MASTER

Real time trajectory control of a RT-robot on a single processor computer platform

van Oosterhout, J.J.A.H.

Award date:
1992

Real time trajectory control of a RT-robot on a single processor computer platform

M. Sc. thesis WPA-1289
J.J.A.H. van Oosterhout

Eindhoven, May 1992
Eindhoven University of Technology,
Department of Mechanical Engineering,
Division of Production Engineering and Automation

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

EINDSTUDIEOPDRACHT

TECHNISCHE UNIVERSITEIT EINDHOVEN 27 september 1991
Faculteit Werktuigbouwkunde
Vakgroep WPA

Student	J.J. van Oosterhout
Hoogleraar	Prof.dr.ir.P.H.J.Schellekens
Begeleider	Ir.P.C.Mulders
Start	juni 1991
Einde	maart 1992
Titel	Implementatie van een nieuw besturingssysteem op de RT robot.

Toelichting

Voor het implementeren en in de praktijk testen van specifieke robot regelalgorithmen is een opstelling beschikbaar, bestaande uit een industriële rotatie-translatie robot met besturingssysteem. Het besturingssysteem -tot voor kort een systeem met een hiërarchische opbouw bestaande uit 4 Intel single board computers samen met een AT 80386 personal computer - is recent gewijzigd en is nu gebaseerd op een systeem met één centrale processor. Dit is gerealiseerd door de aanschaf van enkele specifieke I/O kaarten in de AT 80386 personal computer en de montage van een extra encoder (rotatie opnemer) op de motoras van de liniaire arm.

Opdracht

Bestudeer de huidige opzet van de rotatie-translatie robot met besturingssysteem.
Modificeer de simulatie- en regelmodellen van de robot.

- Maak een nieuwe opzet van de besturingssoftware mede met betrekking tot synchronisatie en interruptverwerking.
- Implementeer een niet-adaptief en een adaptief regelalgorithme in de besturing om de rekencapaciteit van het single processor systeem te testen.
- Vergelijk de simulatie- en praktijkresultaten van de robot bij het doorlopen van een trajectorie.

naam hoogleraar:Prof.dr.ir.P.H.J.Schellekens naam begeleider: Ir.P.C.Mulders

SUMMARY

This report describes the results of a Master of Science research project on the control of a multi Degree Of Freedom robot on a single processor computer system. In order to control such a robot on a single micro-processor, research has been carried out to ensure 'real-time' behaviour (a.o. that all axes have to be controlled simultaneously) on this kind of system.
It turned out that, with the help of specific hardware and by a special implementation architecture of the control software, it is possible to achieve this 'real-time' behaviour. This implies that it is necessary for the hardware to possess simultaneous 'latching' capabilities, while all parts of the control software have to be implemented as 'process-subroutines' which return after they have finished their task.

To test the new control platform, several control laws will be implemented on this platform and used to control a Rotation-Translation (RT)-robot.
Therefore three non-adaptive (PD, PID and Feedforward) and one adaptive (MRAC) control law will be tested first in a numerical simulation, in which a dynamic model of the real RT-robot, called the '**simulation**' model, is used to give an insight in the behaviour of the different control laws.
Some control laws require the use of a (simplified) model of the RT-robot, which is called '**feedforward control**' model, in order to calculate a time dependent steering component. Both models must be different in structure and/or parameter values in order to get meaningful results.

The simulations show that a PID control law offers good results, while the 'tuning' of the adaptive control law is difficult.

After obtaining satisfactory results with the simulation, all four control laws are implemented on the real RT-robot. In general, the results of the simulations and the real RT-robot correspond quite well. The tracking errors, for simulation and implementation are in the same order of magnitude. Even, the maximum applicable 'feedback gains' correspond quite well.
Again the PID control law offers good results, while the adaptive control law is difficult to 'tune'.
This implies that the adaptive control law in its present form is less suitable to control the real RT-robot. Additional research will be necessary in order to find a way to 'tune' the adaptation algorithm and to find the 'performance limits' of the adaptive control law.

SAMENVATTING

Dit rapport bevat de resultaten van een afstudeeronderzoek naar de besturing van een robot met meerdere vrijheidsgraden op een single processor computer systeem. Om zo'n robot met behulp van een single processor systeem te kunnen besturen is onderzocht hoe 'real-time' gedrag (o.a. dat alle assen tegelijkertijd aangestuurd moeten worden) bereikt kan worden op dit systeem.
Het blijkt dat het, met de hulp van specifieke hardware én door een speciale wijze van implementatie van de stuur-software, mogelijk is om dit 'real-time' gedrag te bereiken.
Dit heeft tot gevolg dat het nodig is, dat de hardware simultane 'latching' mogelijkheden biedt en dat alle delen van de besturingssoftware geimplementeerd worden als 'process-subroutines' die terugkeren naar de aanroepende routine, nadat ze hun taak afgerond hebben.

Om het nieuwe computer besturingssysteem te kunnen testen, zijn er verschillende regelwetten op geimplementeerd, die gebruikt worden om een Rotatie-Translatie (RT)-robot te besturen.
Hiertoe zijn eerst drie niet-adaptieve (PD, PID en een Vooruitsturings) en één adaptieve (MRAC) regelwet met behulp van numerieke simulatie getest. In deze simulaties wordt een dynamisch model van de echte RT-robot, '**simulatie**' model geheten, gebruikt om een inzicht te krijgen in het gedrag van de verschillende regelwetten.
Sommige regelwetten vereisen het gebruik van een (vereenvoudigd) model van de RT-robot, '**regel**' model '**met vooruitsturing**' geheten, om een tijdsafhankelijke stuurcomponent te kunnen berekenen. Beide modellen moeten afwijkend zijn, in structuur en/of in parameterwaarden om zinnige resultaten te verkrijgen.
De simulaties laten zien dat een PID regelwet goede resultaten vertoont, terwijl het 'afstellen' van de adaptieve regelwet moeilijk is.

Nadat de simulaties tevredenstellende resultaten opleveren, zijn de vier regelwetten geimplementeerd op de echte RT-robot. In het algemeen stemmen de resultaten van de simulaties en de echte RT-robot aardig overeen. De volgfouten zijn in beide gevallen van dezelfde grootte orde. Zelfs de maximaal bruikbare 'terugkoppel versterkingsfactoren' komen redelijk overeen.
Ook hier levert de PID regelwet goede resultaten, terwijl de adaptieve regelwet moeilijk 'af te stellen' is.
Dit heeft tot gevolg dat de adaptieve regelwet in zijn huidige vorm minder geschikt is om de echte RT-robot te besturen. Extra onderzoek zal nodig zijn om een manier te vinden om het adaptatie algorithme af te stellen en om de maximaal haalbare prestatie van de adaptieve regelwet te ontdekken.

LIST OF SYMBOLS

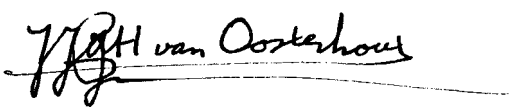
α = adaptation gain for K_p.
α_1 = adaptation gain for K_d.
β = adaptation gain for C.
γ = adaptation gain for B.
δ = adaptation gain for $\underline{F}$.
λ = adaptation gain for A.
ω_i = undamped natural frequency.
ζ_i = damping factor.
$\underline{\Theta}_r$ = reference trajectory (contains $\Phi^R{}_m$, Φ and x).
$\underline{\Theta}_m$ = output of the reference model.
$\underline{\Theta}$ = output of the RT-robot.
$\Phi^R{}_m$ = (rotational) position of the rotational motor.
Φ = (rotational) position of the turntable.
x = (translational) position of the translating arm.
$\underline{e}$ = 'incremental' tracking error (= $\underline{\theta}$ - $\underline{\theta}_r$).
$\underline{E}$ = 'total' tracking error (= $\underline{\Theta}$ - $\underline{\Theta}_r$).
$\underline{F}$ = auxiliary input.
P = nominal operating point.
$\underline{R}$ = adaptation vector.
$\underline{\tau}$ = joint torque vector.
$\underline{T}$ = 'incremental' controller torque.
$\underline{T}_1$ = feedback controller torque.
$\underline{T}_2$ = feedforward controller torque.
$\underline{u}$ = total control effort.
$\underline{u}_{pid}$ = P.I.D. feedback part of control effort.
$\underline{u}_{model}$ = model feedforward part of control effort.
V(..) = Liapunov function.
$\underline{z}$ = position-velocity error.

All other capital letters denote matrices.
A = model matrix (mass part).
B = model matrix (damping part).
C = model matrix (direct steering part).
K_p = proportional amplification matrix.
K_i = integral amplification matrix.
K_d = differential amplification matrix.

PREFACE

This report contains the results of my M.Sc. research at the Eindhoven University of Technology. I would like to thank all people who made it possible for me to develop and realize the control concept presented in this report.

Special thanks goes to my coach Ir. P.C. Mulders, my girlfriend Evelyne and my mother.



Jasper-Jan van Oosterhout

CONTENTS

SUMMARY 3

SAMENVATTING 4

LIST OF SYMBOLS 5

PREFACE 6

1. INTRODUCTION 9

2. ROBOT TRAJECTORY CONTROL 10
2.1. THE RT-ROBOT 10
2.2. THE MEASUREMENT SYSTEMS 11

3. MODELLING OF THE RT-ROBOT 12
3.1. THE NEED FOR DIFFERENT MODELS 12
3.2. DESCRIPTION OF THE SIMULATION MODELS 12
3.3. DESCRIPTION OF THE CONTROL MODELS 13

4. THE COMPUTER PLATFORM 14
4.1. THE POSITION INTERFACES 15
4.2. THE MOTOR INTERFACES 15

5. REAL TIME ASPECTS 16
5.1. THE DATA FLOW DIAGRAM 16
5.2. IMPLEMENTING THE DATA FLOW DIAGRAM 18

6. CONTROL LAWS 19
6.1. CONSIDERATIONS 19
6.2. COMPENSATION OF TRACKING ERRORS 20
6.2.1. P.I.D. CONTROL 20
6.2.2. FEEDFORWARD CONTROL 21
6.2.3. (MODEL REFERENCE) ADAPTIVE CONTROL 22
6.2.3.1. ADAPTATION AND STABILITY 23

7. SIMULATION 31
7.1. PD FEEDBACK CONTROL 32
7.2. PID FEEDBACK CONTROL 32
7.3. FEEDFORWARD CONTROL 33
7.3.1. THE 3-D.O.F. FEEDFORWARD MODEL 33
7.3.2. RESULTS OF 2-D.O.F. FEEDFORWARD CONTROL 35
7.4. ADAPTIVE CONTROL 36

8. IMPLEMENTATION 37
8.1. PD FEEDBACK CONTROL 37
8.2. PID FEEDBACK CONTROL 38
8.3. FEEDFORWARD CONTROL 38
8.4. ADAPTIVE CONTROL 40

9. CONCLUSION AND ADVICE 42

BIBLIOGRAPHY 43

APPENDICES 45

APPENDIX 1: SPECIFICATIONS OF THE MEASUREMENT SYSTEMS 46
APPENDIX 2: THE 11-D.O.F. [R6T5] LUMPED MASS-MODEL 48
APPENDIX 3: THE 5-D.O.F. [R3T2] SIMULATION MODEL 49
APPENDIX 4: THE 4-D.O.F. [R2T2] CONTROL MODEL 52
APPENDIX 5: THE 6-D.O.F. [R3T3] SIMULATION MODEL 56
APPENDIX 6: THE 2-D.O.F. [R1T1] CONTROL MODEL 61
APPENDIX 7: THE 3-D.O.F. [R2T1] CONTROL MODEL 62
APPENDIX 8: MODELS OF THE MOTORS,POWER-AMPS AND DACS 65
APPENDIX 9: THE INTERRUPT ARCHITECTURE 68

1. INTRODUCTION

The group Precision Engineering of the division WPA (Mechanical Production Engineering and Automation) of the Eindhoven University of Technology is researching algorithms (control laws) to control machines and robots. Objective of this research is to minimize the errors that arise when a robot tracks a mathematical defined trajectory (=path) in its work-space.

For testing purposes, a test rig is available consisting of a home-developed, industrial type, Rotation-Translation (RT)-Robot and a computer platform. Recently the computer platform, on which the control algorithm is implemented, has been modified. Instead of four single-board computers, the computer-platform now consists of an industrial standard, AT-386 type personal computer.

The changes in the computer architecture require a new approach in implementing the control algorithm. In implementing, special care must be taken in order to ensure **'real-time'** behaviour of the control program.

Before implementing a control law on the RT-robot, simulation studies are carried out to investigate the behaviour of the control law. For these simulations several models of the RT-robot are needed. Numerical simulation is performed in the software package PC-Matlab©. After satisfactory results of the simulations, the control law can be implemented at the real robot.

To test the performance of the new computer platform, four control laws are implemented and compared. Results of numerical simulation in PC-Matlab and the old computer platform will be taken into account to judge the new platform capabilities.

2. ROBOT TRAJECTORY CONTROL

The main objective in robot trajectory control is to control a mechanical manipulator in order to make its end-effector track some desired path. This path can be anywhere in a part of the real world, often called the work-space of the robot. In this work-space a mathematical path is defined. In the ultimate situation the end-effector should follow this mathematical defined path without deviations (in place and time).

2.1. THE RT-ROBOT

In this case the mechanical manipulator is a Rotation-Translation (RT)-robot. The RT-robot consists of a Rotation module and a Translation module. The end-effector (or tool centre point, TCP) of the RT-robot can, by movement of the two modules - a rotation Φ and a translation x - , track a path in a flat horizontal plane.

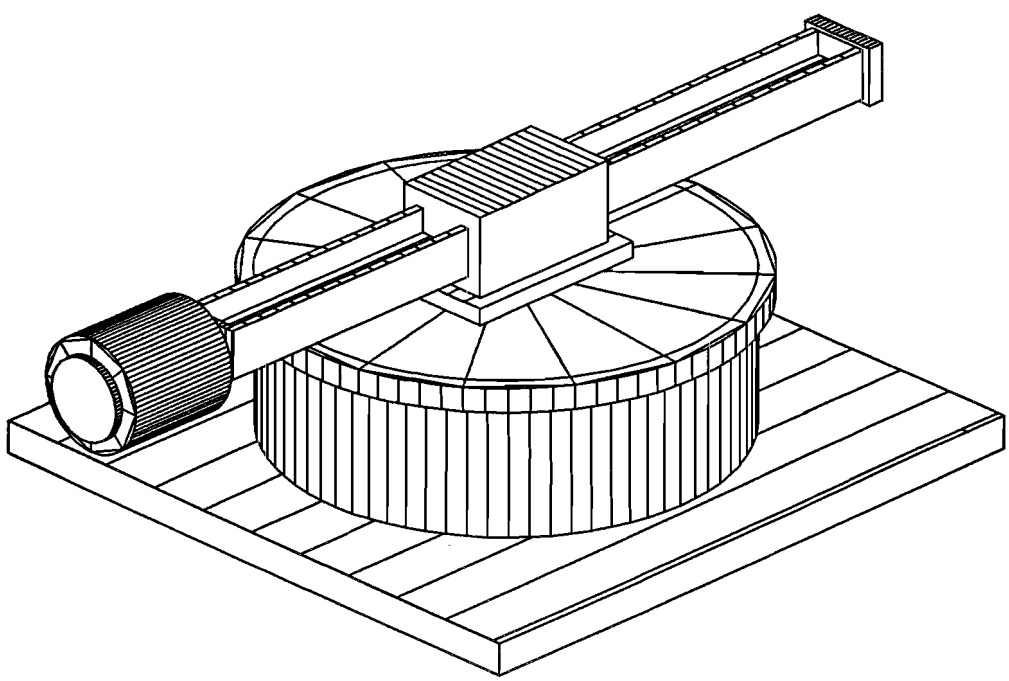


Figure 1: RT-Robot

The Translation module [12] contains a DC motor driving a translating arm by a spindle with a ballscrew nut. The DC motor is of the disc-armature type.
The Rotation module [11] contains an identical DC motor driving the turn table by a four stage toothed wheel combination with divided and preloaded wheels - realized with torsion springs - to eliminate backlash.

The translating arm as well as the Rotation module are restricted in their operating space by Hall-effect switches.

2.2. THE MEASUREMENT SYSTEMS

The Rotation and Translation modules are equipped with a number of incremental measurement systems. These are shown in figure 2 on page 14.
The position of the translating arm is measured via a direct 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. The resolution of the linear encoder is $1{,}6.10^{-6}$ [m]. At the motor-axis of the translation module a rotational optical encoder is mounted. This makes it possible to measure the rotation of the translation-motor. The resolution of the encoder is $2{,}51.10^{-4}$ [rad].
For direct rotational position measurement of the turntable an optical digital incremental encoder as a flexible scale has been mounted along the circumference of the turntable. This measurement-scale moves along a fixed mounted optical head. The resolution of the measurement-scale is $6{,}22.10^{-6}$ [rad]. An rotational optical encoder is mounted at the motor-axis of the rotational module. This makes it possible to measure the rotation of the rotational-motor. The resolution of the encoder is $2{,}51.10^{-4}$ [rad].
See appendix 1 for full specifications of the measurement systems.

3. MODELLING OF THE RT-ROBOT

Bax [13] has developed a lumped-mass model with 11 Degrees Of Freedom **(D.O.F.)** [R6T5] which gives a good description of the dynamic behaviour of the real RT-robot. See appendix 2. This model however, is so complex that it is too difficult to use in practice. For this reason Martens [1] has developed a number of reduced (less D.O.F.) models of the RT-robot.

3.1. THE NEED FOR DIFFERENT MODELS

In order to be able to perform simulations a simulation model is necessary. On one hand this model should describe the properties of the real robot as good as possible, but on the other hand the simulation model should be simple in order to keep the simulation-time low. During simulation the real robot is replaced by the **simulation-**model. (See figure 9 on page 23.)
In some control concepts a (simplified) robot-model is used in order to calculate a time-dependent feedforward steering component in the total control effort $\underline{u}$. The model used to calculate the feedforward component is called **feedforward control-**model. Both models **must** be different in order to get meaningful simulation results. If both models are the same, the feedforward part of the control effort exactly matches the required control effort needed for the simulation model to follow the reference trajectory. Thus, no tracking errors will occur and the simulation is useless.

3.2. DESCRIPTION OF THE SIMULATION MODELS

In the past several simulation models have been developed. In our simulations we use the 5-Degree **O**f **F**reedom [R3T2] **(5-D.O.F.)** model. This model was developed by Martens [2] and contains the three lowest eigenfrequencies present in the model of Bax. The identical 5-D.O.F. model is used in order to make a fair comparison to the results of Voorkamp. See appendix 3 for details on the 5-D.O.F. [R3T2] simulation model.
Because of the mounting of an (additional) encoder at the motor axis of the Translation module, we are able to measure the position at 2 points of the Translation module. Hence, we will need a 2 D.O.F. **control** model for the Translation module to take full advantage of both measurement systems. The use of both measurement systems therefore leads to a 4-D.O.F. [R2T2] **control** model for the entire robot. See appendix 4. The need for a different structure therefore leads to a 6-D.O.F. [R3T3] **simulation** model. This model is presented in appendix 5.

3.3. DESCRIPTION OF THE CONTROL MODELS

The simplest Feedback control model arises from the assumption that both modules of the RT-robot can be regarded as rigid bodies. Hence, the entire robot can be modelled using two Degrees Of Freedom. This leads to the 2-D.O.F. [R1T1] Feedback control model. See appendix 6 for details.

