Adaptive Computed Torque
Computed Reference Control
of a flexible RT-manipulator

Theoretical analysis, simulations and experiments

Eric Tenbult

Professor: prof. dr. ir. J.J. Kok
Coaches: dr. ir. F.E. Veldpaus and ir. P.C. Mulders

WFW, Control and identification of mechanical systems
Department of Mechanical Engineering, Eindhoven University of Technology

Report no.: 93.178
Abstract

In mechanical systems the presence of deformable subsystems (joints) is responsible for an increase of the number of generalized coordinates, i.e. the number of independent coordinates necessary and sufficient to adequately describe the configuration of the system at each time. Consequently the number of degrees of freedom to be controlled/stabilized exceeds the number of control inputs. Furthermore, the desired trajectories of the generalized coordinates are not determined uniquely by the output trajectory. A possibility to tackle this problem, is to apply a Computed Torque Computed Reference Control law, which incorporates nonlinear flexible robot dynamics and computes both the input torques and the a priori unknown references for the generalized coordinates. The controller design according to this algorithm is capable of accomplishing a reasonable trajectory tracking, while keeping the elastic vibrations bounded. Based on Lyapunov's second method global stability is ensured, regardless of the magnitude of the joint stiffness.

In order to improve the performance if some system parameters are unknown an adaptive version of the control law has been derived. Besides the familiar direct adaptation algorithm which extracts parameter information from the tracking errors, a new technique called composite adaptive control has been introduced. This new algorithm extracts parameter information from tracking errors as well as from prediction errors and leads to faster adaptation without incurring significant oscillation in the estimated parameters, thus yielding improved performance. However, both adaptation methods require linear parametrization of the system dynamics.

To illustrate the qualities of the designed controller several simulations under various conditions have been performed on a flexible RT-manipulator model. Although stability combined with asymptotic tracking can only be ensured for simulations with a perfect model, reasonable results can be obtained for the practical situation with time discrete implementation, actuator saturation, measurement noise etc. Similar characteristics and robustness properties can be obtained if the control law is implemented on a real RT-manipulator. Therefore adaptive CTCRC is considered to be a suitable approach for the control of nonlinear mechanical systems with more degrees of freedom than input signals and with uncertain parameters.
Contents

Abstract
Contents
Notation

Chapter 1 Introduction ........................................... 1

Chapter 2 Control of flexible robots ............................. 2
  2.1 The model .............................................. 2
  2.2 Adaptive Computed Torque Computed Reference Control ........ 3
    2.2.1 Stability ......................................... 5
    2.2.2 Direct adaptation algorithm ...................... 6
    2.2.3 Composite adaptation algorithm .................. 7
    2.2.4 Adaptive CTCRC in component form ............... 8

Chapter 3 Simulations .......................................... 12
  3.1 The RT-robot ........................................... 12
  3.2 Simulations .......................................... 13
    3.2.1 Variation of spring constant ..................... 15
    3.2.2 Direct and composite adaptation .................. 17
    3.2.3 Weak differentiation .............................. 19
    3.2.4 Discrete-time simulations ....................... 21
    3.2.5 Supplementary practical conditions ............... 22

Chapter 4 Experiments .......................................... 25

Chapter 5 Conclusions and recommendations ...................... 29

Bibliography ..................................................... 31

Appendices
  Appendix A: Equations of motion of a RT-robot ................ 32
  Appendix B: Variant of adaptive CTCRC ........................ 35
  Appendix C: Discrete-time differentiation .................... 36
  Appendix D: Dynamical model DC-motor ....................... 38
Notation

\( a, \alpha, a_* \) : scalars
\( \mathbf{a} \) : vector
\( A \) : matrix
\( a^T, A^T \) : transposition
\( A^{-1} \) : inversion
\( A(i,j) \) : element of matrix \( A \) on row \( i \), column \( j \)
\( \ddot{a} \) : second order time derivative of \( a \)
\( \dot{a} \) : first order time derivative of \( a \)
\( a^{(n)} \) : \( n \)'th order time derivative of \( a \)
\( \bar{a} \) : estimate of \( a \)
\( |a| \) : norm of \( a \)
\( f = f(t) \) : \( f \) is a function of argument \( t \)
\( g \neq g(t) \) : \( g \) is not a function of argument \( t \)
Chapter 1 Introduction

In order to increase operating speeds, modern robots should be lightweight constructions to reduce the driving torques and to enable the robot to respond faster. This however withdraws the assumption that the links behave as rigid bodies and the transmissions are stiff. Deformation of the links is usually of minor importance compared to the deformation in the joints and for this reason link deformation is omitted. Joint elasticity usually results in lightly damped oscillations which may cause stability problems if the flexibility is neglected in the control system. In order to avoid instability a control design model which accounts for the flexibilities is required. As a consequence of the availability of low cost powerful microprocessors realtime application of more complex control algorithms is possible nowadays.

During the last decade, several control algorithms for flexible manipulators have been developed. One possible approach to control flexible robots is feedback linearization combined with pole placement of the feedback linearized system. Major drawbacks of this method are the large computation time and the requirement for a very accurate model of the system. Another approach is to apply a method that is based on singular perturbation techniques. However, these techniques are restricted to robots with elastic joints only, while global stability is guaranteed for small flexibilities only [Lammerts et al., 1993].

Recently an alternative has been developed at the Eindhoven University of Technology which is based on the familiar computed torque control law. The original version relies on the assumption that all manipulator elements are perfectly rigid and that the number of actuator inputs equals the number of degrees of freedom. Therefore, the presence of flexibilities results in less tracking accuracy and can even lead to instability. In order to improve the performance of a flexible manipulator, the controller must achieve both reasonable trajectory tracking and stabilization of the elastic deflections. A new so-called (adaptive) Computed Torque Computed Reference Control algorithm ensures stability combined with asymptotic tracking for a flexible robot with both joint elasticity and link flexibility regardless of the magnitude of the flexibility. If the system parameters or the payload parameters are unknown, an adaptive version can be used such that trajectory tracking is achieved while all signals within the system remain bounded.

The purpose of the research in this report is to get an impression of the properties of the adaptive CTCRC algorithm. Chapter 2 presents the theoretical analysis of the designed controller. Besides the proof of stability two (on-line) adaptive control schemes have been derived. These schemes are based on the assumption that the structure of the mathematical model is correct, but that some of the model parameters are unknown. Furthermore, it is assumed that these parameters are constant and that the model is linear in these parameters. Chapter 3 presents the simulation results of a flexible RT-manipulator, controlled with an adaptive CTCRC law. The issue of robustness has been examined by performing several simulations under practical conditions. Chapter 4 gives a survey of the experimental results that are obtained by implementing adaptive CTCRC, PD-control and computed torque control on a real RT-manipulator. Chapter 5 offers brief concluding remarks and recommendations.
Chapter 2  Control of flexible robots

2.1 The model

The manipulators considered in this report can be modeled as an open chain of \( n \) rigid links connected by cylindrical, revolute or prismatic joints with one degree of freedom per joint. Several links, however, are driven via an elastic transmission. The base of the manipulator is fixed to the ground while the other end with gripper and payload has to follow a prescribed trajectory. Motor dynamics will not be taken into consideration.

The main difference between a flexible manipulator and its rigid counterpart consists of the fact that the number of degrees of freedom \((n+e)\) to be controlled / stabilized is unequal to the number of actuators \((n)\). The number of extra generalized coordinates \((e)\) for the flexible systems in this report equals the number of elastic transmissions. Neglecting this extra degree(s) of freedom can lead to instability.

The equations of motion (a set of second order differential equations) can be derived by the Lagrange formalism. In general the dynamical model of a flexible manipulator can be written in the following standard form.

\[
M(q,p)\ddot{q} + C(q,\dot{q},p)\dot{q} + g(q,\dot{q},p,t) = H(q)u
\]

\[
\chi = h(q)
\]

(2.1)

Here \( q(t) \in R^{(n+e)} \) is a vector of generalized coordinates, necessary and sufficient to describe the robot configuration, \( p \in R^m \) is a vector of constant, unknown system parameters, \( M(q,p) \in R^{(n+e) \times (n+e)} \) is a symmetric positive definite inertia matrix, \( C(q,\dot{q},p) \in R^{(n+e) \times (n+e)} \) is a matrix which accounts for the Coriolis and centrifugal torques while \( g(q,\dot{q},p) \in R^{(n+e)} \) is a vector which accounts for an eventual external loading (e.g. by gravity) and for all internal torques except actuator torques. These internal torques are due to, for instance damping, friction and joint flexibility. \( H(q) \in R^{(n+e) \times n} \) is a distribution matrix with full rank and \( u \in R^n \) is a vector of actuator forces. The output vector \( \chi(t) \in R^s \) is a smooth bounded vector function of the generalized coordinates.

The given model is applicable for a broad class of flexible manipulators. The matrices \( M \) and \( C \) are not independent since \((M - 2C)\) is skew-symmetric. This implies that at all times \( \dot{\gamma}^T(M - 2C)\dot{\gamma} = 0 \). Often the vector \( g(q,\dot{q},p,t) \) can be written as the sum of a stiffness term \( K_q \), a viscous damping term \( Bq \) and a term which represents the remaining internal torques, Coulomb friction for instance.

\[
g(q,\dot{q},p,t) = B(q)\dot{q} + K_q + g_n(q,\dot{q},p,t)
\]

(2.2)
Chapter 2: Control of flexible robots.

Here $B(y) \in R^{(n+e)(n+e)}$ is a (semi-) positive definite damping matrix, $K \in R^{(n+e)(n+e)}$ is a semi-positive definite stiffness matrix and $g_\bullet (q, \dot{q}, y(t)) \in R^{(n+e)}$ is a vector with the remaining forces. A major restriction of this choice is the assumption that the components of $K$ are exactly known, since $K=K(y)$.

