Eindhoven University of Technology
Department of Electrical Engineering
Measurement and Control Group

Timeoptimal position control of an induction motor based on the Dynamic Contraction Method

by M.v. Duijnhoven

Master of Science thesis

Country:	Poland
Place:	Gliwice
From:	1 may 1998 till 1 september 1998
Supervisors:	Dr. Ir. M. Błachuta, Prof. Dr. Ir. P.P.J. v.d. Bosch

Time optimal position control of an induction motor based on the Dynamic Contraction Method

M. v. Duijnhoven

October 1998

Abstract

Due to the decreasing prices in semi conductors the induction motor is getting more and more popular for controlled speed or position applications.
The induction motor is a nonlinear system and not all state variables are necessarily measurable. The parameters, especially the rotor resistance, vary significantly from their nominal values.
Field oriented control is applied to decouple the control inputs. The first problem is to control T_e, the electrical torque. This is a function of different parameters which are nonlinear dynamically coupled. The Dynamic Contraction Method (DCM) is applied to the AC-motor to control T_e. This is a method similar to Nonlinear Inverse Dynamics (NID) but it can be used in systems with uncertain parameters. A comparison between PI and DCM was made if the input was stepwise. In this case the DCM controller was superior.
After controlling the current and flux with DCM controllers, the motor is described by a two integrator problem with an extra time constant. The goal of this report is to solve this system as a pure two integrator problem without taking resort to the two integrator problem with an extra time constant, which is considerably more difficult to solve and implement. The time optimal solution consists of two intervals of maximum positive/negative electrical torque. This results in maximum acceleration/deceleration. If the position error is large enough there is a third interval with maximum speed and zero acceleration.
If the system can be approximated by a two integrator problem depends on the motor parameters. As will be shown our motor can be approximated by the two integrator problem.
Two control structures, a non-feed forward and a feed forward, which solve the two integrator problem are presented. The non-feed forward structure is superior for larger position errors if torque is not known. This last structure is compared with a control structure from [1]. The article control structure had some disadvantages. Our current and flux controller can use higher gains which result in smaller steady state errors and higher robustness.
As last part of this thesis observers are applied to the motor. The most problematic part is the estimation of the flux angle, if the rotor resistance is different than assumed. All other estimations depend on the proper observation of the flux angle.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The last 25 years there was a fast development of new power semi-conductors, digital signal processors and digital electronics. Due to the availability of these new devices and the price decrease a lot of research is done in the field of controlling AC-drives.
The AC-drive has several advantages above the DC-drive. The AC-drive is brushless and therefore maintenance free (no replacing of the brushes), and has a cheaper construction. This all makes the AC-drive very popular. Therefore it is gradually replacing the DC-drives in applications with controlled drives. However controlling the AC-drive is much more complicated than the DC-drive. It is a nonlinear system and not all state variables are necessarily measurable. The parameters, especially the rotor resistance, vary significantly from their nominal values. The controllers should be robust against these effects. In the DC-drive the electrical torque can be controlled directly by the rotor current however this is not the case in the AC-drive. This all makes it a real challenge to control it. Field oriented control is used on the AC-drive to keep the flux constant. The AC-drive control is now similar to the DC-drive control.
Different approaches have been successfully applied to the AC-drive, sliding mode controllers [2], adaptive controllers [3], and maximization of torque [4].
The motor can be used in different applications. For example in elevators, cranes, or even in not so humane application as a large ammunition loader as described in [5]. The applications can be split into two categories. One with symmetric and one with non-symmetric torque. Friction, or horizontal movements of ballast cause symmetric torque. A typical non-symmetric torque occurs in cranes, and elevators. The non-symmetrical problem is more difficult to control as the torque will not always act in opposite direction of the desired movement. This problem is therefore most interesting, and will be analyzed in detail in this report.
Assuming that T_e is controlled, the motor equations are given by:

$$\begin{aligned} \frac{d\theta}{dt} &= \omega \\ J\frac{d\omega}{dt} &= T_e - T_l, \end{aligned} \tag{1.1}$$

with J inertia, T_e electrical torque, and T_l load torque. A motor has several electrical and mechanical limits, which can be translated to speed and acceleration limits. This system can be solved. The time optimal solution consists of two intervals of maximum positive/negative electrical torque. This results in maximum acceleration/deceleration. If the position error is large enough there is a third interval with maximum speed and zero acceleration.

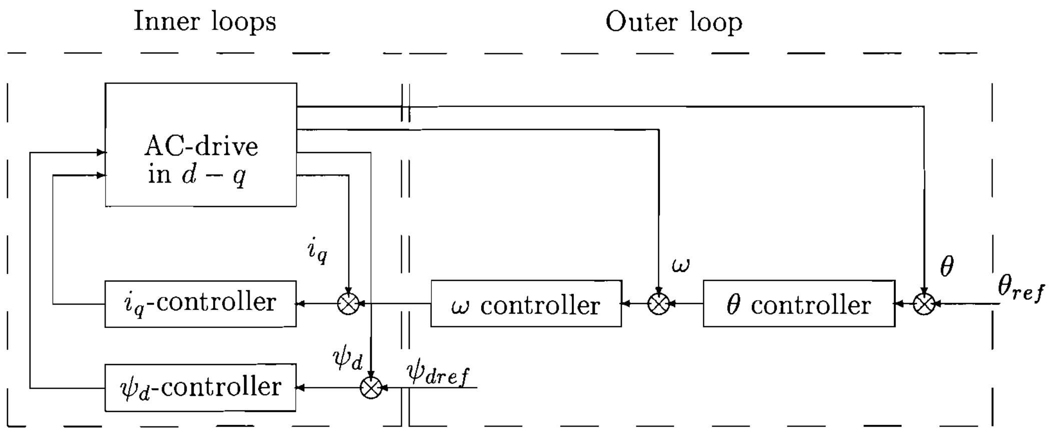


Figure 1.1: AC-motor control problem

In the AC-drive T_e is a function of different parameters, which are nonlinear dynamically coupled. The first problem is to control the T_e. The electrical torque can not be changed stepwise, as the motor is a two integrator system with an extra time constant. The goal of this report is to solve this system as a two integrator problem without taking resort to the two integrator problem with extra time constant, which is considerably more difficult to solve and implement.
In [1] the control problem is solved with PI controllers which can not use as high gains as in our setup.
Recently a new nonlinear control method, the Dynamic Contraction Method (DCM), was developed [6]. Good results were obtained by applying this method on an airplane control problem [7]. This method offers good performance and works well in uncertain nonlinear systems. The DCM method will be applied to the AC-motor to control T_e.
There is assumed that position, and the drive currents are measured. It would be much cheaper to observe position too, but this will degrade our control performance. Flux, flux-angle, speed and torque are observed.
The drive parameters are normally not know. There is assumed that these are known. The identification of these parameters is an other problem, and this will not be treated here (see [8]). The parameters of [3] are used in this report. In chapter 2 the AC-motor model and field oriented control is introduced. In chapter 3 the Dynamic Contraction Method (DCM) is introduced, and applied to the induction motor. The control problem can be split in two different parts as shown in figure 1.1. In chapter 4 the innerloops are analyzed and a comparison is made between DCM and PI controllers. In chapter 5 a time optimal control is designed (the outer loops). In chapter 6 observers are added. The induction motor model is simulated in Simulink ^{TM}.

Chapter 2

The induction motor

2.1 Introduction

A mathematical model for a sinusoidally wound induction motor is presented in this section. This model may also be used for squirrel cage rotors. If the inverter is operating at her voltage maximum limit, the voltages are no longer sinusoidal. This model can also used with these non sinusoidal signals.
After modeling, the theory of field oriented control is introduced and applied to the AC-drive. There is assumed that the AC-drive is fed by a voltage source inverter. This type of inverters have a good dynamic performance and smooth operation at stand still, therefore it is ideal for servo applications.