In the past however, it was observed that the Rotation module of the RT-robot was relatively elastic and wasn't suitably modelled by a rigid body. The use of a 'rigid' model for this module excited instabilities when a PD feedback control law was used.

Therefore a 'flexible' Feedback control model was developed in which the Rotation module was modelled as a flexible body, by using two Degrees Of Freedom. By measuring two Degrees Of Freedom, one at the turntable motor-axis and one at the turntable circumference, the 'wind-up' in the four-stage toothed wheel drive could be measured. Including both degrees of freedom in their own feedback loop resulted in the elastic energy, accumulated in the drive, to stay bounded.
This made the Rotation module more stable than in the case of feedback of the turntable rotation only. Hence, higher feedback gains could be admitted, resulting in a better performance of the Rotation module. This 'flexible' Feedback control model has three Degrees Of Freedom for the entire RT-robot, and is called the 3-D.O.F. [R2T1] Feedback control model. See appendix 7 for details.

In order to be able to calculate the required nominal control effort a 'Feedforward'-model of the real RT-robot is needed. The calculation of the nominal control effort has to be done on-line, so the Feedforward model should be as simple as possible. Voorkamp [2] designed a 3-D.O.F. [R2T1] Feedforward control model. In this research however a 'rigid' 2-D.O.F. [R1T1] Feedforward control model is used to calculate the 'feedforward' component of the nominal control effort in the total control effort $\underline{u}$. Why this is done will be explained later on. See appendix 6.

Also, models of the DC-motors, the power-amplifiers and the Digital to Analog Converters (DACs) are needed. These are described in appendix 8.

4. THE COMPUTER PLATFORM

Recently, the computer platform on which the control laws have been implemented, has been modified. The old platform, consisting of four parallel Intel Single Board Computers (SBC's), did not meet the performance demands required by the applied (adaptive) control algorithm. Besides that, there were problems with the -home made- position interfaces and the implemented algorithms because only integers could be used and no numerical co-processor was available. See Voorkamp [2].
The time needed to make an implementation 'work' was, because of all these reasons, much longer than the time left to 'study' control laws and their aspects.
Therefore in 1991, the control platform was totally modified [3]. Nowadays it is based on a IBM compatible 386 type personal computer with specific interface cards, which plug directly into the computers bus. In this computer, a 387 numerical co-processor is available to speed up floating point calculations. Instead of the (parallel) processors present in the old control system, there is now only one processor. This also facilitates the programming of algorithms because synchronising of the axes is easier due to the presence of the single processor. All the programming is done in Turbo-Pascal©. The picture below describes the new control platform.

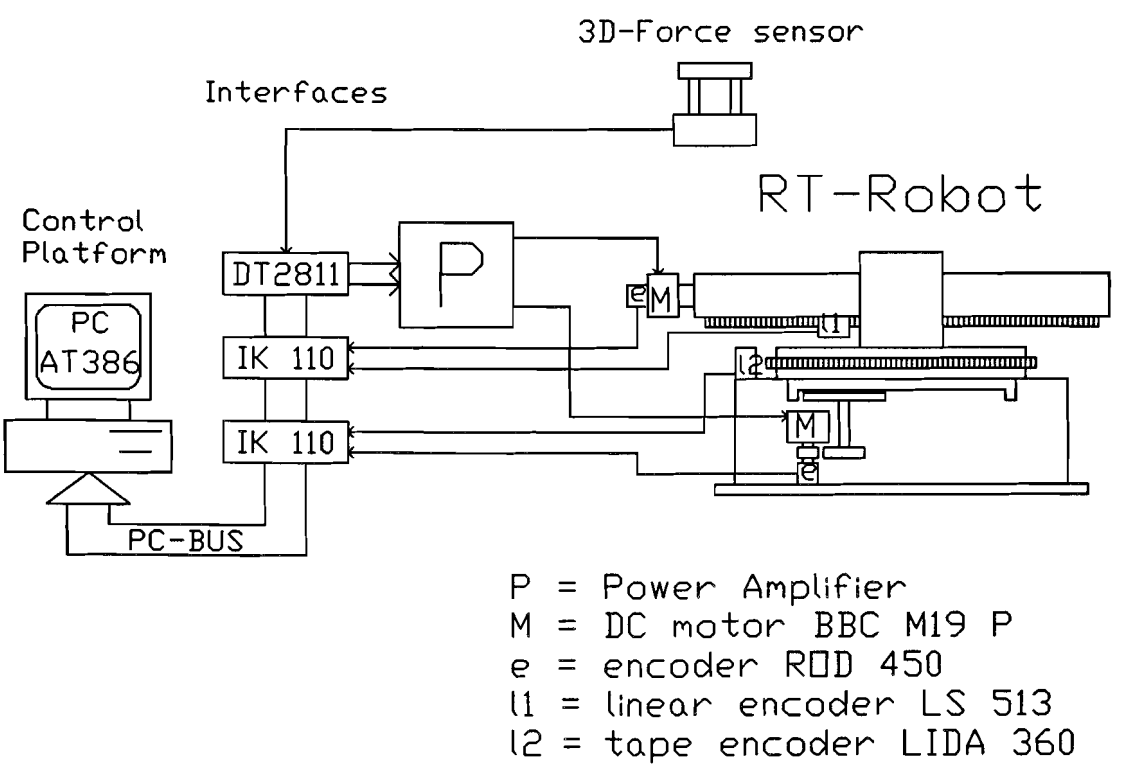


Figure 2: The new control platform

4.1. THE POSITION INTERFACES

The position interfaces consist of two Heidenhain© IK 110 interface boards (IK 110 in figure 2).
Each board contains two 26 bit counters with a 'latch'. Besides this they can read the zero-index signal of the measurement systems. The boards also contain an electronic interpolator, which makes it possible to enlarge the resolution (up to 50 times).
The flexible scale of the Rotation module was used in conjunction with a separate interpolator [11]. In the new control platform this separate interpolation unit has become obsolete, because of the on-board interpolation facilities of the IK 110.
A big disadvantage of the IK 110 boards is that they require a sine-wave input signal. The old measurement systems provided square-wave signals. Fortunately the sine-wave output of the measurement head of the flexible encoder at the rotational module, was converted into a square-wave signal in the separate interpolation unit. Removing the separate interpolation unit resulted in a correct signal.
The measurement head of the translation linear encoder provided more difficulties. Fortunately it was possible to replace only this measurement head for a version that provided sine-wave output signals.
The old rotational encoders have been replaced for new versions providing sine-wave output signals.

More details can be found in [3].

4.2. THE MOTOR INTERFACES

The old - home build - motor interfaces are replaced by the Data-Translation© DT 2811 interface-board (Figure 2). This board contains two 12 bit Digital to Analog Converters (DACs) and Digital Input Output (DIO) facilities.
The DACs are used to control the D.C. motors via an external power amplifier (P in figure 2). Because the power amplifiers of the motors require ± 15 volts for full speed and the DACs only deliver an output signal of ± 5 volts an extra amplifier is used (not shown in figure 2) with a fixed amplification gain of 3.
The Hall-switches (not shown in figure 2) are read-in via the D.I.O. lines of the DT 2811 board. To receive a clear signal, Schmitt-triggers of the type 74LS13, are used between Hall-switches and the board.

More details can be found in [3]

5. REAL TIME ASPECTS

Real time systems tend to be designed for specific applications. They possess special purpose peripheral devices such as sensors and drivers, along with the digital interface hardware which lets them communicate with these devices.
It will be obvious that the RT-robot and the control platform together form a real time system. The measurement systems, such as encoders and linear or flexible encoders form the sensors of the system. The interface boards, such as the IK 110 and DT 2811 (Figure 2), let the control platform communicate with the measurement systems, Hall-switches and power amplifiers.
The essential requirement of any real time system is that it has to respond quickly and correctly to its environment. It will now be explained how a quick and correct response can be obtained on a single processor system.

5.1. THE DATA FLOW DIAGRAM

In the data flow diagram everything is happening at once. Part of the system is for processing, part is for transportation and part is for storage. Bubbles represent processing, and as always the name of the process is written inside the bubble. Arrows represent transportation, and the name of the commodity being transported is written along the arrow. Boxes and parallel horizontal lines represent storage elements. The boxes on the periphery of the system represent the inputs and outputs of the system. This is logical since peripheral devices constitute a systems connection with its environment. Parallel horizontal lines represent a temporary internal storage unit, because it is only connected to other parts of the system.

Bubbles will be used to represent data processing- in other words code. In computer programs there is only one commodity that flows along the arrows: data. The labels on the arrows will tell about the kind of data passing along the arrows between routines, data structures, and I/O devices. The arrows themselves have no direct counterpart. One might say that they represent the passing of parameters.

Figure 3 gives the (simplified) data-flow diagram of the RT-robot:

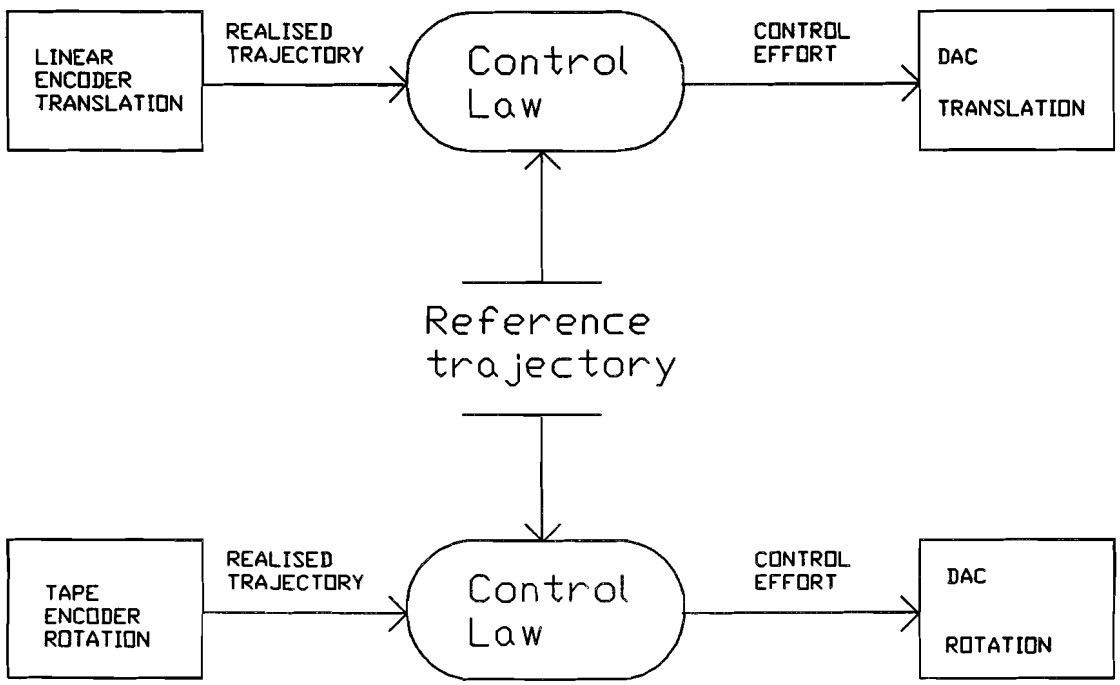


Figure 3: Data flow diagram of the RT-robot

In figure 3 both axes are controlled simultaneously. This implies that several actions must be carried out at the same moment. Fortunately, the hardware assists us at this difficult task. The IK 110 interface boards provide capabilities to read (via a hardware 'latch') the actual positions of all measurement systems at once [3]. It would be ideal if the DACs on the DT 2811 interface board provided the possibility of converting both output values at the same moment. Unfortunately this is not the case, so we will have to use a little trick to minimize the delay here. Conversion of the output values only takes place after the high byte of the 12 bit dataword is written into the corresponding register. By first writing both low-bytes (for Translation and Rotation) and after that writing both high-bytes right after another, the delay between the conversion of the output values of both axes will be minimized. Measurements show that the typical delay is $3{,}19.10^{-6}$. [s.]. This is acceptable. So the sole thing left to worry about is how to program the computer in a way that both control laws are executed simultaneously. This problem will be dealt with in the next chapter.

5.2. IMPLEMENTING THE DATA FLOW DIAGRAM

In practice it is impossible for a (single processor) computer to perform multiple tasks at the same moment. Real-time systems, on the other hand, appear to execute several or many routines at once. This is, of course, an illusion: the illusion of multiple concurrent tasks. It is one of the most powerful illusions of computer science. When all processes are repetitively called for a very short period, this gives the illusion of multiple concurrent tasks.
The goal is to develop a way for a process to 'suspend' itself, assigning the processor to another waiting process. We like to have a subroutine which any process can call, which will suspend the calling process and activate the next dormant process.
This subroutine is called a process manager.
When a process is suspended its entire status must be saved, so that when it is reactivated it can continue as if nothing has happened. From the point of view that processes merely exist to dispatch data, the status of a process only serves the purpose of helping to move and transform the original data from the original input device or queue to an output device or queue. At the moment when the process has finished dispatching, it can be safely suspended without saving any status at all. But we observed that suspending a process means nothing more or less than saving its status and re-activating a process means restoring its status.
Processes can be implemented as subroutines which return when they have finished dispatching. Instead of a process manager all we need is a master top-level routine which will call each process subroutine in turn.
We can now propose the model in figure 4 for processes in a 'streamlined' real-time model. Figure 5 gives us a model for the program's top-level routine.

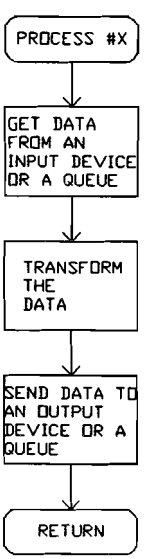


Figure 4: Processes

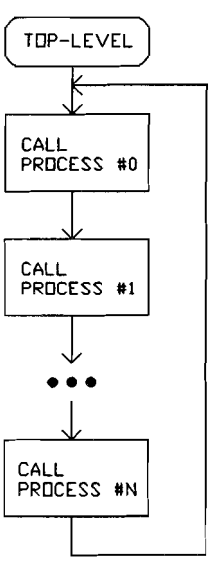


Figure 5: Top-level routine

6. CONTROL LAWS

6.1. CONSIDERATIONS

As mentioned before, the main objective in robot trajectory control is to make a mechanical manipulator track some desired path. Thus, there is need for a 'method' to calculate the control action, necessary to achieve tracking. This 'method' will be called **control law.**
We will now discuss the control of a electromechanical servo, which forms the basis of the rotational and translation module of the RT-robot. Figure 6 gives a simplified description of such an electromechanical servo.

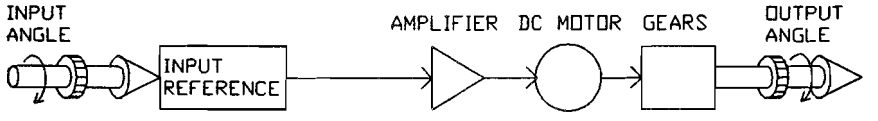


Figure 6: Electromechanical servo

This kind of system is called 'open-loop' system, because there is no connection between the systems output and input. Hence, there is no way to make sure the output follows the input.
In a (robot) servo system however, there will be a way to compensate the occurring deviations. This concept is called 'feed-back'. A so called feed-back loop measures the occurring deviation and controls the systems actuators via a control law in order to compensate these errors.

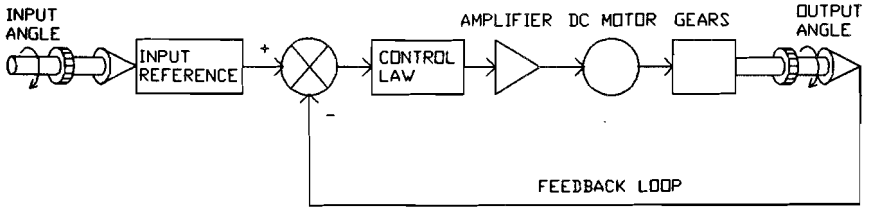


Figure 7: Servo system with feedback

Figure 7 shows the servo system when feedback is applied. The control law calculates the necessary control effort based on the difference of the systems output and input.

A simple analysis shows that, provided that the feedback is in the right sense and that the loop gain is sufficiently high, the output can be made to follow the input more or less exactly. However, the loop gain can't be increased unlimited because of the risk to excite instability in the system.

Most analyses of control systems make the assumption that all elements operate in a linear fashion, but certain non-linearities are very obvious when they are present and their effects can be intuitively understood without complex theoretical analysis.

6.2. COMPENSATION OF TRACKING ERRORS

Almost any practical system in its rudimentary form will display imperfections of performance of one kind or another, which must be removed or reduced by the application of compensation methods. Some possible faults are as follows:

1. Poor positioning accuracy
2. Poor following accuracy
3. Too slow response
4. Too many overshoots
5. Too large overshoots
6. Too long settling time

Some common compensation techniques include:

- P.I.D. control;
- Feed forward control;
- Adaptive control.

These techniques will be discussed in the next chapters.

6.2.1. P.I.D. CONTROL

PID control is the most widely used concept in present days stiff manipulators. For these manipulators PID controllers frequently turn out to perform quite well, in spite of non-linear 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).

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

$$\underline{u} = \underline{u}_{pid} = - K_p \underline{E}(t) - K_i \int_{t_0}^{t} \underline{E}(\tau)\, d\tau - K_d \dot{\underline{E}}(t) \qquad (1)$$

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

There are two main problems in using a PID control law:

Use of the PID control law results in an output signal unequal to zero only if the tracking error $\underline{E}(t)$, its integral or time derivative are unequal to zero. No measures are taken to avoid errors in advance, e.g. by means of a model based time dependent steering component in the total control effort $\underline{u}$.
The PID control concept can be unstable when it is applied to control the position of an end-effector at the end of an open chain of bodies with one or more flexible links. In this case the total number of Degrees Of Freedom (=outputs) is greater than the number of servomotors (=inputs). An example of this case is given by the Rotation module being modelled with two Degrees Of Freedom in the 3-D.O.F. [R2T1] model.

6.2.2.FEEDFORWARD CONTROL

A way to overcome the first problem of avoiding errors in advance when using PID control is to add a 'feedforward' control part to the PID control law. This technique, also called 'Computed Torque' or 'Inverse Dynamics' control is based on the use of an inverse dynamic model of the manipulator to calculate a time dependent feedforward steering component $\underline{u}_{model}$ in the total control effort $\underline{u}$. In this way errors, caused by changes in the desired path, can be avoided in advance. The feedforward time dependent steering component $\underline{u}_{model}$ is calculated by use of the desired path and a model of the manipulator:

$$\underline{u}_{model} = A\, \ddot{\underline{\Theta}}_r(t) + B\, \dot{\underline{\Theta}}_r(t) + C\, \underline{\Theta}_r(t) \qquad (2)$$

A, B and C are the (constant) model matrices.
The total input $\underline{u}$ is now composed out of a feedforward part $\underline{u}_{model}$ and a P.I.D. feedback part $\underline{u}_{pid}$:

$$\underline{u} = \underline{u}_{model} + \underline{u}_{pid} \qquad (3)$$

Figure 8 shows the feedforward concept.

