Desigh of adaptive control for a RT-robot

Citation for published version (APA):

Document status and date:
Published: 01/01/1993

Document Version:
Publisher’s PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:

• A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher’s website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the “Taverne” license above, please follow below link for the End User Agreement:
www.tue.nl/taverne

Take down policy
If you believe that this document breaches copyright please contact us at:
openaccess@tue.nl
providing details and we will investigate your claim.
Design of Adaptive Control
for a RT-robot

Ir. H. Bratadjandra

Professor : Prof.dr.ir. P.H.J. Schellekens
Coach : Ir. P.C. Mulders

Eindhoven University of Technology
Faculty of Mechanical Engineering
Division of Production Technology and Automation
Section Precision Engineering

Eindhoven, oktober 1993
Task description

Student : H. Bratatjandra
Professor : Prof. dr. ir. P.H.J. Schellekens
Coach : ir. P.C. Mulders
Start : October 1992
End : September 1993
Title : Design of Adaptive Control for a RT-robot

Introduction

The section Precision Engineering, of the Faculty of Mechanical Engineering of the Eindhoven University of Technology, conducts research in the field of Mechatronics. This covers the design and construction of machinery and tools as well as the development and implementation of control strategies.

Within this context a RT-robot has been realized, consisting of a Rotation and a Translation module with drives and a control system. The control system includes a force sensor, amplifiers, encoders, measuring systems, interface cards and a PC 80386 SX. This realistic industrial robot configuration is used to design and test adaptive robot control algorithms. The nonlinear behaviour of the RT-robot and the parameters uncertainly results in less tracking accuracy and even may lead to an unstable dynamic behavior. So it is desirable to investigate adaptive control strategies, which take these aspects into account during motion.

Proposal

1. Study of the hardware of the present RT-robot configuration including amplifiers, encoders, measuring system, etc.
2. Study of the software of the RT-robot controllers structure with respect to synchronization and interrupts
3. Study of adaptive control methods, with preference to the combination of an adaptive PD-controller and a computed torque method based on the model reference adaptive control approach.
4. Design of the applicable adaptive control strategy, carried out with computer simulations and control software packages.
5. Implementation of the control strategy on the real RT-robot and comparison of theoretical and practical results of the experiments.

Prof. dr. ir. P.H.J. Schellekens  Ir. P.C. Mulders
### Contents

**Summary** 5

**Preface** 7

1. **Introduction** 8
   1.1 General introduction 8
   1.2 An overview of the report 10

2. **The RT-robot System** 11
   2.1 Introduction 11
   2.2 The RT-robot 11
   2.3 The control system 12
   2.4 The major specifications of the RT-robot modules 15

3. **Modelling of The RT-robot System** 16
   3.1 Introduction 16
   3.2 Modelling of the RT-robot 17
   3.3 Translation of the number of counts of the measurement systems into the correct units 20
   3.4 Modelling of the motors 20
   3.5 Modelling of the amplifiers 21
   3.6 Modelling of the DAC's 23
   3.7 Link from computer to motor 24

   4.1 Introduction 25
   4.2 The computer-controlled system 25
   4.3 Real-time events 26

5. **Control Methods** 28
   5.1 Introduction 28
   5.2 The dynamic model of the robot 32
   5.3 The Seraji method 32
   5.4 The Craig method 36
   5.5 The Slotine & Li method 39
   5.6 The Adaptive PD method 45
   5.6 The PID method 48

6. **Simulation of The Control Algorithms** 50
   6.1 Introduction 50
   6.2 Simulation tools 51
   6.3 The desired trajectory 53
6.4 Simulation results
6.4.1 Rotation module
6.4.2 Translation module
6.5 Conclusions

7. Implementation of The Control Algorithms
7.1 Introduction
7.2 Description of the software
7.3 Implementation results
7.3.1 Implementation results of existing methods
7.3.2 Comparison of Adaptive PD and Craig method
7.3.3 Comparison of the 'basic' and 'composite' version
7.4 Conclusions

7. Implementation of The Control Algorithms

8. Conclusions and Discussion
8.1 Conclusions
8.2 Discussion

Bibliography

List of symbols

Appendices:

Appendix A Adaptation law Seraji method
Appendix B Adaptation law Craig method
Appendix C Adaptation law Slotine & Li method
Appendix D Adaptation law Adaptive PD method
Appendix E Lyapunov stability theory
Appendix F Strictly positive real system
Appendix G Linearizing the 2 DOF nonlinear model
Appendix H Specification of the measurement systems
Appendix I Persistent excitation
Appendix J Modelling of the RT-robot
Appendix K Initial values of the parameters
Summary

The section Precision Engineering, of the Faculty of Mechanical Engineering of the Eindhoven University of Technology, conducts research in the field of Mechatronics. This covers the design and construction of machinery and tools as well as the development and implementation of control strategies.

Within this context a RT-robot has been realized, consisting of a Rotation and a Translation module with drives and a control system. The control system includes a force sensor, amplifiers, encoders, measuring systems, interface cards and a PC 80386 SX. This realistic industrial robot configuration is used to design and test adaptive robot control algorithms. The highly nonlinear behaviour of the RT-robot and the parameters uncertainly results in less tracking accuracy and even may lead to an unstable dynamic behavior. So it is desirable to investigate adaptive control strategies, which take these aspects into account during motion.

Four adaptive control methods from the literature (Seraji, Craig, 'basic’ version and 'composite' version of Slotine & Li) were probed. The adaptive control methods consist of a feedforward model, a feedback model and an adaptation algorithm.

The Seraji method consists of a linearized feedforward model and time-varying 'PD' controllers (feedback model). This method is based on the Adaptive Computed Torque method and the Model Reference Adaptive Control under an assumption of slowly robot motion.

The Craig method and the 'basic’ version of the Slotine & Li method consist of a nonlinear feedforward model and fixed 'PD' controllers. These methods are based on Adaptive Computed Torque method.

The 'composite' version of Slotine & Li method is a combination of the 'basic’ version and the Least-square Estimation method.

Due to the linearization of the feedforward model the Seraji method is very dependent on the adaptation rate of the parameters. The tracking-error performance may be degraded drastically when the linearization operating point moves away from the robot. For this reason an adaptive control method which is based on the Adaptive Computed Torque method and the Model Reference Adaptive Control (similar as the Seraji method) is derived. Unlike the Seraji method this method (Adaptive PD) consists of a nonlinear feedforward model and has no assumption of slowly robot motion.

Due to the verification with a fixed controller, a non-adaptive controller (PID) is also implemented.
To investigate the behaviour of the control algorithms, simulation studies were carried out using the software package PRO-MATLAB. The simulation results show clearly that the PID control method is less accurate than the adaptive control methods. Among the adaptive control methods the nonlinear control methods (Craig, Slotine & Li and Adaptive PD) offer a better tracking accuracy than the Seraji method. The tracking accuracy of the nonlinear controllers are comparable.

Thereupon the control algorithms were implemented, as the performance needed to be experimentally verified with the actual mechanical system that contains all the realities ignored in the simulation. In the implementation the tracking accuracies of the control algorithms at the beginning of the trajectory are less significant with each other but thereafter the nonlinear controllers show a better tracking accuracies. The tracking accuracy in the simulation is better than in the implementation. The difference is caused by all realities (the influences of the DAC's, the sensors, the DC-motors and other degrees of freedom) ignored in the simulation.

Finally it is important to note that the control methods are dependent on the choice of the sampling-time, the tuning of the adaptation gains, the initialization of the parameters and the desired trajectory and only an accurate feedforward model can lead to a high performance of the adaptive methods based on the Adaptive Computed Torque method (see chapter 6 and 7).
Preface

This report is the final part of my post graduate (AI02) study Computer Aided Design and Manufacturing of Discrete Products at the Eindhoven University of Technology.

First of all I want to thank Prof.dr.ir. P.H.J. Schellekens and ir. P.C. Mulders who made it possible for me to do this research and for their many advises and comments. Further, I would like to thank the staff and students of the section Precision Engineering and the control system laboratory ("besturingslaboratorium"). It was a pleasure to work in this group.

Finally, I would like to thank all people (especially Marco) for their many useful suggestions and moral support who made it possible for me to realize this report.

Haryanto Bratadjandra
Chapter 1
Introduction

1.1 General introduction

The group WP A (Production Technology and Automation), section Precision Engineering, at the faculty Mechanical Engineering of the Eindhoven University of Technology, conducts research in the field of Mechatronics. This covers the design and construction of machinery and tools as well as the development and implementation of control strategies.

Within this context a RT-robot with a control system has been realized. The RT-robot consists of a rotation and translation module with drives. The control system includes a force sensor, amplifiers, measurement systems, interfaces cards and a computer (PC 80386 SX, 25 Mhz) with a control algorithm.

The rotation and the translation module can track a trajectory (path) in a flat horizontal plane (2 dimensional). In order to get an accurate tracking of the trajectory the control system is used, which measures the position of the modules and gives the actual positions to the computer. A control algorithm in the computer compares the actual positions with the desired positions in order to compute the control efforts. This control system belongs to the closed-loop feedback control system, as the control system utilizes an additional measure of the actual output in order to compare the actual output with the desired output response.

In order to understand and control a system, one must obtain a quantitative mathematical model of the system. The control algorithm uses the mathematical model in order to compute control efforts. Therefore an accurate mathematical model of the system is very important.

The RT-robot and the control system together form a real-time system. This implies that several computations for the rotation and translation module must be carried out at the same moment, which is impossible for a computer with a single board processor. Thus, the computation has to be mapped into a sequential program in one event and the events are synchronized by an interrupt signal. Chapter 4 treats the RT-robot system as seen from the real-time system.

The aim of the control system is to make the RT-robot motions closely track a desired trajectory. Thus, there is a need for a 'method' to derive a control law to calculate the control efforts.
Obviously, classical control methods (e.g. PID) can be applied to control a mechanical manipulators (like the RT-robot). This class of control methods ignores the important phenomena of nonlinearities of the manipulator and the dynamic coupling between the joint motions. The characteristic of these controllers is that they are fixed (i.e. non-adaptive) controllers. The fixed designs are the most commonly implemented schemes in present day stiff manipulators.

In practice, there are considerable uncertainties in all dynamic manipulator models. For instance, nonlinear behaviour and model parameters such as the mass, inertia, variable pay­
loads and elasticities are either impossible to know precisely or vary very unpredictable. The nonlinear behaviour and the model parameters uncertainties result in less tracking accuracy and even may lead to an unstable dynamic behaviour. A way to deal with these model parameters uncertainties is to apply adaptive-control methods in which the controller is designed to be adjustable to compensate these uncertainties automatically. Generally, adaptive control methods for a manipulator consist of a 'feedback' control part, a 'feedforward' control part and an adaptation part.

The feedforward control part involves a feedforward model, which can be a nonlinear or a linear model of a manipulator. If the feedforward model an 'inverse dynamic' of the manipulator, the adaptive control method is also called the 'Adaptive Computed Torque' method. The feedback control part, which contains a 'PD' controller, is used to compensate the occurring deviations. Together they deliver the total control effort.

The adaptation part consists of an algorithm to adjust the parameters of the feedforward model and/or the PD controller, which can be designed using the Lyapunov stability analysis (see appendix E). If a 'reference model' is used in the Lyapunov stability analysis to derive the adaptation law, the adaptive control method is also called the Model Reference Adaptive Control (MRAC) method. This method uses a 'reference model' to express the desired performance of the manipulator (in terms of the tracking error).