In general, the output (gripper) trajectory $y(t)$ is a smooth function of the generalized coordinates $y=h(q)$. However, for a flexible manipulator only $n$ coordinates of the vector $q(t)$ can be determined from the output trajectory $y(t)$. These coordinates are the components of a vector $q_k$. Now it is possible to partition $q(t)$ in two components:

$$q = L_k q_k + L_u q_u$$

$$q_d = L_d q_{ kd} + L_u q_{ ud}$$ (2.3)

where $[L_k, L_d] \in R^{(n+e)(n+e)}$ is a permutation matrix, $q_k \in R^n$ is a vector with those components of $q(t)$ that are determined from $y(t)$ while $q_d \in R^e$ is a vector with those components of $q(t)$ that cannot be determined from $y(t)$. The index $d$ denotes a desired trajectory.

The components of $q_{kd}$ can be determined from the output equation if $y(t)$ is replaced by the desired output trajectory $y_d(t)$. This is not possible for $q_{ud}$. However, it is assumed that it is possible to compute a smooth and bounded trajectory $q_{ud}$. The determination of such a trajectory is one of the topics of section 2.2.4.

2.2 Adaptive Computed Torque Computed Reference Control

The goal of a controller for a flexible manipulator is to achieve trajectory tracking and to limit the occurring vibrations between acceptable bounds. The control law described in this report is based on a model of the flexible manipulator in order to compensate for non-linearities and to guarantee a desired closed loop behaviour.

Good trajectory tracking is achieved if the position error $e = q_r - q$ and its derivative $\dot{e}$ remain bounded and converge to zero. This condition can be reformulated by the introduction of a virtual reference trajectory $q_r(t)$ with:

$$\dot{q}_r = \dot{q}_d + \Lambda (q_d - q)$$

$$\ddot{q}_r = \ddot{q}_d + \Lambda \dot{e}$$ (2.4)

where, $\Lambda \in R^{(n+e)(n+e)}$ is a positive definite (diagonal) matrix. Since $\ddot{e}$ is defined as $\ddot{e} = \ddot{q}_d - \ddot{q}$ and consequently $\ddot{e} = \ddot{q} + \Lambda \dot{e}$, the requirements with respect to $\dot{e}$ and $\ddot{e}$ are fulfilled if $\ddot{e} \to 0$ for $t \to \infty$. The definition of $\dot{e}$ can be viewed as a stable first-order differential equation in $e$, with $\dot{e}$ as an input.
The control problem now is to derive an algorithm which guarantees that \( \dot{\mathbf{e}} \to \mathbf{0} \) for \( t \to \infty \). In accordance with the basic algorithm of Slotine and Li the next control law has been chosen:

\[
H(q)u = M(q, \dot{q})\ddot{q} - C(q, \dot{q})q - \dot{B}(q)q + Kq + \hat{\mathbf{g}}_n(q, \dot{q}, \dot{\mathbf{e}}) + K\dot{\mathbf{e}},
\]

where \( \hat{\mathbf{g}} \in \mathbb{R}^m \) is a vector of estimated system parameters and \( K, \in \mathbb{R}^{(n+e)x(n+e)} \) is a positive definite diagonal gain matrix. The control algorithm consists of two components i.e. a full dynamics feedforward compensation part and a feedback part \( K\dot{\mathbf{e}} \). Besides for the computation of \( u(t) \), this control law is also used to determine the unknown part \( \mathbf{g}_r \) of \( \mathbf{g}_r = L_u\mathbf{q}_u + L_u\mathbf{q}_u \) from the set of equations (2.5).

This control law can also be analyzed according to the sliding control methodology. Therefore, \( \dot{\mathbf{e}} = \mathbf{e} + \Lambda\mathbf{e} = \mathbf{0} \) is considered to be a set of \( (n+e) \) sliding surfaces, such that sliding mode occurs on the intersection of planes \( \dot{\mathbf{e}}_i = \dot{\mathbf{e}} + \lambda_i \mathbf{e}_i \) with \( i = 1..(n+e) \). The dynamics while in sliding mode \( \dot{\mathbf{e}}_r = \mathbf{0} \) can be written as:

\[
\dot{\mathbf{e}}_r = \mathbf{0}
\]

By solving the above equation formally for the control input an expression for \( u \) called the equivalent control \( \dot{\mathbf{e}}_r = \mathbf{0} \) if the dynamics were exactly known. For simplicity only the non-adaptive case is considered. Substitution of the equations of motion (1.1) in \( \dot{\mathbf{e}}_r = \ddot{\mathbf{q}} - \ddot{\mathbf{q}} + \Lambda\dot{\mathbf{e}} \) yields:

\[
\dot{\mathbf{e}}_r = \ddot{\mathbf{q}} - M^{-1}(\dot{\mathbf{H}}u - C\dot{\mathbf{q}} - B\dot{\mathbf{q}} - K\dot{\mathbf{q}} - \dot{\mathbf{g}}_n) + \Lambda\mathbf{e}
\]

Since \( \dot{\mathbf{H}}u \) is defined as \( \dot{\mathbf{H}}u = \dot{\mathbf{H}}u(\dot{\mathbf{e}}_r = \mathbf{0}) \) following equation can be obtained:

\[
\mathbf{H}u = \mathbf{H}_{eq} + C\mathbf{e}_r + K\mathbf{e}_r + K\dot{\mathbf{e}}_r
\]

Now, it is obvious that \( \dot{\mathbf{H}}u \) is composed of a term which keeps the system in sliding motion and a PI part \( (C + K)\mathbf{e}_r + K\int \dot{\mathbf{e}}_r \, dt \) which forces the system to the sliding surface \( \dot{\mathbf{e}}_r = \mathbf{0} \). Substitution of the control law in the equation (2.6) yields the error equation of the non-adaptive closed-loop system:

\[
M\ddot{\mathbf{e}}_r + C\dot{\mathbf{e}}_r + K\mathbf{e}_r + K\dot{\mathbf{e}}_r = \mathbf{0}
\]

A possible solution of this differential equation combined with a phase plot is given in figure 1. Note that \( \dot{\mathbf{e}}(t) \) intersects the sliding surface several times. Further details about this simulation will be given in section 3.2.
Substitution of the input $u$ according to (2.5) in the equation of motion and substitution of $\dot{q} = \dot{q}_r - \dot{q}_r$ yields the error equation of the closed loop system:

$$M \ddot{e}_r + C \dot{e}_r + K \ddot{e}_r = -[(\hat{M} - M) \ddot{e}_r + (\hat{C} - C) \dot{e}_r + (\hat{B} - B) \ddot{q} + (\hat{g}_n - g_n)]$$

$$= -W_r(q, \dot{q}, \dot{q}_r, t) \ddot{p}$$

where $\ddot{p}(t) = \ddot{q}(t) - p \in R^m$ is a vector of parameter errors while $W_r(q, \dot{q}, \dot{q}_r, t) \in R^{(m+n) \times m}$ is a system matrix. Here, the assumption has been made that it is possible to choose the unknown parameters $\ddot{p}(t)$ such that the system is linear in those parameters $\ddot{p}(t)$.

### 2.2.1 Stability

According to the second lemma of Lyapunov [Slotine and Li, 1991], a system is globally asymptotically stable if there exist a scalar function $V(x)$ of the state $x$, with continuous first order derivatives such that:

$$V(x) > 0 \quad \text{for } x \neq 0$$

$$V(x) < 0 \quad \text{for } x \neq 0$$

$$V(x) \to \infty \quad \text{for } \|x\| \to \infty$$

$$V(0) = 0$$

In order to analyze stability of the closed-loop system the following Lyapunov function has been chosen:

$$V = \frac{1}{2}[\ddot{e}_r^T \check{M} \ddot{e}_r + \ddot{e}_r^T \check{K} \ddot{e}_r + \ddot{p}^T \Gamma^{-1} \ddot{p}]$$

where $\Gamma \in R^{m \times m}$ is a constant, positive definite (diagonal) matrix. This scalar $V$ can be considered as the sum of the mechanical energy of the error system and a positive definite quadratic expression in the parameter error.
Differentiating the Lyapunov function with respect to time yields:

\[ V = \dot{e}_r^T M \ddot{e}_r + \frac{1}{\tau} \dot{e}_r^T M \dot{e}_r + \dot{e}_r^T K \dot{e}_r + \ddot{\theta}^T \Gamma^{-1} \ddot{\theta} \]  \hspace{1cm} (2.15)

Substitution of the closed-loop error equation (2.10) results in:

\[ V = -\dot{e}_r^T K \dot{e}_r + \frac{1}{\tau} \dot{e}_r^T (M - 2C) \dot{e}_r - \dot{e}_r^T W \dot{\theta} + \ddot{\theta}^T \Gamma^{-1} \ddot{\theta} \]  \hspace{1cm} (2.16)

As a consequence of the property of skew-symmetry, the term \( \dot{e}_r^T (M - 2C) \dot{e}_r \) is zero for all \( \dot{e}_r \), so:

\[ V = -\dot{e}_r^T K \dot{e}_r - \dot{\theta}^T (W^T \dot{\theta} - \Gamma^{-1} \ddot{\theta}) \]  \hspace{1cm} (2.18)

By applying one of the following adaptation algorithms for \( \dot{\theta} = p + \ddot{\theta} \) it is possible to guarantee, that \( \dot{e}_r \rightarrow 0 \) for \( t \rightarrow \infty \) and that both \( \dot{e}_r \) and \( \ddot{\theta} \) remain bounded. This implies that \( \dot{g}(t) \rightarrow 0 \) as \( t \rightarrow \infty \) (proof not trivial).