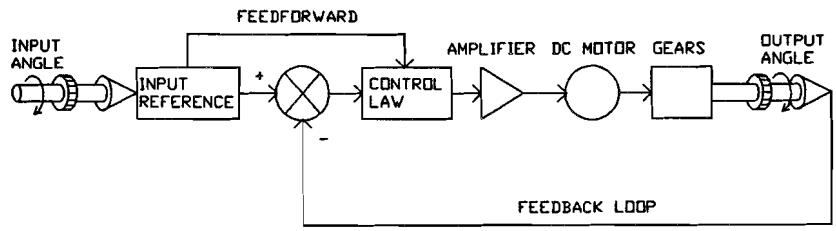


Figure 8: The feedforward electromechanical servo

6.2.3. (MODEL REFERENCE) ADAPTIVE CONTROL

The second problem that is connected with the use of an PID control law applied to a flexible manipulator, can be solved by the use of an 'adaptive' control law. In such an adaptive control law the amplification matrices are **adapted** in order to assure stability and to achieve trajectory tracking of the controlled system. As a result of this adaption the K_p, K_i and K_d amplification matrices are time dependent[1].

Among various alternative methods, the use of a technique known as Model Reference Adaptive Control seems to be one of the most feasible approaches possible for implementation of adaptive control systems. The prime characteristic of such an adaptive system is the presence of a so-called **reference model**. This reference model is used to specify the desired control performance. The adaptive control law will try to adapt the 'control parameters' in such a way that the performance of the controlled manipulator will match the performance of the reference model.

$$\underline{u}_{pid} = - K_p(t)\, \underline{E}(t) - K_i(t) \int_{t_0}^{t} \underline{E}(\tau)\, d\tau - K_d(t)\, \dot{\underline{E}}(t) \qquad (4)$$

$$\underline{u}_{model} = A(t)\, \ddot{\underline{\Theta}}_r(t) + B(t)\, \dot{\underline{\Theta}}_r(t) + C(t)\, \underline{\Theta}_r(t) \qquad (5)$$

$$\underline{u} = \underline{u}_{model} + \underline{u}_{pid} \qquad (6)$$

1- *In some adaptive control approaches, not only the amplification matrices are adapted, but also the (feedforward) model matrices A, B and C. In this case A, B and C will be time dependent. Then the total set of 'control parameters' consists of the time dependent matrices: $K_p(t)$, $K_i(t)$, $K_d(t)$, $A(t)$, $B(t)$ and $C(t)$.*

Adaptation of the 'control parameters' is often based on the difference between the output $\underline{\theta}_m$ of the reference model and the output $\underline{\theta}$ of the real robot. Figure 9 gives insight in the M.R.A.C. method:

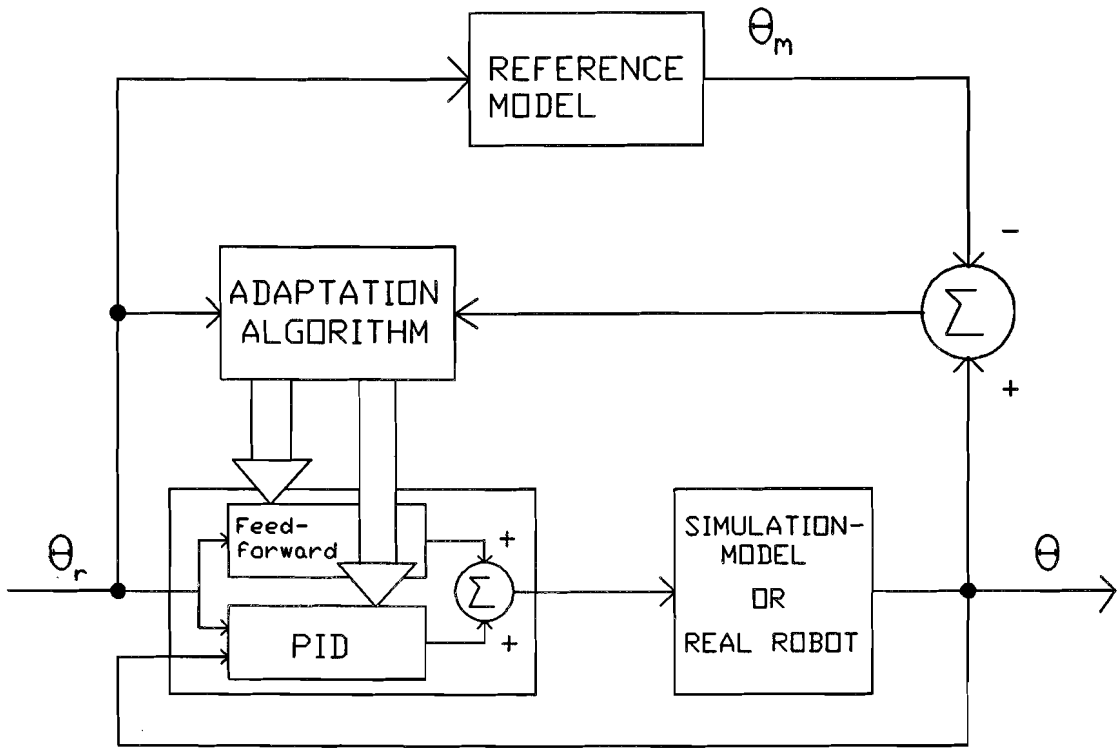


Figure 9: M.R.A.C. method

6.2.3.1. ADAPTATION AND STABILITY

In this chapter the adaptation rules will be derived, according to Seraji [4,5] and a stability analysis will be given according to Liapunov [8].

In general an n-link robot manipulator in which the nx1 joint torque vector $\underline{\tau}(t)$ is related to the nx1 joint angle vector $\underline{\theta}(t)$ by the Euler-Lagrange dynamic equation of motion can be described as:

$$M(\underline{\Theta})\ddot{\underline{\Theta}} + N(\underline{\Theta},\dot{\underline{\Theta}}) + G(\underline{\Theta}) = \underline{\tau} \qquad (7)$$

The elements of M, $\underline{N}$ and $\underline{G}$ are highly complex nonlinear functions of the link configuration $\underline{\theta}$, the speed of motion $\dot{\underline{\theta}}$, and the payload mass.
The incremental dynamic behaviour of the robot for perturbations in the neighbourhood of some nominal operating point P

$$P = [\hat{\underline{\tau}},\hat{\underline{\theta}},\hat{\dot{\underline{\theta}}}] \qquad (8)$$

is obtained by linearizing the nonlinear model (7) to yield:

$$A\ddot{\underline{\theta}}(t) + B\dot{\underline{\theta}}(t) + C\underline{\theta}(t) = \underline{T}(t) \tag{9}$$

where $\underline{\theta}(t) = \underline{\theta}(t) - \hat{\underline{\theta}}$, $\dot{\underline{\theta}}(t) = \dot{\underline{\theta}}(t) - \dot{\hat{\underline{\theta}}}$ and $\underline{T}(t) = \underline{\tau}(t) - \hat{\underline{\tau}}$ are incremental variables. Based on the incremental model of manipulator dynamics, we shall now develop the structure of the joint control system for incremental motion.
A multivariable feedback controller $\underline{T}_1(t)$ is employed for stability and pole-placement. A multivariable feedforward controller $\underline{T}_2(t)$ is used to achieve trajectory tracking.

For the use of stabilization and pole placement, a suitable choice is a proportional-derivative (PD) controller; namely

$$\underline{T}_1(t) = K_p\underline{e}(t) + K_d\dot{\underline{e}}(t) \tag{10}$$

Where $\underline{e}(t) = \underline{\theta}_r(t) - \underline{\theta}(t)$ is the '**incremental**' nx1 tracking-error vector.

The feedforward controller is chosen as the minimal-order inverse of the linearized manipulator model, namely

$$\underline{T}_2(t) = C\underline{\theta}_r(t) + B\dot{\underline{\theta}}_r(t) + A\ddot{\underline{\theta}}_r(t) \tag{11}$$

It must be noted that since the desired velocity and acceleration are directly available, it is not necessary to perform differentiation in implementing the feedforward controller; and hence it is realizable.

The incremental joint control law is given by combining the feedback controller $\underline{T}_1(t)$ and the feedforward controller $\underline{T}_2(t)$

$$\underline{T}(t) = \underline{T}_1(t) + \underline{T}_2(t) = K_p\underline{e}(t) + K_d\dot{\underline{e}}(t) + C\underline{\theta}_r(t) + B\dot{\underline{\theta}}_r(t) + A\ddot{\underline{\theta}}_r(t) \tag{12}$$

When the "incremental" control law is implemented on the non-linear robot, the "**total**" control law can be expressed as the sum of two components. The first component is the value of the joint torque vector at the nominal operating point P namely $\hat{\underline{\tau}}$. The second component is the contribution due to the incremental controllers $\underline{T}_1(t)$ and $\underline{T}_2(t)$.

Thus the "total control law is given by

$$\underline{\tau}(t) = \hat{\underline{\tau}} + \underline{T}(t) = \hat{\underline{\tau}} + K_p(\underline{\theta}_r(t) - \underline{\theta}(t)) + K_d(\dot{\underline{\theta}}_r(t) - \dot{\underline{\theta}}(t)) + C\underline{\theta}_r(t) + B\dot{\underline{\theta}}_r(t) + A\ddot{\underline{\theta}}_r(t) \quad (13)$$

Now let the "total" reference vector be $\underline{\theta}_r(t) = \hat{\underline{\theta}} + \underline{\theta}_r(t)$ and the "total" angle vector be $\underline{\theta}(t) = \hat{\underline{\theta}} + \underline{\theta}(t)$. Substituting these in equation (13) gives the "total" control law in terms of the "total" variables as

$$\underline{\tau}(t) = \underline{\tau}^* + K_p[\underline{\Theta}_r(t) - \underline{\Theta}(t)] + K_d[\dot{\underline{\Theta}}_r(t) - \dot{\underline{\Theta}}(t)] + C\underline{\Theta}_r(t) + B\dot{\underline{\Theta}}_r(t) + A\ddot{\underline{\Theta}}_r(t) \quad (14)$$

It is seen that in addition to the two terms due to the feedback and feedforward controllers, a third term $\underline{\tau}^* = \hat{\underline{\tau}} - C\hat{\underline{\theta}} - B\dot{\hat{\underline{\theta}}} - A\ddot{\hat{\underline{\theta}}}$ reflecting the effect of the operating point P is present in the control law.

Consider now the nonlinear model of manipulator dynamics (7) written as

$$\underline{\tau}(t) = A^*(\underline{\Theta},\dot{\underline{\Theta}})\ddot{\underline{\Theta}}(t) + B^*(\underline{\Theta},\dot{\underline{\Theta}})\dot{\underline{\Theta}}(t) + C^*(\underline{\Theta},\dot{\underline{\Theta}})\underline{\Theta}(t) + \underline{N}(\underline{\Theta},\dot{\underline{\Theta}},t) \quad (15)$$

Where A^*,B^* and C^* are complex nonlinear functions of $\underline{\theta}$ and $\dot{\underline{\theta}}$ The nonlinear time-varying nx1 vector $\underline{N}(\underline{\theta},\dot{\underline{\theta}},t)$ is introduced in equation (15) to represent any unmodelled non-linearities in the manipulator dynamics.

Let us now consider the "**total**" control law for the **non-linear** robot model. In order to account for the variations in the operating point the gains of the feedback and feedforward controllers are varied with time and a time-varying signal $\underline{F}(t)$ is also included in the control law, where $\underline{F}(t)$ is considered as an 'auxiliary' input to be synthesized by the adaptive scheme. This yields the adaptive control law

$$\underline{\tau}(t) = \underline{F}(t) + [K_p(t)\underline{E}(t) + K_d(t)\dot{\underline{E}}(t)] + [C(t)\underline{\Theta}_r(t) + B(t)\dot{\underline{\Theta}}_r(t) + A(t)\ddot{\underline{\Theta}}_r(t)] \quad (16)$$

where $\underline{E}(t) = \underline{\theta}_r(t) - \underline{\theta}(t)$ is the nx1 'total' tracking-error vector.

On applying the adaptive control law to the nonlinear robot model, the error differential equation is obtained:

$$A^*\ddot{\underline{E}}(t) + (B^* + K_d)\dot{\underline{E}}(t) + (C^* + K_p)\underline{E}(t) = (\underline{N}(t) - \underline{F}(t)) + (A^* - A)\ddot{\underline{\Theta}}_r(t) + (B^* - B)\dot{\underline{\Theta}}_r(t) + (C^* - C)\underline{\Theta}_r(t) \quad (17)$$

It is seen that $\underline{\theta}_r(t)$ and its derivatives as well as $[\underline{N}(t) - \underline{F}(t)]$ appear as forcing functions on the right-hand side of the error equation (17). Therefore, if the gains of the feedforward controller $\underline{T}_2(t)$ and $\underline{F}(t)$ are fixed, the solution of equation (17) for the tracking error $\underline{E}(t)$ will no longer tend to zero asymptotically, and will depend on $\underline{\theta}_r(t)$ and $\underline{F}(t)$.

As a result, when the robot model changes in time, it is essential to adapt the feedforward gains and the auxiliary input to cope with these variations so as to accomplish trajectory tracking. The feedback gains will also be adapted to ensure closed-loop stability with desired transient performance.

Now, let us define the 2nx1 position-velocity error vector z(t) as:

$$z(t)=\begin{pmatrix}\underline{E}(t)\\ \dot{\underline{E}}(t)\end{pmatrix} \tag{18}$$

And rewrite the equation (17) in state-space-format:

$$\dot{z}(t) = \begin{pmatrix}0 & I\\ -[A^*]^{-1}[C^*+K_p] & -[A^*]^{-1}[B^*+K_d]\end{pmatrix}z(t) + \begin{pmatrix}0\\ [A^*]^{-1}[\underline{N}-\underline{F}]\end{pmatrix} + \begin{pmatrix}0\\ [A^*]^{-1}[C^*-C]\end{pmatrix}\underline{\Theta}_r(t)$$

$$+ \begin{pmatrix}0\\ [A^*]^{-1}[B^*-B]\end{pmatrix}\dot{\underline{\Theta}}(t) + \begin{pmatrix}0\\ [A^*]^{-1}[A^*-A]\end{pmatrix}\ddot{\underline{\Theta}}_r(t) \tag{19}$$

Equation (19) constitutes the "adjustable system" in the M.R.A.C. framework.

Now the "reference model" shall be defined which embodies the desired performance of the system in terms of the tracking error $\underline{E}(t)$. In the case of a robot manipulator, the most desirable situation is that each tracking-error $E_i(t)$ should be decoupled from the others and must satisfy a second-order homogeneous differential equation of the form:

$$\ddot{E}_i(t) + 2\xi_i\omega_i\dot{E}_i(t) + \omega_i^2E_i(t) = 0; \qquad i=1,...,n \tag{20}$$

where ω_i and ξ_i are the undamped natural frequency and the damping factor specified by the designer.
Equation (20) can be written in the vector form:

$$\ddot{\underline{E}}_m(t) + D_2\dot{\underline{E}}_m(t) + D_1\underline{E}_m(t) = \underline{0} \tag{21}$$

where

$$D_2 = diag(2\xi_i\omega_i) \quad , \quad D_1 = diag(\omega_i^2) \tag{22}$$

are constant nxn diagonal matrices and the subscript 'm' denotes the reference model. Equation (21) can be put in the standard state space format:

$$\dot{z}(t) = \begin{pmatrix} 0 & I \\ -D_1 & -D_2 \end{pmatrix} z_m(t) \tag{23}$$

Since the reference model is stable, according to Liapunovs second method there exists a symmetric positive-definite 2nx2n constant matrix P which satisfies the Liapunov equation [8].

$$P = \begin{pmatrix} P_1 & P^T_2 \\ P_2 & P_3 \end{pmatrix} \tag{24}$$

The method to derive this matrix P will now be given. Equation (23) describes an autonomous linear time-invariant system and can be written as:

$$\dot{z}_m(t) = Dz_m(t) \tag{25}$$

where the elements of D are constant. Consider now a general quadratic Liapunov function:

$$V(z_m) = z_m^T P z_m \tag{26}$$

In which P is a positive-definite symmetric matrix. The function $\dot{V}(z_m)$ is:

$$\dot{V}(z_m) = \dot{z}_m^T P z_m + z_m^T P \dot{z}_m \tag{27}$$

Substituting (25) into (27) gives

$$\dot{V}(z_m) = z_m^T (D^T P + PD) z_m \tag{28}$$

Equation (28) can be written as:

$$\dot{V}(z_m) = -z_m^T Q z_m \tag{29}$$

The quantity Q therefore is given by:

$$Q = -(D^T P + PD) \tag{30}$$

According to Liapunovs second method this system will be stable as long as $\dot{V}(x)$ is at least semi-definite. In equation (25) D is completely defined by the reference-systems equations. Hence if an arbitrary positive-definite or positive semi-definite matrix is selected for Q and the equation (30) is solved for P, asymptotic stability is concluded by the positive definiteness of P. An infinite number of V-functions and their corresponding definite or semi-definite time derivatives are capable of determining whether or not global asymptotic stability exists.
The adaptation laws can now be derived which ensure that, for any reference trajectory $\underline{\Theta}_r(t)$, the state of the system (19) tends to that of the reference model asymptotically. The robot parameters A^*, B^*, C^* and $\underline{N}$ are treated as unknown and "slowly time-varying" in comparison with the adaptation scheme.
The adaptation laws are obtained as:

$$\dot{\underline{E}}(t) = \delta\left\{P_2\left[\underline{E}(t)-\underline{E}_m(t)\right]+P_3\left[\dot{\underline{E}}(t)-\dot{\underline{E}}_m(t)\right]\right\} \tag{31}$$

$$\dot{K}_p(t) = \alpha\left\{P_2\left[\underline{E}(t)-\underline{E}_m(t)\right]+P_3\left[\dot{\underline{E}}(t)-\dot{\underline{E}}_m(t)\right]\right\}\underline{E}^T(t) \tag{32}$$

$$\dot{K}_d(t) = \alpha_1\left\{P_2\left[\underline{E}(t)-\underline{E}_m(t)\right]+P_3\left[\dot{\underline{E}}(t)-\dot{\underline{E}}_m(t)\right]\right\}\dot{\underline{E}}^T(t) \tag{33}$$

$$\dot{C}(t) = \beta\left\{P_2\left[\underline{E}(t)-\underline{E}_m(t)\right]+P_3\left[\dot{\underline{E}}(t)-\dot{\underline{E}}_m(t)\right]\right\}\Theta^T{}_r(t) \tag{34}$$

$$\dot{B}(t) = \gamma\left\{P_2\left[\underline{E}(T)-\underline{E}_m(t)\right]+P_3\left[\dot{\underline{E}}(t)-\dot{\underline{E}}_m(t)\right]\right\}\dot{\underline{\Theta}}^T_r(t) \tag{35}$$

$$\dot{A}(t) = \lambda\left\{P_2\left[\underline{E}(t)-\underline{E}_m(t)\right]+P_3\left[\dot{\underline{E}}(t)-\dot{\underline{E}}_m(t)\right]\right\}\ddot{\underline{\Theta}}^T_r(t) \tag{36}$$

Where $\{\delta,\alpha,\alpha_1,\beta,\gamma,\lambda\}$ are the positive adaptation gains specified by the designer. The solution of the homogeneous reference model can be expressed as:

$$\underline{z}_m(t) = \exp[Dt]\cdot\underline{z}_m(0) \tag{37}$$

where $\underline{z}_m(0)$ is the initial state of the reference model.
Since the initial values of the reference- and actual- trajectories are often the same, the initial error is usually zero; i.e. $\underline{z}_m(0) = \underline{0}$.
Hence from equation (37), $\underline{z}_m(t) \equiv 0$ for all t.