In this research 4 adaptive control methods are investigated and studied. They are based on the Model Reference Adaptive Control (MRAC) method and Adaptive Computed Torque method.

The adaptive control methods are:

1. The Seraji method.

2. The Craig method.

3. The Slotine & Li methods.
Slotine & Li [23] based their methods (basic and composite version) on the Adaptive Computed Torque method using a nonlinear feedforward model.

The Adaptive PD method (see appendix D) is based on the Adaptive Computed Torque method and the Model Reference Adaptive Control method using a nonlinear feedforward model.

Due to the verification of the adaptive control methods, a non-adaptive controller (PID) is also implemented.

Before implementing a control method on the RT-robot, simulations are carried out to investigate the behavior of the control method. Each method has been simulated on the same trajectory and the performance is evaluated based on the tracking accuracy and the trajectory error dynamics. The numerical simulation is performed in the software package Pro-Matlab.

As the performance needs to be experimentally verified with the actual mechanical system that contains all the realities ignored in the simulations, the implementation of the control methods was done at the RT-robot.

1.2 An overview of the report

After the general introduction in paragraph 1.1 this report presents in Chapter 2 the description of the RT-robot and the control system.
Chapter 3 presents the models of the components of the RT-robot system which are used to compute the control efforts.
The description of the control methods (the adaptive control methods and the PID) are given in Chapter 5.
Chapter 6 presents the description and the results of the simulation of the control methods. The results in Chapter 6 needs to be experimentally verified with the actual mechanical system that contains all the realities ignored in the simulations. Chapter 7 describes the experimental implementation of the control methods at the RT-robot.
Finally, conclusions and a discussion are presented in Chapter 8.
Chapter 2
The RT-robot System

2.1 Introduction

The RT-robot system includes the RT-robot and the control system. The RT-robot consists of a rotation module and a translation module with drives, while the control system includes a computer (PC-386 SX, 25 Mhz) with control algorithms, interface-cards and a measurement system.

The RT-robot is a mechanical handling device that can be manipulated under computer control. The computation of the control efforts is done by the computer, which contains the control algorithms. The control algorithms enable to control the motion of the both modules (the rotation and the translation module) of the RT-robot. The communication between the computer and the RT-robot occurs through the measurement systems and the computer interfaces. Before describing the investigated control methods (see Chapter 5), a description is given of the RT-robot system and it's facilities.

2.2 The RT-robot

The RT-robot consists of a rotation module and a translation module. The translation module designed by van Bommel [3]. The DC motor is of the disc-armature type drives the translation arm of the translation module by means of a spindle with a ballscrew nut. The rotation module is designed by Kreffer [11], contains an identical DC motor as in the translation module, which drives the turn-table by means of a four stage toothed wheel combination with divided and preloaded wheels, realized with torsion springs, to eliminate backlash.

The end-effector (or tool center point, TCP) of the RT-robot can, by movement of the two modules, track a path in a flat horizontal plane.

The RT-robot is schematically shown in figure 2.1.
Both the translation motion and the rotation motion are restricted in their movement by Hall-Switches. In case that the Hall-Switches fail, extra protection is provided by mechanical end-switches which directly disconnect the power amplifiers (the power amplifiers are shut-down by being activated the end-switches).

2.3 The control system

In 1991 a new control system was implemented by van Oosterhout [19]. It is based on a IBM compatible 386 SX type personal computer with specific interface-cards, which are plugged directly into the computers bus. In this computer, a 387 numerical co-processor is available to speed up floating point calculations. The communication between the interfaces and the personal computer is done directly by the PC-bus.

The structure of the control system is shown in figure 2.2.
The measurement systems

The rotation and translation modules are equipped with a number of incremental measurement systems. They are shown in figure 2.2.

The position of the translating arm is measured by an incremental measurement system. This system consists of an optical linear encoder, mounted under the translating arm which is moving along a fixed mounted optical head with the resolution of $2.0 \times 10^{-7}$ [m].

At the motor-axis of the translation module a rotational optical encoder is mounted, which measures the rotation of the translation-motor with the resolution of $2.51 \times 10^{-4}$ [rad].
For rotational position measurement of the turntable an optical digital incremental encoder has been mounted along the circumference of the turntable. This measurement-scale moves along a fixed mounted optical head. The resolution of this measurement-scale is $1.6 \times 10^{-6} \text{[rad]}$.

An rotational optical encoder with the resolution of $2.51 \times 10^{-4} \text{[rad]}$ is mounted at the motor-axis of the rotational module, which measures the rotation of the rotational-motor.

The position interfaces

The position interfaces consist of two Heidenhain IK 110 interface boards (see IK 110 in figure 2.1).

Each board contains:
- two 26 bits counters with a 'latch',
- an electronic interpolator to enlarge the resolution (up to 50 times).

The position interfaces can read the zero-index signal of the measurement systems. The number of counts which are given by the IK110 must be translated into the correct units, meter (for the translation module) or radian (for the rotation module).

The motor interface

The communication between the computer and the motors takes place by the Data-Translation DT 2811 interface-board (see figure 2.2). This board contains two 12 bits Digital to Analog Converters (DAC's) and Digital Input Output (DIO) facilities.

The DAC's are used to control the DC motors via an external power amplifier (P in figure 2.2). Because the power amplifiers of the motors require ± 15 volts for full speed and the DAC's only deliver an output signal of ± 5 volts an extra amplifier is used (not shown in figure 2.1) with a fixed gain of 3.

The Hall-switches interface

The Hall-switches (not shown in figure 2.2) are readed-in via the DIO lines of the DT 2811 interface-board. To receive a clear signal, Schmitt-triggers of the type 74LS13, are used between Hall-switches and the interface-board.
The force-sensor interface

The force-sensor is readed-in by the ADC (Analog to Digital Converter) of the DT 2811 interface-board. Because of the maximal range of the ADC of the DT 2811 ± 5 volt, the maximal output of the force-sensor may not be higher than ± 5 volt. This can be done by the tuning of the gain amplifier of the force-sensor. The force-sensor is not used in this research.

More details on these subjects can be found in van Oosterhout [18].

2.3 The major specifications of the RT-robot modules

The major specifications of the translation module are shown in table 1

<table>
<thead>
<tr>
<th>Specification</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum velocity</td>
<td>1 m/s</td>
</tr>
<tr>
<td>Maximum acceleration</td>
<td>5 m/s²</td>
</tr>
<tr>
<td>Maximum load</td>
<td>50 kg</td>
</tr>
<tr>
<td>Stroke</td>
<td>1 m</td>
</tr>
<tr>
<td>Position accuracy</td>
<td>10 μm</td>
</tr>
<tr>
<td>Position measuring system</td>
<td>Heidenhain LS 513</td>
</tr>
<tr>
<td>Power source</td>
<td>DC-motor BBC MC 19 P</td>
</tr>
</tbody>
</table>

Table 1. Specifications of the translation module

The major specifications of the rotation module are shown in table 2

<table>
<thead>
<tr>
<th>Specification</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum angular velocity</td>
<td>$\pi/2$ rad/s</td>
</tr>
<tr>
<td>Maximum angular acceleration</td>
<td>$\pi/2$ rad/s²</td>
</tr>
<tr>
<td>Maximum angular range</td>
<td>$\pi$ rad</td>
</tr>
<tr>
<td>Position accuracy</td>
<td>6.22 μrad</td>
</tr>
<tr>
<td>Position measuring system</td>
<td>Heidenhain LIDA 360</td>
</tr>
<tr>
<td>Power source</td>
<td>DC-motor BBC MC 19 P</td>
</tr>
</tbody>
</table>

Table 2. Specifications of the rotation module
Chapter 3
Modelling of The RT-robot System

3.1 Introduction

In order to understand and control the system, it is important to have quantitative mathematical models of the system-components. The mathematical models are used together with the control law to calculate the control efforts \( u(t) \).

For the RT-robot system, we need mathematical models of the RT-robot, the measurement systems, the motors, the power-amplifiers and the Digital to Analog Converters (DAC's).

We need two dynamic models of the RT-robot, namely a RT-robot model of 2 degrees of freedom (the 2 DOF RT-robot model) and a RT-robot model of 5 degrees of freedom (the 5 DOF RT-robot model) (see Martens [14]). The 2 DOF RT-robot model is used as a feedforward model of the control algorithm and for the simulation we need the 5 DOF RT-robot model as a simulation model to simulate the behaviour of the real RT-robot.

The two dynamic models are simplified models of the RT-robot model of 11 degrees of freedoms (see Bax [2]). The reasons of the simplification are:

- The calculation of the control algorithm has to be done on-line. Thus the feedforward model should be as simple as possible.
- The simulation-time has to be kept low. Thus the simulation model should be simple as the computation of the solution of differential equations is a time consuming computation.

The RT-robot model of 2 degrees of freedom is used either in the simulation or in the implementation, while the RT-robot of 5 degrees of freedom is only used in the simulation. In the implementation the RT-robot model of 5 degrees of freedom is replaced by the real RT-robot.

In the implementation we need the mathematical models of the measurement systems, the motors, the power-amplifiers and the Digital to Analog Converters (DAC's) to compensate the behaviour of these components.
3.2 Modelling of the RT-robot

The 2 DOF RT-robot model

The 2 DOF-robot model is modelled using 2 degrees of freedom:
- $\theta^R$: rotation of the turntable
- $\theta^T$: position of the translating arm,
with the assumption that both modules of the RT-robot can be regarded as rigid bodies. The lumped-mass representation of the 2 DOF RT-robot model is shown in figure 3.1.

![Lumped-mass representation of the 2 DOF model](image)

Figure 3.1 The lumped-mass representation of the 2 DOF model

This robot model is used as a feedforward model in order to calculate a time-dependent feedforward steering component in the control efforts $u(t)$ (see figure 3.3 and figure 3.4).

The 5 DOF RT-robot model

The 5 DOF-robot model is modelled using 5 degrees of freedom (2 DOF of the translation module and 3 DOF of the rotation module). The 5 DOF are:
- $\theta_m^R$: rotation of the rotation motor.
- $\theta_l^R$: rotation of the composed mass.
- $\theta^R$: rotation of the turntable.
- $\theta_m^T$: rotation of the translation motor.
- $\theta^T$: position of the translating arm.

This model is modelled with the assumption that some components of the modules of the RT-robot can be regarded as rigid bodies (see Martens [14]) and consists of the three lowest
natural frequencies present in the model of Bax [2]. The lumped-mass representation of the 5 DOF RT-robot model is shown in figure 3.2.

![Figure 3.2 The lumped-mass representation of the 5 DOF model](image)

This robot model is used as a simulation model in order to simulate the behaviour of the real RT-robot (see figure 3.4).

The differential equations of the 2 DOF and 5 DOF RT-robot model are given in appendix J.

The feedforward model and the simulation model as a part of a control system are shown in figure 3.3 and figure 3.4.

The feedforward time dependent control effort is calculated using the feedforward model with the desired trajectory as the input. This kind of system is called 'open-loop' system, as there is no connection between the system output and input.

A so called 'closed-loop' (feedback) system measures the occurring deviation and calculates the feedback time dependent control effort in order to compensate these errors.