### 2.2.2 Direct adaptation algorithm

According to the second lemma of Lyapunov, a global stable system can be obtained if at all times \( V < 0 \) for \( \dot{e}_r \neq 0 \). This requirement will be met if the second term of the formula (2.18) is equal to 0. In the direct adaptation algorithm the parameters \( \dot{\theta}(t) \) are adapted according to:

\[ \dot{\theta} = \Gamma W_r^T \dot{e}_r \]  \hspace{1cm} (2.19)

with \( \dot{\theta} = \ddot{\theta} \). The second lemma of Lyapunov then states that \( \dot{e}_r \) converges to 0, i.e. the tracking error \( g(t) \) becomes 0 for \( t \rightarrow \infty \). Further, the adaptation algorithm states that \( \ddot{\theta} \) converges to zero. This implies that \( \dot{g}(t) \) becomes constant for \( t \rightarrow \infty \). However, there is no guarantee that estimates \( \ddot{\theta} \) resemble the true constant values \( \ddot{\theta} \) for \( t \rightarrow \infty \). In general it is not a real problem that \( \ddot{\theta} \neq 0 \) for \( t \rightarrow \infty \), since it is guaranteed that the tracking errors \( g(t) \rightarrow 0 \) for \( t \rightarrow \infty \). Parameter convergence is assured only under certain persistent excitation conditions, i.e. if the reference trajectory is sufficiently rich [Slotine and Li, 1991]. Note that if \( \dot{\theta}(t) \) doesn't converge to a constant value significant unmodelled dynamics must be present in the system.
2.2.3 Composite adaptation algorithm

Parameter estimation is based on extracting information from available system data, by using an estimation model. This estimation model doesn't have to be the same as the model used for control purpose. A fairly general model for parameter applications is in the linear parametrization form:

\[ z(t) = Y(t)p \]  \hspace{1cm} (2.20)

where \( z \in \mathbb{R}^k \) is a vector containing the outputs of the system, \( p \in \mathbb{R}^m \) the parameter vector and \( Y(t) \in \mathbb{R}^{k \times m} \) is a signal matrix. It is obvious at this point that the output vector of the estimation model doesn't have to be the same as the output in a control problem.

A possible approach to obtain a suitable estimation model is to reformulate the equations of motion.

\[
\begin{align*}
Hu - Kg &= M\ddot{q} + C\dot{q} + B\ddot{q} + g_n \\
Hu - Kg &= Y_m(q,\dot{q},\ddot{q},t)p + Y_m(q,\dot{q},\ddot{q},t)
\end{align*}
\]  \hspace{1cm} (2.21)

Note that \( z(t) \) and \( Y(t) \) are required to be known from continuous measurement of the system signals. Since \( p \) is a constant parameter vector and equation (2.21) must hold for all \( t \), the estimation model is an overdetermined set of equations. Now it is possible to predict the value of \( z(t) \) at time \( t \), by substituting the parameter estimate \( \hat{p}(t) \) in equation (2.20). This yields:

\[ \hat{z}(t) = Y(t)\hat{p}(t) \]  \hspace{1cm} (2.22)

where \( \hat{z}(t) \) is the predicted value at time \( t \). The difference between the predicted output and the measured output is called the prediction error, denoted by \( e_p(t) \), i.e.:

\[ e_p(t) = \hat{z}(t) - z(t) \]  \hspace{1cm} (2.23)

Now, a relation between the prediction error and the parameter error can be obtained.

\[ e_p(t) = Y(t)\hat{p}(t) - Y(t)p = Y(t)\tilde{p} \]  \hspace{1cm} (2.24)
In order to reduce the prediction error, it is possible to derive a (gradient) estimator which updates the parameters in the converse direction of the gradient of the squared prediction error, with respect to the parameters:

\[ \hat{\lambda} = -\gamma \frac{\partial[e^T \hat{e}_p]}{\partial \lambda} \]  

(2.25)

This equation is equivalent with:

\[ \hat{\lambda}(t) = -\gamma Y(t) \tilde{e}_p \]  

(2.26)

According to [Slotine and Li, 1991], it is possible to derive an adaptation algorithm which extracts information about the parameters from the tracking errors \( \hat{e}(t) \), as well as from the prediction errors \( \tilde{e}_p \). This so-called composite adaptation algorithm can further improve the performance of the adaptive controller. A combination of the direct adaptation algorithm and the gradient estimator leads to the following composite adaptation algorithm:

\[ \hat{\lambda} = \Gamma(W^T \hat{e}_r - Y^T \tilde{e}_p) \]  

(2.27)

This adaptation law simply adds an extra term in the expression of \( \hat{V} \) (2.18).

\[ \hat{V}(t) = -\hat{e}_r^T K \hat{e}_r - \hat{e}_p^T Y(t) \tilde{e}_p \]  

(2.28)

Now, the second lemma of Lyapunov states that both the tracking errors \( \hat{e}(t) \) and the prediction errors \( \tilde{e}_p \) converge to 0. Similar to the direct algorithm, the composite adaptation law only guarantees that \( \hat{\lambda}(t) \to 0 \) for \( t \to \infty \). If in addition, the trajectories are persistently exciting and uniformly continuous, the estimated parameters asymptotically converge to the true parameters.

### 2.2.4 Adaptive CTCRC in component form

So far it is assumed that \( g = L \dot{q} + L \ddot{q} \) is completely known. However, the trajectory \( g_{\text{out}} \) is unknown beforehand. Without loss of generality it is possible to write the equations of motion for a manipulator with elastic joints,

\[ M(q,p)\ddot{q} + C(q,q,p)\dot{q} + B(p)\dot{q} + Kq + g_n(q,q,p,t) = H(q)u \],  

(2.30)
in component form as

\[
\begin{bmatrix}
M_{ee} & M_{we} & 0 & \hat{q}_e \\
M_{we} & M_{ee} & 0 & \hat{q}_e \\
0 & 0 & M_{uu} & \hat{u}_u
\end{bmatrix}
\begin{bmatrix}
\frac{\partial}{\partial t} \hat{q}_e \\
\frac{\partial}{\partial t} \hat{q}_e \\
0 & 0 & \hat{u}_u
\end{bmatrix}
= \begin{bmatrix}
B_e & 0 & 0 & \hat{q}_e \\
0 & B_e & 0 & \hat{q}_e \\
0 & 0 & B_u & \hat{u}_u
\end{bmatrix}
\begin{bmatrix}
0 & 0 & \hat{q}_e + \hat{q}_e \\
0 & 0 & \hat{q}_e + \hat{q}_e \\
0 & -K & \hat{q}_e + \hat{q}_e
\end{bmatrix}
= \begin{bmatrix}
I & 0 \\
0 & I
\end{bmatrix}
\begin{bmatrix}
\hat{u}_u \\
\hat{u}_e
\end{bmatrix}
\]

where \( \hat{q}_e^T = [q_e^T, q_e^T] \in R^n \) is a vector of generalized link coordinates, \( q_e \in R^{(n-e)} \) is a vector of link coordinates with a stiff transmission, \( q_e \in R^e \) is a vector of link coordinates with an elastic transmission and \( q_e \in R^e \) is a vector of coordinates necessary and sufficient to describe the deformation of the elastic transmission (motor coordinates). \( q_e \in R^{(n-e)} \) is a vector component of \( \hat{u} \) directly acting on \( q_e \), while \( u_e \) is a vector component of \( \hat{u} \) acting on \( q_e \) via an elastic transmission.

For a further elaboration the adaptive Computed Torque Computed Reference Control algorithm,

\[
H(q)\hat{u} = \hat{\dot{q}}(q, \hat{q}, \hat{q}, \hat{q}) + \hat{\dot{\dot{q}}}(q, \hat{q}) + K q_r + \hat{\dot{\dot{q}}} n(q, \hat{q}, \hat{q}, \hat{q}) + K_r \hat{e}_r
\]

is formulated in component form as

\[
H_u = \begin{bmatrix}
\hat{u}_s & [M_{es}, \dot{M}_{es}] \hat{q}_s & + [\hat{C}_{es}, \dot{C}_{es}] \hat{q}_s & + \hat{\dot{B}} \hat{q}_s & + \hat{\dot{\dot{q}}} s & + K_r \hat{e}_r \\
0 & [M_{es}, \dot{M}_{es}] \hat{q}_s & + [\hat{C}_{es}, \dot{C}_{es}] \hat{q}_s & + \hat{\dot{B}} \hat{q}_s & + \hat{\dot{\dot{q}}} s & + K_r \hat{e}_r \\
\hat{u}_e & M_{uu} \hat{u}_u & + \hat{\dot{B}} \hat{u}_u & - K(q_{ur} - q_{ur}) & + \hat{\dot{\dot{q}}} e & + K_r \hat{e}_u
\end{bmatrix}
\]

where \( q_{ur}^T = [q_{ur}^T, q_{ur}^T] \in R^n \) is a vector of link reference coordinates, \( q_{ur} \in R^{(n-e)} \) is a vector of stiff link reference coordinates, \( q_{ur} \in R^e \) is a vector of elastic link reference coordinates and \( q_{ur} \in R^e \) is a vector of motor reference coordinates.