Therefore, the adaptation laws simplify to:

$$\dot{E}(t) = \delta \underline{R}(t) \qquad \dot{K}_p(t) = \alpha \underline{R}(t) \underline{E}^T(t)$$

$$\dot{K}_d(t) = \alpha_1 \underline{R}(t) \dot{\underline{E}}^T(t) \qquad \dot{C}(t) = \beta \underline{R}(t) \underline{\Theta}^T{}_r(t)$$

$$\dot{B}(t) = \gamma \underline{R}(t) \dot{\underline{\Theta}}^T_r(t) \qquad \dot{A}(t) = \lambda \underline{R}(t) \ddot{\underline{\Theta}}^T_r(t)$$

(38)

where $\underline{R}(t)$ is an nx1 vector as:

$$\underline{R}(t) = P_2 \underline{E}(t) + P_3 \dot{\underline{E}}(t)$$

(39)

Thus the required auxiliary input and the controller gains are given by:

$$\underline{F}(t) = \underline{F}(0) + \delta \int_0^t \underline{R}(t)\, dt \qquad K_p(t) = K_p(0) + \alpha \int_0^t \underline{R}(t) \underline{E}^T(t)\, dt$$

$$K_d(t) = K_d(0) + \alpha_1 \int_0^t \underline{R}(t) \dot{\underline{E}}^T(t)\, dt \qquad C(t) = C(0) + \beta \int_0^t \underline{R}(t) \underline{\Theta}^T_r\, dt$$

$$B(t) = B(0) + \gamma \int_0^t \underline{R}(t) \dot{\underline{\Theta}}^T_r\, dt \qquad A(t) = A(0) + \lambda \int_0^t \underline{R}(t) \ddot{\underline{\Theta}}^T_r\, dt$$

(40)

Figure 10 on page 30 gives the adaptive joint control scheme.

Seraji makes the following remarks regarding the above adaptive control scheme:

1. The adaptation laws (40) do not require the knowledge of the robot parameters or the payload, and are based entirely on the reference and actual trajectories which are both directly available.

2. The adaptation method has *guaranteed convergence* in that the actual trajectory $\underline{\theta}(t)$ tracks the reference trajectory $\underline{\theta}_r(t)$ for any arbitrary time function $\underline{\theta}_r(t)$. The assurance of asymptotic tracking is the main attraction of the model reference adaptive control laws and relieves the designer from the complicated analysis of the stability problem.

3. The rate of convergence, i.e. the speed of adaption, depends on the magnitudes of the adaption gains $\{\delta,\alpha,\alpha_1,\beta,\gamma,\lambda\}$ and on the reference trajectory $\underline{\theta}_r(t)$. To maintain a high speed of adaption for all $\underline{\theta}_r(t)$, it is necessary to choose reasonable large values for the adaption gains.

4. The convergence of the adaptive scheme is *independent* of the initial values $\{\underline{F}(0),\underline{K}_p(0),\underline{K}_d(0),C(0),B(0),A(0)\}$. Therefore, it is not necessary to set these initial values on basis of the robot parameters.

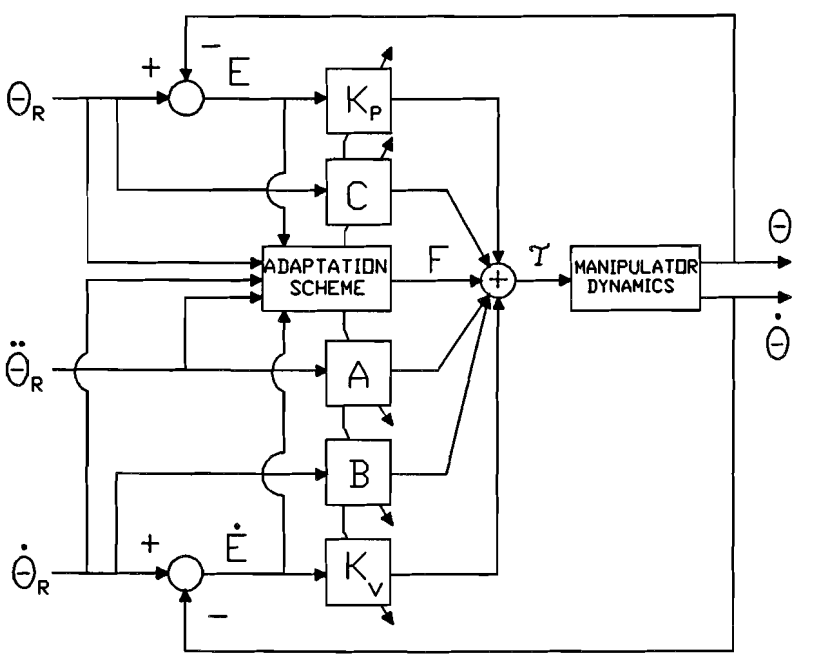


Figure 10: Adaptive joint control scheme

7. SIMULATION

In order to get insight in the performance of the different control laws a number of numerical simulations are carried out before implementation on the real RT-robot. This makes it possible to predict aspects such as stability, robustness and the maximum usable feedback gains. All simulations are performed in the software package PC-Matlab©

In order to compare the performance of the new control system relative to the old system we will consider the same reference trajectory as in [2]. Both parts consist of a skew-sine.

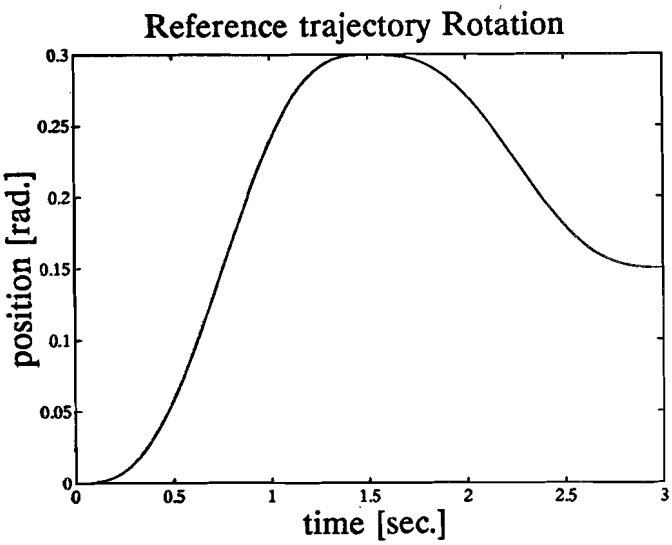


Figure 11

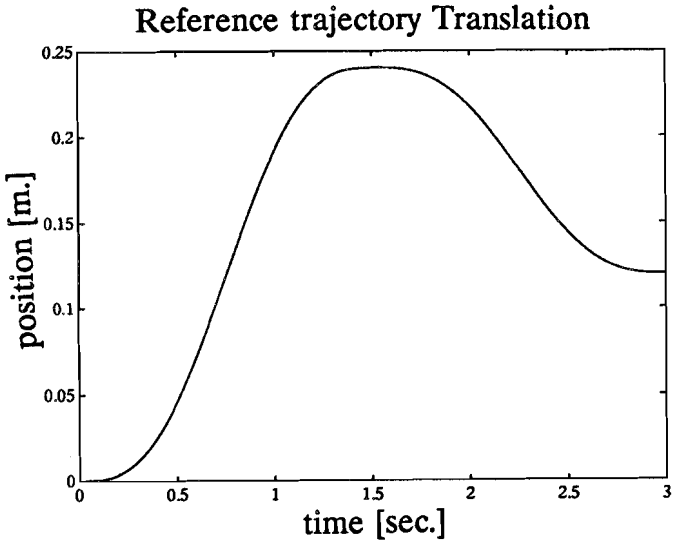


Figure 12

The next chapters give the results of the simulations. In all simulation the same reference trajectory, as given in figure 11 and 12, has to be tracked. No pay-load mass is present in the simulations. During all simulations the sampletime has been set to 5 [ms.], the same as used in implementation. At the rotational module only the **'output'** (the rotational position Φ of the turntable) is used for feedback. This is different from the situation described in [1] and [2]. In all simulations the feedback gains used, are made as high as possible without inducing instability. The choice of feedback gains is a trial and error process. Every time the maximum applicable feedback gains will be written above the corresponding figures.

Four control laws are regarded :
- PD feedback;
- PID feedback;
- Feedforward (and PD feedback);
- Adaptive control.

7.1. PD FEEDBACK CONTROL

In this case PD feedback control is regarded as a kind of reference, a standard to compare results of the other control laws. Figure 13 and 14 show the results of PD feedback control, with maximum feedback gains applied so the robot is behaving stable. It must be noted that the maximum usable feedback gains for the rotational module are clearly lower than those for the translation module. The rotational module is relative elastic and therefore starts to show unstable behaviour at lower feedback gains. Figure 13 shows a small vibration superimposed on top of the tracking error for the rotational module, in the second part of the movement. Again, this is caused by the difference in stiffness. (rotational module ± 2.9 * 10^5 [Nm/rad] , translation module ± 2.2 * 10^7 [N/m])

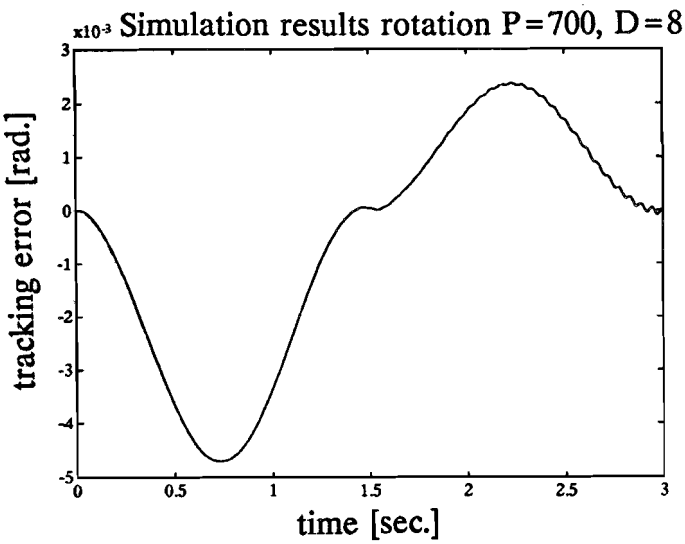


Figure 13

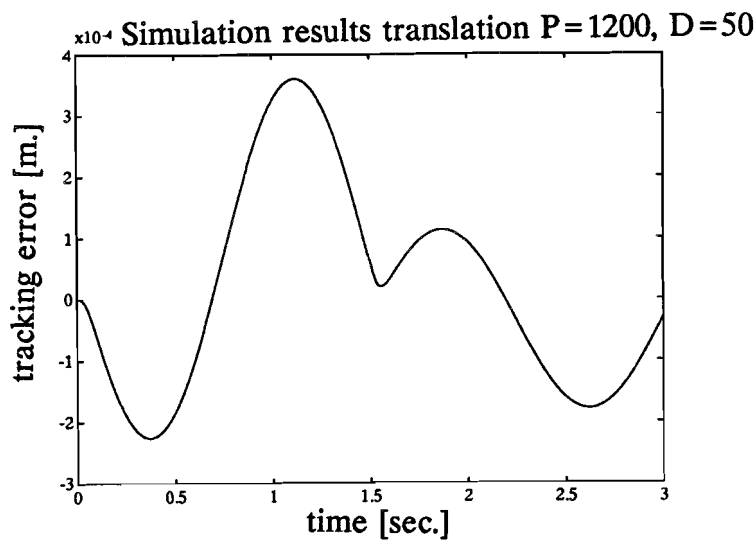


Figure 14

7.2. PID FEEDBACK CONTROL

Adding an Integral term to the control law results in a more stable behaviour of the robot. With an extra integral term the maximum usable proportional and differential feedback gains can be raised without stability problems. Figure 15 and 16 show the results when P.I.D. feedback is applied. The results show an improvement by about a factor two. Besides this, the small vibration, present on top of the tracking error of the rotational module in figure 13 has disappeared. Also, the shapes of the tracking errors have changed. This is because a different control law is used.

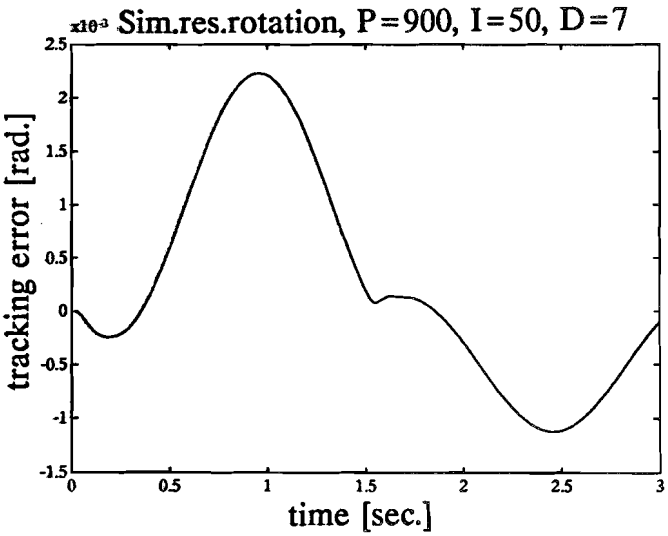


Figure 15

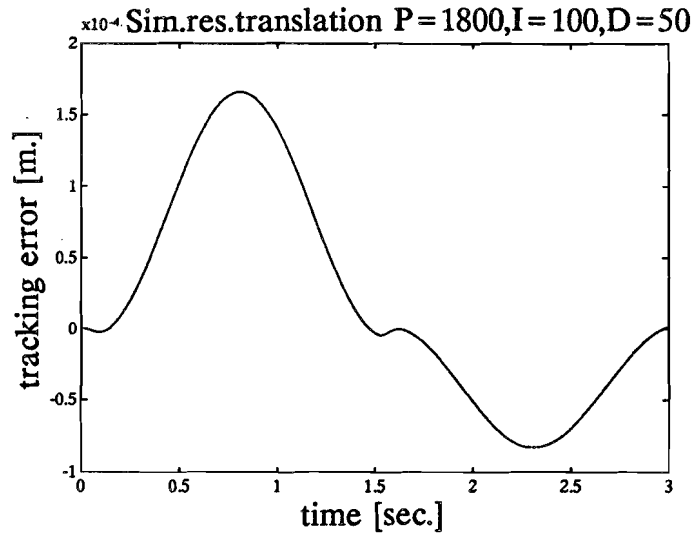


Figure 16

7.3. FEEDFORWARD CONTROL

7.3.1. THE 3-D.O.F. FEEDFORWARD MODEL

Voorkamp [2] designed a 3-D.O.F. [R2T1] **feedforward** model to be used in the 3-D.O.F. control model. It now will be explained why such a 'flexible feedforward' model cannot work when applied in the 'normal' way (as done by Voorkamp [2, Appendix C]). Figure 17 shows the lumped-mass representation of this model:

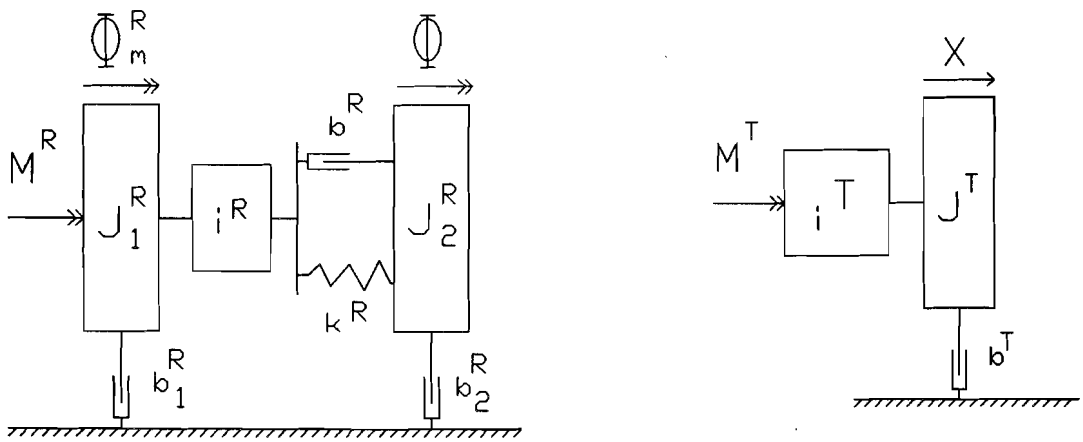


Figure 17: Lumped-mass repr. 3-D.O.F. model [R2T1]

According to Voorkamp the nominal control effort (for the rotation module) is calculated by substituting the desired trajectory (Φ and Φ_m) in the formula below:

$$M^R = J_1^R \ddot{\phi}_m^R + \left[\frac{b^R}{(i^R)^2} + b_1^R\right]\dot{\phi}_m^R - \frac{b^R}{i^R}\dot{\phi} + \frac{k^R}{(i^R)^2}\phi_m^R - \frac{k^R}{i^R}\phi \qquad (41)$$

However, at the 'free' D.O.F. Φ (=the turntable), the position, velocity and acceleration can't be **directly** forced. This is only possible for a D.O.F. where a motor is present. Only then the specific D.O.F. can be forced to follow the desired trajectory.
When the 3-D.O.F. [R2T1] feedforward control model is used in this way, the inertia J_2^R is exactly following the desired trajectory without any part, corresponding with J_2^R, being present in the total control effort. Hence, the spring k^R and the damping b^R and b_2^R <u>never</u> perform work.
The part of the lumped-mass model right of inertia J_1^R can be left out without <u>any</u> influence on the calculated nominal control effort.
In this way the actual representation of the lumped-mass model used <u>for the Rotation module</u> looks as follows:

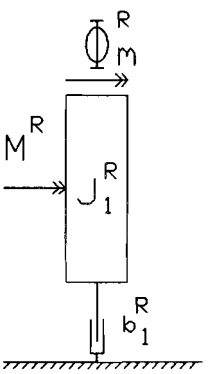
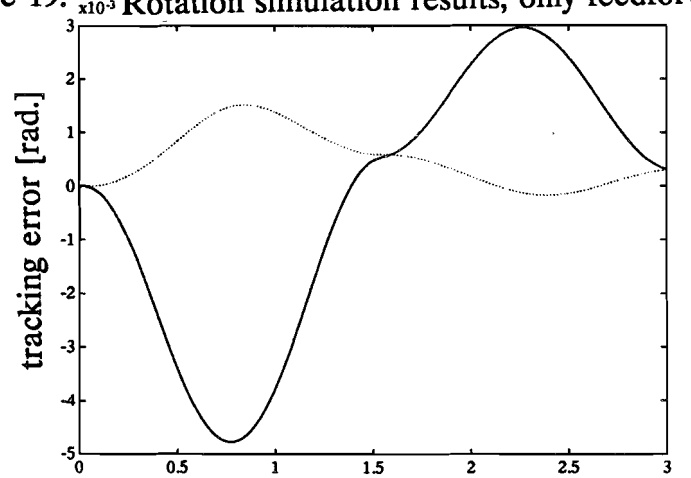


Figure 18: Actual lumped-mass representation

Therefore the 3-D.O.F. [R2T1] flexible feedforward model offers a bad description of the real robot. Even a simpler 2-D.O.F. [R1T1] rigid feedforward control model will perform better. This is illustrated by a simulation of the Rotation module where both models are tested and in which no feedback is applied. The tracking errors therefore give an indication on the performance of the feedforward control model. The results are given in figure 19.