The sum of the control efforts is the total control effort \( u(t) \).

Figure 3.3 shows a situation for the implementation, while figure 3.4 shows a situation for the simulation, where the real RT-robot is replaced by the simulation model.
Chapter 3 Modelling of The RT-robot System

Figure 3.3 Feedforward model as a part of a control system

Figure 3.4 Simulation model as a part of a control system
3.3 Translation of the number of counts of the measurement systems into the correct units

**Translational measurement system**

The translational measurement system has for 1 m $= 2.5 \times 10^4$ counts. Higher accuracy can be achieved by setting an interpolation factor. By setting 50 fold interpolation (so that 200 counts per grating period occur) we get:

$$1 \text{ m} = 200 \times 2.5 \times 10^4 \text{ counts} \approx 5.0 \times 10^6 \text{ counts.}$$

**Rotational measurement system**

The rotational measurement system has for $2\pi \text{ rad} = 20200$ counts. By setting 50 fold interpolation we get:

$$1 \text{ rad} = \left(200 \times 20200 \right) / \left(2\pi \right) \text{ counts} \approx 6.4298597 \times 10^5 \text{ counts.}$$

More information see appendix H.

3.4 Modelling of the motors

The general motor-equation for a DC-motor is:

$$L \frac{\partial I_a(t)}{\partial t} + R_a I_a(t) + K_e \dot{\theta}_m(t) = u_o(t) \quad (3.5)$$

A part of the total voltage $u_o(t)$ at the motor will be lost in the brushes. This part is called the brush-contact loss voltage $u_b$.

As the first term is negligibly small ($L$ in $\mu$H), the motor-equation becomes as follows:

$$R_a I_a(t) + K_e \dot{\theta}_m(t) = u_o(t) - u_b \quad (3.6)$$

The torque delivered by a DC-motor is linear in the armature current $I_a$:

$$T(t) = K_T I_a(t) \quad (3.7)$$

20
Substitution of equation (3.7) in equation (3.6) delivers:

\[
\frac{R_a}{K_T} T(t) + K_e \dot{\theta}_m(t) = u_o(t) - u_b
\]  (3.8)

By definition \(K_T\) equals \(K_e\), so the delivered torque of the DC-motor is equal to:

\[
T(t) = \frac{K_e}{R_a} [u_o(t) - u_b] - \frac{K_e^2}{R_a} \dot{\theta}_m(t)
\]  (3.9)

with:
- \(u_o(t)\): input voltage of the DC-motors.
- \(u_b\): brushes contact-loss voltage.
- \(R_a\): armature resistance.
- \(K_e\): (constant) motor-parameter.
- \(\dot{\theta}_m(t)\): rotational velocity of the motor.

The motor parameters are given by:
- \(R_a = 0.46\) [Ohm]
- \(u_b = 1.8 \text{ sign}(u_o)\) [Volts]
- \(K_e = 0.266\) [Vs/rad]

### 3.5 Modelling of the amplifiers

**The extra amplifier**

The extra amplifier is required to make the maximum output voltage range of the DAC’s on the DT2811 equal to the range of the input signal of the power-amplifiers. The range of the input signal of the power-amplifiers is from -15 Volts to +15 Volts for full motor speed, while the maximum output voltage range of the DAC’s on the DT2811 interface board reaches only from -5 Volts to +5 Volts.

Therefore, an extra amplifier is used with gain factor of 3.
\[ u_i = 3 * u_{dac} \]  \hspace{1cm} (3.10)

where:
- \( u_i \) = input voltage of power-amplifiers.
- \( u_{dac} \) = output voltage of DAC multiplied by 3.

**The power-amplifiers**

A power-amplifier is used to supply the DC-motors, as the voltage delivered by the extra amplifier is not enough for full motor speed steering.

Hence, it is important to have exact knowledge of the behaviour of the power amplifiers.

Both power-amplifiers are of the type BBC LV05. However, their characteristics are a little bit different. The characteristic of the power-amplifiers can be given by

\[ u_u = K_{ver} u_i + u_{off} \]  \hspace{1cm} (3.11)

where:
- \( u_u \) = output voltage of power-amplifier.
- \( u_i \) = input voltage of DC-motor.
- \( u_{off} \) = offset voltage.
- \( K_{ver} \) = amplification.

The offset voltage only shows effect when the input voltage is higher than 0.5 - 1 volts. The amplification gain is different for low- and high input voltages.

**Rotational amplifier:**

- high \( u_i \) : \( K_{ver} = 2.8 \) \hspace{1cm} \( u_{off} = 0.75 \, \text{sign}(u_i) \) \hspace{1cm} [Volts]
- low \( u_i \) : \( K_{ver} = 3.4 \) \hspace{1cm} \( u_{off} = 0 \) \hspace{1cm} [Volts]

**Translational amplifier:**

- high \( u_i \) : \( K_{ver} = 2.8 \) \hspace{1cm} \( u_{off} = 0.45 \, \text{sign}(u_i) \) \hspace{1cm} [Volts]
- low \( u_i \) : \( K_{ver} = 3.4 \) \hspace{1cm} \( u_{off} = 0 \) \hspace{1cm} [Volts]
3.6 Modelling of the DAC’s

The Digital to Analog Converters (DAC’s) convert a digital value (generated by the computer) to an analog voltage (required by the amplifiers).

The dataword has a length of 12 bits, conversion of the output value takes place after the high-byte of the 12 bit dataword is written in the corresponding register.

DATA = 000 H - 7FF H : delivers a negative voltage.
DATA = 800 H - FFF H : delivers a positive voltage.

The decimal dataword value and the output voltage of the DAC’s (with the range from -5 Volt to 5 Volt) are related as follows:

**Negative voltage**

DATA = 000 H (0) delivers the minimal voltage of -5 Volt

\[ K_{dac} = \frac{u_{dac}}{DATA - 2047} = \frac{-5}{-2047} \approx 2.5 \times 10^{-3} \text{ [Volt]} \]  \hspace{1cm} (3.12)

**Positive voltage**

DATA = FFF H (4095) delivers the maximal voltage of 5 Volt

\[ K_{dac} = \frac{u_{dac}}{DATA - 2048} = \frac{5}{2047} \approx 2.5 \times 10^{-3} \text{ [Volt]} \]  \hspace{1cm} (3.13)

Note:

H means **hexadecimal** value
3.7 Link from computer to motor

A control law calculates the required control effort in [Nm], to be delivered by the DC-motors. The input voltage of the DC-motor must be calculated in a way that the output-axis of the DC-motor delivers the required torque. figure 3.3 shows all components in the link from computer to the motor.

Figure 3.3 Link from computer to motor

This link is only used in the implementation (real-time).
Chapter 4
Real-time Aspects of The Computer-controlled System

4.1 Introduction

The digital computer plays an important role in our system configuration. The digital computer receives the positions from the measurement systems and calculates the error, \( E(t) \), which is then used to perform calculations in order to provide the control effort \( u(t) \). For a system controlled by a computer it is important to understand the computer-controlled system well.

4.2 The computer-controlled system

A computer-controlled system can be schematically described as in figure 4.1.

![Figure 4.1 Schematic diagram of a computer-controlled system](image-url)
The output from the RT-robot $\theta(t)$ is a continuous-time motion of both modules. The positions of the motions are measured and converted into digital form by the measurement systems (digital incremental encoders). The data’s from the measurement systems are read-in at the sampling times $t_k = kT_s$, where $k=1...n$.

The data’s from the measurement system of the rotation module and translational module are read-in at the same time by simultaneous latching the counters of both modules. This ability does not exist in the hardware of the DT2811 interface-card. The delay time is minimized by first writing the low-order bytes and then the high-order bytes of both modules in the DAC’s (DT2811). This delay time is negligible small ($3.19 \times 10^{-6}$ second) (van Oosterhout [19]).

The computer interprets the digital signal, $\{\theta(t_k)\}$, as a sequence of numbers, processes the measurements using an algorithm (control algorithm), and gives a new sequence of numbers, $\{u(t_k)\}$. This sequence is converted into an analog signal by a digital-to-analog (DAC) converter.

Note that the system runs open loop (without feedback) in the interval between the read-in of the measurement systems and the DA conversion (the interval is equal to 1 sample-time ($T_s$)). To get equal intervals the events are synchronized by the real-time clock in the computer.

The digital computer operates sequentially in time and each operation takes the same time. The DA converter must, however, produce a continuous-time signal. This is done by keeping the control signals $u(t_k)$ constant between the conversion.

The computer-controlled system contains both continuous-time and sampled or discrete-time signals. Such system is called a sampled-data system or a computer-controlled system (Astrom and Wittenmark [1]).

4.3 Real-time events

It is natural to think about control loops as concurrent activities that are running in parallel. However, the digital computer operates sequentially in time (especially the digital computer with single board processor). Thus the key problem is to map a number of parallel, concurrent activities into a sequential program. It is important to note that the activities must be completed in the required sample-time ($T_s$).
The RT-robot and the control system together form a real-time system. This implies that several concurrent activities (for the translation and rotation module) must be carried out at the same moment. Thus the concurrent activities have to be mapped into a sequential program. The sequential activities in the program form together an event and the events are synchronized by the real-time clock in the computer by interrupt signals. The period of the interrupt signals is equal to the sample-time $T_s$ (see figure 4.2). After an interrupt signal is generated, the status of the computation is tested. If the computation for the control efforts has already been finished, the next events can be started which implies the measurement systems are read-in and the next computation can be started. Otherwise all actions have to be stopped as the sample-time $T_s$ is too small.

Figure 4.2 Real-time events synchronized by interrupt signals
Chapter 5
Control Methods

5.1 Introduction

The robot trajectory control problem can generally be stated as designing the actuator torques so that the robot motions closely track a desired trajectory. Thus, there is a need for a 'method' to calculate the control effort, necessary to achieve the tracking. This 'method' will be called control method.

To compensate the presence of nonlinearities and uncertainties in robot dynamic models and payloads, the concept of adaptive control is used. Generally, an adaptive control method for a robot consists of:
- a feedforward control part,
- a feedback control part,
- a control law,
- an adaptation part
- and a method to derive the adaptation part.

The feedforward control part consist of a feedforward model and uses the desired trajectory to calculate the feedforward control effort. If the feedforward model is an 'inverse dynamic' of the robot, the adaptive control method is also called the 'Adaptive Computed Torque' method (this method is schematically shown in figure 5.1). The adaptive control methods used in this research are the Adaptive Computed Torque method.

The feedback control part usually contains 'PD' gains and uses the difference between the desired trajectory and the actual trajectory to calculate the feedback control effort.

The control law contains an algorithm (consists of the feedforward control part and the feedback control part) to calculate the control effort.

The adaptation part contains an adaptation law, which uses an algorithm to adapt (modify) the parameters of the feedforward model and/or the 'PD' gains and guarantees the stability of the control system.

The adaptation law in the adaptive control methods used in this research designed using the Lyapunov stability analysis (Lyapunov second method).
In Adaptive PD and Seraji methods a 'reference model' is also used to express the desired
performance of the robot in terms of the tracking error. The method with a 'reference model' and an 'inverse dynamic' model of the manipulator is called the **Model Reference Adaptive Control (MRAC)** based **Adaptive Computed Torque** method (this method is schematically shown in figure 5.2).