Characteristic for this control law is the on-line determination of \( q_{ur} \) from the second set of equations. This gives:

\[
q_{ur} = K^{-1} \left\{ [M_{es}, \dot{M}_{es}] \hat{q}_s + [\hat{C}_{es}, \dot{C}_{es}] \hat{q}_s + \hat{\dot{B}} \hat{q}_s + \hat{\dot{\dot{q}}} s + K_r \hat{e}_r \right\} + q_{er}
\]
Double differentiation of this equation yields respectively an equation for \( \ddot{q}_{ur} \) and \( \dddot{q}_{ur} \) necessary to compute \( \ddot{u} \). From the equation above it follows that:

\[
\begin{align*}
\ddot{q}_{ur} &= \ddot{q}_{ur}(q, \dot{q}, \dot{d}, \ddot{d}, \dot{\theta}, \ddot{\theta}) \\
\dddot{q}_{ur} &= \dddot{q}_{ur}(q, \dot{q}, \dot{d}, \ddot{d}, \dot{\theta}, \ddot{\theta}) \\
\dddot{q}_{ur} &= \dddot{q}_{ur}(q, \dot{q}, \dot{d}, \ddot{d}, \dot{\theta}, \ddot{\theta})
\end{align*}
\]  

(2.35)

Since \( \ddot{q} = f(q, \dot{q}, \ddot{q}) \) and \( \dddot{q} = h(q, \dot{q}, \dot{\theta}, \ddot{\theta}) \), it seems impossible to obtain an analytical (explicit) equation for \( \dddot{u} \). However, it is assumed that:

\[
\begin{align*}
M_k &= \begin{bmatrix} M_{ss} & M_{se} \\ M_{es} & M_{ee} \end{bmatrix} = M_k(q, \dot{q}, \ddot{q}) \\
C_k &= \begin{bmatrix} C_{ss} & C_{se} \\ C_{es} & C_{ee} \end{bmatrix} = C_k(q, \dot{q}, \ddot{q}) \\
\hat{g}_k &= \begin{bmatrix} \hat{g}_s \\ \hat{g}_e \end{bmatrix} = \hat{g}_k(q, \dot{q}, \ddot{q}, t)
\end{align*}
\]

(2.36)

where \( \hat{p} \in \mathbb{R}^{(n-r)} \) is a vector component of the parameter vector \( \hat{p} \), which accounts for all link and payload parameters. Thus

\[
\begin{align*}
\ddot{q}_{ur} &= \ddot{q}_{ur}(q, \dot{q}, \dot{d}, \ddot{d}, \dot{\theta}, \ddot{\theta}) \\
\dddot{q}_{ur} &= \dddot{q}_{ur}(q, \dot{q}, \dot{d}, \ddot{d}, \dot{\theta}, \ddot{\theta}) \\
\dddot{q}_{ur} &= \dddot{q}_{ur}(q, \dot{q}, \dot{d}, \ddot{d}, \dot{\theta}, \ddot{\theta})
\end{align*}
\]  

(2.37)

Since these vector functions consists of some undetermined terms \( q^{(3)}_k, q^{(4)}_k, \dot{q}^{(3)}_k, \ddot{q}^{(3)}_k \), preliminary calculations have to be made. The terms \( \ddot{q}_k \) and \( \dddot{q}_k \) (necessary to compute \( q^{(3)}_k \) and \( q^{(4)}_k \)) can be obtained by the partitioned equations of motion.

\[
\begin{align*}
\ddot{q}_k &= \begin{bmatrix} \ddot{q}_s \\ \ddot{q}_e \end{bmatrix} = \begin{bmatrix} M_{ss} & M_{se} \\ M_{es} & M_{ee} \end{bmatrix}^{-1} \begin{bmatrix} \dot{u}_s \\ \dot{u}_e \end{bmatrix} - \begin{bmatrix} C_{ss} & C_{se} \\ C_{es} & C_{ee} \end{bmatrix} \begin{bmatrix} \ddot{q}_s \\ \ddot{q}_e \end{bmatrix} - \begin{bmatrix} \dddot{q}_s \\ \dddot{q}_e \end{bmatrix} \\
\dddot{q}^{(3)}_k &= \begin{bmatrix} q^{(3)}_s \\ q^{(3)}_e \end{bmatrix} = \begin{bmatrix} M_{ss} & M_{se} \\ M_{es} & M_{ee} \end{bmatrix}^{-1} \begin{bmatrix} \dot{u}_s \\ \dot{u}_e \end{bmatrix} - \frac{d}{dt} \begin{bmatrix} C_{ss} & C_{se} \\ C_{es} & C_{ee} \end{bmatrix} \begin{bmatrix} \dddot{q}_s \\ \dddot{q}_e \end{bmatrix} - \frac{d}{dt} \begin{bmatrix} \dddot{q}_s \\ \dddot{q}_e \end{bmatrix} - \frac{d}{dt} \begin{bmatrix} M_{ss} & M_{se} \\ M_{es} & M_{ee} \end{bmatrix} \begin{bmatrix} \dddot{q}_s \\ \dddot{q}_e \end{bmatrix}
\end{align*}
\]
It is noted that this is only possible in simulations, since the true parameters are unknown. The necessary expression for $\dot{u}_s$ and $\ddot{u}_s$ can be obtained from the first set of equations of the control algorithm. Now an equation for $q'_{kr}$ and $q''_{kr}$ can be derived:

\[
q'_{kr} = q''_{kr} + \Lambda(q_{kr} - q_s) \\
q''_{kr} = q'''_{kr} + \Lambda(q_{kr} - q''_{kr})
\]  

(2.39)

It is obvious that the necessary expression for $\dot{\theta}_s$ and $\ddot{\theta}_s$ can be obtained from the direct or the composite adaptation algorithm, respectively equation (2.40)

\[
\dot{\theta} = \Gamma W^T \dddot{\theta}_r \\
\ddot{\theta} = \Gamma(W^T \dddot{\theta}_r + W^T \dddot{\theta}_s)
\]  

(2.40)

and equation (2.41)

\[
\dot{\theta} = \Gamma(W^T \dddot{\theta}_r - Y^T \dddot{\theta}_p) \\
\ddot{\theta} = \Gamma(W^T \dddot{\theta}_r + W^T \dddot{\theta}_s - Y^T \dddot{\theta}_p - Y^T \dddot{\theta}_p)
\]  

(2.41)

Here, the assumption has been made that $\dot{\theta}_s = \dot{\theta}_s(q_s \dddot{q}_s, \dddot{q}_s, \dot{q}_s, \dddot{q}_s, t)$. This requirement will be met if $K \neq K(\mu)$. 

Chapter 3  Simulations

3.1 The RT-robot

In order to obtain an impression of the performance of the adaptive Computed Torque Computed Reference Control law several simulations have been performed. The simulations concern a two dimensional rotation-translation (RT-) robot of figure 2 with 3 degrees of freedom ($n+e=3$). The rotation module is driven via an elastic motor transmission that is modeled as a linear massless torsional spring.

Using the Lagrange equations, it is possible to derive the dynamic equations of the flexible RT-robot (appendix 1):

$$
\begin{bmatrix}
p(1) & 0 & 0 & 0 \\
0 & p(1)q_e^2 - 2p(2)q_e + p(3) & 0 & 0 \\
0 & 0 & p(4) & 0
\end{bmatrix}
\begin{bmatrix}
\dot{q}_e \\
\dot{q}_r \\
\dot{q}_u
\end{bmatrix}
+
\begin{bmatrix}
0 & -\mu(t)q_e & 0 \\
\mu(t)q_e & 0 & 0 \\
0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
q_e \\
q_r \\
q_u
\end{bmatrix}
+
\begin{bmatrix}
b_d \dot{q}_e \\
b_d \dot{q}_r \\
b_d \dot{q}_u - k(q_e - q_u)
\end{bmatrix}
=
\begin{bmatrix}
F \\
0 \\
0
\end{bmatrix}
$$

where $\mu(t) = p(1)q_e - p(2)$, $q_e [m]$ is the horizontal position of the payload, $q_r [rad]$ is the rotation angle of the rotation module, $q_u [rad]$ is the rotation angle of the motor rotor, $F [N]$ is the force acting on the translation module via a stiff transmission, $M [Nm]$ is the torque acting on the rotation module via an elastic transmission and $p(1)...p(4)$ are components of the parameter vector $p$. The parameter $k [Nm/rad]$ represents the torsional spring constant.

From the equations of motion it is seen that the RT-manipulator has significant non-linearities due to the Coriolis and centrifugal effects. All inertia properties are considered to be unknown. Furthermore, no actuator dynamics are taken into account.
Unless stated otherwise, the following (realistic) parameter values have been chosen:

\[ P = \begin{bmatrix} 3 \text{[kg]} \\ 3 \text{[Kgm]} \\ 15 \text{[Kgm}^2] \\ 5 \text{[Kgm}^2] \end{bmatrix}, \quad b = \begin{bmatrix} b_s \\ b_s \\ b_s \\ b_s \end{bmatrix} = \begin{bmatrix} 1 \text{[Ns/m]} \\ 1 \text{[Nms/\text{rad}]} \end{bmatrix}, \quad l = 2 \text{ [m]} \quad k = 10 \text{ [Nm]} \]

(3.2)

The control objective is to let the manipulator track a desired trajectory \( q_d(t) \), while keeping all generalized coordinates bounded. The desired trajectory itself must be chosen smooth enough not to excite high frequency dynamics. The desired trajectory for simulation purpose is:

\[ q_{kd} = \begin{bmatrix} q_{sd} \\ q_{nd} \end{bmatrix} = \begin{bmatrix} A_{s0} + A_s \cos(\omega_s t) \\ A_s \sin(\omega_s t) \end{bmatrix} \]

(3.3)

with \( A_{s0} = 0.5 \text{ [m]}, A_s = 0.5 \text{ [m]}, A_s = 1 \text{ [rad]}, \omega_s = 1 \text{ [rad / s]} \) and \( \omega_s = 1 \text{ [rad / s]} \).