2.2 Modeling the AC-drive

An induction motor normally consists of three stator windings and three rotor windings. Kraus and Thomas [9] introduced a two phase equivalent representation which is used in this report. The stator current i_s can be written as:

$$i_s(t) = i_{s1} + i_{s2}e^{j\gamma} + i_{s3}e^{j2\gamma} = i_{sa} + ji_{sb} \tag{2.1}$$

$$\vec{i_s} = \begin{pmatrix} i_{sa} \\ i_{sb} \end{pmatrix} = i_{s1}\begin{pmatrix} 1 \\ 0 \end{pmatrix} + i_{s2}\begin{pmatrix} -\frac{1}{2} \\ \frac{1}{2}\sqrt{3} \end{pmatrix} + i_{s3}\begin{pmatrix} -\frac{1}{2} \\ -\frac{1}{2}\sqrt{3} \end{pmatrix} \tag{2.2}$$

This can also be rewritten as:

$$\begin{pmatrix} i_{sa} \\ i_{sb} \end{pmatrix} = \mathcal{A}_p \begin{pmatrix} i_{s1} \\ i_{s2} \\ i_{s3} \end{pmatrix}, \tag{2.3}$$

where matrix $\mathcal{A}_p$ is given by:

$$\mathcal{A}_p = \sqrt{\frac{2}{3}}\begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{1}{2}\sqrt{3} & -\frac{1}{2}\sqrt{3} \end{bmatrix} \tag{2.4}$$

The factor $\sqrt{2/3}$ ensures power invariance of a three-phase system with the two-phase equivalent. The inverse transformation can be calculated from $\mathcal{A}_p^T$. The two phase equivalent will be used to model and simulate the motor. As the motor's power is supplied by a Voltage

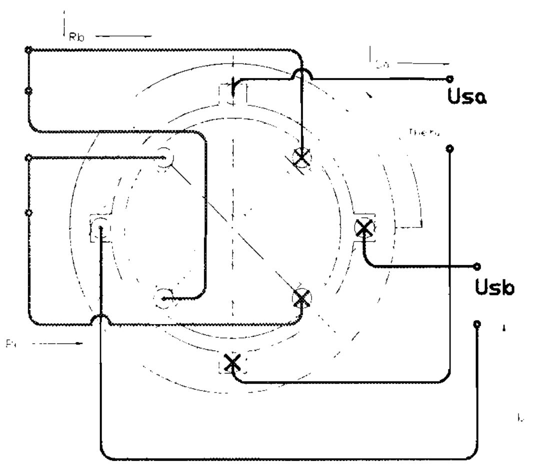


Figure 2.1: Schematic representation of a 2-phase induction motor

Source Inverter the control inputs are stator voltages. The dynamics of the induction motor are described by:

$$\begin{aligned} R_s i_{sa} &+ \frac{d\psi_{sa}}{dt} = u_{sa} \\ R_s i_{sb} &+ \frac{d\psi_{sb}}{dt} = u_{sb} \\ R_r i_{rd'} &+ \frac{d\psi_{rd'}}{dt} = 0 \\ R_r i_{rq'} &+ \frac{d\psi_{rq'}}{dt} = 0 \end{aligned} \tag{2.5}$$

u_s=	stator voltage [V]
R=	resistance [Ω]
ψ=	flux [Wb]
ω=	speed [$rads^{-1}$]
n_p=	the number of pole pairs

(a,b) denotes the components of a vector with respect to a fixed stator reference frame. (d',q') denotes the components of a vector with respect to a frame rotating at speed $n_p w$.
The angle-speed condition is given by:

$$\begin{aligned} \frac{d\delta}{dt} &= n_p\omega \\ \delta(0) &= 0 \end{aligned} \tag{2.6}$$

The vectors $(i_{rd'}, i_{rq'})$, $(\psi_{rd'}, \psi_{rq'})$ in the rotating frame (d', q') are transformed to the vectors (i_{ra}, i_{rb}), (ψ_{ra}, ψ_{rb}) in the stationary frame (a, b).

$$\begin{aligned}\begin{pmatrix}\psi_{rd'}\\ \psi_{rq'}\end{pmatrix} &= \begin{bmatrix}\cos(\delta) & \sin(\delta)\\ -\sin(\delta) & \cos(\delta)\end{bmatrix}\begin{pmatrix}\psi_{ra}\\ \psi_{rb}\end{pmatrix}\\ \begin{pmatrix}i_{rd'}\\ i_{rq'}\end{pmatrix} &= \begin{bmatrix}\cos(\delta) & \sin(\delta)\\ -\sin(\delta) & \cos(\delta)\end{bmatrix}\begin{pmatrix}i_{ra}\\ i_{rb}\end{pmatrix}\end{aligned} \tag{2.7}$$

Applying the transformation matrices (2.7) to Eq. (2.5) and using Eq. (2.6) gives:

$$\begin{aligned}R_s i_{sa} + \frac{d\psi_{sa}}{dt} &= u_{sa}\\ R_s i_{sb} + \frac{d\psi_{sb}}{dt} &= u_{sb}\\ R_r i_{ra} + \frac{d\psi_{ra}}{dt} + n_p\omega\psi_{rb} &= 0\\ R_r i_{rb} + \frac{d\psi_{rb}}{dt} - n_p\omega\psi_{ra} &= 0\end{aligned} \tag{2.8}$$

If saturation and iron losses are neglected the fluxes can be written as:

$$\begin{aligned}\psi_{sa} &= L_s i_{sa} + M i_{ra}\\ \psi_{sb} &= L_s i_{sb} + M i_{rb}\\ \psi_{ra} &= L_r i_{ra} + M i_{sa}\\ \psi_{rb} &= L_r i_{rb} + M i_{sb}\end{aligned} \tag{2.9}$$

Eliminating i_{ra}, i_{rb} and ψ_{sa}, ψ_{sb} in Eq. (2.8) by using Eq. (2.9) gives:

$$\begin{aligned}R_s i_{sa} + \frac{M}{L_r}\frac{d\psi_{ra}}{dt} + \left(L_s - \frac{M^2}{L_r}\right)\frac{di_{sa}}{dt} &= u_{sa}\\ R_s i_{sb} + \frac{M}{L_r}\frac{d\psi_{rb}}{dt} + \left(L_s - \frac{M^2}{L_r}\right)\frac{di_{sb}}{dt} &= u_{sb}\\ \frac{R_r}{L_r}\psi_{ra} - \frac{R_r}{L_r}M i_{sa} + n_p\omega\psi_{rb} &= 0\\ \frac{R_r}{L_r}\psi_{rb} - \frac{R_r}{L_r}M i_{sb} - n_p\omega\psi_{ra} &= 0\end{aligned} \tag{2.10}$$

In Eq. (2.10) the ω terms are the speed voltage terms. Multiplying the speed voltage terms with their phase currents and adding these two terms gives the power absorbed in the rotor.

$$P_{rotor} = n_p\omega\psi_{rb}i_{ra} - n_p\omega\psi_{ra}i_{rb} \tag{2.11}$$

The rotor power equals:

$$P_{rotor} = \omega T_e \tag{2.12}$$

Where T_e is the electrical torque. Eliminating the rotor currents using Eq. (2.9) (2.12) gives:

$$T_e = \frac{n_p M}{L_r}(\psi_{ra}i_{sb} - \psi_{rb}i_{sa}) \tag{2.13}$$

Substituting (2.13) in the well known torque equation

$$J\frac{d\omega}{dt} = T_e - T_L, \tag{2.14}$$

gives:

$$\frac{d\omega}{dt} = \frac{n_p M}{J L_r}(\psi_{ra} i_{sb} - \psi_{rb} i_{sa}) - \frac{T_L}{J} \tag{2.15}$$

where T_L is the load torque. Rearranging Eq. (2.10) and adding the torque Eq. (2.15) gives the 6th order asynchronous motor model:

$$\begin{aligned}
\frac{d\theta}{dt} &= \omega \\
\frac{d\omega}{dt} &= \frac{n_p M}{J L_r}(\psi_{ra} i_{sb} - \psi_{rb} i_{sa}) - \frac{T_L}{J} \\
\frac{d\psi_{ra}}{dt} &= -\frac{R_r}{L_r}\psi_{ra} - n_p\omega\psi_{rb} + \frac{R_r}{L_r}M i_{sa} \\
\frac{d\psi_{rb}}{dt} &= -\frac{R_r}{L_r}\psi_{rb} + n_p\omega\psi_{ra} + \frac{R_r}{L_r}M i_{sb} \\
\frac{di_{sa}}{dt} &= \frac{M R_r}{\sigma L_s L_r^2}\psi_{ra} + \frac{n_p M}{\sigma L_s L_r}\omega\psi_{rb} \\
&\quad - \left(\frac{M^2 R_r + L_r^2 R_s}{\sigma L_s L_r^2}\right) i_{sa} + \frac{1}{\sigma L_s}u_{sa} \\
\frac{di_{sb}}{dt} &= \frac{M R_r}{\sigma L_s L_r^2}\psi_{rb} - \frac{n_p M}{\sigma L_s L_r}\omega\psi_{ra} \\
&\quad - \left(\frac{M^2 R_r + L_r^2 R_s}{\sigma L_s L_r^2}\right) i_{sb} + \frac{1}{\sigma L_s}u_{sb}
\end{aligned} \tag{2.16}$$