![Figure 5.1 The Adaptive Computed Torque method](image)

The simulation model or the real robot in figure 4.1 represents the input-output relationship of the control effort $u(t)$ and the actual trajectory $\theta(t)$. $u(t)$ is calculated by the feedforward and the feedback model using the informations from the desired ($\theta_d(t)$) and the actual ($\theta(t)$) trajectory. To minimize the tracking error the adaptation mechanism adapt the parameters of the feedforward model.

In figure 5.2 the adaptation mechanism adapts the parameters of the feedforward as well as the feedback model.

The adaptation mechanism calculates the parameter adaptation using the informations from the desired trajectory $\theta_d(t)$, the actual trajectory $\theta(t)$ and the reference model.

The reference model uses only the tracking error $E(t)$ at $t=0$ (for more information see appendix A)
In this research four adaptive-control methods (Seraji, Craig, Slotine & Li and Adaptive PD) and a PID control method are investigated and studied.

The PID control method is the most widely used concept in present day stiff manipulators. This method performs quite well for the manipulators without nonlinear dynamic behaviours.

In practice, there are considerable uncertainties in all dynamic manipulator models. The RT-robot contains nonlinear dynamic behaviours and model parameters uncertainties, which result in a less tracking accuracy and even may lead to an unstable dynamic behaviour. A way to deal with the model parameters uncertainties is to apply adaptive-control methods, which are linear or nonlinear controllers.

The aim of the nonlinear controllers is to compensate the nonlinearities in order to improve the control systems.

Seraji designed his model using a linearized 'inverse dynamic' model of a robot. By using the linear model this method relies on the assumptions of a small range operation and a 'slowly-time' varying for the linear model to be valid.
The Craig, Slotine & Li and Adaptive PD method belong to the nonlinear controllers, which can deal with the nonlinearities. Like the Seraji method the Adaptive PD method adapts the parameters of the feedforward and feedback model, while the Craig and Slotine & Li method adapt only the parameters of the feedforward model.

The aim of the parameter adaptation of the feedforward and feedback model is to get a better tracking accuracy.

The methods are in short presented in the next paragraphs, in order to get a better view. For a fully description of the methods see appendices (A,B,C and D).
Chapter 5 Control Methods

5.2 The dynamic model of the robot

The dynamic model of a robot plays an important role in order to derive the adaptation law of an adaptive control method. Generally, in absence of friction and other disturbance, the dynamics for a robot can be written as a matrix equation (Schwartz [25]),

$$T(t) = M(\dot{q}(t)) \ddot{q}(t) + C(\dot{q}(t), \ddot{q}(t)) \dot{q}(t) + G(\dot{q}(t))$$  \hspace{1cm} (5.1)

where:

- $T(t)$ is a nx1 vector of joint torques and forces.
- $M(\dot{q}(t))$ is the nxn mass matrix.
- $\dot{q}(t)$ is a nx1 vector of joint positions.
- $C(\dot{q}(t), \ddot{q}(t))$ represents the nxn matrix of coriolis, centrifugal and friction terms.
- $G(\dot{q}(t))$ is the nx1 vector of gravity terms.

5.3 The Seraji method

The Seraji method [22] is derived from linear multivariable theory and simply consists of feedforward from the reference trajectory, feedback from the actual trajectory, and an auxiliary input. The feedforward and feedback gains and the auxiliary input are adapted using simple equations derived from model reference adaptive control theory. The control scheme is computationally fast and does not require a priori knowledge of the physical parameters of the robot and the payload. Fully description of the Seraji method is given in appendix A.

1. Structure of the feedback controller

The feedback controller generates the control torque $T_1(t)$ by acting on the error vector $E(t) = \dot{q}_r(t) - \dot{q}(t)$; that is

$$T_1(t) = K_p(t) E(t) + K_v(t) \dot{E}(t)$$  \hspace{1cm} (5.2)

where,

- $K_p(t)$ and $K_v(t)$ are the time-dependent 'PD' gain matrices. The adaptation of these parameters are given in equation (5.6).
2. Structure of the feedforward controller

The feedforward controller generates the control torque $T_2(t)$ by acting on the desired trajectory vector $\dot{\Theta}_d(t)$, that is

$$T_2(t) = A(t)\ddot{\Theta}_d(t) + B(t)\dot{\Theta}_d(t) + C(t)\dot{\Theta}_d(t)$$

(5.3)

The $A(t)$, $B(t)$ and $C(t)$ matrices are obtained by linearizing the nonlinear model (5.1) in some nominal operating point $P$ (see appendix G) and adapted by equation (5.6).

3. Control law

The control law is formed by combining the feedback controller and the feedforward controller. The control law is given by

$$\ddot{\Theta}(t) = \dot{E}(t) + A(t)\ddot{\Theta}_d(t) + B(t)\dot{\Theta}_d(t) + C(t)\dot{\Theta}_d(t) + [K_p\dot{E}(t) + K_v\dot{E}(t)]$$

(5.4)

where,

$\dot{E}(t)$ is a time-varying auxiliary input to compensate any unmodelled nonlinearities in the robot dynamics.

4. Adaptation law

The adaptation law of the Seraji method is written as,

$$F(t) = F(0) + \delta\int_0^t \dot{E}(t)\,dt$$

$$K_p(t) = K_p(0) + \alpha\int_0^t \dot{E}(t)\dot{E}^T(t)\,dt$$

33
where \( \{\delta, \alpha, \beta, \gamma, \lambda\} \) are the positive adaptation gains specified by the designer and 

\[ R(t) = P_2(t) + P_3(t). \]

\( P_2 \) and \( P_3 \) are the elements of a symmetric positive-definite constant matrix \( P \) in equation (A.17) in appendix A.

The control law in equation (5.5) is derived with the assumption that the elements in the \( M(t), C(t), \dot{C}(t), G(t) \) matrices in equation (5.1) are unknown and 'slowly time-varying' in comparison with the adaptation scheme.

5. Adaptation method

The adaptation law in equation (5.5) is designed by the MRAC based Adaptive Computed Torque method. The Lyapunov function to derive the adaptation algorithm is not published by Seraji in Seraji[21,22] and not accessible for this research. The Lyapunov function in equation (A.22) in appendix A is used to derive the adaptation algorithm of the Seraji method.
6. Adaptive control scheme

Figure 5.3 shows the block diagram indicating the structure of the adaptive controller.

![Block diagram of adaptive robot control scheme](image)

Figure 5.3 Adaptive robot control scheme Seraji method

The matrices A, B and C are the components of the feedforward model which use the desired trajectory \( \dot{\theta}_d(t) \) to calculate the input torque. 

\( K_p \) and \( K_v \) are the 'PD' gains of the feedback model. The total input torque \( T(t) \) is delivered by the feedforward and the feedback model.

In order to get a better tracking accuracy the adaptation law adapts the parameters of the feedforward- and the feedback model.

\( E(t) \) is a time-varying auxiliary input to compensate any unmodelled nonlinearities in the robot dynamics.
5.4 The Craig method

The Craig method is derived from nonlinear control theory. It consists of feedforward from the reference trajectory and 'internal PD' state feedback from the actual trajectory. In this method only the parameters of the feedforward model are adapted. The adaptation algorithm is driven by the trajectory tracking errors. The computations of the Craig method are more time consuming than the Seraji method, because the Craig method is based on a nonlinear model. For more information see appendix B.

1. Structure of the feedback controller

The feedback controller generates the control torque $T_1(t)$ by acting on the error vector $E(t) = \theta_d(t) - \theta(t)$; that is

$$T_1(t) = \dot{M}(\theta(t))K_p(t)E(t) + \dot{M}(\theta(t))K_v(t)\dot{E}(t) \quad (5.6)$$

where:
- $K_p$ and $K_v$ are the fixed positive diagonal matrices ('PD' term).
- $\dot{M}(\theta(t))$ is the estimation of $M(\theta(t))$.

2. Structure of the feedforward controller

The feedforward controller generates the control torque $T_2(t)$ by acting on the desired trajectory vector $\theta_d(t)$, that is

$$T_2(t) = \ddot{M}(\theta(t))\ddot{\theta}_d(t) + \dot{C}(\theta(t),\dot{\theta}(t))\dot{\theta}(t) + \dot{G}(\theta(t)) \quad (5.7)$$

where:
- $\ddot{M}(\theta(t))$ is the estimation of $M(\theta(t))$,
- $\dot{C}(\theta(t),\dot{\theta}(t))$ is the estimation of $C(\theta(t),\dot{\theta}(t))$ and
- $\dot{G}(\theta(t))$ is the estimation of $G(\theta(t))$. 

36
3. Control law

The control law is formed by combining the feedback controller and the feedforward controller,

$$ T(t) = \dot{M}(\Theta(t))\ddot{\Theta}(t) + \dot{C}(\Theta(t), \dot{\Theta}(t))\dot{\Theta}(t) + C(\Theta(t)) $$

(5.8)

where,

$$ \ddot{\Theta}(t) = \ddot{\Theta}_d(t) + K_p \dot{E}(t) + K_p E(t) $$

(5.9)

4. Adaptation law

The adaptation law will compute how to change the estimated parameters as a function of a filtered servo error signal. The filtered servo error is given by

$$ E_i(t) = \dot{E}(t) + \Psi E(t) $$

(5.10)

where \( \Psi = \text{diag}(\psi_1, \psi_2, ..., \psi_n) \).

The adaptation law is,

$$ \dot{\hat{\Theta}}(t) = \Gamma W^T(\Theta(t), \dot{\Theta}(t), \ddot{\Theta}(t)) \dot{M}^{-1}(\Theta(t)) E_i(t) $$

(5.11)

where,

- \( \hat{\Theta}(t) \) is a vector containing the estimated parameters of the feedforward model.
- \( W(\Theta(t), \dot{\Theta}(t), \ddot{\Theta}(t)) \) is a signal matrix (see equation (B.6)).
- \( \Gamma \) is the adaptation gain matrix (constant positive diagonal gain matrix).

Note, equation (5.11) is only valid if the inverse of the mass matrix is a symmetric matrix (see equation (B.17) and equation (B.18) in appendix B).
5. Adaptation method

The adaptation law in equation (5.11) is designed using the Lyapunov second method. The Lyapunov function is given in equation (B.13).

6. Adaptive control scheme

In the next figure (figure 5.4) a block diagram indicates the structure of the controller that makes use of a dynamic model of the robot. The adaptation law observes servo errors and adjusts the parameters that appear in equation (5.8).

\[ M(\dot{\theta}(t)) + Q(\dot{\theta}(t), \ddot{\theta}(t)) \text{ are the components of the nonlinear feedforward model.} \]

The total input torque \( T(t) \) is delivered by the feedforward model and the feedback model. Unlike the Seraji method the feedback model in the Craig method is an internal 'PD' feedback, which means that the feedback model is included in the feedforward model.

Note, \( \dot{Q}(\dot{\theta}(t), \ddot{\theta}(t)) \) is equal to \( \dot{C}(\dot{\theta}(t), \ddot{\theta}(t)) + \dot{G}(\dot{\theta}(t)) \) in equation (5.8)
5.5 The Slotine & Li method

In this section, the methods developed by Slotine & Li are presented. These methods can be used to control a robot under certain parametric uncertainties. Slotine & Li derived first a simple, globally tracking-convergent, direct adaptive manipulator algorithm. This adaptive control algorithm has fully dynamic feedforward compensation with 'internal PD' state feedback while the robot parameters are estimated on-line.