### 3.2 Simulations

In this paragraph an overview is given of the numerical simulation results. In order to avoid large computation times, it is noted that all system parameters with exception of the parameters of \( M(q,p) \) and \( C(q,q,p) \) are exactly known. This implies that \( B=B(p) \). Besides the assumption has been made that the full state vector \( x^T=[q^T \dot{q}^T] \) is available, while for the theoretical situation the second and third derivative (respectively \( \ddot{q} \) and \( \dot{q}^{(3)} \)) can be determined exactly from the equations of motion. For this purpose the true parameters \( p \) are substituted in the equations of motion. Obviously this method can only be applied during simulations.

Because it is impractical to assume that the initial position of the end effector always correspond to the desired position, several simulations have been performed with large initial position errors. The presence of relatively large initial position errors \( \dot{e}(0)=[8A_s, 2A_s] \) results in rapidly fluctuating parameter estimations, due to the fact that the system matrices \( W=W(q,q,q,\ddot{q}) \) and \( Y=Y(q,q,q,t) \) and the vector \( \dot{e}_c = \dot{e} + \Lambda \dot{e} \) consists of large components. This phenomenon is caused by the gradient nature of the adaption algorithm, i.e. too large adaptation gains or reference signals lead to an oscillatory behaviour of \( \dot{b}(t) \). Several approaches are available to obtain a smoother behaviour of the parameter vector \( \dot{b}(t) \).
The simplest possibility is to drastically decrease $\Gamma$, since $\dot{\theta} = \Gamma W^T \dot{\xi}$ or $\dot{\theta} = \Gamma(W^T \dot{\xi} - Y^T \dot{\xi})$. This however results in a very slow adaptation process when the system reaches the sliding surface ($\dot{e} = 0$). Note that $\dot{e}(t) = 0$ for $t \to \infty$ as a result of the discontinuous character of the simulations. Therefore it is possible to extract parameter information from the trajectory errors even when $t \to \infty$. The upper-boundary $\Phi$ with $|\dot{e}_i| \leq \Phi$ for $t \to \infty$ is a function of the numerical integration tolerance.

Another and more effective approach is to decrease the components $\lambda_i$ of the matrix $\Lambda$, since it reduces the vector $\dot{e} = \dot{e} + \Lambda \dot{e}$ as well as the system matrices $W = W(g, \dot{q}, \ddot{q}, \dot{\theta})$ and $Y = Y(q, \dot{q}, \ddot{q}, t)$, as a result of the reduction in control activity. However this results in large tracking errors $e(t)$ for $t \to \infty$, since $e \leq \Phi/\lambda_i$.

In order to achieve good trajectory tracking combined with a smooth adaptation process $\Lambda$ is chosen to be a time-varying diagonal matrix $\Lambda = \Lambda(t)$. Assume the system is in sliding motion ($\dot{e}_r = 0$) for $t > t_s$. Then the general solution of $\dot{e}_r + \lambda_i(t)e_i = 0$ is:

$$e_i(t) = e_i(t_s) \exp \left[ - \int_{t_s}^{t} \lambda_i(\tau) d\tau \right]$$  \hspace{1cm} (3.4)

This equation is stable for all $t > t_s$ if $\lambda_i(t) > 0$. It is asymptotically stable if $\int_{t_s}^{\infty} \lambda_i(\tau) d\tau = 0$.

For simulations with large initial errors function (3.5) has been applied:

$$\Lambda(t) = \begin{cases} \Lambda_{\max} t - \frac{\Lambda_{\max}}{2\pi} \sin \left( \frac{2\pi}{t_{\max}} t \right) & \text{for } t \leq t_{\max} \\ \Lambda_{\max} & \text{for } t > t_{\max} \end{cases}$$  \hspace{1cm} (3.5)

An asymptotically stable error function can be obtained, if $\Lambda_{\max}$ is a positive definite diagonal matrix. By applying this function a much smoother system can be obtained. In order to confirm this statement, an identical simulation as in section 2.2 has been performed.

$k = 10, \dot{\theta}^T(0) = [4, 2], \dot{\theta}^T(0) = [0, 1], \dot{\theta}(0) = \hat{\theta}, \quad t_{\max} = 10, \Lambda_{\max} = 20, K_r = 20, \Gamma = 0, tol. = 1e^{-3}$

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure3.png}
\caption{Reference velocity error and phase plot.}
\end{figure}
3.2.1 Variation of spring constant

In the first example the influence of the magnitude of the torsional elastic transmission is considered. Therefore $k$ is varied from $10$ to $10.000$ [Nm/rad]. Figures 4-7 present the simulation results for respectively $k=10$ and $k=10.000$ [Nm/rad]. Here, the direct adaptation algorithm has been applied, while in the simulations the analytical equations for $\dot{q}_{ur}$ and $\ddot{q}_{ur}$ have been used. Further, it is assumed that $\dot{q}$ and $q^{(3)}$ are exactly known.

As stated in [Lammerts et al., 1991] the range in which the resulting position errors fluctuate is independent of the magnitude of the stiffness $k$. This can be confirmed by comparing figure 5 and 7. Thus there seems to be no restriction to the upper-boundary of the stiffness $k$. However, a very stiff transmission ($k \to \infty$) causes very high frequencies in the input vector $[F^T \ 0^T]^T$. In practice this is undesirable because it can excite unmodeled dynamics. For simulation this results in large cpu times.

$k=10$, $\dot{e}(0)=4 \ 2$, $e(0)=0$, $\dot{\theta}(0)=0$, $\theta(0)=0$, $t_{\text{max}}=10$, $A_{\text{max}}=20$, $K_r=20$, $\Gamma=10$, $\text{tol.}=1e^{-3}$

figure 4: Simulation results for $k=10$ and without initial tracking errors.

$k=10$, $\dot{e}(0)=4 \ 2$, $e(0)=0$, $\dot{\theta}(0)=0$, $\theta(0)=0$, $t_{\text{max}}=10$, $A_{\text{max}}=20$, $K_r=20$, $\Gamma=10$, $\text{tol.}=1e^{-3}$

figure 5: Simulation results for $k=10$ and with large initial tracking errors.
Although still some rest-oscillation occurs, the time-varying sliding surface results in a much smoother behaviour, as is confirmed by comparison with corresponding simulations with a constant sliding surface. Further it appears (figure 6) that a large spring constant $k$ results in the impossibility to estimate the true motor rotor inertia $p(4)$. However, the total constant inertia $(p(3)+p(4)=20)$ is estimated correctly.

$k=1e4, \hat{e}(0)=0, \dot{e}(0)=0, \ddot{e}(0)=0, t_{\max}=0, \Lambda_{\max}=20, K_r=20, \Gamma=10, tol.=1e-3$

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure6.png}
\caption{Simulation results for $k=10000$ and without initial tracking errors.}
\end{figure}

Since the effort to reduce $\dot{e}_r$ is a function of the spring constant $k$, the components of the system matrices $W=W(q, \dot{q}, \ddot{q}, \dot{\theta})$ and $Y=Y(q, \dot{q}, \ddot{q}, t)$ and the vector $\dot{e}_r=\dot{e}+\Lambda e$ increase with the magnitude of the spring constant. This leads for $k=10,000$ to more rapidly fluctuating parameters as can be seen in figure 7.

$k=1e4, \hat{e}^r(0)=[4 2], \dot{\hat{e}}^r(0)=[0 1], \ddot{\hat{e}}(0)=0, t_{\max}=10, \Lambda_{\max}=20, K_r=20, \Gamma=10, tol.=1e-3$

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure7.png}
\caption{Simulation results for $k=10000$ and with large initial tracking errors.}
\end{figure}
3.2.2 Direct and Composite adaptation

In this section a comparison will be made between the direct and the composite adaptation algorithm. In order to obtain a fair comparison, identical (control) parameters have been used. Nevertheless, the adaptation gain matrix $\Gamma_{comp}$ is chosen $\Gamma_{comp} = 0.1 \times \Gamma_{direct}$, in order to avoid numerical difficulties, which are caused by the extreme value of the derivative $\dot{p}(4)$.

The simulations are performed for the theoretical situation. This implies that the true parameter vector $p$ is used for the reconstruction of $\dot{q}$ and $q^{(3)}$, as the analytical equations for the derivatives $\dot{q}_{ur}$ and $\ddot{q}_{ur}$ are applied.

The simulations are performed for the theoretical situation. This implies that the true parameter vector $p$ is used for the reconstruction of $\dot{q}$ and $q^{(3)}$, as the analytical equations for the derivatives $\dot{q}_{ur}$ and $\ddot{q}_{ur}$ are applied.
The estimation model to obtain a composite adaptation algorithm can be chosen fairly arbitrary. However, for the simulations described here the equations of motion are used. The transformation of the equations of motion in a suitable estimation model is described in section 2.2.3. Figures 10 and 11 give an impression of the performance of both the direct and the composite controller under almost identical circumstances.

Both figures show that the application of a composite adaptation algorithm results in a smoother behaviour of the parameter vector $\hat{\theta}$ combined with faster convergence. The smoother behaviour could be ascribed to the lower adaptation gain. However, smoothness appears to be characteristic for the composite controller. Therefore it is possible to use higher adaptation gains in comparison with the direct adaptation algorithm, without getting oscillatory behaviour. Due to the possibility of using high adaptation gains, smaller tracking errors and faster parameter convergence can be obtained, without exciting the high frequency unmodeled dynamics. This is a significant advantage in a practical situation.

The effect of composite adaptation can be given a simple intuitive interpretation. The direct adaptation algorithm has a gradient-like nature, because it can be written as:
3.2 Weak differentiation

In the previous sections the analytical first and second order time-derivatives of the reference trajectory \( q_{ur} \) have been applied for the computation of the input \( u_r \). However, this results in rather complex relations for \( \dot{q}_{ur} \) and \( \ddot{q}_{ur} \). Besides it is usually impractical or even impossible to reconstruct or measure the quantities \( \ddot{q} \) and jerks \( \dddot{q} \). A more practical solution is to use weak differentiators in order to determine \( q_{ur} \) and \( \dot{q}_{ur} \).