These equations can be rewritten in the form:

$$\begin{aligned}
\frac{d\theta}{dt} &= \omega \\
\frac{d\omega}{dt} &= \mu(\psi_{ra} i_{sb} - \psi_{rb} i_{sa}) - \frac{T_L}{J} \\
\frac{d\psi_{ra}}{dt} &= -\eta\psi_{ra} - n_p\omega\psi_{rb} + \eta M i_{sa} \\
\frac{d\psi_{rb}}{dt} &= -\eta\psi_{rb} + n_p\omega\psi_{ra} + \eta M i_{sb} \\
\frac{di_{sa}}{dt} &= \eta\beta\psi_{ra} + \beta n_p\omega\psi_{rb} - \gamma i_{sa} + \frac{1}{\sigma L_s}u_{sa} \\
\frac{di_{sb}}{dt} &= -\beta n_p\omega\psi_{ra} + \eta\beta\psi_{rb} - \gamma i_{sb} + \frac{1}{\sigma L_s}u_{sb},
\end{aligned} \tag{2.17}$$

with

$$\sigma = 1 - \frac{M^2}{L_s L_r}$$

Table 2.1: AC-drive parameter

R_s	stator resistance	0.18 Ω
R_r	rotor resistance	0.15 Ω
L_s	stator inductance	0.0699 H
L_r	rotor inductance	0.0699 H
M	mutual inductance	0.0680 H
J	rotor inertia	0.0568 kgm^2
n_p	number of pole pairs	1
T	electric motor torque	15kW

Table 2.2: Constants

σ	0.0536
η	2.1459
β	259.5
μ	8.3
γ	85.89

$$\begin{aligned}\eta &= \frac{R_r}{L_r}\\ \beta &= \frac{M}{\sigma L_s L_r}\\ \mu &= \frac{n_p M}{J L_r}\\ \gamma &= \frac{M^2 R_r}{\sigma L_r^2 L_s} + \frac{R_s}{\sigma L_s}\end{aligned} \tag{2.18}$$

Eq. (2.17) is called the $a-b$ motor model. The most commonly used approach is to keep flux constant using field oriented control. This eliminates the coupling between the two control inputs, u_{sa} and u_{sb}. The parameters of 2.18 have a known meaning. σ is the total leakage factor, $1/\eta$ is the rotor time constant, $1/\beta$ a parameter. $1/\mu$= the mechanical constant, $1/\gamma$ the stator time constant.
The AC-drive parameters given in table 2.1 are used in simulation in this report.

2.3 Field oriented control

The idea of field oriented control is from Blaschke [10]. The AC-drive is controlled in such a way that it behaves like a DC-drive. With coil 1 a constant field is generated. To produce torque, current is applied to the rotor. The rotor, coil 3 produces a field which is perpendicular to the field generated by coil 1. This brings the field out of the ideal position. This is unwanted because the torque will decrease. Compensating this by coil 2 ($i_2 = -i_3$) keeps the field into ideal position.
The situation in the AC-drive is different. In the AC-drive the rotor current is caused by the difference in electrical stator frequency and the mechanical rotor speed. Current is applied

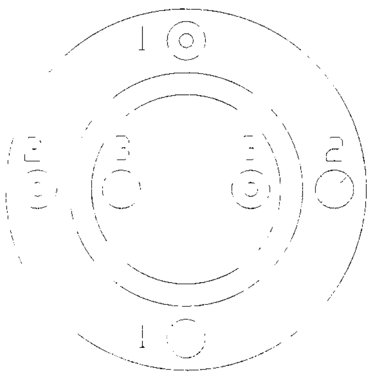


Figure 2.2: Schematic representation of a drive

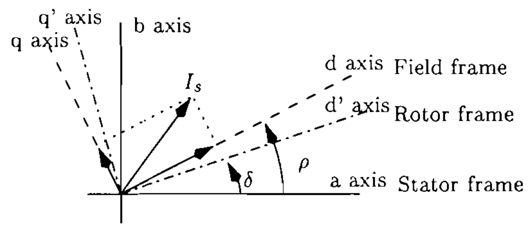


Figure 2.3: Relation between the different coordinate systems

to stator coil 2. As the current rises in coil 2 a reverse current is generated in the rotor coil. First the current in the rotor is exactly opposed to the current in coil 2. This is the same situation as in a DC-motor. However the rotor induction current will change the field and bring it out of ideal position. To keep the ideal situation the current in coil 1 and 2 have to be changed in such way that the field is again perpendicular to the rotor current. This is the idea of field oriented control.
The currents can be transformed to two currents i_d and i_q which are perpendicular and parallel to the field.
The following transformation is called the DQ transformation:

$$\begin{pmatrix} i_d \\ i_q \end{pmatrix} = \begin{bmatrix} \cos\rho & \sin\rho \\ -\sin\rho & \cos\rho \end{bmatrix} \begin{pmatrix} i_a \\ i_b \end{pmatrix} \tag{2.19}$$

$$\begin{pmatrix} \psi_d \\ \psi_q \end{pmatrix} = \begin{bmatrix} \cos\rho & \sin\rho \\ -\sin\rho & \cos\rho \end{bmatrix} \begin{pmatrix} \psi_a \\ \psi_b \end{pmatrix} \tag{2.20}$$

$$\begin{aligned} i_d &= \frac{\psi_a i_a + \psi_b i_b}{|\psi|} \\ i_q &= \frac{\psi_a i_b - \psi_b i_a}{|\psi|} \\ \psi_d &= \sqrt{\psi_a^2 + \psi_b^2} = |\psi| \\ \psi_q &= 0 \end{aligned} \tag{2.21}$$

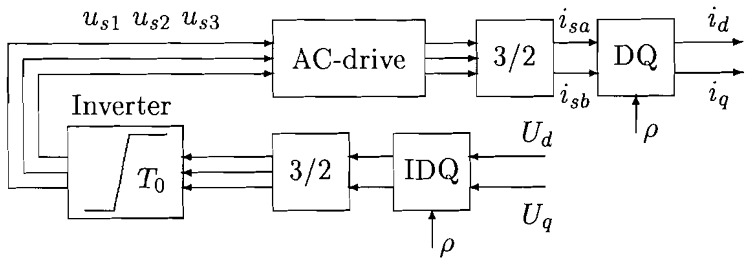


Figure 2.4: Block diagram of induction motor in field coordinates

The information about ρ can be obtained by an observer.

$$\rho = \arctan \frac{\psi_b}{\psi_a} \tag{2.22}$$

Applying this transformation to Eq. (2.17) and translating the field oriented control to the following state feedback

$$\begin{pmatrix} u_a \\ u_b \end{pmatrix} = |\psi| \begin{bmatrix} \psi_a & \psi_b \\ -\psi_b & \psi_a \end{bmatrix}^{-1} \begin{pmatrix} u_d \\ u_q \end{pmatrix} \tag{2.23}$$

gives:

$$\begin{aligned}
\frac{d\omega}{dt} &= \mu\psi_d i_q - \frac{T_L}{J} \\
\frac{d\psi_d}{dt} &= -\eta\psi_d + \eta M i_d \\
\frac{di_d}{dt} &= -\gamma i_d + \eta\beta\psi_d + n_p\omega i_q + \eta M \frac{i_q^2}{\psi_d} + \frac{1}{\sigma L_s} u_d \\
\frac{di_q}{dt} &= -\gamma i_q - \beta n_p \omega \psi_d - n_p \omega i_d - \eta M \frac{i_q i_d}{\psi_d} + \frac{1}{\sigma L_s} u_q \\
\frac{d\rho}{dt} &= n_p\omega + \eta M \frac{i_q}{\psi_d}
\end{aligned} \tag{2.24}$$

This model is called the $d - q$ model or $d - q$ coordinate system. Eq. (2.24) is implemented in a Simulink$^{\text{TM}}$ S-function. (For the S-function see Appendix D) The AC-drive structure to be controlled is shown in figure 2.4.

2.4 Limitations in the model

Saturation is not modeled, but it is possible. In this case a good approximation can be obtained by measuring the saturation curve and linearize it in different pieces. This is described in [11]. In the model a lumped inertia is used. Especially in elevator and crane systems, where the rope is elastic this is not true, and much more complicated. In this case extra equations are needed to describe the elastic rope between the elevator mass and the motor mass. Another non-ideal effect appears when the induction motor is supplied with non-sinusoidal waveforms [8] the rotor resistance will increase and the inductance will decrease due to the skin effect.