The algorithm was further refined into a 'composite' version, whose adaptation law is driven by both tracking error in the joint motion and prediction error in the joint torques. The prediction error in the joint torques is used to estimate the robot parameters using the Least-Square Estimation method. Therefore the composite version represents a combination of a direct and an indirect approach. The description over this method is given in appendix C.

1. Structure of the feedback controller

The feedback controller generates the control torque $T_1(t)$ by acting on the error vector $E(t) = \theta_d(t) - \theta(t)$; that is

$$T_1(t) = [\dot{\hat{\theta}}(t) + \hat{C}(t)\hat{\dot{\theta}}(t) + \sqrt{K_p}] E(t) + \ddot{\hat{\theta}}(t) \dot{E}(t)$$  \hspace{1cm} (5.12)

where:
- $K_p = \Lambda^2$.
- $K_v = 2\Lambda$.
- $\Lambda$ is a constant positive diagonal gain matrix.

2. Structure of the feedforward controller

The feedforward controller generates the control torque $T_2(t)$ by acting on the desired trajectory vector $\theta_d(t)$, that is

$$T_2(t) = \dddot{\theta}(t) \ddot{\theta}(t) + \dddot{\theta}(t) \dot{\theta}(t) + C(\theta(t), \dot{\theta}(t))$$  \hspace{1cm} (5.13)
where:
- $\dot{M}(\theta(t))$ is the estimation of $M(\theta(t))$.
- $\dot{C}(\theta(t),\dot{\theta}(t))$ is the estimation of $C(\theta(t),\dot{\theta}(t))$ and $\dot{G}(\theta(t))$ is the estimation of $G(\theta(t))$.

3. Control law

The control law is formed by combining the feedback controller and the feedforward controller,

$$T(t) = \dot{M}(\theta(t))[\dot{\theta}(t) - \Lambda \hat{s}(t)] + \dot{C}(\theta(t),\dot{\theta}(t))\dot{\theta}(t) + \dot{G}(\theta(t))$$  \hspace{1cm} (5.14)

where,
- $\hat{s}(t) = \hat{E}(t) + \Lambda \hat{E}(t)$ is the sliding-surface (Slotine and Li [23]).
- $\hat{\dot{\theta}}(t) = \dot{\theta}(t) - \Lambda \dot{\theta}(t)$ is the reference trajectory.
- $\hat{E}(t) = \dot{\theta}(t) - \theta(t)$ is the tracking error.

4. Adaptation law

The 'basic' version

The adaptation law of the 'basic' version has the following form,

$$\dot{\hat{\theta}}(t) = -\Gamma^{-1} Y(\theta(t),\dot{\theta}(t),\theta(t),\dot{\theta}(t))_m^T \hat{s}(t)$$  \hspace{1cm} (5.15)

where,
- $\hat{\theta}(t)$ is a vector containing the estimated parameters of the feedforward model.
- $Y(\theta(t),\dot{\theta}(t),\theta(t),\dot{\theta}(t))_m$ is the signal matrix (see equation (C.11)).
- $\Gamma$ is the adaptation gain matrix (constant positive diagonal gain matrix).
The 'composite' version

The adaptation law of the 'composite' version is,

\[
\dot{\theta}(t) = P(t)[Y(\theta(t),\dot{\theta}(t),\ddot{\theta}(t),\dot{\theta}_r(t),\ddot{\theta}_r(t))]^T s(t) + W_f^T(\theta(t),\dot{\theta}(t)) R_i(t) E_u(t)
\]

(5.16)

where,

- \(P(t)\) is the time-varying adaptation gain matrix (see equation (C.31)).
- \(W_f^T(\theta(t),\dot{\theta}(t))\) is the transpose of the filtered signal matrix \(W(\theta(t),\dot{\theta}(t),\ddot{\theta}(t))\) by a first order filter (see equation (C.34)).
- \(R_i(t)\) is a uniformly positive definite weighting matrix indicating how much attention the adaptation law should pay to the parameter information in the prediction error.
- \(E_u(t)\) is the filtered prediction error in the joint torques (see equation (C.35)).

Note, the earlier 'basic' version simply corresponds to

\(\forall t \geq 0, R_i(t) = 0\) and \(P(t) = \Gamma^{-1}\)

5. Adaptation method

The 'basic' version

In this method the adaptation law extracts information about the parameters from the joint tracking errors. To derive the control law for this 'tracking-error-based' adaptation law, Slotine & Li choose the Lyapunov function in equation (C.1).

Note:
To derive the adaptation law, Slotine & Li have used the property of the skew-symmetry of the matrix \(Q = M(\theta(t)) - 2C(\theta(t),\dot{\theta}(t))\) (see appendix C.4).

The 'composite' version

In the 'tracking-error-based' adaptation law, parameter adaptation is driven by the motion tracking error \(s(t)\). Actually, the prediction errors on the joint torques also contain
Chapter 5 Control Methods

parameter information. The parameter estimation driven by the prediction errors is derived using the least-square estimator with exponential forgetting (see appendix C.2.3).

Equation (C.20) - (C.30) are derived to verify the gain update law of the time-varying adaptation gain matrix $P(t)$.

6. Adaptive control scheme

This section gives the adaptive control scheme of the Slotine & Li methods.

6.1 Scheme of the 'basic' version adaptive control method

![Diagram](image)

Figure 5.5 The 'basic' version Slotine & Li method scheme

Block A contains the components of the nonlinear feedforward model, which are adapted by the adaptation law in order to minimize the tracking error $E(t)$.

The components of the feedback model, which consists of the sliding control term $s(t)$ and the 'internal PD' feedback ($K_p = \Lambda^2$ and $K_v = 2\Lambda$), is shown in block B.

The feedback and the feedforward model deliver the total input torque $T(t)$.
6.2 Scheme of the prediction-error-based adaptive control method

The model contains the 'inverse dynamic' of the nonlinear robot model and calculates the joint torque \( \mathbf{T}(t) \).

Using the actual trajectory we can reconstruct the joint torque \( \hat{\mathbf{T}}(t) \). If the model is an exact description of the dynamic behaviour of the robot the joint torque \( \hat{\mathbf{T}}(t) \) is equal to \( \mathbf{T}(t) \) and the joint torque error \( \epsilon(t) \) is zero.

As the reconstruction is done using a filter the joint torque \( \mathbf{T}(t) \) must also be filtered in order to calculate the joint torque error \( E_{lf}(t) \). The filter is used to avoid the acceleration term which can usually not be measured directly.

The Least-Square Estimator estimates the parameters of the nonlinear model in order to minimize the joint torque error \( E_{lf}(t) \).
6.3 Adaptive control scheme of the 'composite' adaptation law

The 'composite' adaptation law is a combination of the 'basic' version adaptation law (tracking-error-based adaptation law) and the prediction-error-based adaptation law.

Figure 5.7 The 'composite' version Slotine & Li method scheme
5.5 The Adaptive PD method

This method is designed using the MRAC based Computed Torque Control method. Like the other methods this method consists of a feedforward control part, an internal 'PD' feedback part, a control law part and an adaptation part (adaptation law). The feedforward part is a nonlinear feedforward model. The feedback control part consists of 'PD' gains and is included in the feedforward part. We call this feedback control an internal 'PD' feedback. In this method, both the parameters of the feedforward model and the 'PD' gain matrices are adapted. The adaptation algorithm is driven by the trajectory tracking errors.

In contrast with the Seraji method which is also designed using the MRAC based Computed Torque Control method, this method is derived without the 'slowly-time varying' assumption of the elements of the $M(\theta(t))$, $C(\theta(t),\dot{\theta}(t))$ and $G(\theta(t))$ matrices in equation (5.1). This method can also be used to derive an adaptation law for a control law without an internal 'PD' feedback (like in the Seraji method). In this case we need the inverse of the mass matrix $M(\theta(t))$ to adapt the 'PD' gain matrices. This model is further not used.

1. Structure of the feedback controller

The feedback controller generates the control torque $T_1(t)$ by acting on the error vector $E(t)=\dot{\theta}_d(t)-\dot{\theta}(t)$; that is

$$T_1(t) = \dot{M}(\theta(t)) K_p(t) E(t) + \dot{M}(\theta(t)) K_d(t) \dot{E}(t)$$  \hspace{1cm} \text{(5.17)}$$

where:
- $K_p(t)$ and $K_d(t)$ are the time-dependent positive diagonal matrices ('PD' term).
- $\dot{M}(\theta(t))$ is the estimation of $M(\theta(t))$.

2. Structure of the feedforward controller

The feedforward controller generates the control torque $T_2(t)$ by acting on the desired trajectory vector $\theta_d(t)$, that is
where:

- $\hat{M}(\theta(t))$ is the estimation of $M(\theta(t))$,
- $\hat{C}(\theta(t), \dot{\theta}(t))$ is the estimation of $C(\theta(t), \dot{\theta}(t))$ and
- $\hat{G}(\theta(t))$ is the estimation of $G(\theta(t))$.

3. Control law

The control law is formed by combining the feedback controller and the feedforward controller,

$$T(t) = \hat{M}(\theta(t)) \ddot{\theta}(t) + \hat{C}(\theta(t), \dot{\theta}(t)) \dot{\theta}(t) + \hat{G}(\theta(t))$$

(5.19)

where $\ddot{\theta}(t)$ is equal to equation (5.9).

4. Adaptation law

The parameter adaptation law will compute how to change the parameters estimates as a function of the error signal.

$$K_p(t) = K_p(0) + \alpha \int_0^t E(t)^T(t)$$

$$K_q(t) = K_q(0) + \beta \int_0^t \dot{E}(t)^T(t)$$

(5.20)

$$\dot{\theta}(t) = \dot{\theta}(0) + \gamma \int_0^t W(\theta(t), \dot{\theta}(t), \ddot{\theta}(t)) [\hat{M}(\theta(t))^{-1}]^T R(t)$$

The $R(t)$ component is the same as the $R(t)$ component in the Seraji method and is comparable with the filtered error signal $\tilde{E}_q(t)$ in the Craig method.

Further we can see that the adaptation algorithm of this method is similar to a combination of the parameter adaptation of the feedforward model of the Craig method and the 'PD' adaptation of the Seraji method.
6. Adaptive control scheme

In the next figure (figure 5.8), a scheme of the Adaptive PD is shown. This scheme is comparable with the scheme of the Craig method (see figure 5.4). In contrast with the Craig method the 'PD' gain matrices in this method are also adapted by the control law.

\[
\begin{align*}
\dot{\theta}_d(t) & \quad + \quad \Sigma \\
\dot{\theta}_d(t) & \quad + \quad \Sigma \\
\dot{\theta}_d(t) & \quad - \quad \Sigma \\
\theta_d(t) & \quad + \quad \Sigma \\
\end{align*}
\]

\[
\begin{align*}
K_v & \quad \hat{M}(\theta(t)) \\
K_p & \quad \hat{Q}(\theta(t), \dot{\theta}(t)) \\
\end{align*}
\]

\[
\begin{align*}
\dot{T}(t) & \quad + \quad \Sigma \\
\dot{T}(t) & \quad + \quad \Sigma \\
\dot{T}(t) & \quad - \quad \Sigma \\
T(t) & \quad + \quad \Sigma \\
\end{align*}
\]

\[
\begin{align*}
\dot{\theta}(t) & \quad \rightarrow \\
\dot{\theta}(t) & \quad \rightarrow \\
\dot{\theta}(t) & \quad \rightarrow \\
\theta(t) & \quad \rightarrow \\
\end{align*}
\]

Figure 5.6 The Adaptive PD control scheme

\(\dot{M}(\theta(t))\) and \(\hat{Q}(\theta(t), \dot{\theta}(t))\) are the components of the nonlinear feedforward model. The total input torque \(T(t)\) is delivered by the feedforward model and the feedback model. Like the Craig method the feedback model in this method is an internal 'PD' feedback, which means that the feedback model is included in the feedforward model.
5.6 The PID method

Many control problems can be solved using a PID controller. The PID controller is also the most widely used concept in present day stiff manipulators. For these manipulators PID controllers frequently turn out to perform quite well, in spite of nonlinear system dynamics. The position of stiff manipulators can often adequately be predicted by a rigid model. In this model the number of degrees of freedom (=outputs) is equal to the number of servomotors (=inputs) and there is no coupling between the modules (arms).