This gradient accounts for the poor tracking convergence when a large adaptation gain is used. Since \( \hat{p} = \dot{p} \) the composite adaptation law can be written in the following form:

\[
\dot{\hat{p}} = -\gamma \frac{\partial H_u}{\partial \hat{p}} \hat{e}_r
\]

This equation represents a time-varying low pass filter. This means that the parameter errors are now a filtered version of the gradient direction. Thus the parameter search in the composite adaptation goes along a filtered or averaged direction. This indicates that the parameter and tracking error convergence in composite adaptation control can be smoother and faster than in direct control. This conclusion is confirmed by the results represented in the figures above.

**3.2.3 Weak differentiation**

In the previous sections the analytical first and second order time-derivatives of the reference trajectory \( q_{ur} \) have been applied for the computation of the input \( u_r \). However, this results in rather complex relations for \( \dot{q}_{ur} \) and \( \ddot{q}_{ur} \). Besides it is usually impractical or even impossible to reconstruct or measure the quantities \( \ddot{q} \) and jerks \( \dddot{q} \). A more practical solution is to use weak differentiators in order to determine \( q_{ur} \) and \( \dot{q}_{ur} \).

\[
k = 10, \ \phi(0) = 0, \ \dot{\phi}(0) = 0, \ \ddot{\phi}(0) = 0, \ t_{\text{max}} = 0, \ \Lambda_{\text{max}} = 20, \ K_r = 20, \ \Gamma = 10, \ \alpha = 100
\]

\[
\begin{align*}
\text{tracking error} & \quad \text{ref. velocity error} \\
\text{ Input vector } u & \quad \text{estimated parameters}
\end{align*}
\]

**Figure 12:** Weak differentiation without initial tracking errors.
Transformation of a first order differentiator to the Laplace domain yields:

\[ y = \frac{d}{du} \rightarrow \mathcal{L} \rightarrow Y(s) = sU(s) = H(s)U(s) \]  

(3.7)

If the function \( H(s) = s \) is replaced by:

\[ H'(s) = \frac{\alpha s}{\alpha + s} \quad \text{with} \quad \alpha > 0 \]  

(3.8)

the following differential equation will be obtained:

\[ \alpha y + \dot{y} = \alpha u \]

\[ (y - \alpha u) + \alpha(y - \alpha u) = -\alpha^2 u \]  

(3.9)

Now by solving this first order differential equation (on-line), a simple algebraic equation for \( y \) can be obtained. A major disadvantage of this method is the necessity of using large gains (\( \alpha \)) which can result in large simulating times.

\[ h = 10, \; \dot{e}^T(0) = [4, 2], \; e^T(0) = [0, 1], \; \dot{\rho}(0) = 0, \; t_{\text{max}} = 10, \; \Lambda_{\text{max}} = 20, \; K_\tau = 20, \; \Gamma = 10, \; \alpha = 100 \]

\[ \text{figure 13: Weak differentiation with large initial tracking errors.} \]

If the initial errors are small, weak differentiation yields similar results as analytical differentiation, for reasonable values of \( \alpha \). However, large accelerations of \( q_u \) as may occur if large initial errors are present, require a very large break-frequency \( \alpha \) which will result in a considerable extension of the simulating time. On the other hand too small values of \( \alpha \) will result in a performance reduction as can be seen in figure 13.
3.2.4 Discrete-time simulations

In the preceding simulations only the continuous case is considered. In a practical situation however, this is not realizable since the computer need a certain amount of time for computing a new input. Meanwhile the manipulator is in motion. In order to examine this effect of discontinuity on the controller performance several simulations have been performed. In order to exclude other influences, the first simulation has been performed under theoretical circumstances.

\[ k=10, \; g(0)=0, \; \dot{g}(0)=0, \; \ddot{g}(0)=0, \; t_{\max}=0, \; A_{\max}=20, \; K_x=20, \; \Gamma=10, \; \Delta t=0.01 \]

![Figure 14: Discrete-time simulation results.](image)

Obviously, the effect of discontinuity results in a slight performance reduction. It is not astonishing that the time interval \( \Delta t \), in which the input vector \( u \) has a constant value, is bounded by a maximum. Although this kind of simulations produces some interesting results, it does not resemble reality, since the true parameter vector \( \theta \) is used for reconstruction. There are two possibilities to avoid the use of \( \theta \). The first method is based on the replacement of \( \theta \) by \( \hat{\theta} \). This possibility, however, is only applicable if an accurate estimate of \( \theta \) is available and the adaptation gains are small. These requirements are a result of the fact that the reconstruction of \( \hat{\theta} \) and \( \dot{\theta}^{(0)} \) is based on matrix inversion of \( \hat{M}(\theta, \hat{\theta}) \).

The second method is based on numerical differentiation of \( q_{\theta} \). For this purpose there are several algorithms available. Two possible approaches are described in appendix [C].
Figure 15 gives an impression of the results obtained from discrete time simulations with the Lagrange approximation algorithm (appendix [C]). Although the spring stiffness has been increased to \( k = 1000 \), in order to obtain a stable behaviour, still a fairly bad performance may be obtained. This is mainly due to the differentiating imprecision \( O(h^2) \), as can be concluded from comparison with the analytical discrete-time simulation.

### 3.2.5 Supplementary practical conditions

Besides the discontinuity effect and the impossibility of exactly reconstructing \( \dot{g} \) and \( g^{(3)} \), there are other circumstances in which the practical situation differs from the theoretical one, namely the presence of measurement noise, actuator saturation and model imperfection.

In order to get an impression of the robustness against measurement noise, actuator saturation and model imperfection several theoretical simulations have been performed. Figure 16 represents the effect of both measurement noise \( x_i = x_i + V_{w \text{rand}(normal)} \) with \( x^T = [q^T \dot{q}^T] \) and actuator saturation.
Chapter 3: Simulations.

$k = 10$, $e(0) = 0$, $\dot{e}(0) = 0$, $\beta(0) = 0$, $t_{\text{max}} = 0$, $\Lambda_{\text{max}} = 20$, $K_r = 20$, $\Gamma = 10$, $\nu w = 0.001$, $\vec{u}^T = [10 \, 50]$

![Graphs showing tracking error, reference velocity error, input vector, and estimated parameters.]

**Figure 16:** The effect of actuator saturation and measurement errors.

It appears that the robustness against measurement noise and actuator saturation is good. However, it is mentioned that the absolute saturation value $|\vec{u}_{\text{sat}}|$ is bounded by a minimum in order to avoid an unacceptable system behaviour. This manifests itself in a rapidly fluctuating input $\vec{u}$ and an increasing tracking error $\vec{e}$.

Model imprecision may come from actual uncertainty about the manipulator (e.g. unknown parameters) or from the purposeful choice of simplified representation of the system's dynamics (e.g. modelling friction as linear, or neglecting structural modes in a reasonably rigid mechanical system). It is usually appropriate in control design to distinguish between structured and unstructured modelling uncertainties. Structured uncertainties correspond to inaccuracies in the parameters of the model or to additive disturbances (e.g. stiction), while unstructured uncertainties reflect errors on the system order, i.e. unmodeled dynamics (e.g. neglecting motor dynamics). If modelling is adequate, the unmodeled dynamics are of high frequency.

Suppose that the distinction between the true manipulator model and the model used for control purpose is represented by:

$$\eta(\vec{q}, \dot{\vec{q}}, \ddot{\vec{q}}, \vec{p}, t)$$

(3.10)

This yields the following expression for $\bar{V}$:

$$\bar{V} = -\vec{e}^T K \vec{e} - (\bar{\beta}^T Y(t)^T Y(t) \dot{\vec{q}}) + \dot{\vec{e}}^T \eta$$

(3.11)
Obviously it is impossible to guarantee stability if structured uncertainties are present. Figure 17 represents the results in case the presence of Coulomb friction in the true manipulator model is neglected.

\[ k = 10, \quad \dot{e}(0) = 0, \quad \dot{\theta}(0) = 0, \quad \lambda_{\max} = 0, \quad \Lambda_{\max} = 20, \quad K_r = 20, \quad \Gamma = 10, \quad \mu = [0.5, 0.5, 0.5] \]

![Simulation results for model errors.](image)

**Figure 17**: Simulation results for model errors.

Though global stability can't be ensured, the robustness properties for structured uncertainties are good. Furthermore, \( \dot{\theta}(t) \) does not converge to a constant value, since an additive disturbance \( \ell_{\omega} = \mu \cdot \text{sign}(\dot{q}) \) is present in the manipulator model.
Chapter 4 Experiments

Because the simulations in the previous chapter showed reasonable results, it is worthwhile to test the adaptive CTCRC law under practical circumstances. The practical use of adaptive CTCRC has been verified by means of experiments on a flexible RT-manipulator. This mechanical system isn't deliberately constructed with a torsional transmission. Therefore the spring constant $k$ isn't adjustable and is rather large ($k = 2 e^6 [Nm/rad]$). Although a large spring constant is in accordance with the reality, it does not clearly express the benefits of a flexible control law. For the experiments the same manipulator model as for the simulations has been used. Obviously this model does not exactly describe the reality, since there are structured uncertainties as well as unmodeled dynamics.

Here only an global survey is given of the most important implementation issues. For a more detailed description of the test apparatus and the controller hardware, the reader is referred to, for instance, [v. Oosterhout, 1992]. In order to ensure a constant timing an interrupt timer is used, which issues an interrupt after a specified time interval $\Delta t$. This interrupt activates the corresponding interrupt-service routine which will be described later on. To ensure that the timer interrupt has the highest priority, some other interrupt requests must be inhibited. During all experiments the same sample time (5ms) has been used in order to obtain a "fair" comparison between the different control laws. This sample time is the maximal achievable on a 386 sx 16 personal computer, in case the adaptive CTCRC algorithm is applied.

$$u = K_r (\dot{e} + \Lambda \dot{e}) = 5000 (\dot{e} + 20\dot{e})$$

\[\begin{array}{c}
\text{trans. error} \\
\text{rot. error} \\
\text{trans. input} \\
\text{rot. input}
\end{array}\]

\textit{figure 18:} Experimental results for PD-control.