2.5 Mechanical basics

In a motor systems it is important to know some mechanical background. The elevator system is used as an example in this section. The real elevator system is more complex, than the simplified simulation model. The ropes and the elevator can be seen as a mass-spring system. The torque does not only dependent on the load, but also on friction. In our case the system is simplified to a stiff system with 'stiff cables', and friction is neglected. In this case the mass of the elevator can be added into the motor's inertia. The effective inertia can be calculated as follow:
Newton's law for rotational motion is given by:

$$T_e - T_l = \frac{dJ\omega}{dt} = J\frac{d\omega}{dt} + \omega\frac{dJ}{dt} = J_0\frac{d\omega}{dt}, \tag{2.25}$$

with

T_e=	electrical torque
T_l=	load torque
J=	inertia

The inertia can be calculated from the following formula:

$$J = \int r^2 dm, \tag{2.26}$$

with m the mass.

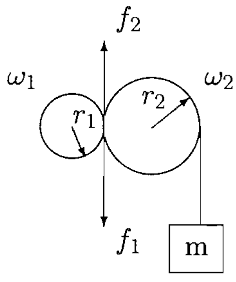


Figure 2.5: Elevator system

Figure 2.5 shows our simplified elevator system. Assuming no friction, backlash or slip Eq. (2.25) gives for the motor wheel:

$$T_e - r_1 f_1 = J_1\frac{d\omega_1}{dt} \tag{2.27}$$

The elevator mass can be simplified to a point mass at radius r_2. Using Eq. (2.26) gives:

$$J_2 = J_{r2} + mr_2^2 \tag{2.28}$$

Applying Eq. (2.25) to wheel 2 gives:

$$r_2 f_2 - mgr_2 = J_2\frac{d\omega_2}{dt} \tag{2.29}$$

Since the two wheels are in balance and there is no slip, $f_1 = -f_2$ and $-r_1\omega_1 = r_2\omega_2$. Eliminating f_1 in Eq. (2.27) with:

$$f_2 = \frac{J_2 d\omega}{r_2 dt} + mg \tag{2.30}$$

gives:

$$T_e - mgr_1 = J_1\frac{d\omega_1}{dt} + \frac{r_1^2}{r_2^2}J_2\frac{d\omega_2}{dt} = J_{1e}\frac{d\omega_1}{dt}, \tag{2.31}$$

with

$$J_{1e} = J_1 + \frac{r_1^2}{r_2^2}\left(J_{r2} + mr_2^2\right). \tag{2.32}$$

The effective inertia in the simplified case depends on the elevator mass, and the inertia of wheel 2. The effective inertia seen by the motor will therefore be larger than the inertia of the rotor itself.

Chapter 3

Nonlinear control

3.1 Introduction

In this chapter the Dynamic Contraction Method is introduced. It is a nonlinear control method, which is based on designing desired trajectories for systems.
Vectors and matrices are represented by the characters in bold. Let us consider a nonlinear time-varying system in the following form:

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(t,\boldsymbol{x}) + \boldsymbol{B}(t,\boldsymbol{x})\boldsymbol{u}(t), \qquad \boldsymbol{x}(0) = \boldsymbol{x}_0 \tag{3.1}$$

$$\boldsymbol{y} = \boldsymbol{g}(t,\boldsymbol{x}) \tag{3.2}$$

where $\boldsymbol{x}(t)$ is an n-dimensional state vector, $\boldsymbol{y}(t)$ is a p-dimensional output vector and $\boldsymbol{u}(t)$ is a p-dimensional control vector.
Each output y_i can be differentiated α_i times until the control input appears. Which results in the following equation:

$$\boldsymbol{y}_* = \boldsymbol{c}^*(t,\boldsymbol{x}(t)) + \boldsymbol{B}^*(t,\boldsymbol{x})\boldsymbol{u}, \qquad \boldsymbol{x}(0) = \boldsymbol{x}_0, \tag{3.3}$$

where $\boldsymbol{y}_* = \left[y_1^{(\alpha_1)}, y_2^{(\alpha_2)}, \ldots, y_p^{(\alpha_p)}\right]$.
Assume that a reference model for transients of $y_i^{(\alpha_i)}(t)$ is given in the following vector differential equation:

$$y_i^{(\alpha_i)}(t) = F_i(\boldsymbol{y_i}(t), \boldsymbol{r_i}(t)), \tag{3.4}$$

where F_i is called the desired dynamics of $y_i(t)$, r_i the reference input, and

$$\boldsymbol{y_i} = \left[y_i, y_i^{(1)}, \ldots, y_i^{(\alpha_i-1)}\right]' \tag{3.5}$$

$$\boldsymbol{r_i} = \left[r_i, r_i^{(1)}, \ldots, r_i^{(p_i-1)}\right]'$$

In case the components $y_i^{(\alpha_i)}$ are mutual independent and the control inputs u_i are decoupled the ith element of $\boldsymbol{y}_*$ could be written as:

$$y_i^{(\alpha_i)} = -a_{\alpha_i-1}y_i^{(\alpha_i-1)} \ldots - a_0 y_i + b_{\alpha_i-1}r_i^{(p_i-1)} + \ldots + b_0 r_i \tag{3.6}$$

The difference between the desired and the actual response of the system is defined as:

$$\boldsymbol{\Delta} = \boldsymbol{F} - \boldsymbol{y}_* \tag{3.7}$$

The control system to be designed must provide the following condition:

$$\lim_{t\to\infty} \boldsymbol{\Delta} = 0 \tag{3.8}$$

The control problem can be solved with the following Nonlinear Inverse Dynamics (NID) control algorithm $u(t) = u^a(t)$:

$$\boldsymbol{u}^a(t) = \boldsymbol{B}^*(t,\boldsymbol{x})^{-1}[\boldsymbol{F} - \boldsymbol{c}^*(t,\boldsymbol{x}(t))] \tag{3.9}$$

However this control law may be used only if there is complete information about disturbances, model parameters and the state of the system. The algorithm has therefore no real practical value.

3.2 The Dynamic Contraction Method (DCM)

Assuming that $\boldsymbol{u}(t)$ changes much faster than $\boldsymbol{x}(t), \boldsymbol{y}(t)$, and $\vec{\boldsymbol{r}}(t)$ our control problem can be translated into a two time scale problem. One for the fast motions (the control) and one for the slow motions (the states). The new fast time scale is defined as:

$$\tau = \mu^{-1}t, \tag{3.10}$$

where μ is a small positive parameter. Let us introduce nonsingular matrices $\boldsymbol{K}_0$, and $\boldsymbol{K}_1$ whose meaning will be discussed later. $\boldsymbol{K}_1$ is usually a diagonal matrix. In the next equation the new control input $\boldsymbol{v}$ is defined as:

$$\boldsymbol{u}(t) = \boldsymbol{K}_0\boldsymbol{K}_1\boldsymbol{v}(t) \tag{3.11}$$

The following equation was discussed in [6]:

$$\boldsymbol{\mu}^q\boldsymbol{D}_q\boldsymbol{u}^{(q)} + \boldsymbol{\mu}^{q-1}\boldsymbol{D}_{q-1}\boldsymbol{u}^{(q-1)} + \ldots + \boldsymbol{\mu}\boldsymbol{D}_1\boldsymbol{u}^{(1)} + \boldsymbol{D}_0\boldsymbol{u} = k\boldsymbol{\Delta}, \qquad \vec{\boldsymbol{u}}(0) = \vec{\boldsymbol{u}}_0, \tag{3.12}$$

with $\boldsymbol{\mu} = diag\{\mu_1, \mu_2, \ldots, \mu_p\}$. μ_i are small positive parameters, and $\boldsymbol{D}_{q-1}, \ldots, \boldsymbol{D}_0$ are diagonal matrices and $\vec{\boldsymbol{u}}(t) = \left[\boldsymbol{u}', \boldsymbol{u}^{(1)\prime}, \ldots, \boldsymbol{u}^{(q-1)\prime}\right]'$. Using Eq. (3.11) for each ith component can be written:

$$\mu_i^{q_i} d_{i,q_i} v^{(q_i)} + \mu_i^{q_i-1} d_{i,q_i-1} v^{(q_i-1)} + \ldots + \mu d_{i,1} v_i^{(1)} + d_{i,0} v_i = k_i \Delta_i, \qquad \boldsymbol{v_i}(0) = \boldsymbol{v}_{i,0}, \tag{3.13}$$

where $\boldsymbol{v_i} = \left[v_i, v_i^{(1)}, \ldots, v_i^{(q_i-1)}\right]'$. Assuming that $q_i \geq \alpha_i$ the system is proper and realizable without differentiation.