The PID control law, that attempts to let the output $\theta(t)$ track a desired path $\theta_d(t)$, can now be expressed in the tracking error $E(t) = \theta_d(t) - \theta(t)$ as follows:

$$ T_{PID} = K_p E(t) + K_i \int_{t_0}^{t} E(\tau) d\tau + K_v \dot{E}(t) $$

(5.21)

$K_p$, $K_i$ and $K_v$ are the Proportional, Integral and Differential amplification matrices, which are chosen positive definite and diagonal.

The 'PID' control law scheme is shown in figure 5.7. The RT-robot consists of the rotation and the translation module with the joint torques, $T_{\text{translation}}(t)$ and $T_{\text{rotation}}(t)$, as the inputs. The movements of the modules depend on the joint torques which calculate using the PID control law.

For the RT-robot the PID control law calculates the inputs separately. In practice, the movement of the rotation module acts upon the translation module and vice versa.
In chapter 6 and chapter 7, the simulation- and the implementation results of this methods are given.
Chapter 6
Simulation Of The Control Algorithms

6.1 Introduction

Simulation is a tool for obtaining responses of nonlinear models in order to analyze and understand their dynamic behavior. If models are linear, many methods are available for calculating the time responses, for example the inverse Laplace transformation and the Z-transformation, which transforms a continuous model into a discrete one after which the time responses can be calculated quite easily.

When nonlinearities come into focus or when a model consists of both continuous and discrete parts (for example the simulation model is a continuous model and the control model is discrete), the techniques intended for linear models are no longer a suitable tool. Further, simulation is used if no analytical solution of a complex model can be obtained and if experiments with a real system are dangerous.

Simulation using the digital computer will yield the following problems, namely:

1. In the digital computer variables are represented by means of numbers. These numbers have a finite accuracy. This finite accuracy sometimes introduces problems.

2. A model is assumed to represent a parallel system. Digital computers have only one or a finite number of processors. Consequently, the parallel-described simulation model has to be calculated with a sequential-oriented computer. This conversion from parallel to sequential can introduce problems.

3. Continuous models are described with differential equations. It will turn out that these differential equations can only be solved approximately. This approximation depends to a high degree on the selection of the integration methods and the size of the integration interval.

However, simulation using the digital computer is one of the most wide-spread tools for solving technical problems. It poses almost no restriction on the description of the mathematical model. As long as a model can be described, simulation is nearly always possible.
6.2 Simulation tools

Simulation hardware

The simulation is done using a DEC station 2100 machine at the 'Reken Centrum TUE'. This machine is chosen (and not the PC) due to the performance of this machine (simulation run-time and possibility to run more than one program at the same time).

Simulation software

The simulation software is the software-package PRO-MATLAB V 3.5e. This version runs on the DEC station with a UNIX operating-system. Due to the high accuracy (double precision) of the software-package and the facility to solve differential equations (standard sub-routines), this software package can reasonable deal with the accuracy problems (see paragraph 6.1). Unlike PC-MATLAB version, PRO-MATLAB has no limit on the elements number. The individual variables on PC-MATLAB are limited to 8188 elements each.

Flow-diagram simulation programs

The simulation program can be written in several parts. First the initialization of the parameters used in the program. After that, the positions of the desired trajectory are calculated for each sampling time. This results in a number of points. Due to the real-time behaviour of the system, the control efforts must be calculated for each point based on the response of the simulation model. To do this a loop is required. In the loop the adaptation algorithm, the control law and the simulation model are computed. Based on the response of the simulation model, the tracking-errors can be calculated. This information of the tracking-errors is required for the computation of the next point. The loop is repeated until the last point is achieved.

The flow-diagram of the simulation program is shown in figure 6.1.

After each loop the progress of the simulation is tested. If the end of the trajectory (the last point) is reached the simulation is stopped, otherwise the simulation is continued.

In each loop the adaptation law is calculated and the parameters of the control model are adapted. Using the control law the control efforts for the next point are calculated.

The solutions of the computation of the differential equations in the simulation model represent the realized trajectories. The tracking-error is the numerical difference between
the desired trajectory and the realized trajectory.

Figure 6.1 Flow-diagram of the simulation program
6.3 The desired trajectory

A skew-sine function is chosen to generate the desired trajectory for both modules. The desired trajectories for the rotation and the translation module are shown in figure 6.2 and figure 6.3 respectively. These desired trajectories are chosen as they have no jump in the position, velocity and acceleration of the motion, which results in a smooth motion of the RT-robot. For the implementation this property is important as the jump in the position, the velocity of the acceleration may lead to instability of the dynamic behaviour of the RT-robot. This dynamic behaviour must be avoided due to the mechanical parts of the robot.

![Desired trajectory rotation](image)

Figure 6.2 Desired trajectory rotation.
Chapter 6 Simulation of The Control Algorithms

Figure 6.3 Desired trajectory translation.
6.4 Simulation results

The simulations are done with a sampling time $T_s$ of 5 [ms.]. Several simulations are carried out to get optimal adaptation of the gain matrices. It is important to note that the results depend on the sampling time, the tuning of the adaptation gain matrices, the initialization of the parameters of the system and the desired trajectory.

The sampling time chosen in the simulations, due the lowest sampling time in the implementation, is 5 [ms.]. The sampling time in the implementation depends on the computing power of the computer (PC 386 SX, 25 Mhz). The accuracy of the results can be improved by choosing the lowest sampling time as the robot runs open-loop in the interval between the sampling moments (see Chapter 4).

The adaptation gain matrices are important for the parameter adaptation. With the high adaptation gains, performance is maximized but it can lead to instability of the dynamic behaviour of the RT-robot if the adaptation gains are too high.

For Craig, Slotine & Li and Adaptive PD the initialization of the feedforward model parameters is equal to the parameters in the 2 DOF RT-robot model (see appendix J). The initialization of the parameters in the Seraji method is chosen in the operating point at $t = 0.75$ second as this operating point is situated in the middle between the minimum and the maximum position in the trajectory. By this choice the influence of the adaptation gain matrices is less important for the stability of the system than in the case that the operating point is chosen at $t = 0$ second. The linearization of the 2 DOF RT-robot model is presented in appendix G.

The desired trajectory plays an important role due to the stability of the system. A smooth motion of a manipulator is achieved when the trajectory has the property that the positions, the velocities and the accelerations are continuous on the desired trajectory (see reference [12]). Therefore the sine-skew function is used.
Further the motion is constrained by the maximum velocity and acceleration.

As the methods depend on different control parameters it is difficult to make a fair comparison of the tracking-error accuracy of the nonlinear controllers with either the PID- or Seraji method (as there is no fixed method to determine the parameters values and it requires a trial and error period for each method to get a stable system with a 'minimum' tracking-error). The PID- and the Seraji method were already implemented by Van
Oosterhout [19]. As his simulation- and real-time programs are not available (have been lost) these methods are implemented again in this research.

A comparison can well be made among the nonlinear controllers as they have the same concept and the same feedforward- and feedback model. The differences of the control law and the adaptation law of the nonlinear controllers can be used to judge the performance of the methods.

The initial values of the parameters are given in appendix K.
6.4.1 Rotation module

In figure 6.4 the simulation results of the rotation module are given. We can see from the figure that the nonadaptive controller PID gives the worst tracking-error accuracy than the adaptive controllers. Among the adaptive controllers the Slotine and Li methods and the Craig method give a comparable result and better than the Seraji method. These methods are nonlinear controllers based on a nonlinear model, while the Seraji method is based on a linearized model of the nonlinear model.

![Simulation results rotation](image)

Figure 6.4 Simulation results rotation module

These results are conform the expected results. The adaptive controller gives a better performance that the nonadaptive. Among the adaptive controllers the nonlinear adaptive controllers are better than the controllers with a linearized control model.
In figure 6.4 the Craig method gives the best performance. In the next figure the Craig method is compared with the Adaptive PD method.

By the adaptation of the 'PD' parameters the tracking-error accuracy can be slightly improved.
Both methods show the same tracking-error until $t=0.3$ second but there after by the adaptation of the PD parameters the tracking-error of the Adaptive PD method can be slightly improved.
6.4.2 Translation module

As can be seen from figure 6.6 the nonlinear controllers show comparable results. For the translation module the PID performs a comparable result with the nonlinear controllers and the Seraji method shows at the first half of the trajectory a less tracking-error accuracy than the PID. Due to the adaptation of the parameters of the feedforward model the Seraji method performs there after a better tracking-error accuracy.

![Simulation results translation module](image)

Figure 6.6 Simulation results translation module

The improvement of the nonlinear controllers for the translation module is not remarkable as for the rotation module. It can be caused by the coupling between the rotation- and the translation module in the feedforward model which makes the tracking-errors of the modules depend on each other (a decreasing of the tracking-error of the rotation module results in an increasing of the tracking-error of the translation module and vice versa). This coupling is depended on the choice of the control parameters. For the nonlinear controllers they are so chosen to get the same tracking-error magnitudes of both modules.
By the adaptation of the 'PD' parameters the tracking-error accuracy can be improved. Both methods show the same tracking-error until \( t=0.3 \) second but there after by the adaptation of the PD parameters the tracking-error of the Adaptive PD method can be clearly improved.
6.5 Conclusions

From the experience during the simulations and the results of the various methods we can obtain the following conclusions:

There is no fixed procedure to determine the parameter adjustment of the control methods (see also paragraph 6.1). It requires a trial and error period for each method to get a stable system with a 'minimum' tracking-error. As the methods depend on different control parameters it is difficult to make a fair comparison of the tracking-error accuracy of the nonlinear controllers with either the PID- or the Seraji method.

The tracking-error is dependent on the sampling-time and the desired trajectory. To avoid the effects of these elements the sampling-time is held at 5 [ms.], and the same desired trajectory is used.

The performance of the control methods are dependent on the following parameters:

- **PID**:
  1. The parameters of the feedback model (P, I and D gains)

- **Seraji**:
  1. The initial value of the parameters of the feedback model (P and D gains).
  2. The initial value of the parameters of the feedforward model (linearized model of the RT-robot model of 2 DOF).
  3. The Adaptation gains for the feedforward and feedback model.

- **Craig and Slotine & Li**:
  1. The parameters of the 'internal' feedback model (P and D gains).
  2. The initial value of the parameters of the feedforward model (nonlinear model of the RT-robot model 2 DOF).
  3. The Adaptation gains for the feedforward model.