The interrupt-service routine sequentially performs the following tasks. First it is verified whether a new input voltage is available. If not the program is terminated. Hereafter a signal is send to the hardware in order to latch the link positions.
However, it is not possible to latch $q_s$ and $q_u$ simultaneously, since both positions are stored on a different interface board. This fault results in the following fictive relative replacement:

$$\Delta q = (q_u - q_s)_{nc} = q_s \Delta \Delta q_{latch} \tag{4.1}$$

After successively sending the motor voltages and reading the link positions a new input vector $y^T = [F^T 0 M]$ is computed. This input vector is transformed in a motor voltage, using a dynamical motor model (appendix [D]).

Another important aspect concerns the fact that the manipulator is not equipped with tachometers. Therefore the velocities have to be reconstructed on-line. Here a discrete implementation of a filtered differentiator, as described in appendix [C], is used. This approximate procedure can actually be interpreted as building a reduced-order observer for the system. However, it does not use an explicit model, so that the system can be nonlinear and its parameters unknown [Slotine and Li, 1991].

Apart from adaptive CTCRC, a number of other control techniques has been implemented, (such as PD-control and the control law of Slotine and Li) to obtain a clear impression of the properties of the new control algorithm. Both adaptive controllers use the direct adaptation algorithm as described in section 2.2.2. Figures 18-20 represent the characteristic tracking results that are obtained with the three different control algorithms.
The control objective for those experiments is to track the following desired trajectory in joint-space:

\[
\begin{bmatrix}
q_{ad} \\
q_{ad}
\end{bmatrix} =
\begin{bmatrix}
A_t \sin(\omega_t t) \\
A_r \sin(\omega_r t)
\end{bmatrix}
\]  

(4.2)

with \(A_t = 0.1, A_r = 0.2, \omega_t = 0.5\) and \(\omega_r = 0.1\). Since the manipulator is initially at rest, initial velocity errors \(\dot{\mathbf{e}}^T = [A_t \omega_t, A_r \omega_r] \) are present. The initial position errors \(\mathbf{e}\) on the other hand are equal to zero.

Adaptive CTCRC (fig. 20) results in a small increase of the tracking error of the rotation module in comparison with the other two control laws (fig. 18 and 19). This phenomenon is probably caused by the numerical differentiating imprecision combined with the positive effect of the large spring stiffness on the tracking accuracy in case the flexibility is not taken into account. Furthermore the fictive replacement \((q_u - q_e)^{fe}\) affects the tracking accuracy in the negative, since it represents a fictive torque \(k(q_u - q_e)^{fe}\) on the rotation module.

Although rather small operating speeds have been used for the above comparison, similar results can be obtained for fast movements. Figure 21 gives an example of such a movement while figure 22 presents the corresponding parameter estimates \(\hat{\theta}(t)\). It appears that the estimated parameters fluctuate around a constant value, therefore unmodeled dynamics or structured uncertainties must be present in the system. Structured uncertainties may be caused by inaccuracies in the constant "known" system parameters \(k\) and \(b\) or by the presence of Coulomb friction. The initial value of the estimated parameter vector \(\hat{\theta}(0)\) has been obtained by off-line identification.
Chapter 4: Experiments.

Though the tracking errors increase drastically for fast movements, the robustness for unmodeled dynamics etc. appears to be good. On the whole the performance of the closed-loop system strongly depends on the control and adaptation parameters. Increasing these parameters leads to higher control activity, quicker convergence of the parameter estimates and a better tracking accuracy.

\[
\begin{bmatrix} 150 & 100 & 200 & 50 \end{bmatrix}, \Gamma = 10I, K_c = 500I, \Lambda = 5I
\]

![Figure 21: Experimental results for the ACTCRC algorithm.](image)

However, increasing them too much led to instability, because in that case unmodeled high-frequency dynamics are excited too much. To avoid stability problems a compromise with respect to the parameter choice has been made between tracking accuracy and robustness for measurement noise, disturbances and unmodeled dynamics. Here, the parameter choice is bounded by a safety-loop which stops the experiment if too large derivatives in \(u(t)\) occur (in this case at \(t=0\)).

\[
\begin{bmatrix} 150 & 100 & 200 & 50 \end{bmatrix}, \Gamma = 10I, K_c = 500I, \Lambda = 5I
\]

![Figure 22: Estimated parameters.](image)

The experiments have been performed using a PC-386 sx 16. Faster control hardware offers the opportunity to use a higher sampling rate and, thus, the opportunity to increase the control and adaptation gains. Improved controller hardware is expected to yield improved tracking performance.
Chapter 5 Conclusions and recommendations

- From simulations and experiments it appeared that CTCRC is a suitable algorithm for controlling manipulators which are driven via elastic motor transmissions. An important property of the controller is its robustness for unmodeled dynamics, measurement noise and other model imperfections. Furthermore, the magnitude of the spring constant $k$ does not influence the range in between the tracking errors and their time derivatives fluctuate. However, for a rather stiff manipulator unacceptable high frequency signals may occur in the motor inputs.

- To cope with the parametric uncertainty an adaptive version of CTCRC can be applied. Adaptive control is an appealing approach for controlling parameter-uncertain systems. During the simulations two possibilities have been considered, i.e. direct and composite adaptation. The latter not only extracts parameter information from the tracking errors but from the prediction errors as well. The direct adaptation law has a gradient-like nature which accounts for the poor parameter convergence when a large adaptation gain is used. Composite adaptation goes along a filtered, or averaged gradient direction. This indicates that the parameter and tracking error convergence in composite adaptation can be smoother and faster then in direct adaptive control. This smoothness is of great advantage in case unmodeled dynamics are present.

- If the first and the second derivatives of the unknown reference coordinates $\dot{q}_u$ are derived by numerical differentiation full state availability is required. Use of the analytical derivative also demands the availability of $\dot{q}_u^{(3)}$ and $\dot{q}_u^{(4)}$. For practical implementation full state availability is a necessary and sufficient requirement. If not all state variables are measured a suitable observer must be designed to estimate the unknown deformations and velocities.

- In order to obtain a smooth system behaviour in case large initial position errors are present, a time depending sliding surface has been chosen: $\dot{e} = \dot{e} + \Lambda(t)e$

- The performance of the system strongly depends on an appropriate tuning of the control gains. As a general theory for obtaining control gains is not (yet) available in nonlinear control system design, the simulations are performed with gains obtained by trail and error.

- The transmission flexibility is bounded by a minimum for adaptive CTCRC to be advantageous. This is mainly due to the differentiation imprecision.
Chapter 5: Conclusions and recommendations.

- Since differentiating $q_r$ is the bottleneck of the algorithm, more attention should be paid to the derivation of precise differentiation schemes.

- Another topic for future research, is to compare CTCRC with the input-output feedback linearization approach and the method of singular perturbation.
Bibliography


With the Lagrange formalism it is possible to derive the equations of motion. The Lagrange equations are:

\[
\frac{d}{dt} \left( \frac{\partial T}{\partial \dot{q}} \right) - \frac{\partial T}{\partial q} + \frac{\partial V}{\partial q} = Q^*
\]  (A.1)

where \( T=T(q,\dot{q},\dot{p}) \) represents the kinetic energy of the mechanical system, while \( V=V(q,p) \) accounts for the potential (gravitational and elastical) energy. The vector \( Q^*=Q^*(q,\dot{q},\dot{p},t) \) accounts for all non-conservative torques (internal as well as external).

![Figure 23: RT-manipulator.](image)

Kinetic energy of the mechanical system:

\[
T = T_{\text{trans.motor}} + T_{\text{trans.module}} + T_{\text{payload}} + T_{\text{rot.module}} + T_{\text{transmission}}
\]  (A.2)

For example:

\[
\xi = l - q_s
\]

\[
T_{\text{trans.motor}} = T_z + T_{\text{rounds}}
\]

\[
L_z(q) = \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a - \xi \cos(q_s) \\ b - \xi \sin(q_s) \end{bmatrix}
\]  (A.3)

\[
u_z(q,q) = \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} -\xi \cos(q_s) + \xi \sin(q_s)q_z \\ -\xi \sin(q_s) - \xi \cos(q_s)q_z \end{bmatrix}
\]
The kinetic energy is a homogeneous quadratic function of the generalized velocities $q$ and can be written as:

$$T = \frac{1}{2}(J_m + J_i + J_t)q_e^2 + \frac{1}{2}(m_m + m_i + m_1)q_s^2 + \frac{1}{2}m_m(q_e - l)^2q_e^2 + \frac{1}{2}m_i(q_s - l)^2q_s^2 + \frac{1}{2}m_s^2q_e^2 + \frac{1}{2}J_{rot}q_e^2 + \frac{1}{2}J_{irm}q^2$$

(A.4)

This symmetric inertia matrix $M$ is positive definite because the kinetic energy is strictly positive unless the manipulator is at rest. It is noted that $M$ may not explicitly depend on time $t$, since in that case the Lagrange formalism in the form as stated here cannot be used. With $C = \frac{1}{2}M + \frac{1}{2}(G - G^T)$ and $g_i = \frac{\partial M}{\partial q_i}$ an expression for the $C$ matrix can be obtained [Lammerfs, 1993]:

$$G = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 2(\alpha q_s - \beta q_e) & 0 \\ 0 & 0 & 0 \end{bmatrix} \Rightarrow C = \begin{bmatrix} 0 & -2(\alpha q_s - \beta q_e) & 0 \\ 2(\alpha q_s - \beta q_e) & 2(\alpha q_s - \beta q_e) & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

(A.6)

This $C$ matrix meets the requirement that $(C - \frac{1}{2}M)$ is a skew-symmetric matrix.