3.2.1 Fast motions

From equations (3.1), (3.3), (3.7), (3.11), (3.12) the closed loop system can be rewritten as:

$$\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{f}(t,\boldsymbol{x}) + \boldsymbol{B}(t,\boldsymbol{x})\boldsymbol{K}_0\boldsymbol{K}_1\boldsymbol{v}, \qquad \boldsymbol{x}(0) = \boldsymbol{x}_0 \\ \boldsymbol{\mu}^q\boldsymbol{D}_q\boldsymbol{v}^{(q)} &+ \boldsymbol{\mu}^{q-1}\boldsymbol{v}^{(q-1)}\boldsymbol{D}_{q-1} + \ldots + \boldsymbol{\mu}\boldsymbol{D}_1\boldsymbol{v}^{(1)} + \boldsymbol{\Gamma}\boldsymbol{v} \\ &= k\left[\boldsymbol{F} - \boldsymbol{c}^*(t,\boldsymbol{x})\right], \qquad \vec{\boldsymbol{v}}(0) = \vec{\boldsymbol{v}}_0, \end{aligned} \tag{3.14}$$

where $\boldsymbol{\Gamma}$ is defined as:

$$\boldsymbol{\Gamma} = \boldsymbol{D}_0 + k\boldsymbol{B}^*(t,\boldsymbol{x})\boldsymbol{K}_0\boldsymbol{K}_1, \qquad \boldsymbol{K}_1 = diag\{k_1, \ldots, k_p\} \tag{3.15}$$

Taking lim $\mu \to 0$ and returning to the primary time scale using $t = \mu\tau$ we obtain the fast motion time system which is defined by:

$$\begin{aligned} 0 &= \boldsymbol{f}(t,\boldsymbol{x}) + \boldsymbol{B}(t,\boldsymbol{x})\boldsymbol{K}_0\boldsymbol{K}_1\boldsymbol{v}, \qquad \boldsymbol{x}(0) = \boldsymbol{x}_0 \\ \mu^q D_q \boldsymbol{v}^{(q)} &+ \mu^{q-1}\boldsymbol{v}^{(q-1)}\boldsymbol{D}_{q-1} + \ldots + \mu \boldsymbol{D}_1 \boldsymbol{v}^{(1)} + \boldsymbol{\Gamma v} \\ &= k\left[\boldsymbol{F} - \boldsymbol{c}^*(t,\boldsymbol{x})\right], \qquad \vec{\boldsymbol{v}}(0) = \vec{\boldsymbol{v}}_0, \end{aligned} \tag{3.16}$$

where $\boldsymbol{F} - \boldsymbol{c}^*$, and $\dot{\boldsymbol{x}}$ is assumed to be constant if there is good time separation between the fast and the slow motions.

3.2.2 Slow motions

The slow motion system is found by taking the lim $\mu \to 0$ using equations (3.3), (3.11), (3.14).

$$\boldsymbol{y}_* = \boldsymbol{F} + k^{-1}\boldsymbol{D}_0\left[k^{-1}\boldsymbol{D}_0 + \boldsymbol{B}^*(t,\boldsymbol{x})\boldsymbol{K}_0\boldsymbol{K}_1\right]^{-1}\left[\boldsymbol{c}^*(t,\boldsymbol{x}) - \boldsymbol{F}\right] \tag{3.17}$$

The steady state of the fast motion time system is defined by:

$$\boldsymbol{v}^s = k\boldsymbol{\Gamma}^{-1}\left[\boldsymbol{F} - \boldsymbol{c}^*\right] \tag{3.18}$$

Using $\boldsymbol{u}^a = \boldsymbol{K}_0\boldsymbol{K}_1\boldsymbol{v}^a(t)$ can be rewritten into the following form:

$$\begin{aligned} \boldsymbol{v}^s &= \boldsymbol{v}^a + \boldsymbol{\Gamma}^{-1}\boldsymbol{D}_0\left[\boldsymbol{B}^*\boldsymbol{K}_0\boldsymbol{K}_1\right]^{-1}\left[\boldsymbol{c}^*(t,\boldsymbol{x}) - \boldsymbol{F}\right] \\ &= \boldsymbol{v}^a + \Lambda\left[\boldsymbol{c}^*(t,\boldsymbol{x}) - \boldsymbol{F}\right], \end{aligned} \tag{3.19}$$

where $\boldsymbol{v}^a$ is the NID solution.
If $\boldsymbol{D}_0 = 0$ or $\boldsymbol{D}_0 \neq 0$ but $k \to \infty$ then $\boldsymbol{v}^s = \boldsymbol{v}^a$ and $\boldsymbol{y}_* \to \boldsymbol{F}$. (Eq. (3.9)). The DCM method will converge to the NID solution.

3.2.3 Conclusions

The DCM method can be used in systems with incomplete information and varying parameters. The matrix $\boldsymbol{K}_0$ is often chosen as $\boldsymbol{B}^{-1}$ to simplify the equations. Matrix $\boldsymbol{K}_1$ can be used to tune the control inputs.

3.3 Applying the DCM to the induction motor

The goal of this section is to apply the DCM method to our MIMO motor system, and design a current and flux controller. Rewriting the motor equations (2.24) in states space gives:

$$\dot{\boldsymbol{x}} = \begin{bmatrix} \mu x_2 x_4 - \frac{T_L}{J} \\ -\eta x_2 + \eta M x_3 \\ -\gamma x_3 + \eta\beta x_2 + n_p x_1 x_4 + \eta M \frac{x_4^2}{x_2} \\ -\gamma x_4 - \beta n_p x_1 x_2 - n_p x_1 x_3 - \eta M \frac{x_3 x_4}{x_2} \\ n_p x_1 + \eta M \frac{x_4}{x_2} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \frac{1}{\sigma L_s} \\ \frac{1}{\sigma L_s} \\ 0 \end{bmatrix} \boldsymbol{u}, \tag{3.20}$$

with $x_1 = \omega$, $x_2 = \psi_d$, $x_3 = i_d$, $x_4 = i_q$, $x_5 = \rho$, $u_1 = u_q$, $u_2 = u_d$, $y_1 = i_q$, and $y_2 = \psi_d$. Assuming that i_q is measured, differentiating once gives:

$$\dot{y}_1 = \frac{di_q}{dt} = \frac{dx_4}{dt} = -\gamma x_4 - \beta n_p x_1 x_2 - n_p x_1 x_3 - \eta M \frac{x_3 x_4}{x_2} + \frac{1}{\sigma L_s} u_2 \tag{3.21}$$

There is assumed that ψ_d is a measurable output. Later in chapter 6 an observer for ψ_d is constructed. The flux dynamics are given by Eq. (2.24):

$$\begin{aligned}\frac{d\psi_d}{dt} &= -\eta\psi_d + \eta M i_d \\ \frac{di_d}{dt} &= -\gamma i_d + \eta\beta\psi_d + n_p\omega i_q + \eta M\frac{i_q^2}{\psi_d} + \frac{1}{\sigma L_s}u_d\end{aligned} \tag{3.22}$$

Differentiating the flux output two times gives the differential equation with the control input.

$$\begin{aligned}y_2 &= \psi_d \\ \dot{y}_2 &= \dot{x}_2 = \frac{dx_2}{dt} = -\eta x_2 + \eta M x_3 \\ \ddot{y}_2 &= -\eta\dot{x}_2 + \eta M\dot{x}_3 \\ &= \eta(x_2 + x_3) \\ &\quad +\eta M(-\gamma x_3 + \eta\beta x_2 x_3 + n_p x_1 x_4 + \eta M\frac{x_4^2}{x_2} + \frac{1}{\sigma L_s}u_1)\end{aligned} \tag{3.23}$$