- **Adaptive PD**:
  1. The initial value of the parameters of the 'internal' feedback model (P and D gains).
  2. The initial value of the parameters of the feedforward model (nonlinear model of the RT-robot model 2 DOF).
  3. The Adaptation gains for the feedforward and feedback model.
High feedback gains can minimize the tracking-errors but it can also lead to instability of the system. The values of the 'internal' feedback gains are dependent on the mass matrix of the control model. In general it should be larger than the values of the 'external' feedback gains.

The results can be made quite different depending on how the adaptation gains are chosen. With the high adaptation gains, the tracking-error can be minimized but it can lead to instability of the system if the adaptation gains are too high.

Due to the linearization of the feedforward model of the Seraji method, the dynamic behaviour of the feedforward model is very dependent on the rate of the adaptation (the value of the adaptation gains). The tracking-error performance, especially in a system with feedback (PD) adaptation, may be degraded drastically when the linearization operating point, which incrementally is adapted, moves away from the robot. For this reason the Adaptive PD method is derived based on a nonlinear feedforward model.

The tracking-error of the nonlinear controllers (Slotine & Li, Craig and Adaptive PD method) are comparable as these methods belong to the same control method. The Adaptive PD method gives the best tracking-error accuracy due to the adaptation of the feedback gains.

Due to the coupling between the rotation and translation module in the feedforward model the tracking-errors of the modules are dependent on each other (decreasing the tracking-error of the rotation module increases the tracking-error of the translation module and vice versa).

The PID and Seraji method are less accurate than the nonlinear controllers. The Seraji method gives a better tracking-error accuracy for the rotation module than the PID, but for the translation module the PID is worse.
Chapter 7
Implementation Of The Control Algorithms

7.1 Introduction

After the simulations we are ready to implement the control algorithms at the real RT-robot as the performance needs to be experimentally verified with the actual mechanical system. The actual mechanical system contains all the realities ignored in the simulations.

The feedforward model uses in the implementation is the same as in the simulation (see Chapter 6). In the implementation the positions of the rotation- and the translation module are measured by the measurement systems and the tracking-errors of the modules are the difference between the desired trajectories and the realized trajectories which are measured by the sensors with the accuracy of ±10 µm for the translation module and ±6.22 µrad for the rotation module. The measurement system accuracies are lower than the realized tracking-errors (about 5% of the tracking-error translation module and 0.3% of the tracking-error rotation module).

In the implementation the control of the RT-robot is synchronized each sampling time ($T_i$). That means that each $T_i$ seconds the positions are read-in and the motor voltages are sent-out at the same moment. In order to ensure a constant timing during the replay of the desired trajectory an interrupt-timer is used. This interrupt activates the corresponding 'interrupt-service routine'. After an interrupt the routine issues a command to the hardware to latch the positions and voltage values and then calls all the process subroutines (e.g. read-in the latched positions, sent-out the voltages values, calculate the required control efforts, convert the values to voltage and wait until the next interrupt) in turn.

During implementation the sampling time is held at 5 [ms.] for all algorithms except for the Slotine & Li method ('Composite' version) where the sampling time is held at 7 [[ms.]. Also the parameter adaptation of the 'composite' version is done each eight sampling times 56 [ms.] due to the restriction of the computing power.

7.2 Description of the software

The software for the real-time control is written in the software package TURBO PASCAL 6.0. The subroutines for the data interfaces and the interrupt are an adapted version from the IK110 demo software of the firma Heidenhain.
Chapter 7 Implementation Of The Control Algorithms

The global flow-diagram of the software is given in figure 7.1. After the initialization of the parameters the desired trajectory is read-in from the disk (the positions of the desired trajectory are calculated off-line). Before the replay the interrupt must be installed. The interrupt takes care of the synchronization during the replay. If the interrupt occurs, the status of the variable sample ready is tested. In case that 'sample ready' is true the next computation can be done, otherwise the control efforts are set to zero and the replay is terminated. The flow diagram of the interrupt is shown in figure 7.2.

![Flow-diagram of the software](image)

Figure 7.1 Flow-diagram of the software

If 'sample ready' is true the process subroutines are called. The positions from the measurement systems are read-in and the control efforts are written-out to the DAC's. Using the information from the desired- and the realized trajectories the adaptation law
calculates the 'new' parameters of the control model, which then are used to calculate the control efforts of the RT-robot. This computations are done on-line and are synchronized by the interrupt signals. The flow diagram of the interrupt is shown in figure 7.2.

'Sample ready' is a variable which gives the status of the computations. This status is tested each time where the interrupt signal occurs. As shown in figure 7.2 the next computations can be done if 'sample ready' is true otherwise the replay must be stopped because the computations take more time than is allowed by the sample time. After the testing and before the next computations 'sample ready' gets the status false. If the computations have been finished 'sample ready' achieves the status true again and the program has to wait until the next interrupt signal. Thus 'sample ready' is being false if the interrupt signal occurs during the computations.
7.3 Implementation results

The desired trajectory used in the implementation is the same as in the simulation (see paragraph 6.3).

As in the simulation, the results of the implementation also depend on the sampling time, the tuning of the adaptation gain matrices, the initialization of the parameters and the desired trajectory (see also paragraph 6.4). The initial values of the parameters are given in appendix K.

The results are presented in three sub-paragraphs:

1. Results of the existing methods (Seraji, Craig, 'basic' version of Slotine & Li and PID). These are implemented with a sampling-time of 5 [ms.].

2. Results of the Adaptive PD and Craig method. In this sub-paragraphs the Adaptive PD is compared with the Craig method which gives the best results.

3. Results of the comparison of the 'basic' and 'composite' version of Slotine & Li method with a sampling-time of 7 [ms.].

In paragraph 7.4 the conclusions and an explanation of the results are given.
Chapter 7 Implementation Of The Control Algorithms

7.3.1 Implementation results of existing methods with a sampling-time of 5 [ms].

The tracking-errors of existing methods with a sampling-time of 5 [ms.] are shown in figure 7.3 for the rotation module and figure 7.4 for the translation module.

![Figure 7.3 Rotation results of existing methods](image)

As in the simulation results the nonlinear controllers perform a comparable tracking-errors. The tracking-error behaviours of the PID method and the Seraji method are closer with each other than with the nonlinear controllers as the PD parts of the PID method comparable with the 'external' PD feedback of the Seraji method while the nonlinear controllers have an 'internal' PD feedback.
Figure 7.4 Translation results of existing methods

After the first half of the replay the nonlinear controllers show a better tracking-error accuracy than the PID- and the Seraji method, especially for the translation module. The tracking-errors of the translation module are about a factor four lower than those of the rotation module as the maximum usable feedback gains for the rotation module are clearly lower than for the translation module. The rotation module is relative elastic and therefore starts to show unstable behaviour at lower feedback gains.
7.3.2 Comparison of Adaptive PD and Craig method with a sampling-time of 5 [ms].

Figure 7.4 and figure 7.6 show the results of the Craig method and Adaptive PD method. These results are the means of the tracking-errors of three times replays. The results of both methods in the implementation are comparable while in the simulation the results of the Adaptive PD are better. Increasing the adaptation gains of the PD parameters leads to unstable behaviour of the RT-robot.

![Graph showing real-time results rotation](image)

Figure 7.5 Rotation results of Adaptive PD and Craig method (5 [ms.])
Figure 7.6 Translation results of Adaptive PD and Craig method (5 [ms.])

The Adaptive PD is implemented 6 months later after the others methods were implemented. It is important to note that the tracking-errors of the Craig method is now different from the previous replay and the replay is not always stable. To get a stable replay another initial value of the $K_v$ parameter (110) of the rotation module is chosen. The $K_v$ parameter of the previous replay was 120 (see appendix K.2).
7.3.3 Comparison of the 'basic' and 'composite' version of Slotine & Li method with a sampling-time of 7 [ms.]

Due to the restriction of the computing power of the computer the sampling-time of the 'composite' version of the Slotine & Li method is held at 7 [ms.] and the parameter adaptation is done each eight sampling times (56 [ms.]).

To make a reliable comparison between the 'basic' version and the 'composite' version of the Slotine & Li methods, the 'basic' version is also implemented with the same condition.

Figure 7.7 Rotation results of 'basic' and 'composite' version (7 [ms.])
Figure 7.8 Translation results of 'basic' and 'composite' version (7 [ms.])

The tracking-errors of both versions are comparable as shown in figure 7.7 and 7.8. The influence of the sampling-time can be seen from the tracking-errors of the 'basic' version with the sampling-time of 5 [ms.] which are shown in figure 7.3 and 7.4 and with the sampling-time of 7 [ms.] in figure 7.7 and 7.8.
Chapter 7 Implementation Of The Control Algorithms

7.4 Conclusions

As in the simulations, the implementation of the methods requires a trial and error period to obtain the 'optimal' system-parameters with a 'minimum' tracking-error. The following conclusions and explanations are based on the experience during the implementation and the comparison of it with the simulations.

The results of the implementation of the control algorithms are worse than the simulations. The parameter values obtained from the simulations must be readjusted to get a 'minimum' tracking-error.

This can be explained as follows:

- In the simulation we used a simulation model which contains only the five lowest natural frequencies of the RT-robot. This model is derived from the reduction of the robot model with 11 Degrees Of Freedom.
- The motors, amplifiers and other elements from the sensors are not included in the simulations.
- In the implementation we deal with the real RT-robot system with all it’s realities ignored in the simulations.

As in the simulation, the tracking-errors are dependent on the sampling-time, the desired trajectory and the control parameters. The sampling time is held at 5 [ms.] (except for the 'composite' version of the Slotine & Li, where the sampling time is 7 [ms.] due to the computing power) and the same desired trajectory as in the simulations is used.

The control parameters which have influence on the tracking-errors are the same as in the simulations:

- PID:
  1. The parameters of the feedback model (P, I and D gains)

- Seraji:
  1. The initial value of the parameters of the feedback model (P and D gains).
  2. The initial value of the parameters of the feedforward model (linearized model of the RT-robot model of 2 DOF).
  3. The Adaptation gains for the feedforward and feedback model.
Chapter 7 Implementation Of The Control Algorithms

- Craig and Slotine & Li:
  1. The parameters of the 'internal' feedback model (P and D gains).
  2. The initial value of the parameters of the feedforward model (non-linear model of the RT-robot model 2 DOF).
  3. The Adaptation gains for the feedforward model.

- Adaptive PD:
  1. The initial value of the parameters of the 'internal' feedback model (P and D gains).
  2. The initial value of the parameters of the feedforward model (non-linear model of the RT-robot model 2 DOF).
  3. The Adaptation gains for the feedforward and feedback model.

High feedback gains can minimize the tracking-errors but it can also lead to instability of the system. The values of the 'internal' feedback gains are dependent on the mass matrix of the control model. In general it should be larger than the values of the 'external' feedback gains.

The results can be made quite different depending on how the adaptation gains are chosen. With the high adaptation gains, the tracking-error can be minimized but it can lead to instability of the system if the adaptation gains are too high.

Due to the linearization of the feedforward model of the Seraji method, the dynamic behaviour of the feedforward model is very dependent on the rate of the adaptation (the value of the adaptation gains). The tracking-error performance, especially in a system with feedback (PD) adaptation, may be degraded drastically when the linearization operating point, which incrementally is adapted, moves away from the robot.