.... 1 D.O.F. feedforward ---- 2 D.O.F. feedforward

Figure 19

As can be seen in figure 19 the tracking errors of the simpler, rigid 2-D.O.F. feedforward model are clearly smaller, so the 2-D.O.F. [R1T1] feedforward model offers a better description of the real robot.
In theory a 3 [R2T1] (or even higher) -D.O.F. feedforward control model could work better if it was possible to make 'good' estimates **for the future** of the position, velocity and acceleration at the 'free' degrees of freedom. In praxis however this will be difficult.

7.3.2 RESULTS OF 2-D.O.F. FEEDFORWARD CONTROL

Because of the reasons mentioned before, the 2-D.O.F. [R1T1] model is used as (feedforward) control model. The results of a simulation with 2-D.O.F. [R1T1] feedforward control are given in figure 20 and 21. Compared to the results of PD feedback control in figure 13 and 14 it shows that the use of a 'feedforward' control model <u>can</u> improve the results. The improvement isn't as spectacular however as reported by Voorkamp [2]. Voorkamp reports an improvement by a factor 10 while these simulations show an improvement by a factor two to three. Comparing the results of the feedforward control simulation in figure 20 and 21 with those of PID feedback control in figure 15 and 16, again shows the relative small contribution of adding a feedforward part to the control law to the improvement of the results. Compared with PID feedback control, Feedforward control only shows a marginal improvement of results for the Rotation module, and a clear worsening for the Translation module.

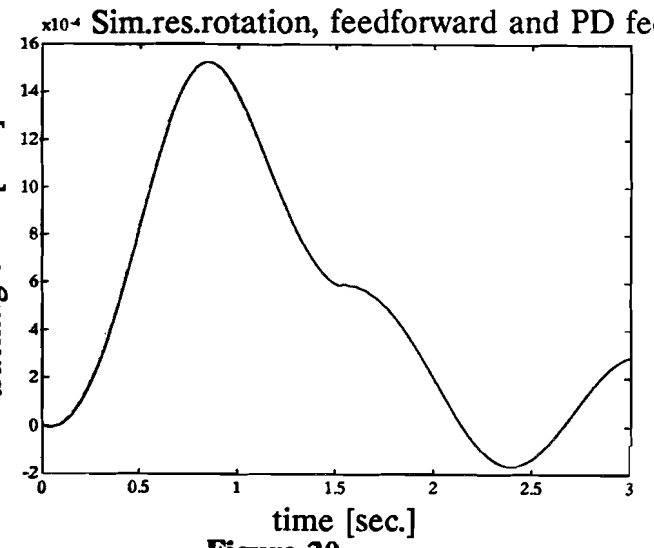


Figure 20

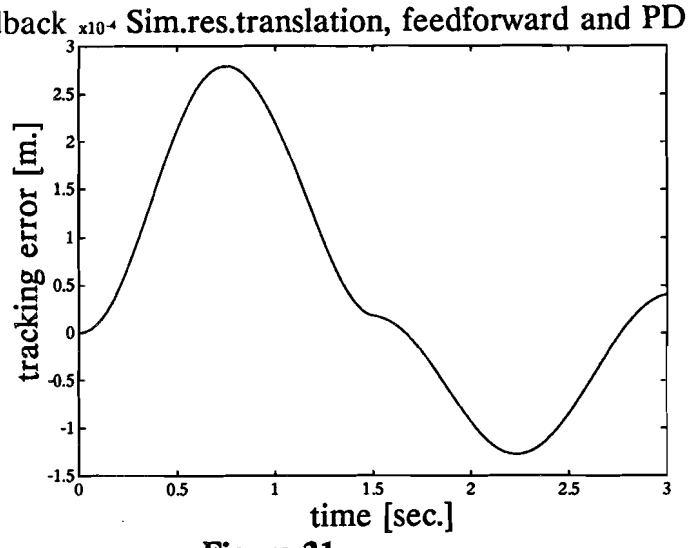


Figure 21

7.4. ADAPTIVE CONTROL

In the adaptive control simulation, the initial 'control' parameters are chosen the same as in the feedforward control simulation. This is different form Seraji's fourth consideration (page 30). The results of the adaptive control situation are given in figure 22 and 23. In this case adaptive control offers only a very small improvement of the results. This is in accordance with the results reported by Voorkamp [2].

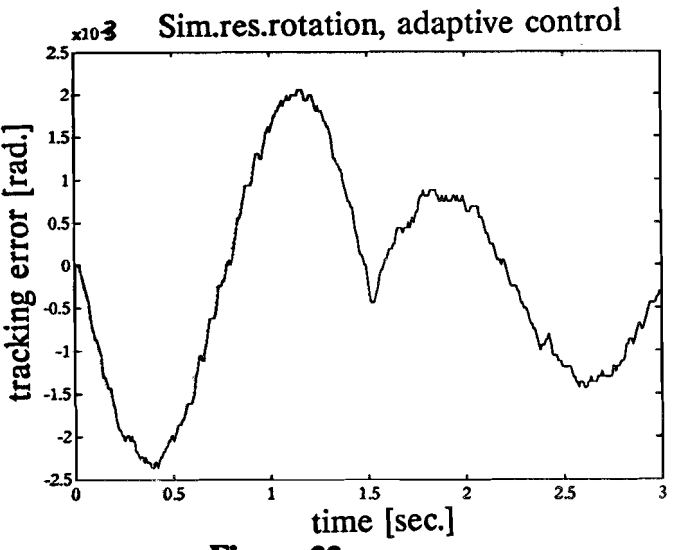


Figure 22

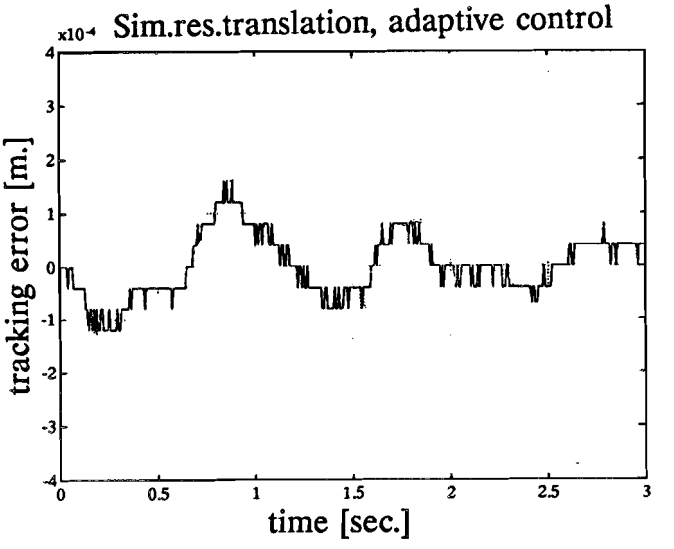


Figure 23

The horizontal pieces in figure 23 are caused by the limited accuracy, of the 'old' measurement system for the Translation module, being modelled. This is only done in this simulation for the Translation module in order to measure the result of the limited measurement accuracy on the adaptation process. The influence is neglectable.

In general the following remarks regarding adaptive control according to Seraji's method can be made from experimental results:

1- Stability isn't guaranteed for all magnitudes of the adaptation gains. The choice of 'too large' adaptation gains leads to instability, because of unbounded growth of the magnitude of the feedforward parameters or too abrupt change of the feedback amplification gains.

2- As opposed to Seraji's third consideration the magnitude of the adaptation gains can't be freely chosen '*reasonably large*' to ensure a fast speed of adaption. The magnitude of the adaption gains can't be chosen random because of remark 1.

3- Choosing the adaptation gains is a very time-consuming trial-and-error process. Many simulations have to be carried out in order to find '**usable**' adaptation gains.

4- Tuning the adaptation algorithm is difficult, hence it is difficult to find the maximum achievable performance of the adaptive control law.

8. IMPLEMENTATION

Now that we have satisfying results of the simulation we are ready to implement the control algorithms at the real RT-robot.
As mentioned before in chapter 5, the hardware of the computer platform provides capabilities to read-in positions and send-out motor voltages at the 'same' moment.
The control laws for the rotational and translation module will be implemented as mentioned in chapter 5.2..
In order to ensure a constant timing during the replay of the desired trajectory, an interrupt-timer is used. This timer counts down from a specified initial value and then issues an interrupt at reaching zero. This interrupt activates the corresponding 'interrupt-service routine' which contains the 'top-level' routine (Figure 5).
The top-level routine first issues a command to the hardware to latch the position and voltage values and then calls all the process subroutines (e.g. read out the 'latched' position, calculate the required control effort, convert to voltage) in turn.
To ensure that the top-level timer-interrupt has the highest priority in the system some other interrupt requests must be inhibited. See appendix 9 for details.
The timer is clocked at a constant interval, hence by varying the specified initial count-down value, the sample time can be varied. During implementation the sample time is held at 5 [ms.] This is the maximum achievable on a 386 SX 16 type of Personal Computer and with an adaptive control law. With a PD feedback control law a sampletime of 2 [ms.] can be achieved.
The reference trajectory used at the implementation is the same as used at the simulations (figure 11 and 12).

8.1. PD FEEDBACK CONTROL

Figure 24 and 25 show the results of the implemented PD feedback control with the same feedback gains as used in simulation.
It can be seen that both tracking errors show a reasonable large initial error, hence the real position of the RT-robot at the start of the replay doesn't correspond exactly with the desired position. Furthermore the implementation results show much more vibrations superimposed, due to unmodelled dynamics, stick-slip effects, etc., etc..

The order of magnitude of the tracking errors corresponds reasonable with the simulation results.

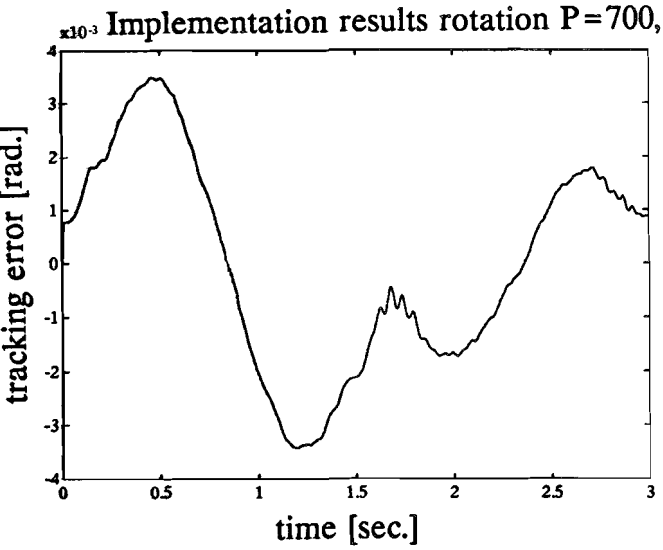


Figure 24

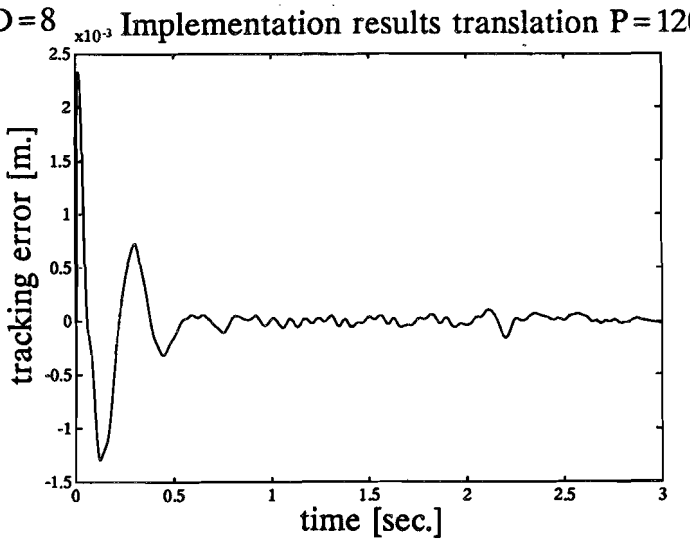


Figure 25

8.2. PID FEEDBACK CONTROL

Figure 26 and 27 show the results when PID feedback is applied. Again, the same situation as in the simulations, the results show an improvement by about a factor two. Also the maximum usable feedback gains are clearly higher than in the case of PD feedback.

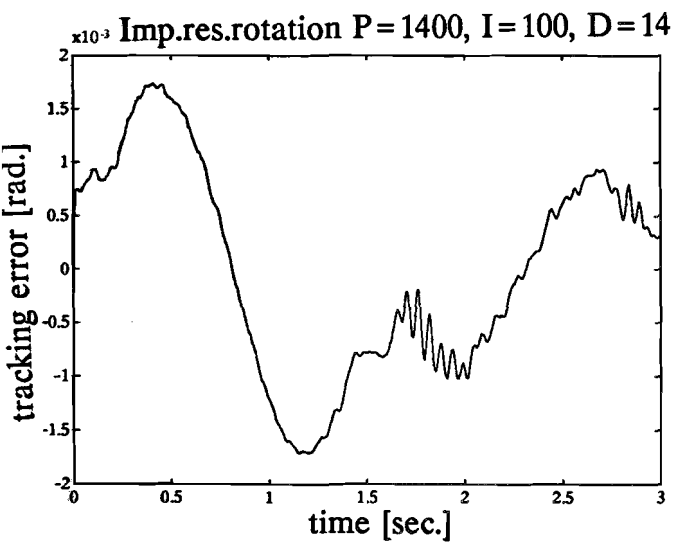


Figure 26

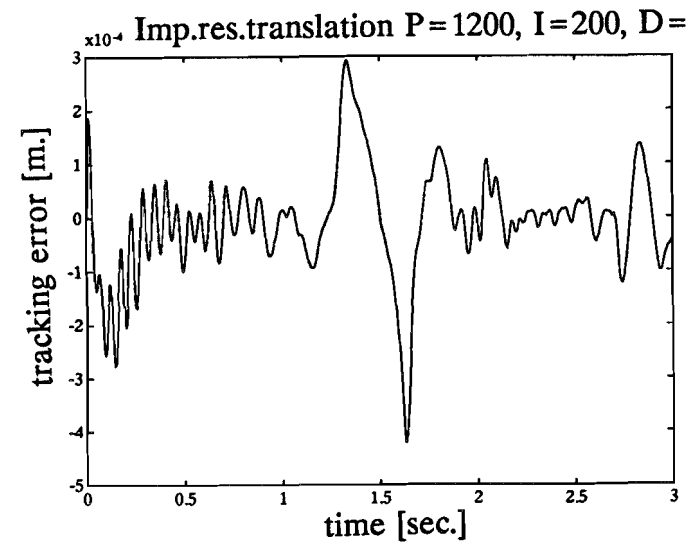


Figure 27

8.3. FEEDFORWARD CONTROL

Figure 28 and 29 show the results when feedforward (and feedback) control is applied. The results show a slight improvement compared with PD feedback control in figure 24 and 25, but if compared with the PID feedback control in figure 26 and 27 the improvement is not significant.

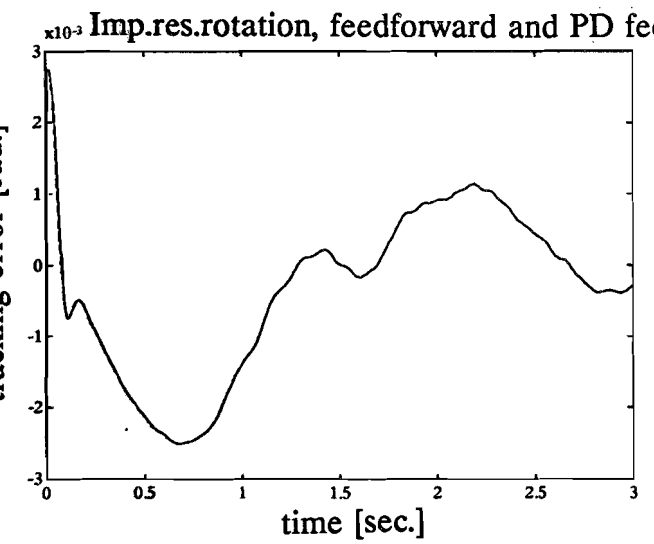


Figure 28

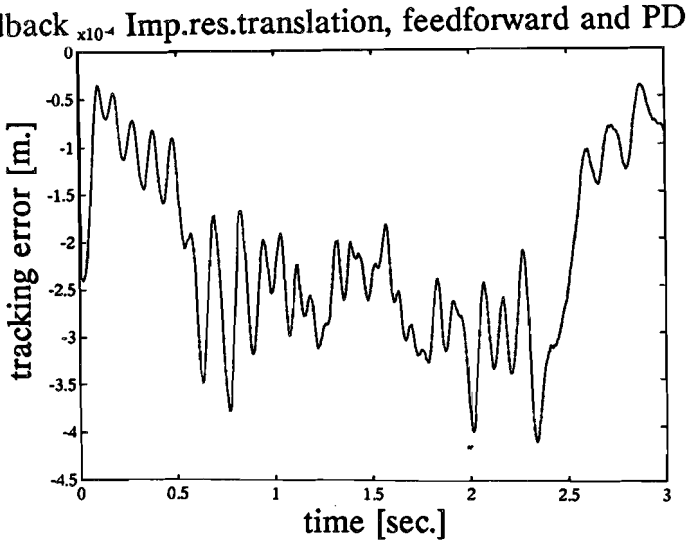


Figure 29

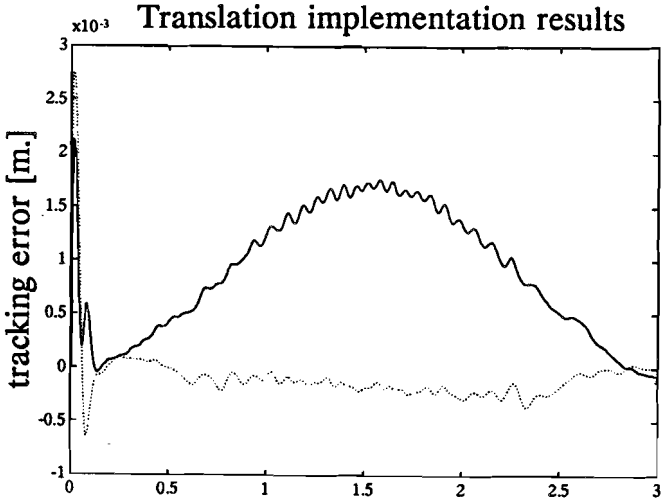
Again, the same as in simulation, the improvement isn't as evident as the results reported by Voorkamp [2].

Now, an explanation will be given for this difference in results.

The code used by Voorkamp to implement the (feedforward) control algorithm shows that in the case of PD feedback, a control signal is delivered where the input voltage of the DC-motors is linear in the tracking error E(t) and its derivative. Hence, the required control effort at the motor-axis in [Nm] was **non-linear** in the tracking error E(t) and its derivative. Only in the feedforward part of the control effort the **non-linear** characteristic of the DC-motors was taken into account.

The clear improvement reported, by the use of a feedforward part in the control effort is based on a misinterpretation.

In reality this clear improvement is only caused by the characteristic of the DC-motors being taken into account. This can be shown by a test in which two situations are regarded, one with modelled motor characteristic, and one without this. Figure 30 shows the results.



--- without motor model .... with motor model

Figure 30

The improvement is exactly in the order of magnitude as the improvement reported by Voorkamp and ascribed to the feedforward control. In reality, the improvement was caused by modelling the DC motor and **not** by adding a feedforward part to the implemented control law.

8.4. ADAPTIVE CONTROL

In testing the adaptive control law at the real RT-robot a different reference trajectory was used by Voorkamp [2]. In order to make a fair comparison the same reference trajectory will be used, with the first half being exactly the same as the reference trajectory used before. Again, all parts consist of skew sine waves. The new reference trajectory is given in the figure 31 and 32.