**Potential energy of the mechanical system:**

The potential energy $V$ for a rigid manipulator with flexible joints consists of a contribution $V_{el}$ due to the deformation of the joints plus a contribution $V_{cons}$ due to the
conservative external loading, such as gravity.

\[ V = V_u + V_{\text{cons}} = \frac{1}{2}k(\phi - \phi_m) \] (A.7)

The elastic energy can be written as a product of a symmetric, semi-positive definite stiffness matrix and the generalized coordinates \( \phi \):

\[ V = \frac{1}{2}q^T K q \quad \Rightarrow \quad K = \begin{bmatrix} 0 & 0 & 0 \\ 0 & k & -k \\ 0 & -k & k \end{bmatrix} \] (A.8)

Virtual work:

The virtual work \( \delta A = \delta q^T \dot{Q}^* \) done by all non-conservative torques during a variation \( \delta q \) of \( q \) can be split up in a component \( \delta A_u \) generated by the input vector \( u \) and a component \( \delta A_i \) representing all other (non-conservative) torques in and on the manipulator.

\[ \delta A = \delta A_i + \delta A_u \]

\[ \delta A_i = -b_i \dot{q}_i \delta q_i - b_e \dot{q}_e \delta q_e - b_u \dot{q}_u \delta q_u \]

\[ \delta A_u = F \delta q_s + M \delta q_u \] (A.9)

This can also be formulated as \( \delta A_i = -\delta q^T \dot{B} \dot{q} \) and \( \delta A_u = \delta q^T H u \) where:

\[ B = \begin{bmatrix} b_i & 0 & 0 \\ 0 & b_e & 0 \\ 0 & 0 & b_u \end{bmatrix}; \quad H = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}; \quad u = \begin{bmatrix} F \\ T \end{bmatrix} \] (A.10)

Substitution of (A.5), (A.6), (A.8) and (A.10) in the Lagrange equations yields the following equations of motion:

\[ M \ddot{q} + C \dot{q} + B \dot{q} + K q = Hu \] (A.11)
Appendix B: Variant of adaptive CTCRC

In this appendix a variant of the adaptive CTCRC algorithm as described in section 2.2 will be examined. The variation is based on the introduction of a supplementary term $M_{\tilde{e}_r}$. This yields:

$$Hu = M(q,p)\tilde{q}_r + C(q,p) + B\tilde{q} + K\tilde{e}_r + M_{\tilde{e}_r}$$  \hspace{1cm} (B.1)

where $M_r \in R^{(n \times n \times n)}$ is a positive definite matrix. Further it is required that $M_r$ has the following form:

$$M_r = \begin{bmatrix}
M_{ru} & 0 & 0 \\
0 & M_{re} & M_{reu} \\
0 & M_{reu} & M_{ru}
\end{bmatrix}$$  \hspace{1cm} (B.2)

Now the error equation results in:

$$(M + M_r)\tilde{e}_r + (C + K_r)\tilde{e}_r + K\tilde{e}_r = -W_{r,\tilde{e}}$$  \hspace{1cm} (B.3)

In accordance with the standard adaptive CTCRC algorithm it is also possible to guarantee asymptotical stability for this error equation. By writing equation (B.1) in the component form following equations can be obtained:

$$Hu = \begin{bmatrix}
\tilde{u}_s = [\tilde{M}_s \tilde{C}_s] \tilde{q}_r + [\tilde{C}_s \tilde{C}_e] \tilde{q}_u + \tilde{B} \tilde{q}_s + \tilde{g}_s + K_{s,e} \tilde{e}_{se} + M_{s,e} \tilde{e}_e \\
\tilde{u}_e = [\tilde{M}_e \tilde{C}_e] \tilde{q}_r + [\tilde{C}_e \tilde{C}_e] \tilde{q}_u + \tilde{B} \tilde{q}_s + K(q_{er} - q_{ur}) + \tilde{g}_e + K_{e,e} \tilde{e}_{se} + M_{e,e} \tilde{e}_e + M_{e, u} \tilde{e}_{u} \\
\tilde{u}_u = \tilde{M}_{uu} \tilde{q}_u + \tilde{B}_u \tilde{q}_u + K(q_{er} - q_{ur}) + \tilde{g}_u + K_{u,e} \tilde{e}_{se} + M_{u,e} \tilde{e}_e + M_{u, u} \tilde{e}_{u}
\end{bmatrix}$$

The advantage of this new algorithm should consist of the fact that $\tilde{q}_u$ is directly available in the second equation, since $\tilde{e}_u = \tilde{q}_u - \tilde{q}_r$. Therefore the necessity to differentiate $\tilde{q}_r$ twice disappears. However, since $\tilde{u}_u$ now becomes a function of $\tilde{q}_u$ and $\tilde{q}_u = f(q_u, \tilde{q}_u, \tilde{u}_u)$ the problem arises that $\tilde{q}_u$ doesn't necessarily have to be bounded.
Appendix C: Discrete-time differentiation

Numerical integration and differentiation are sometimes explicit parts of a controller design. Numerical differentiation may avoid the complexity of constructing the whole system state based on partial measurements (the non-linear observer problem).

Numerical differentiation can, for instance, be performed by applying a filtered differentiation of the form:

\[
\frac{s}{s+\alpha} = 1 - \frac{\alpha}{s+\alpha}
\]  

(C.1)

where \( s \) is the Laplace variable and \( \alpha \geq 1 \). The time discrete implementation of above equation, assuming e.g. a zero-order hold, is simply an addition:

\[
\dot{g}_{\text{new}} = a_1 \dot{g}_{\text{old}} + a_2 \dot{g}
\]  

(C.2)

where the constants \( a_1 \) and \( a_2 \) are defined as:

\[
a_1 = -\alpha e^{-\alpha T} \quad a_2 = 1 - \alpha (1 - e^{-\alpha T})
\]  

(C.3)

and \( T \) is the sampling period. Note that this approximate procedure can actually be interpreted as building a reduced order observer for the system. However, it does not use an explicit model, so that the system can be non-linear and its parameters unknown.

Another differentiation algorithm is based on the approximation of a function \( f(x) \) by Lagrange's form of the unique interpolating polynomial for the \( n+1 \) sampled nodes:

\[
f(x) = p_n(x) = f(x_0)L_0(x) + \ldots + f(x_n)L_n(x)
\]  

(C.4)

where \( L_i(x) \), the \( i \)'th Lagrange polynomial, is given by:

\[
L_i(x) = \prod_{\substack{j=0 \atop j \neq i}}^{\text{n}} \frac{(x-x_j)}{(x_i-x_j)} \quad \text{for} \quad i=0..n
\]  

(C.5)
By differentiating the Lagrange polynomial \( k \) times an approximate for the \( k \)'th derivative can be obtained:

\[
f^{(k)}(z) = p_n^{(k)}(z) = L_0^{(k)}(z)f(x_0) + \ldots + L_n^{(k)}(z)f(x_n)
\]  
(C.6)

For any given \( n+1 \) sampled nodes \( x_i \), the above \((n+1)\)-point approximation formula has exactness degree at least \( n \). The guaranteed exactness when \( f(x) \) is a polynomial of degree \( \leq n \) follows immediately from the fact that \( p_n(x) \) must equal \( f(x) \) in this case. Formula (C.6) is very general and can be used whether or not the \( x_i \)'s are equispaced, and whether or not \( z \) is a sampled node \( x_i \).

In case the sampled nodes are equispaced, following \( O(h^2) \) backward difference formula in which \( f_j = f(z + jh) \) can be obtained.

\[
\begin{align*}
\hat{f}(z) &= \frac{1}{2h} [-3f_0 + 4f_1 - f_2] \\
\hat{f}(z) &= \frac{1}{h^3} [2f_0 - 5f_1 + 4f_2 - f_3]
\end{align*}
\]  
(C.7)
Appendix D: Dynamical model DC-motor

The general model which describes the dynamical behaviour of a DC-motor is given in equation (D.1).

\[ L \frac{d}{dt} i_a(t) + R_a i_a(t) + K_e \phi_m(t) = u(t) \]  
\[ \text{(D.1)} \]

where \( u(t) \) is the input voltage of the DC-motor, \( R_a \) is the armature resistance, \( K_e \) a constant motor parameter, \( i_a(t) \) is the armature current and \( \phi_m(t) \) is the rotational velocity of the motor.

A part of the total input voltage \( u(t) \) at the motor will be lost in the brushes. This part is called the brush-contact loss voltage \( u_b \). Further it is assumed that the first term is negligible small. Then the motor equation becomes as follows:

\[ R_a i_a(t) + K_e \phi_m(t) = u(t) - u_b \]  
\[ \text{(D.2)} \]

The torque delivered by a DC-motor is linear in the armature current \( i_a(t) \):

\[ M(t) = K_T i_a(t) \]  
\[ \text{(D.3)} \]

Substitution of equation (D.3) in equation (D.2) delivers:

\[ \frac{R_a}{K_T} M(t) + K_e \phi_m(t) = u(t) - u_b \]  
\[ \text{(D.4)} \]

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

\[ M(t) = \frac{K_e}{R_a} [u(t) - u_b] - \frac{K_e^2}{R_a} \phi_m(t) \]  
\[ \text{(D.5)} \]

During the experiments this formula was used to calculate the necessary input voltage \( u(t) \) on basis of a desired torque \( M(t) \).