999.
- [13] H.J. Sussmann, P.V. Kokotovic, 'The Peaking Phenomenon and the Global Stabilization of Nonlinear Systems', IEEE Transaction on Automatic Control, Vol. 36, No.4, September 1991.
- [14] R. Gourdeau, 'Adaptive Control of Robotic Manipulators', Ph. D. Thesis, Carleton University, Ottawa, Canada, April 1991.
- [15] G. Warshaw, 'Investigations of Adaptive Control for a Direct Drive Robotic Manipulator', Master's Thesis, Carleton University, Ottawa, Canada, January 1990.
- [16] A. Robertsson, 'On Observer-Based Control of Nonlinear Systems', Ph. D. Thesis, Lund Institute of Technology, Lund, Sweden, 1999.
- [17] J.E. Slotine, W. Li, *Applied Nonlinear Control*, Prentice Hall, Englewood Cliffs, NJ, 1991.
- [18] K.S. Narendra, A.M. Annaswamy, *Stable Adaptive Systems*, Prentice Hall, Englewood Cliffs, NJ, 1998.
- [19] K.J. Astrom, B. Wittenmark, *Adaptive Control*, Addison-Wasley, Reading, MA, 1995.
- [20] M.W. Spong, M. Vidyasagar, *Robot Dynamics and Control*, John Wiley & Sons, New York, NJ, 1989.
- [21] K. Ogata, *Modern Control Engineering*, Prentice Hall, Upper Saddle River, NJ, 2002.
- [22] H.K. Khalil, *Nonlinear Systems*, Prentice Hall, Upper Saddle River, NJ, 2002.
- [23] J.J. Craig, *Adaptive Control of Mechanical Manipulators*, Addison-Wasley, Reading, MA, 1988.
- [24] J.J. Craig, *Introduction to Robotics: Mechanics and Control*, Addison-Wasley, Reading, MA, 1989.
- [25] A. Isidori, *Nonlinear Control Systems*, Springer-Verlag, New York, NJ, 1989.

- [26] D.P. Stoten, *Model Reference Adaptive Control of Manipulators*, Research Studies Press LTD., Taunton, England, 1990.