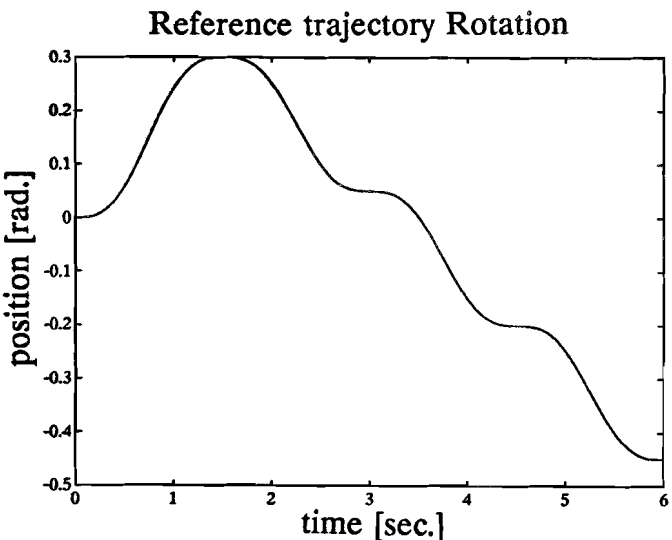


Figure 31

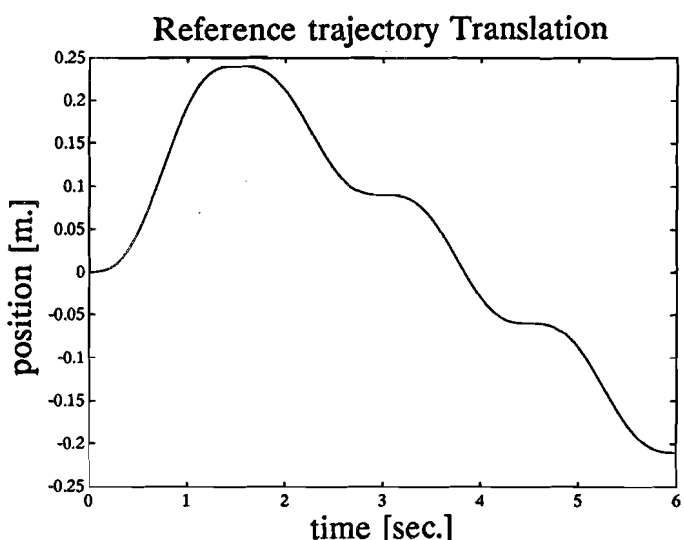


Figure 32

The initial 'control' parameters are the same as used in the feedforward control case. Figure 33 and 34 show the results of the adaptive control.

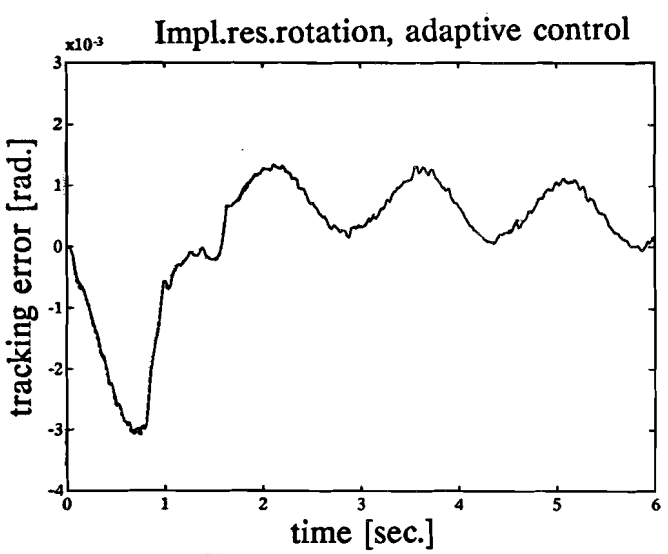


Figure 33

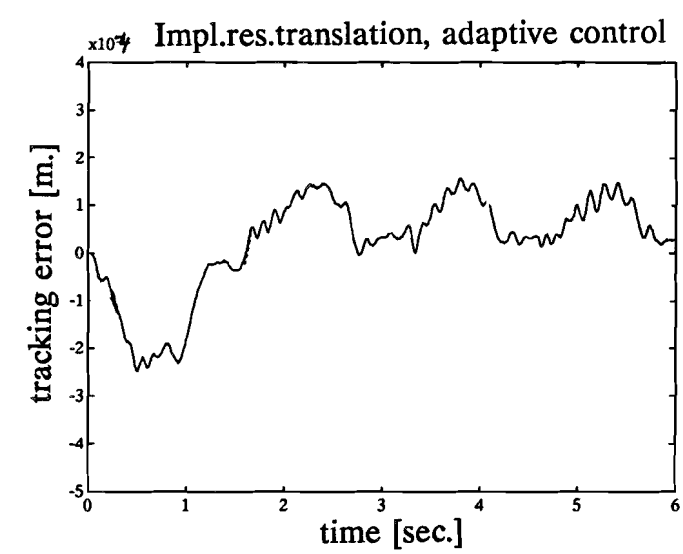


Figure 34

In the implementation, the results of adaptive control showed no improvements if compared with PID feedback control (see figure 26 and 27).
The remarks made for the simulation of an adaptive control law (page 36), hold for the implementation of such a control law.
During the replay the total set of 'control' parameters was adapted, hence non of the adaptation gains was equal to zero. This is the same situation as in simulation.
Voorkamp [2] only used an adaptation of the damping model parameters in the implementation.

The required computational effort causes the minimal sampletime to be 5 [ms.] for an adaptive control law on the present 386SX16 computer. This sampletime could be decreased by using a faster 486-50 computer or by adding so called co-processor boards to the system.

9. CONCLUSION AND ADVICE

- The use of a single processor computer as a basis for the computer-platform of the RT-robot requires a new approach at implementing the control program. With specific interface boards, which provide simultaneous 'latching' capabilities real-time behaviour can be attained.
  Synchronising can be done on interrupt basis and controls both latching and calling process-subroutines.

- Research on control algorithms can now be carried out effective, due to easy change and exchange of control laws in Turbo-Pascal©.

- Three non-adaptive (PD, PID and Feedforward) control laws and one adaptive MRAC) control law have been simulated and implemented on the new computer platform.
  From the three non-adaptive control laws the PID feedback control law offers the best results, both in simulation and implementation, while the improvements of Feedforward control are marginal.
  The adaptive control law, has serious problems regarding the tuning of the adaptation gains. This easily leads to instability, due to excessive adaptation of the 'control' parameters. Therefore much time was used, to find 'usable' adaptation gains, while aspects as '**maximum usable**' adaptation gains and '**maximum achievable performance**' couldn't be researched.

- It is necessary that all parts in the system have known behaviour and that their models are included in the control program when implementing a control algorithm on the real RT-robot.

- The required computational effort leads to an achievable sample time of 2 [ms.] for a 'simple' PD control algorithm and an achievable sampletime of 5 [ms.] for a computational intensive adaptive control algorithm. The use of a faster microprocessor (e.g. 486-50) or an additional co-processor board (e.g. Intel 860) could clearly decrease these sampletimes.

- Further research is required in :

    - Feedforward control (model matching, higher D.O.F. Feedforward models).

    - Adaptive control (stability aspects, 'tuning' adaptation gains, performance limits of adaptive control).

BIBLIOGRAPHY

[1] Martens, A.P.M.A.
Vergelijkingsstudie en implementatie van de trajectory-sturing van een RT-robot.
Eindhoven University of Technology, M.Sc. report WPA-0955, 1990.

[2] Voorkamp, R.J.
Studie naar de adaptieve trajectory sturing van een rotatie-translatie robot.
Eindhoven University of Technology, M.Sc. report WPA-1076, 1991.

[3] Oosterhout, J.J.A.H.
Modificatie van het besturingssysteem van de RT-robot.
Eindhoven University of Technology, Student research report WPA-1038, 1991.

[4] Seraji, H.
Design of adaptive joint controllers for robots.
In: Recent trends in Robotics p.251-260, Elsevier Science Publishers B.V., Amsterdam, 1986.

[5] Seraji, H.
Adaptive control of robot manipulators.
In: Proceedings IEEE International Conference on Robotics and Automation, p.565-571, San Francisco, 1986.

[6] Heeren, T.A.G.
ON CONTROL OF MANIPULATORS.
Eindhoven University of Technology, Graduation Report 1989.

[7] Weaver, L.E.
Reactor Dynamics and Control.
American elesevier publishing company, inc., New York, 1968.

[8] Lefschets, S.
STABILITY OF NONLINEAR CONTROL SYSTEMS.
Academic press, New York - London, 1965.

[9] Unbehauen, H.
Methods and Applications in Adaptive Control.
Springer-Verlag, Berlin 1980.

[10] Gupta, M.M.
Adaptive Methods for Control System Design.
IEEE Press, New York 1986.

[11] Kreffer, G.
Een ontwerp van een rotatiemodule en een studie naar een adaptieve regeling van de RT-robot.
Eindhoven University of Technology. M. Sc. report WPA-0575, 1988.

[12] van Bommel,L.V.M.
Ontwerp, produktie en de dynamische analyse van een lineaire actuator.
Eindhoven University of Technology. M. Sc. report WPA-0067, 1984.

[13] Bax, W.
Studie van het dynamisch model van een RT-robot.
Eindhoven University of Technology. M.Sc. report WPA-0787, 1989.

APPENDICES

APPENDIX 1: SPECIFICATIONS OF THE MEASUREMENT SYSTEMS

Each module of the RT-robot contains three measurement systems. At each motor-axis an optical encoder and a tachogenerator is mounted. The encoders measure the (rotational) position of the motor-axis, while the tachogenerators measure the motor-speed. In this case the tachogenerators aren't used because the motorspeed is calculated by numerical differentiation of the position signal. On the translating arm a linear optical encoder is mounted to measure the tangential position directly. Around the circumference of the Rotational module an optical flexible scale is mounted to measure the rotational position of the module.

Encoders

Type:	Heidenhain ROD 450
Number of line counts	1000 [counts/revolution]
Maximum rotational velocity	½π . 180,72 [rad./s.]
Maximum frequency IK 110	46 [kHz]
Single count of IK 110	interpolation 1 (adjustable)
Number of counts	1000 counts/revolution
Interpolation IK 110	25
Precision	2.51e-4 [rad] at motor-axis => 1.39e-6 [rad] at turntable

Translational linear encoder

Type:	Heidenhain LS 513
Distance of line counts	0.04 [mm.]
Number of line counts	2.5e4 [counts/m.]
Maximum velocity	1 [m./s.]
Maximum frequency IK 110	25 [kHz]
Single count of IK 110	interpolation 1 (adjustable)
Interpolation IK 110	25
Precision	1.6e-6 [m.]

Rotational flexible scale

Type:	Heidenhain LIDA 360
Distance of line counts	0.01 [mm.]
Number of line counts	20200 [counts/revolution]
Diameter turn-table	642.68 [mm.]
Maximum rotational velocity	½π [rad./s.]
Maximum frequency IK 110	5 kHz
Double count of IK 110	interpolation 2 (adjustable)
Interpolation IK 110	25
Number of counts	1010000 [counts/revolution]
Precision	6.22e-6 [rad.] (at turn-table)

APPENDIX 2: THE 11-D.O.F. [R6T5] LUMPED-MASS MODEL

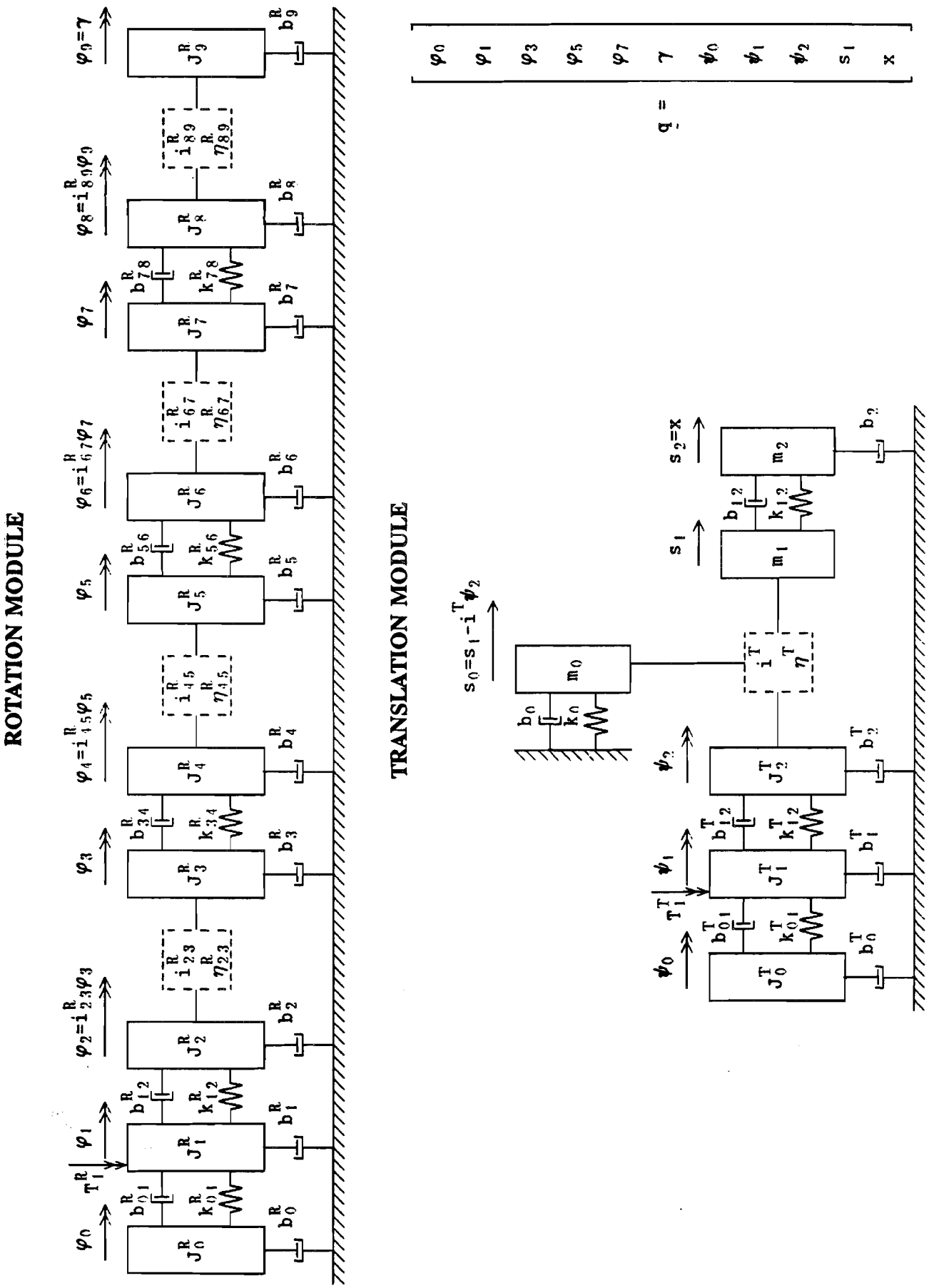


Figure 35 : 11-D.O.F. dynamic model of the RT-Robot

APPENDIX 3: THE 5-D.O.F. [R3T2] SIMULATION MODEL

This model contains 5 Degrees Of Freedom :

Φ_m^R	: rotation of the Rotation motor.
Φ_1	: rotation of the composed mass.
Φ	: rotation of the turntable.
Φ_m^T	: rotation of the Translation motor.
x	: position of the translating arm.

Figure 36 gives the lumped-mass representation of the 5-D.O.F. model.

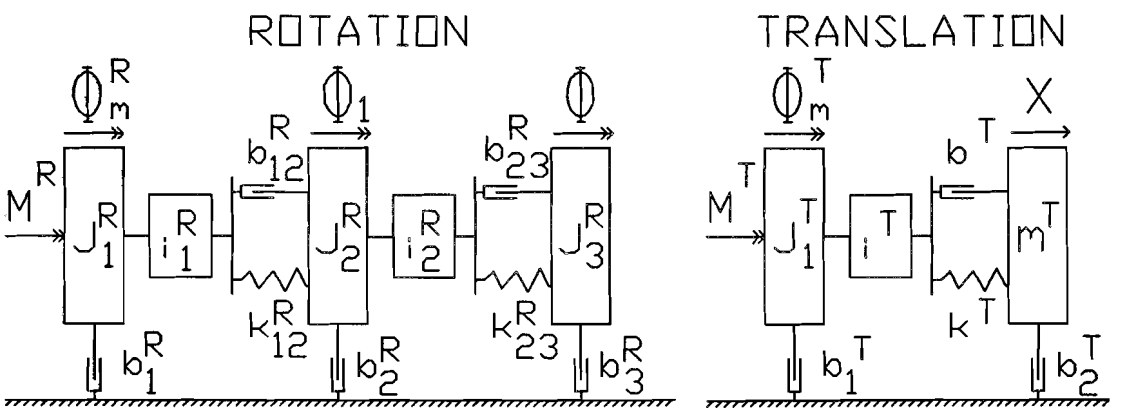


Figure 36: Lumped-mass representation 5-D.O.F. [R3T2] model.