The tracking-error of the nonlinear controllers (Slotine & Li, Craig and Adaptive PD method) are comparable as these methods belong to the same control method. In the implementation the nonlinear controllers do not show the improvement of the tracking-error accuracy, in comparison with the other controllers, as in the simulations.

In the simulations the simulation-model (5 DOF) and the feedforward model (2 DOF) are derived from the robot model with 11 DOF (see Chapter 2). Therefore they match quite close with each other. However in the implementation the simulation-model is replaced by the real RT-robot which ends the possible matching of both models. Thus only an accurate feedforward model can lead to a high-performance of the Adaptive
Computed Torque method, especially for the Adaptive PD method where the time-varying PD controller is also used.

The Seraji method gives a better tracking-error accuracy for the rotation module than the PID, but for the translation module the PID is worse. Due to the coupling between the rotation and translation module in the feedforward model the tracking-errors of the modules are dependent on each other (decreasing of the tracking-error of the rotation module gives increasing of the tracking-error of the translation module and vice versa).

The tracking accuracies of the control algorithms at the beginning of the trajectory is less significant with each other but thereafter the nonlinear controllers show better tracking accuracies.

As can be seen from the comparison of the 'basic' version of Slotine & Li method and other implementations (not discussed) the tracking-error accuracy increases while the sampling-time decreases.

It is interesting to investigate the repeatability of the RT-robot in the future research.
Chapter 8
Conclusions and Discussion

8.1 Conclusions

Computer programs to investigate the control algorithms for the RT-robot are realized. The computer programs for simulation are written in PRO-MATLAB V.3.5e and can be used to analyze and understand the behaviour of the RT-robot control system in a safe way.

Due to the verification of the performance with the actual mechanical construction, the computer programs in TURBO PASCAL 6.0 written by van Oosterhout [19] have been modified and corrected. A research for other control algorithms can be carried out due to change and exchange of the control method part in the computer programs.

Four adaptive control algorithms which have been proposed in the literature are investigated. The Seraji method based on the Adaptive Computed Torque method and the Model Reference Adaptive Control. This method uses a linearized feedforward model and time-varying PD controllers and derived under the assumption of slow robot motion. Due to the linearization the performance may be degraded drastically when the linearization operating point moves away from the real robot.

The Craig method and the 'basic' version of the Slotine & Li method are based on the Adaptive Computed Torque method. This method uses a nonlinear feedforward model and fixed PD controllers.

The 'composite' version of the Slotine & Li method is a combination of the 'basic' version and the Least-square Estimation method. Due to the Least-square Estimation part the computation of this method is very time consuming.

An adaptive control algorithm based on the Adaptive Computed Torque method and the Model Reference Adaptive Control (as the Seraji method) has been studied and derived. The differences between this Adaptive PD method and the Seraji method are:

- This method is derived without assumption of a slowly motion of the robot.
The feedforward model of this method is a nonlinear model as the linearization may be degraded the performance when the linearization operating point moves away from the real robot.

Due to the verification of the adaptive control methods, a non-adaptive controller (i.e PID) is also implemented.

The simulation results show that the non-adaptive (PID) control algorithm is less accurate than the adaptive control algorithms.

The tracking-errors of the nonlinear control algorithms (Craig, Slotine & Li, Adaptive PD) show clearly improvement if compared with the linearized control algorithm (Seraji). The Adaptive PD method has the best tracking-error accuracy if compared with the other adaptive control algorithms.

After the simulations the control algorithms are implemented at the real RT-robot. In the implementation the tracking-error accuracies of the control algorithms at the beginning of the trajectory are less significant with each other but thereafter the nonlinear controllers show better tracking-error accuracies.

The tracking-errors accuracy of the control algorithms in the simulation are better than in the implementation. The difference is caused by all the realities ignored in the simulations. For example, the influences of the DAC's, the sensors, the DC-motors and other degrees of freedom are not included in the simulation.

Only an accurate feedforward model can lead to a high-performance of the Adaptive Computed Torque method, especially for the Adaptive PD method where the time-varying PD controllers are used.

It is also important to note that the results are dependent on the sampling time, the tuning of the adaptation gains, the initialization of the parameters of the system and the desired trajectory (see Chapter 6 and 7).
8.2 Discussion

The improvement of the tracking-errors accuracy of the RT-robot could be achieved by the improvement of the hardware components of the RT-robot system and the control algorithms.

Some important aspects to improve the hardware components are:

- A faster microprocessor (e.g. 486 - 66) or an additional co-processor board. The control algorithms are derived as a continuous-time system. With a faster microprocessor the sample time could clearly decrease, where the discretization effects could be reduced and matched more with the Shannon's sampling theorem.
- Addition of a velocity or acceleration measurement system. Direct measurement of the velocity or acceleration is better than the numerical derivation from the position. As most of the control algorithms include a position- and velocity error, the velocity measurement system has a higher priority than the acceleration measurement system.

Some aspects of the control algorithms:

- Accurate feedforward model. A feedforward model with a higher degree of freedom gives more realistic description of the real RT-robot, but it is also more complex and needs more computation power.
- As there is no fix method (procedure) to determine the 'optimal' parameters adjustment of the control methods, a research on it could be useful as the adjustment of the parameters is very time consuming.
- To deal with the stability of the system, one can try to limit the range where the parameters can adapt. By this high adaptation gains can be used without the parameters in the control model are out of range which can lead to instability.
- To deal with the flexibility of the rotation module of the RT-robot it is interesting to investigate a method that can deal with the flexibility in a manipulator.
BIBLIOGRAPHY


*John Wiley & Sons Ltd.*, (1974).


*Lecture Notes 113137, Department of Mechanical Engineering*, Twente University (1990).

Lecture Notes 5733, Department of Electrical Engineering, Eindhoven University of Technology (1990).


Lecture Notes 4603, Department of Mechanical Engineering, Eindhoven University of Technology (1991).


[22] Seraji, H., Linear Multivariable Control of Robot Manipulators.  


[26] Voorkamp, R.J., Studie naar de Adaptieve Trajectory Sturing van een Rotatie-Translatie Robot.  


*Data Translation, Inc.*, (1990).


*Dr. Johannes Heidenhain GmbH*, Traunreut, West Germany (1990).
List of symbols

\( b^R \) = damping of the composed mass of the rotation module
\( b_1^R \) = damping of the composed mass 1 of the rotation module
\( b_2^R \) = damping of the composed mass 2 of the rotation module
\( b_3^R \) = damping of the composed mass 3 of the rotation module
\( b_{12}^R \) = material damping between the composed mass 1 and 2 of the rotation module
\( b_{23}^R \) = material damping between the composed mass 2 and 3 of the rotation module
\( b^T \) = damping of the composed mass of the translation module
\( b_1^T \) = damping of the composed mass 1 of the translation module
\( b_2^T \) = damping of the composed mass 2 of the translation module
\( b_{12}^T \) = material damping between the composed mass 1 and 2 of the translation module
\( i^R \) = transmission ratio of the rotation module
\( i_{12}^R \) = transmission ratio between the composed mass 1 and 2 of the rotation module
\( i_{23}^R \) = transmission ratio between the composed mass 2 and 3 of the rotation module
\( i^T \) = transmission ratio of the translation module
\( i_{12}^T \) = transmission ratio between the composed mass 1 and 2 of the translation module
\( k_{12}^R \) = torsional stiffness between the composed mass 1 and 2 of the rotation module
\( k_{23}^R \) = torsional stiffness between the composed mass 2 and 3 of the rotation module
\( k_{12}^T \) = torsional stiffness between the composed mass 1 and 2 of the translation module
\( m^T \) = composed mass of the translation module
\( p(t) \) = parameter vector feedforward model
\( \dot{p}(t) \) = estimation parameters of \( p(t) \)
\( s(t) \) = sliding-surface term
\( u_b \) = brushes contact-loss voltage
\( u_{\text{dac}} \) = output voltage of DAC
\( u_i \) = input voltage of power-amplifiers
\( u_{\text{off}} \) = offset voltage.
\( u(t) \) = input voltage of the DC-motors

\( A(t) \) = linearized model matrix Seraji (mass part)
\( B(t) \) = linearized model matrix Seraji (damping part)
\( C(t) \) = linearized model matrix Seraji (direct steering part)
\( C(\theta(t),\dot{\theta}(t)) \) = matrix of coriolis, centrifugal and friction terms.
\( \hat{C}(\theta(t),\dot{\theta}(t)) \) = estimation of \( C(\theta(t),\dot{\theta}(t)) \)
\( E(t) \) = tracking-error
\( E_{\text{sf}}(t) \) = filtered servo error
\( E_{\text{fp}}(t) \) = filtered prediction error in the joint torques
\( F(t) \) = time-varying auxiliary input
\( G(\theta(t)) \) = vector of gravity terms.
\( \hat{G}(\theta(t)) \) = estimation of \( G(\theta(t)) \)
\( J_{1R}^{R} \) = moment inertia of the composed mass 1 of the rotation module
\( J_{2R}^{R} \) = moment inertia of the composed mass 2 of the rotation module
\( J_{3R}^{R} \) = moment inertia of the composed mass 3 of the rotation module
\( J_{1T}^{T} \) = moment inertia of the composed mass 1 of the translation module
\( K_v \) = differential amplification matrix
\( K_v(t) \) = time-dependent amplification matrix
\( K_e \) = (constant) motor-parameter amplification.
\( K_i \) = integral amplification matrix
\( K_p \) = proportional amplification matrix
\( K_p(t) \) = time-dependent amplification matrix
\( M(\theta(t)) \) = mass matrix
\( \hat{M}(\theta(t)) \) = estimation of \( M(\theta(t)) \)
\( P \) = nominal operating point
\( P(t) \) = time-varying adaptation gain matrix
\( R(t) \) = filtered error signal
\( R_a \) = armature resistance
\( R_s(t) \) = weighting matrix 'composite' version Slotine & Li method
\( T^R \) = input torque of the rotation module
\( T^T \) = input torque of the translation module
\( \mathbf{T}(t) \) = vector of joint torques and forces.
\( T_1(t) \) = feedback controller torque
\( T_2(t) \) = vector feedforward controller torque
\( \mathbf{W}(t) \) = signal matrix
\( \mathbf{W}_r(t) \) = filtered signal matrix \( \mathbf{W}(t) \)
\( \mathbf{Y}(t) \) = signal matrix Slotine & Li method
\( \alpha \) = adaptation gain for \( K_p \) and \( K_r \)
\( \beta \) = adaptation gain for \( C(t) \)
\( \Gamma \) = adaptation gain matrix
\( \gamma \) = adaptation gain for \( B(t) \)
\( \delta \) = adaptation gain for auxiliary input
\( \theta(t) \) = joint positions.
\( \theta^R \) = rotation of the composed mass
\( \dot{\theta} \) = desired trajectory
\( \theta_m \) = rotation of the motor
\( \dot{\theta}_m(t) \) = rotational velocity of the motor
\( \dot{\theta}_r(t) \) = reference trajectory
\( \theta^R \) = rotation of the turntable
\( \theta^T \) = position of the translating arm
\( \Lambda \) = constant positive diagonal gain matrix.
\( \lambda \) = adaptation gain for \( A(t) \)