Equations (3.22) and (3.23) are rewritten into the following form:

$$\boldsymbol{y}_* = \boldsymbol{c}^*(t,\boldsymbol{x}) + \boldsymbol{B}^*(t,\boldsymbol{x})\boldsymbol{u}, \tag{3.24}$$

with

$$\begin{aligned}\boldsymbol{c}^*(t,\boldsymbol{x}) &= \begin{bmatrix}-\gamma x_4 - \beta n_p x_1 x_2 - n_p x_1 x_3 - \eta M x_3 x_4/x_2 \\ \eta M(-\gamma x_3 + \eta\beta x_2 + n_p x_1 x_4 + \eta M x_4^2/x_2)\end{bmatrix} \\ \boldsymbol{B}^*(t,\boldsymbol{x}) &= \begin{bmatrix}0 & 1/(\sigma L_s) \\ \eta M/(\sigma L_s) & 0\end{bmatrix}.\end{aligned} \tag{3.25}$$

$$\begin{aligned}\mu^2\boldsymbol{D}_2\boldsymbol{v}^{(2)} + \mu^1\boldsymbol{D}_1\boldsymbol{v}^{(1)} + \boldsymbol{D}_0\boldsymbol{v} &= k\left[\boldsymbol{F} - \boldsymbol{y}_*\right] \qquad (3.26) \\ &= k\left[\boldsymbol{F} - \boldsymbol{c}^* - \boldsymbol{B}^*(t,\boldsymbol{x})\boldsymbol{K}_0\boldsymbol{K}_1\boldsymbol{v}\right] \\ \mu^2 D_2\boldsymbol{v}^{(2)} + \mu^1\boldsymbol{D}_1\boldsymbol{v}^{(1)} + \boldsymbol{D}_0\boldsymbol{v} + \boldsymbol{B}^*(t,\boldsymbol{x})\boldsymbol{K}_0\boldsymbol{K}_1\boldsymbol{v} &= k\left[\boldsymbol{F} - \boldsymbol{c}^*\right] \\ \mu^2\boldsymbol{D}_2\boldsymbol{v}^{(2)} + \mu^1\boldsymbol{D}_1\boldsymbol{v}^{(1)} + \boldsymbol{\Gamma}\boldsymbol{v} &= k\left[\boldsymbol{F} - \boldsymbol{c}^*\right],\end{aligned}$$

$$\boldsymbol{\Gamma} = \boldsymbol{D}_0 + k\boldsymbol{B}^*(t,\boldsymbol{x})\boldsymbol{K}_0\boldsymbol{K}_1. \tag{3.27}$$

Using $\boldsymbol{K}_1 = \boldsymbol{I}$, and $\boldsymbol{K}_0 = (\boldsymbol{B}^*)^{-1}$, Eq. (3.27) can be written as:

$$\Gamma = \boldsymbol{D}_0 + k \tag{3.28}$$

3.3.1 Designing the desired trajectories

The current trajectory

Defining the desired trajectory as:

$$r - \alpha_0 y - \tau\dot{y} = 0 \tag{3.29}$$

With Eq. (3.12) the current controller equation becomes:

$$\mu d_1 \dot{v} + d_0 v = k\left[r - \alpha_0 y - \tau \dot{y}\right] \tag{3.30}$$

Rewriting this in the form with $d_1 = 1$ and $\alpha_0 = 1$ gives:

$$v(s) = k\left[\frac{1}{\mu s + d_0} r(s) - \frac{(1+\tau s)}{\mu s + d_0} y(s)\right] \tag{3.31}$$

The flux trajectory
Using Eq. (3.6) for $q = 2$ the form of the desired trajectory can be written as:

$$y^{(2)}(t) = -\alpha_0 - \alpha_1 \dot{y} + \beta_0 + \beta_1 \dot{r} \tag{3.32}$$

These parameters make not much sense in this form. Therefore mostly second order systems are written in the following form:

$$H(s) = \frac{k}{\tau^2 s^2 + 2\alpha\tau s + 1} \tag{3.33}$$

In differential form this is:

$$\tau^2 \frac{dy^2}{dt^2} + 2\alpha\tau\frac{dy}{dt} + y = kv \tag{3.34}$$

The parameters of (3.33) have a known meaning, τ is the time constant and α is the damping of the system.
The flux controller equation can be written as:

$$\mu^2 v_\psi^{(2)} + 2d_1\mu v_\psi^{(1)} + d_0 v_\psi = k\left[\psi_d^{(2)} + \tau_\psi^{-2}(2\alpha_\psi\tau_\psi\psi_d^{(1)} + \psi_d - \psi_{dref})\right] \tag{3.35}$$

3.3.2 Controller design

The parameters that determine the fast dynamics of the controller can now be chosen.
The current controller
The characteristic equation for the current equation can be written as (see Eq. 3.27):

$$\mu d_1 s + d_0 + k = 0, \tag{3.36}$$

with $d_0 = 0$ and $\mu = 1$ this equation can be written as:

$$\frac{d_1}{k}s + 1 = 0 \tag{3.37}$$

$$\mu_i \lambda + 1 = 0$$

$$\mu_i = \frac{d_1}{k}, \tag{3.38}$$

using

$$k = \frac{k'}{\tau_c}, \tag{3.39}$$

and $d_1 = 1$ gives:

$$\mu_i = \frac{\tau_c}{k'}. \tag{3.40}$$

To have a good time separation the following condition must be fulfilled:

$$\mu_i \leq 0.1\tau_i \tag{3.41}$$

In table 3.1 some current controller parameters are given which will be checked later.

Table 3.1: DCM-current controller parameters

	case I	case II
τ_i	1e-3	1e-3
μ	1	1
d_0	0	0
d_1	1	1
k_i'	50	25

The flux controller

The characteristic equation for the flux controller is:

$$\frac{\mu_\psi^2}{d_0 + k_\psi}\lambda^2 + \frac{2d_{\psi,1}\mu_\psi}{d_0 + k_\psi}\lambda + 1 = 0 \tag{3.42}$$

This equation can be rewritten in the following form:

$$\mu_{0\psi}^2\lambda^2 + 2d_{0\psi}\mu_{0\psi}\lambda + 1 = 0, \tag{3.43}$$

with

$$\mu_{0\psi} = \frac{\mu_\psi}{d_0 + k_\psi}\sqrt{d_0 + k_\psi}, \tag{3.44}$$

and

$$d_{0\psi} = \frac{d_1}{d_0 + k_\psi}\sqrt{d_0 + k_\psi}. \tag{3.45}$$

To have a good time separation the following condition must be fulfilled:

$$\mu_{0\psi} \leq 0.1\tau_\psi \tag{3.46}$$

Some values which fulfill this equation will be simulated later and are given in table 3.2.

Table 3.2: DCM-flux controller parameters

	case I	case II
τ_ψ	1e-2	1e-2
μ	0.001	0.01
d_0	0	0
d_1	1.4	1.4
k_ψ	1.6	1.6

Chapter 4

Design of inner loops

4.1 Introduction

In this chapter the inner loops are analyzed in more detail as was done in chapter 3. It consists of two parts. The first part analyzes the current controller and the last part the flux controller. The influence of different disturbances are analyzed for both controllers. There is explained how to choose the parameters of the DCM controllers and why the DCM controller is better than normal PI.

4.1.1 Analysis of the current controller

To give some guidelines on how to choose parameters the current equation is translated to a model where the non-linear terms are collected in a disturbance term.
The current equation with i_q was:

$$\frac{di_q}{dt} = -\gamma i_q - \beta n_p \omega \psi_d - n_p \omega i_d - \eta M \frac{i_q i_d}{\psi_d} + \frac{1}{\sigma L_s} u_q \tag{4.1}$$

The terms $-\beta n_p \omega \psi_d - n_p \omega i_d$ are called the speedterm. Assuming that the flux is between $\psi_d = 0.1 \ldots 1$ Wb, constant and using

$$\dot{\psi}_d = -\eta \psi_d + \eta M i_d \tag{4.2}$$

gives:

$$\psi_d = M i_d \tag{4.3}$$

Eq. (4.1) can be rewritten as:

$$\frac{di_q}{dt} = (-\gamma - \eta) i_q - (\beta M - 1) n_p \omega i_d + \frac{1}{\sigma L_s} u_q \tag{4.4}$$

Taking $\omega = 0$, the speed term is zero. The system reduces to a linear equation. The system is shown in figure 4.1. With $\tau_1 = 1/(\gamma + \eta)$, $d(s) = (\beta M - 1) n_p \omega i_d$ and $B_1 = \frac{1}{(\gamma+\eta)(\sigma L_s)}$. The block B_1^{-1} is used to normalize the gain, and is the new input for the controller. The parameters relevant for the current loop are given in table 4.1. If the drive is turning the speed term is no longer zero, and forms an important disturbance in the system. In this case

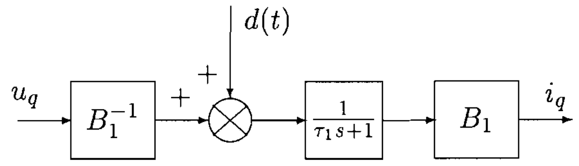


Figure 4.1: Block diagram of current model with the nonlinear terms collected in the disturbance term

Table 4.1: Current dynamics parameters

Parameters	Values
γ	85.8927
η	2.14
n_p	1
β	259.5321
τ_l	11.4e-3
B_l	3.0303

changing μ or k has the same effect, therefore μ is chosen 1. Choosing $d_0 = 0$ the controller contains an integrator. With $\mu = 1$, $d_0 = 0$, the DCM controller equation (3.31) becomes:

$$u(s) = k \cdot \left[\frac{1}{s} \cdot r(s) - \frac{\tau_c s + 1}{s} \cdot y(s)\right] \tag{4.5}$$

For the sake of analysis the DCM controller is defined by Eq. (3.30) and is of order one. The DCM controller can be translated to a PI controller with a filter at the reference input. The block diagram in figure 4.2 is equivalent to Eq. (3.31) only for zero initial conditions and its order is two.