The vector containing the Degrees Of Freedom is:

$$q = \left(\Phi_m^R \; \Phi_1 \; \Phi \; \Phi_m^T \; x \right)^T \tag{42}$$

The non-linear state-equations can be obtained by defining the state-space vector as follows:

$$\vec{x}^T = \begin{bmatrix} \vec{q} \\ \dot{\vec{q}} \end{bmatrix} \tag{43}$$

Hence, the equations of motion can be written as:

$$\dot{\vec{x}} = f(\vec{x}(t), \vec{u}(t), t)$$

$$\vec{y} = g(\vec{x}(t), \vec{u}(t), t)$$

(44)

This delivers 10 **non-linear** state-space equations:

$$\dot{x}_1 = \dot{\Phi}_m{}^R$$

$$\dot{x}_2 = \dot{\Phi}_1$$

$$\dot{x}_3 = \dot{\Phi}$$

$$\dot{x}_4 = \dot{\Phi}_m{}^T$$

$$\dot{x}_5 = \dot{x}$$

$$\dot{x}_6 = \frac{1}{J_1{}^R}\left[M^R - \left[b_1{}^R + \frac{b_{12}{}^R}{(i_1{}^R)^2} \right] \dot{\Phi}_m{}^R + \frac{b_{12}{}^R}{i_1{}^R}\dot{\Phi} + \frac{k_{12}{}^R}{i_1{}^R}\Phi_1 - \frac{k_{12}{}^R}{(i_1{}^R)^2}\phi_m{}^R \right]$$

$$\dot{x}_7 = \frac{1}{J_2{}^R}\left[-\left[b_2{}^R + b_{12}{}^R + \frac{b_{23}{}^R}{(i_2{}^R)^2} \right] \dot{\Phi}_1 + \frac{b_{12}{}^R}{i_1{}^R}\dot{\Phi}_m{}^R + \frac{b_{23}{}^R}{i_2{}^R}\dot{\Phi} - \left[k_{12}{}^R + \frac{k_{23}{}^R}{(i_2{}^R)^2} \right] \Phi_1 + \frac{k_{12}{}^R}{i_1{}^R}\Phi_m{}^R + \frac{k_{23}{}^R}{i_2{}^R}\Phi \right]$$

$$\dot{x}_8 = \frac{1}{J_3{}^R}\left[-\left[b_3{}^R + b_{23}{}^R \right] \dot{\Phi} + \frac{b_{23}{}^R}{i_2{}^R}\dot{\Phi}_1 - k_{23}{}^R\Phi + \frac{k_{23}{}^R}{i_2{}^R}\Phi_1 - \frac{\partial J_3{}^R}{\partial t}\dot{\Phi} - M_3{}^R \right]$$

$$\dot{x}_9 = \frac{1}{J_1{}^T}\left[M^T - \left[b_1{}^T + \frac{b^T}{(i_2{}^T)^2} \right] \dot{\Phi}_m{}^T + \frac{b^T}{i^T}\dot{x} + \frac{k^T}{i^T}x - \frac{k^T}{(i^T)^2}\Phi_m{}^T - M_{wl}{}^T \right]$$

$$\dot{x}_{10} = \frac{1}{m^T}\left[\frac{1}{2}\frac{\partial J_3{}^R}{\partial x}\dot{\Phi}^2 - b_2{}^T\dot{x} - k^T x + \frac{k^T}{i^T}\Phi_m{}^T - b^T\dot{x} + \frac{b^T}{i^T}\dot{\Phi}_m{}^T - F_w{}^T \right]$$

(45)

The parameters in the formulas **(45)** above are given by:

m_l	= pay-load mass.	[kg]
M^R	= external torque on the Rotation module.	[Nm]
M^T	= external torque on the Translation module.	[Nm]
J_1^R	= 1.33e-3	[kgm^2]
J_2^R	= 1.18e-2	[kgm^2]
J_3^R	= $(83.06+m_l)\,x^2 + (1.48m_l-19.09)\,x + 32.56 + 0.55m_l$	[kgm^2]
k_{12}^R	= 11925.31	[Nm/rad]
k_{23}^R	= 310512.50	[Nm/rad]
b_{12}^R	= 8.7e-6	[Nms/rad]
b_{23}^R	= 2.6e-5	[Nms/rad]
b_1^R	= 6.21e-2	[Nms/rad]
b_2^R	= 1.77e-1	[Nms/rad]
b_3^R	= 1.59e-1	[Nms/rad]
i_1^R	= 8.47	[-]
i_2^R	= 21.34	[-]

J_1^T	= 1.71e-3	[kgm^2]
m^T	= $83.07+m_l$	[kg]
k^T	= $(2.72e15+1.28e15\,x)/(1.22e8+4.35e7\,x)$	[N/m]
b^T	= 0.018	[Ns/m]
b_1^T	= 0.020	[Nms/rad]
b_2^T	= 120	[Ns/m]
i^T	= 251.33	[rad/m]

APPENDIX 4: THE 4-D.O.F. [R2T2] CONTROL MODEL

This model contains 4 Degrees Of Freedom :

- Φ_m^R : rotation of the Rotation motor.
- Φ : rotation of the turntable.
- Φ_m^T : rotation of the Translation motor.
- x : position of the translating arm.

Figure 37 gives the lumped-mass representation of the 4-D.O.F. model.

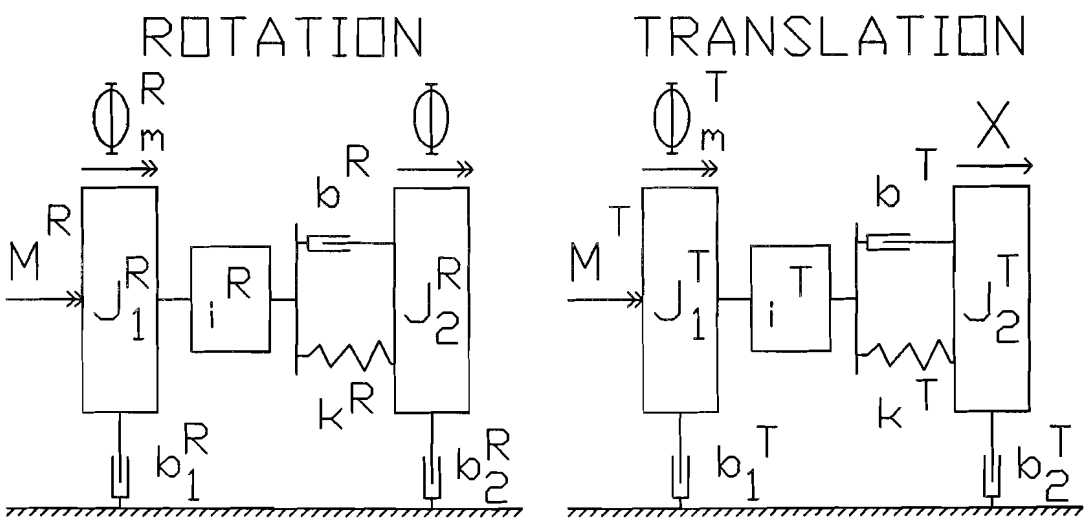


Figure 37: Lumped-mass representation 4-D.O.F. [R2T2] model.

The equations of motion can be derived using Lagrange

$$\frac{d}{dt}\left(E_{kin_{\dot{q}}}\right)-E_{kin_q}+E_{pot_q}=Q^* \qquad (46)$$

First the vector q̲ containing the Degrees Of Freedom is defined as

$$\underline{q}^T = \left[\ \varphi_m{}^R\ \varphi\ \varphi_m{}^T\ x\ \right] \qquad (47)$$

The kinetic energy is given by

$$E_{kin}=\frac{1}{2}J_1{}^R\left(\dot{\varphi}_m{}^R\right)^2+\frac{1}{2}J_2{}^R(\dot{\varphi})^2+\frac{1}{2}J_1{}^T\left(\dot{\varphi}_m{}^T\right)^2+\frac{1}{2}J_2{}^T(\dot{x})^2 \tag{48}$$

While the potential energy is given by

$$E_{pot}=\frac{1}{2}k^R\left(\varphi^2-2\varphi\frac{\varphi_m{}^R}{i^R}+\frac{\left(\varphi_m{}^R\right)^2}{\left(i^R\right)^2}\right)+\frac{1}{2}k^T\left(x^2-2x\frac{\varphi_m{}^T}{i^T}+\frac{\left(\varphi_m{}^T\right)^2}{\left(i^T\right)^2}\right) \tag{49}$$

Partial differentiation of the kinetic energy to the terms of $\underline{q}$ and their derivatives delivers

$$\frac{\partial E_{kin}}{\partial\dot{\varphi}_m{}^R}=J_1{}^R\dot{\varphi}_m{}^R\rightarrow\frac{d}{dt}\left(\frac{\partial E_{kin}}{\partial\dot{\varphi}_m{}^R}\right)=J_1{}^R\ddot{\varphi}_m{}^R\quad,\quad\frac{\partial E_{kin}}{\partial\varphi_m{}^R}=0$$

$$\frac{\partial E_{kin}}{\partial\dot{\varphi}}=J_2{}^R\dot{\varphi}\rightarrow\frac{d}{dt}\left(\frac{\partial E_{kin}}{\partial\dot{\varphi}}\right)=J_2{}^R\ddot{\varphi}+\frac{dJ_2{}^R}{dt}\dot{\varphi}=J_2{}^R\ddot{\varphi}+\frac{dJ_2{}^R}{dx}\dot{x}\dot{\varphi}\quad,\quad\frac{\partial E_{kin}}{\partial\varphi}=0$$

$$\frac{\partial E_{kin}}{\partial\dot{\varphi}_m{}^T}=J_1{}^T\dot{\varphi}_m{}^T\rightarrow\frac{d}{dt}\left(\frac{\partial E_{kin}}{\partial\dot{\varphi}_m{}^T}\right)=J_1{}^T\ddot{\varphi}_m{}^T\quad,\quad\frac{\partial E_{kin}}{\partial\varphi_m{}^T}=0$$

$$\frac{\partial E_{kin}}{\partial\dot{x}}=J_2{}^T\dot{x}\rightarrow\frac{d}{dt}\left(\frac{\partial E_{kin}}{\partial\dot{x}}\right)=J_2{}^T\ddot{x}\quad,\quad\frac{\partial E_{kin}}{\partial x}=\frac{1}{2}\frac{dJ_2{}^R}{dx}(\dot{\varphi})^2 \tag{50}$$

Differentiation of the potential energy to the terms of $\underline{q}$ delivers

$$\frac{\partial E_{pot}}{\partial\varphi_m{}^R}=k^R\left(\frac{\varphi_m{}^R}{\left(i^R\right)^2}\right)-k^R\frac{\varphi}{i^R}$$

$$\frac{\partial E_{pot}}{\partial\varphi}=k^R\varphi-k^R\frac{\varphi_m{}^R}{i^R}$$

$$\frac{\partial E_{pot}}{\partial \varphi_m{}^T} = k^T \left(\frac{\varphi_m{}^T}{(i^T)^2} \right) - k^T \frac{x}{i^T}$$

$$\frac{\partial E_{pot}}{\partial x} = k^T x - k^T \frac{\varphi_m{}^T}{i^T}$$

(51)

The virtual work is defined by

$$\delta W = \left[M^R - b_1{}^R \dot{\varphi}_m{}^R + \frac{b^R}{i^R} \left(\dot{\varphi} - \left(\frac{\dot{\varphi}_m{}^R}{i^R} \right) \right) \right] \delta \varphi_m{}^R + \left[-b_2{}^R \dot{\varphi} - b^R \left(\dot{\varphi} - \left(\frac{\dot{\varphi}_m{}^R}{i^R} \right) \right) \right] \delta \varphi +$$

$$+ \left[M_T - b_1{}^T \dot{\varphi}_m{}^T + \frac{b^T}{i^T} \left(\dot{x} - \left(\frac{\dot{\varphi}_m{}^T}{i^T} \right) \right) \right] \delta \varphi_m{}^T + \left[-b_2{}^T \dot{x} - b^T \left(\dot{x} - \left(\frac{\dot{\varphi}_m{}^T}{i^T} \right) \right) \right] \delta x$$

(52)

Substituting these terms **(50,51,52)** in the equation of Lagrange **(46)** delivers the equations of motion for the 4-D.O.F. [R2T2] model **(53)**. (See next page)

The parameters of the 4-D.O.F. model **(53)** are given by:

b^R	= 2.17e-5	[Nms/rad]
k^R	= 2.94e5	[Nm/rad]
$b_1{}^R$	= 6.46e-2	[Nms/rad]
$b_2{}^R$	= 0.45	[Nms/rad]
$J_1{}^R$	= 1.48e-3	[kgm^2]
$J_2{}^R$	= $(83.06+m_l)\ x^2 + (1.48m_l-19.09)\ x+33.10+0.55m_l$	[kgm^2]
i^R	= 180.72	[-]
b^T	= 0.018	[Ns/m]
k^T	= $(2.72e15+1.28\ x)/(1.22e8+4.35e7\ x)$	[N/m]
$b_1{}^T$	= 0.020	[Nms/rad]
$b_2{}^T$	= 120	[Ns/m]
$J_1{}^T$	= 1.71e-3	[kgm^2]
$J_2{}^T$	= $83.07+m_l$	[kg]
i^T	= 251.33	[m/rad]
m_l	= pay-load mass.	[kg.]

Equations of motion (53) for the 4.D.O.F. [R2T2] model.

ROTATION MODULE

$$M^R = J_1{}^R * \ddot{\varphi}_m{}^R + \left[\frac{b^R}{(i^R)^2} + b_1{}^R\right] * \dot{\varphi}_m{}^R - \left[\frac{b^R}{i^R}\right] * \dot{\varphi} + \left[\frac{k^R}{(i^R)^2}\right] * \varphi_m{}^R - \left[\frac{k^R}{i^R}\right] * \varphi + M_{w1}{}^R$$

$$0 = J_2{}^R * \ddot{\varphi} + \frac{\partial J_2{}^R}{\partial t} * \dot{\varphi} + \left[b^R + b_2{}^R\right] * \dot{\varphi} - \left[\frac{b^R}{i^R}\right] * \dot{\varphi}_m{}^R - \left[\frac{k^R}{i^R}\right] * \varphi_m{}^R + k^R * \varphi + M_{w2}{}^R$$

TRANSLATION MODULE

$$M^T = J_1{}^T * \ddot{\varphi}_m{}^T + \left[\frac{b^T}{(i^T)^2} + b_1{}^T\right] * \dot{\varphi}_m{}^T - \left[\frac{b^T}{(i^T)}\right] * \dot{x} + \left[\frac{k^T}{(i^T)^2}\right] * \varphi_m{}^T - \left[\frac{k^T}{i^T}\right] * x + M_{w1}{}^T$$

$$0 = J_2{}^T * \ddot{x} - \frac{1}{2} * \frac{\partial J_2{}^R}{\partial x} * \dot{\varphi}^2 + \left[b^T + b_2{}^T\right] * \dot{x} - \left[\frac{b^T}{i^T}\right] * \dot{\varphi}_m{}^T - \left[\frac{k^T}{i^T}\right] * \varphi_m{}^T + k^T * x + M_{w2}{}^T$$

(53)

APPENDIX 5: THE 6-D.O.F. [R3T3] SIMULATION MODEL

This model contains 6 Degrees Of Freedom :

- Φ_m^R : rotation of the Rotation motor.
- Φ_1 : rotation of the composed inertia.
- Φ : rotation of the turntable.
- Φ_m^T : rotation of the Translation motor.
- Φ_2 : rotation of the composed inertia.
- x : position of the translating arm.

Figure 38 gives the lumped-mass representation of the 6-D.O.F. model.

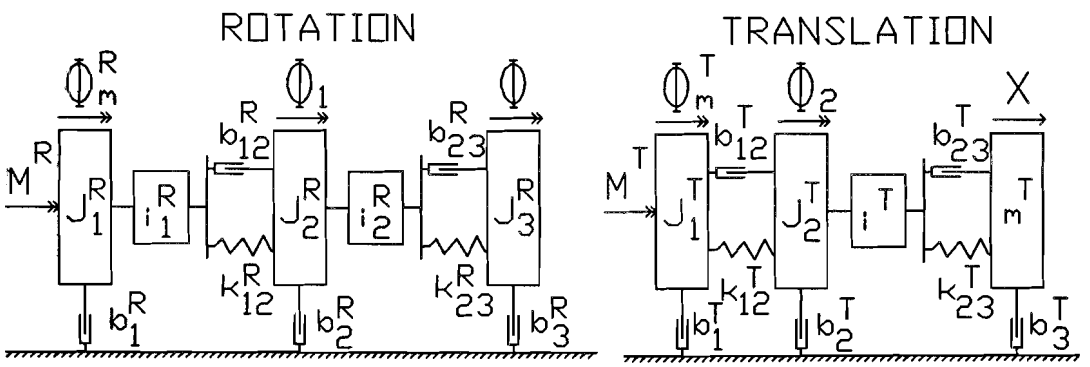


Figure 38: Lumped-mass representation 6-D.O.F. [R3T3] model.

To derive the equations of motion, first the $\underline{q}$ vector is defined

$$\underline{q}^T = \left[\varphi_m^R \; \varphi_1 \; \varphi \; \varphi_m^T \; \varphi_2 \; x \right] \tag{54}$$

The kinetic energy is given by

$$E_{kin} = \frac{1}{2} J_1^R (\dot{\varphi}_m^R)^2 + \frac{1}{2} J_2^R (\dot{\varphi}_1)^2 + \frac{1}{2} J_3^R (\dot{\varphi})^2 + \frac{1}{2} J_1^T (\dot{\varphi}_m^T)^2 + \frac{1}{2} J_2^T (\dot{\varphi}_2)^2 + \frac{1}{2} m^T (\dot{x})^2 \tag{55}$$

Partial differentiation of the kinetic energy to the terms of $\underline{q}$ and their derivatives delivers

$$\frac{\partial E_{kin}}{\partial \dot{\varphi}_m{}^R} = J_1{}^R \dot{\varphi}_m{}^R \rightarrow \frac{d}{dt}\left(\frac{\partial E_{kin}}{\partial \dot{\varphi}_m{}^R}\right) = J_1{}^R \ddot{\varphi}_m{}^R \quad , \quad \frac{\partial E_{kin}}{\partial \varphi_m{}^R} = 0$$

$$\frac{\partial E_{kin}}{\partial \dot{\varphi}_1} = J_2{}^R \dot{\varphi}_1 \rightarrow \frac{d}{dt}\left(\frac{\partial E_{kin}}{\partial \dot{\varphi}_1}\right) = J_2{}^R \ddot{\varphi}_1 \quad , \quad \frac{\partial E_{kin}}{\partial \varphi_1} = 0$$

$$\frac{\partial E_{kin}}{\partial \dot{\varphi}} = J_3{}^R \dot{\varphi} \rightarrow \frac{d}{dt}\left(\frac{\partial E_{kin}}{\partial \dot{\varphi}}\right) = J_3{}^R \ddot{\varphi} + \frac{dJ_3{}^R}{dt}\dot{\varphi} = J_3{}^R \ddot{\varphi} + \frac{dJ_3{}^R}{dx}\dot{x}\dot{\varphi} \quad , \quad \frac{\partial E_{kin}}{\partial \varphi} = 0$$

$$\frac{\partial E_{kin}}{\partial \dot{\varphi}_m{}^T} = J_1{}^T \dot{\varphi}_m{}^T \rightarrow \frac{d}{dt}\left(\frac{\partial E_{kin}}{\partial \dot{\varphi}_m{}^T}\right) = J_1{}^T \ddot{\varphi}_m{}^T \quad , \quad \frac{\partial E_{kin}}{\partial \varphi_m{}^T} = 0$$

$$\frac{\partial E_{kin}}{\partial \dot{\varphi}_2} = J_2{}^T \dot{\varphi}_2 \rightarrow \frac{d}{dt}\left(\frac{\partial E_{kin}}{\partial \dot{\varphi}_2}\right) = J_2{}^T \ddot{\varphi}_2 \quad , \quad \frac{\partial E_{kin}}{\partial \varphi_2} = 0$$

$$\frac{\partial E_{kin}}{\partial \dot{x}} = m^T \dot{x} \rightarrow \frac{d}{dt}\left(\frac{\partial E_{kin}}{\partial \dot{x}}\right) = m^T \ddot{x} \quad , \quad \frac{\partial E_{kin}}{\partial x} = \frac{1}{2}\frac{dJ_3{}^R}{dx}(\dot{\varphi})^2 \qquad \textbf{(56)}$$

Differentiation of the potential energy to the terms of $\underline{q}$ delivers

$$\frac{\partial E_{pot}}{\partial \varphi_m{}^R} = k_{12}{}^R\left(\frac{\varphi_m{}^R}{(i_1{}^R)^2}\right) - k_{12}{}^R\left(\frac{\varphi_1}{i_1{}^R}\right)$$

$$\frac{\partial E_{pot}}{\partial \varphi_1} = k_{12}{}^R \varphi_1 - k_{12}{}^R\left(\frac{\varphi_m{}^R}{i_1{}^R}\right) + k_{23}{}^R\left(\frac{\varphi_1}{(i_2{}^R)^2}\right) - k_{23}{}^R\left(\frac{\varphi}{i_2{}^R}\right)$$

$$\frac{\partial E_{pot}}{\partial \varphi} = k_{23}{}^R \varphi - k_{23}{}^R \frac{\varphi_1}{i_2{}^R}$$

$$\frac{\partial E_{pot}}{\partial \varphi_m{}^T} = -k_{12}{}^T \varphi_2 + k_{12}{}^T \varphi_m{}^T$$

$$\frac{\partial E_{pot}}{\partial \varphi_2} = k_{12}{}^T \varphi_2 - k_{12}{}^T \varphi_m{}^T + k_{23}{}^T \left(\frac{\varphi_2}{(i^T)^2} \right) - k_{23}{}^T \frac{x}{i^T}$$

$$\frac{\partial E_{pot}}{\partial x} = k_{23}{}^T x - k_{23}{}^T \left(\frac{\varphi_2}{i^T} \right)$$

(57)

The virtual work is defined by

$$\partial W = \left[M^R - b_1{}^R \dot{\varphi}_m{}^R + \frac{b_{12}{}^R}{i_1{}^R} \left(\dot{\varphi}_1 - \left(\frac{\dot{\varphi}_m{}^R}{i_1{}^R} \right) \right) \right] \delta \varphi_m{}^R + \ldots$$

$$\ldots + \left[-b_2{}^R \dot{\varphi}_1 - b_{12}{}^R \left(\dot{\varphi}_1 - \left(\frac{\dot{\varphi}_m{}^R}{i_1{}^R} \right) \right) + \frac{b_{23}{}^R}{i_2{}^R} \left(\dot{\varphi} - \left(\frac{\dot{\varphi}_1}{i_2{}^R} \right) \right) \right] \delta \varphi_1 + \ldots$$

$$\ldots + \left[-b_3{}^R \dot{\varphi} - b_{23}{}^R \left(\dot{\varphi} - \left(\frac{\dot{\varphi}_1}{i_2{}^R} \right) \right) \right] \delta \varphi + \ldots$$

$$\ldots + \left[M^T - b_1{}^T \dot{\varphi}_m{}^T + b_{12}{}^T \left(\dot{\varphi}_2 - \dot{\varphi}_m{}^T \right) \right] \delta \varphi_m{}^T + \ldots$$

$$\ldots + \left[-b_2{}^T \dot{\varphi}_2 - b_{12}{}^T \left(\dot{\varphi}_2 - \dot{\varphi}_m{}^T \right) + \frac{b_{23}{}^T}{i^T} \left(\dot{x} - \left(\frac{\dot{\varphi}_2}{i^T} \right) \right) \right] \delta \varphi_2 + \ldots$$

$$\ldots + \left[-b_3{}^T \dot{x} - b_{23}{}^T \left(\dot{x} - \left(\frac{\dot{\varphi}_2}{i^T} \right) \right) \right] \delta x$$

(58)

Substituting these terms **(56,57,58)** in the equation of Lagrange delivers the equations of motion of the 6 D.O.F. model **(60)** :

$$q^T = \left[\varphi_m{}^R \; \varphi_1 \; \varphi \; \varphi_m{}^T \; \varphi_2 \; x \right]$$

(59)

ROTATION MODULE

$$M^R = J_1{}^R * \ddot{\varphi}_m{}^R + \left[\frac{b_{12}{}^R}{(i_1{}^R)^2} + b_1{}^R\right] * \dot{\varphi}_m{}^R - \left[\frac{b_{12}{}^R}{i_1{}^R}\right] * \dot{\varphi}_1 - \left[\frac{k_{12}{}^R}{i_1{}^R}\right] * \varphi_1 + \left[\frac{k_{12}{}^R}{(i_1{}^R)^2}\right] * \varphi_m{}^R + M_{w1}{}^R$$

$$0 = J_2{}^R * \ddot{\varphi}_1 + \left[b_2{}^R + b_{12}{}^R + \frac{b_{23}{}^R}{(i_2{}^R)^2}\right] * \dot{\varphi}_1 - \left[\frac{b_{12}{}^R}{i_1{}^R}\right] * \dot{\varphi}_m{}^R - \left[\frac{b_{23}{}^R}{i_2{}^R}\right] * \dot{\varphi} + \left[k_{12}{}^R + \frac{k_{23}{}^R}{(i_2{}^R)^2}\right] * \dot{\varphi}_1 - \left[\frac{k_{12}{}^R}{i_1{}^R}\right] * \varphi_m{}^R - \left[\frac{k_{23}{}^R}{i_2{}^R}\right] * \varphi + M_{w2}{}^R$$

$$0 = J_3{}^R * \ddot{\varphi} + \frac{\partial J_3}{\partial t} * \dot{\varphi} + \left[b_3{}^R + b_{23}{}^R\right] * \dot{\varphi} - \left[\frac{b_{23}{}^R}{i_2{}^R}\right] * \dot{\varphi}_1 + k_{23}{}^R * \varphi - \left[\frac{k_{23}{}^R}{i_2{}^R}\right] * \varphi_1 + M_{w3}{}^R$$

TRANSLATION MODULE

$$M^T = J_1{}^T * \ddot{\varphi}_m{}^T + \left[b_{12}{}^T + b_1{}^T\right] * \dot{\varphi}_m{}^T - b_{12}{}^T * \dot{\varphi}_2 - k_{12}{}^T * \varphi_2 + k_{12}{}^T * \varphi_m{}^T + M_{w1}{}^T$$

$$0 = J_2{}^T * \ddot{\varphi}_2 + \left[b_2{}^T + b_{12}{}^T + \frac{b_{23}{}^T}{(i^T)^2}\right] * \dot{\varphi}_2 - b_{12}{}^T * \dot{\varphi}_m{}^T - \frac{b_{23}{}^T}{i^T} * \dot{x} + \left[k_{12}{}^T + \frac{k_{23}{}^T}{(i^T)^2}\right] * \varphi_2 - k_{12}{}^T * \varphi_m{}^T - \left[\frac{k_{23}{}^T}{i^T}\right] * x + M_{w2}{}^T$$

$$0 = m^T * \ddot{x} - \frac{1}{2}\frac{\partial J_3{}^R}{\partial x} * \dot{\varphi}^2 + \left[b_3{}^T + b_{23}{}^T\right] * \dot{x} - \left[\frac{b_{23}{}^T}{i^T}\right] * \dot{\varphi}_2 + k_{23}{}^T * x - \left[\frac{k_{23}{}^T}{i^T}\right] * \varphi_2 + M_{w3}{}^T$$

(60)

The parameters of the 6-D.O.F. model **(60)** are given by:

b_1^R	= 6.21e-2	[Nms/rad]
b_2^R	= 1.77e-1	[Nms/rad]
b_3^R	= 1.59e-1	[Nms/rad]
b_{12}^R	= 8.70e-6	[Nms/rad]
b_{23}^R	= 2.6e-5	[Nms/rad]
k_{12}^R	= 11925.31	[Nm/rad]
k_{23}^R	= 310512.50	[Nm/rad]
J_1^R	= 1.33e-3	[kgm^2]
J_2^R	= 1.18e-2	[kgm^2]
J_3^R	= $(83.06+m_l)x^2+(1.48m_l+19.09)x+32.56+0.55m_l$	[kgm^2]
i_1^R	= 8.47	[-]
i_2^R	= 21.34	[-]
b_{12}^T	= 8e-7	[Ns/m]
b_{23}^T	= 0.05	[Ns/m]
k_{12}^T	= 1.46e3+668 x	[N/m]
k_{23}^T	= 5.13e7+5.2e-4 x	[N/m]
b_1^T	= 4.29e-3	[Nms/rad]
b_2^T	= 1.6e-2	[Ns/m]
b_3^T	= 120	[Ns/m]
J_1^T	= 1.51e-3-1.03e-4 x	[kgm^2]
J_2^T	= 2.16e-4+1.03e-4 x	[kgm^2]
m^T	= 83.07 + m_l	[kg]
i^T	= 251.33	[m/rad]
m_l	= pay-load mass.	[kg.]

APPENDIX 6: THE 2-D.O.F. [R1T1] CONTROL MODEL

This model contains 2 Degrees Of Freedom :

- Φ : rotation of the turntable.
- x : position of the translating arm.

Figure 39 gives the lumped-mass representation of the 2-D.O.F. model.

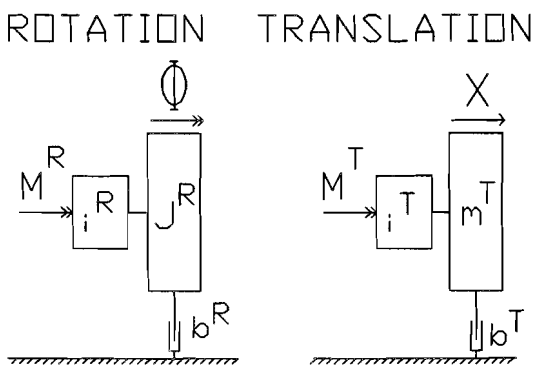


Figure 39: Lumped-mass representation 2-D.O.F. [R1T1] model.

The equations of motion are given by

$$M^R = \frac{1}{i^R}\left[J^R\ddot{\Phi} + \frac{\partial J^R}{\partial t}\dot{\Phi} + b^R\dot{\Phi} + M_w{}^R \right]$$

$$M^T = \frac{1}{i^T}\left[m^T\ddot{x} - \frac{1}{2}\frac{\partial J^R}{\partial t}\dot{\Phi}^2 + b^T\dot{x} + F_w{}^T \right] \qquad \textbf{(61)}$$

The parameters of the 2-D.O.F. [R1T1] model **(61)** are given by:

J^R	$= (83.06+m_l)x^2 + (1.48m_l-19.09)x + 81.43 + 0.55\ m_l$	[kgm^2]
b^R	$= 2110.10$	[Nm/rad]
i^R	$= 180.72$	[-]
m^T	$= 192.35 + m_l$	[kg]
b^T	$= 1383.3$	[Ns/m]
i^T	$= 251.33$	[m/rad]

APPENDIX 7: THE 3-D.O.F. [R2T1] CONTROL MODEL

This model contains 3 Degrees Of Freedom :

- Φ : rotation of the Rotation motor.
- $\Phi_m{}^R$: rotation of the turntable.
- x : position of the translating arm.

Figure 40 gives the lumped-mass representation of the 3-D.O.F. model.

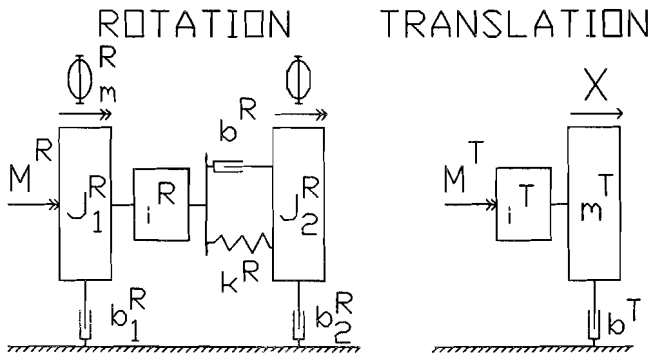


Figure 40: Lumped-mass representation 3-D.O.F. [R2T1] model.

The equations of motion are given by

$$M^R = J_1{}^R \ddot{\Phi}_m{}^R + \left[\frac{b^R}{(i^R)^2} + b_1{}^R\right]\dot{\Phi}_m{}^R - \frac{b^R}{i^R}\dot{\Phi} + \frac{k^R}{(i^R)^2}\Phi_m{}^R - \frac{k^R}{i^R}\Phi$$

$$0 = J_2{}^R \ddot{\Phi} + \frac{\partial J_2{}^R}{\partial t}\dot{\Phi} + \left[b^R + b_2{}^R\right]\dot{\Phi} - \frac{b^R}{i^R}\dot{\Phi}_m{}^R + k^R\Phi - \frac{k^R}{i^R}\Phi_m{}^R$$

$$M^T = \frac{1}{i^T}\left[m^T\ddot{x} - \frac{1}{2}\frac{\partial J_2{}^R}{\partial x}\dot{\Phi}^2 + b^T\dot{x}\right] \qquad (62)$$

The parameters in **(62,65)** are given by

m_l	= pay-load mass	[kg]
M^R	= external torque on the Rotation module	[Nm]
M^T	= external torque on the Translation module	[Nm]
J_1^R	= 1.48e-3	[kgm^2]
J_2^R	= $(83.06+m_l)x^2+(1.48m_l-19.09)x+33.10+0.55m_l$	[kgm^2]
k^R	= 2.94e5	[Nm/rad]
b^R	= 2.17e-5	[Nms/rad]
b_1^R	= 6.46e-2	[Nms/rad]
b_2^R	= 0.45	[Nms/rad]
i^R	= 180.72	[-]
m^T	= $192.35+m_l$	[kg]
b^T	= 1383.3	[Ns/m]
i^T	= 251.33	[rad/m]

The non-linear equations of motion can be written as non-linear state-space equations by defining the state-space vector $\underline{x}$

$$\vec{x}^T = \begin{bmatrix} \vec{q} \\ \dot{\vec{q}} \end{bmatrix} \tag{63}$$

Hence, the equations of motion can be written as

$$\dot{\vec{x}} = f(\vec{x}(t), \vec{u}(t), t)$$

$$\vec{y} = g(\vec{x}(t), \vec{u}(t), t) \tag{64}$$

With u(t) being the input signal (input vector), x(t) the state-space vector and y(t) the output vector.

This delivers 6 **non-linear** state-space equations:

$$\dot{x}_1 = \dot{\Phi}_m{}^R$$

$$\dot{x}_2 = \dot{\Phi}$$

$$\dot{x}_3 = \dot{x}$$

$$\dot{x}_4 = \frac{1}{J_1{}^R}\left[M_1{}^R - \left[\frac{b^R}{(i^R)^R} + b_1{}^R\right]\dot{\Phi}_m{}^R + \frac{b^R}{i^R}\dot{\Phi} - \frac{k^R}{(i^R)^2}\Phi_m{}^R + \frac{k^R}{i^R}\Phi - M_{w1}{}^R\right]$$

$$\dot{x}_5 = \frac{1}{J_2{}^R}\left[-\frac{\partial J_2{}^R}{\partial t}\dot{\Phi} - \left[b^R + b_2{}^R\right]\dot{\Phi} + \frac{b^R}{i^R}\dot{\Phi}_m{}^R - k^R\Phi + \frac{k^R}{i^R}\Phi_m{}^R - M_{w2}{}^R\right]$$

$$\dot{x}_6 = \frac{1}{m^T}\left[i^T M^T + \frac{1}{2}\frac{\partial J_2{}^R}{\partial t}\dot{\Phi}^2 - b^T\dot{x} - F_w{}^T\right] \qquad \textbf{(65)}$$

APPENDIX 8: MODELS OF THE MOTORS,POWER-AMPS AND DACS

A control law calculates the required control effort in [Nm], to be delivered by the DC-motors, to follow the desired trajectory.
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 41 shows all components in the traject from computer to the motor.

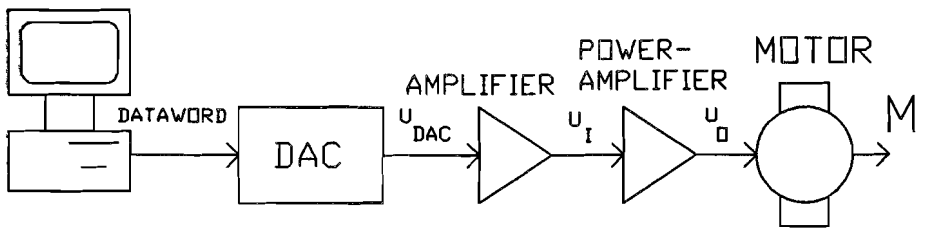


Figure 41: Traject from computer to motor

THE DACs

The Digital to Analog Converters (DACs) 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 only takes place after the high-byte of the 12 bit dataword is written in the corresponding register.

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

The (decimal) dataword value and the output voltage of the DACs are related as follows:

$$K_{dac} = \frac{u_{dac}}{DATA} = 2.5\ 10^{-3} \qquad (66)$$

THE EXTRA AMPLIFIER

The maximum output voltage range of the DACs on the DT 2811 interface board reaches from -5 to +5 Volts.
The power-amplifiers however require an input signal range from -10 to +10 Volts for full motor speed. Therefore, an extra amplifier is used with a fixed (low frequency) gain factor of 3. More details can be found in [3].

$$u_i = 3 * u_{dac} \tag{67}$$

THE POWER-AMPLIFIERS (PWR-AMPS)

The DC-motors can't be supplied by the voltage delivered by the DACs and the extra amplifiers. An extra power-amplifier is used to supply the DC-motors.
Hence, it is important to have exact knowledge on the behaviour of the power-amplifiers. Both power amplifiers are of the type BBC LV05. However, their characteristics are a littlebit different. The characteristic can be given by

$$u_o = K_{ver}.u_i + u_{off} \tag{68}$$

where

u_o = output voltage of pwr-amp = input voltage of DC-motor.
u_i = input voltage of pwr-amp = output voltage of DAC multiplied by 3.
u_{off} = offset voltage
K_{ver} = amplification gain

For small input voltages the offset voltage is zero. 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. This is corrected in the control program.

ROTATIONAL AMPLIFIER:
high u_i: $K_{ver} = 2.8$ $u_{off} = 0.75$ sign (u_i) [Volts]
low u_i: $K_{ver} = 3.4$ $u_{off} = 0$ [Volts]

TRANSLATIONAL AMPLIFIER:
high u_i: $K_{ver} = 2.8$ $u_{off} = 0.45$ sign (u_i) [Volts]
low u_i: $K_{ver} = 3.4$ $u_{off} = 0$ [Volts]

THE DC-MOTORS

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

$$L\frac{dI_a(t)}{dt} + R_a I_a(t) + K_e \dot{\Phi}_m(t) = u_o(t) \tag{69}$$

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.
When we suppose the first term is neglectible small, the motor-equation becomes as follows:

$$R_a I_a(t) + K_e \dot{\Phi}_m(t) = u_o(t) - u_b \tag{70}$$

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

$$M(t) = K_T I_a(t) \tag{71}$$

Substitution of equation (71) in equation (70) delivers:

$$\frac{R_a}{K_T} M(t) + K_e \dot{\Phi}_m(t) = u_o(t) - u_b \tag{72}$$

By definition K_T equals K_e, so the delivered torque is:

$$M(t) = \frac{K_e}{R_a}\left[u_o(t) - u_b\right] - \frac{K_e^2}{R_a} \dot{\Phi}_m(t) \tag{73}$$

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.
$\Phi_m(t)$: rotational velocity of the motor.

The motor parameters are given by:

R_a = 0.46 [Ω]
u_b = 1.8 sign (u_o) [Volts]
K_e = 0.226 [Vs/rad]

APPENDIX 9: THE INTERRUPT ARCHITECTURE

* Type 8259 (Intel)
* Addresses: Master 8259 Base = 20 H.
  Slave 8259 Base = A0 H.
* Assignment of interrupts:

Level		Function
Microprocessor NMI		Parity or I/O Channel Check
Interrupt Controllers		
CTRL1	CTRL2	
IRQ 0		Timer Output 0
IRQ 1		Keyboard (Output Buffer Full)
IRQ 2 ←		Interrupt from CTRL 2
	IRQ 8	Realtime Clock Interrupt
	IRQ 9	Software redirected to INT 0AH (IRQ 2)
	IRQ 10	Reserved
	IRQ 11	Reserved
	IRQ 12	Reserved
	IRQ 13	Coprocessor
	IRQ 14	Fixed Disk Controller
	IRQ 15	Reserved
IRQ 3		Serial Port 2
IRQ 4		Serial Port 1
IRQ 5		Parallel Port 2
IRQ 6		Diskette Controller
IRQ 7		Parallel Port 1

* Masking of individual interrupt lines:

A logical "1" in written in the Interrupt Mask Register (I.M.R.) blocks the corresponding incoming interrupt request.

* I.M.R. Master 8259 : 254
  Slave 8259 : 255 (Only during the replay)