Using the model of figure 4.1 the influence of measurement noise and speed disturbances are examined. The current plant model is shown in figure 4.3. The controller is translated to the general PI form:

$$u(s) = k\frac{\tau_c s + 1}{s} = k'\frac{\tau_c s + 1}{\tau_c s} = k'(1 + \frac{1}{\tau_c s}), \tag{4.6}$$

with $k' = k\tau_c$.

The two different time scales in the system are seen in this figure. The filter is the slow time scale, and the control loop the fast time scale.

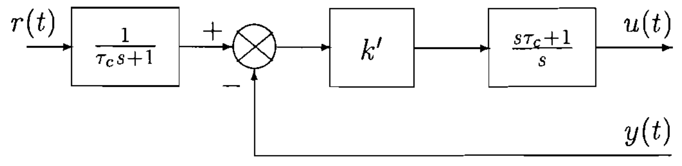


Figure 4.2: The DCM-controller translated to PI with pre-filter

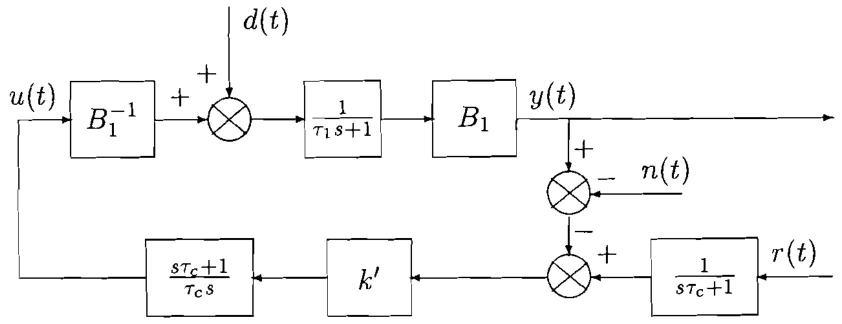


Figure 4.3: The current plant

Response time:
Assuming zero noise and zero disturbances Eq. (3.31) gives:

$$y(s) = k'\frac{1}{\tau_c(s\tau_1+1)}\left[\frac{1}{s}r(s) - \frac{s\tau_c+1}{s}y(s)\right] \tag{4.7}$$

$$y(s) = k'\frac{1}{\tau_c s(s\tau_1+1)+(s\tau_c+1)k'}\cdot r(s) \tag{4.8}$$

$$y(s) = \frac{1}{\mu_n^2 s^2 + 2d_n\mu_n s + 1}\cdot r(s), \tag{4.9}$$

with $\mu_n = \sqrt{\tau_1\tau_c/k'}$, and $d_n = (k'+1)/(2\sqrt{k'})$.
With large enough gain this equation reduces to:

$$y(s) = \frac{1}{(s\tau_c+1)}\cdot r(s) \tag{4.10}$$

and is independent of the plant time constant τ_1. When k is large enough the system can be made faster by decreasing τ_c.
With τ_c the filter time constant and $\sqrt{\tau_1\tau_c/k'}$ the fast motion time constant the time scale separation can be defined by $\tau_c/\sqrt{(\tau_1\tau_c/k')}$. The controller is well tuned if there is sufficient time scale separation.

Speed term disturbance:
Most of the time the motor is accelerating or decelerating. Therefore the speed terms form a ramp wise disturbance ($d(s) = (\beta M - 1)n_p i_d/s^2$). The steady state error due to the speed ramp is defined by:

$$y(s) = \frac{B_1}{s\tau_1+1}\left[d(s) - k'\frac{s\tau_c+1}{\tau_c B_1 s}y(s)\right] \tag{4.11}$$

$$\lim_{s\to 0} s\cdot Y(s) = s\cdot\frac{\tau_c B_1 s}{\tau_c s(s\tau_1+1)+(s\tau_c+1)k'}\cdot\frac{(\beta M-1)n_p i_d}{s^2} \tag{4.12}$$

$$= \frac{\tau_c B_1(\beta M-1)n_p i_d}{k'} \tag{4.13}$$

The gain k' must be large enough to minimize the speed term disturbances which cause a steady state error.

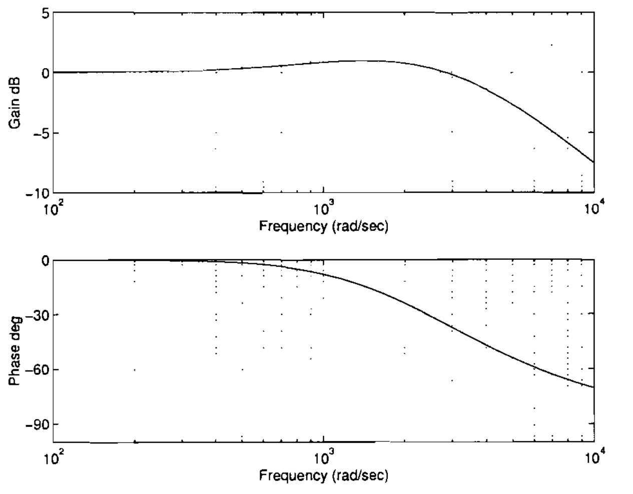


Figure 4.4: The complementary sensitivity function for the current plant, k'=50 and τ_c =1e-3

Sensor noise:

The complementary sensitivity transfer function is defined by:

$$C(s) = \frac{y(s)}{n(s)} = \frac{k'(s\tau_c + 1)}{\tau_c s(s\tau_l + 1) + (\tau_c s + 1)k'} \tag{4.14}$$

If τ_c equals to τ_l the closed loop system reduces to a first order system and the complementary sensivity equation reduces to:

$$C(s) = \frac{1}{\frac{\tau_c}{k'}s + 1} \tag{4.15}$$

The complementary sensitivity function in figure 4.4 shows the plot for the second order system.

Eq. (4.11) and Eq. (4.15) confront us with a classical problem. The gain must be large enough to track the reference trajectory, and have a small steady state error. Choosing the gain larger, the system becomes more sensitive to high frequency noise. As always it is looking for a compromise between robustness and sensitivity. Noise is not simulated, because we do not know in what frequency area noise will be important. Another practical reason for not simulating noise are the long simulation times in Simulink^{TM}.

The time scale separation is easily seen from the root locus plot, with o=the zeros, x=the open loop poles, +=the closed loop poles. Figure 4.5 shows the root locus if the controller zero cancels the current pole. The closed loop pole for our gain $p_{cl} = -42000$ is not displayed in this picture.

If the controller zero does not cancel the current pole (τ_c smaller for faster response) the root locus will look like figure 4.6. Making the system faster makes the damping worse if the gain is not increased (see also Eq. 4.9). This is due to the larger circle radius for smaller τ_c.

The following design rules can be defined:

- To have a steady state error equal or smaller than p percent in i_q:

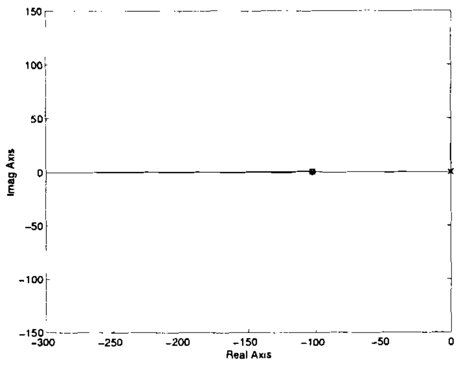


Figure 4.5: Root locus if the controller cancels the current pole

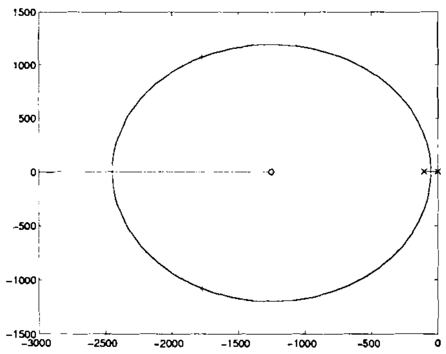


Figure 4.6: Root locus if the controller does not cancel the current pole

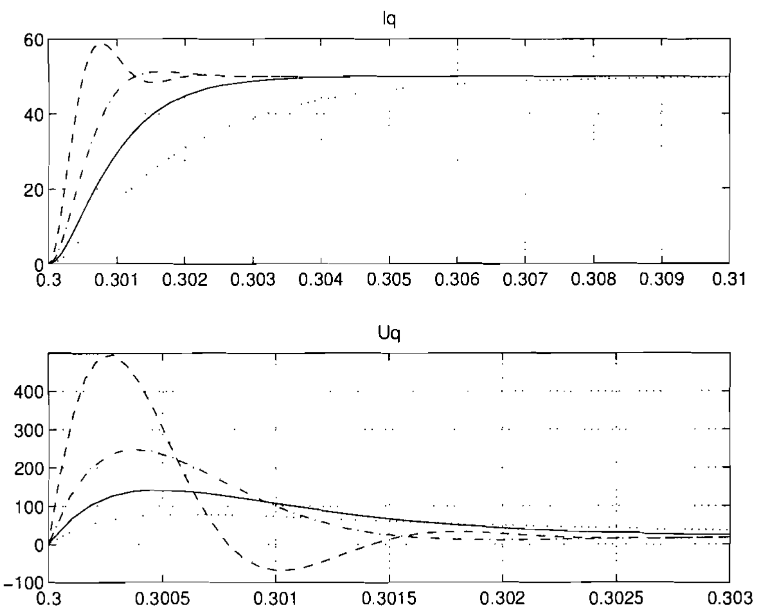


Figure 4.7: Simulation plot for k'=50 and different τ_c =.2e-3 (dashed), .5e-3 (dash-dot), 1e-3 (solid), 2e-3 (dots).

$$k' > 100\tau_c \frac{B_1}{(\beta M - 1)n_p i_d i_q p} \tag{4.16}$$

- If the controller zero cancels the current pole:
  $\tau = \tau_c/k' \leq 0.1\tau_1$
  General:
  $\tau_n = \sqrt(\tau_1 \tau_c/k') \leq 0.1\tau_1$

- If the controller zero does not cancel the current pole:
  $d_n = (k'+1)/(2\sqrt(k')) > 0.5$, no large bumps in $C(s)$ characteristic, and good damping of the fast motion system.

- Noise sensitivity:
  $\tau_n > \tau_{noise}$

Figure 4.7 shows the simulation result for different τ_c and constant k. From figure 4.8 the influence is shown for choosing different values of k. If the controller is too slow to realize the fast current dynamics, this results in overshoot in the current. The controller parameters are shown in table 4.2. Changing τ_c has the most influence on the responses of the current controller. The controller responses are almost the same for the different k' values. For larger k' the system follows better the trajectory, is more robust against disturbances, and has a smaller steady state error.

4.1.2 Conclusions

As shown viewing systems in different time scales is a good start for designing a control. The principle of separation in different time scales works.

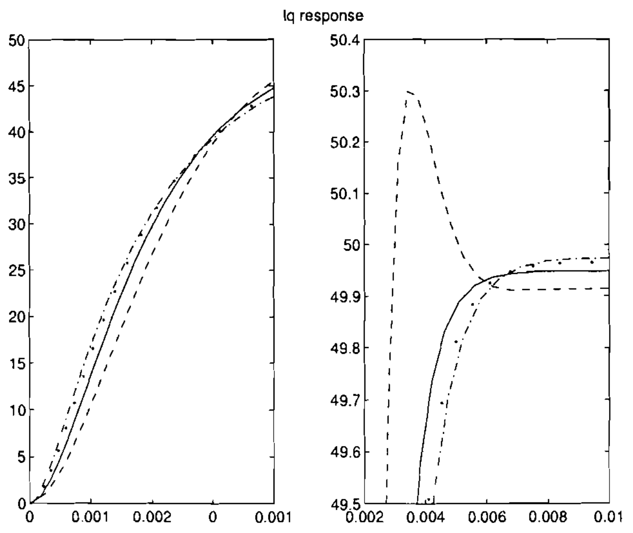


Figure 4.8: Simulation plot for different k' and constant τ, τ_c=1e-3, k'=30 (dashed), k'=50 (solid), k'=75 (dots), k'=100 (dash-dot).

Table 4.2: DCM current controller parameters

Parameters	Values
d_0	0
μ	1
α	1
τ_c	1e-3
k'	50

Table 4.3: Simulation parameters, in comparison between PI-DCM

τ_{dcm}	10e-4
DCM gain	50
τ_{PI}	τ_1=11.4e-3
PI gain	10

The first order DCM controller can be translated to a PI controller with prefilter. The current system can be analyzed by a linear system with a disturbance term which contains the nonlinear terms. Some basic calculations can then be made on how to choose parameters. We distinguished two different cases: a first order and a second order case. The two root locus plots are different. In the first case the system had no complex poles, and in the second case it has. Therefore the first case can not be used to approximate the system if $\tau_c \neq \tau_1$. If the guidelines for designing the controller are followed the system will follow the disered trajectory.

4.2 Comparison between PI and DCM

There are some important differences between normal PI and a DCM controller. In the DCM case the controller design is based on predefined trajectories which is not the case with normal PI. The difference is that the DCM controller controls the error between the output and the desired trajectory. The PI controller is based only on controlling the error between input and output to zero.
The PI controller equation is:

$$u(s) = k(1 + \frac{1}{\tau_c s}), \tag{4.17}$$

τ_c is chosen to cancel the current pole.
Increasing the gain in the PI case will speed up the response time. The close loop equation is:

$$H(s)_{cl} = \frac{1}{\frac{\tau_c}{k}s + 1} \tag{4.18}$$

In DCM case this equation is an approximation, in PI it is exact. To make a good comparison between PI and DCM the time constants should be the same. To have the same time constant as in the DCM case ($\tau_{dcm} = 1e-3$), the PI gain must be $k_{PI} = 10$. Normally in PI controller design no prefilter is used. If step responses are used in PI case to tune the controller the gains can not be as high as in the DCM case otherwise the PI controller oversteer the stator voltages. This means a slower system. The prefilter prevents the DCM controller against step inputs and therefore the DCM control structure makes no spikes. Another difference is the indepancy of gain and time constant in the DCM case. The gain in DCM controller can be higher than in the PI case for the same time constant. This makes the DCM controller more robust agains disturbances, and causes a smaller steady state error.
A simulation in Simulink^{TM}in figure 4.9 shows this effect. The steady state error is larger than in DCM case.

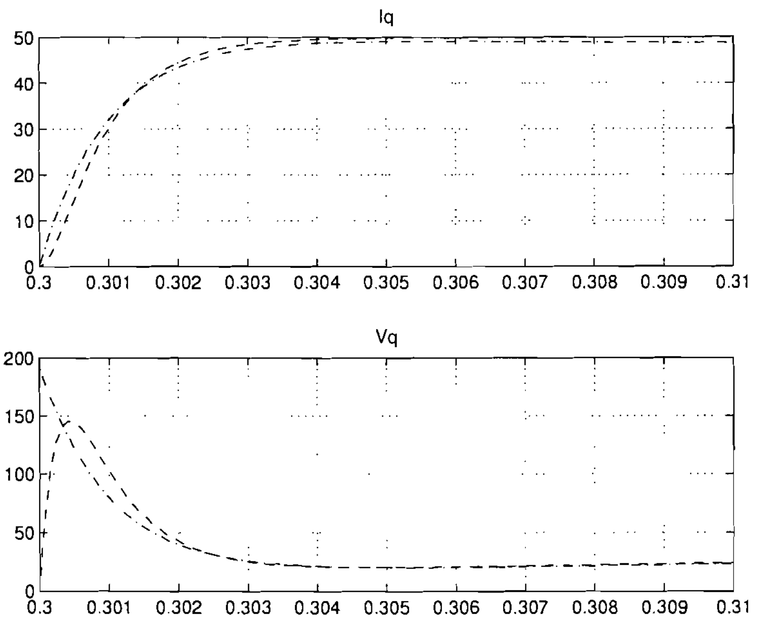


Figure 4.9: Comparison in responces between PI-DCM, dashed=DCM and dashed-dot=PI

4.2.1 Conclusion: DCM controller is better than PI controller

As in the DCM controller higher gain can be used with the same time constant it will have better performance than a conventional PI controller. As shown in figure 4.9 the PI controller introduces spikes in the control signals if steps are applied to the controller input. In DCM case the control signal is still smooth.
The DCM controller will therefore be superior to a conventional PI controller.

4.2.2 Analysis of the flux controller

From Eq. (2.24), the equations for flux and current are:

$$\begin{aligned}\frac{d\psi_d}{dt} &= -\eta\psi_d + \eta M i_d \\ \frac{di_d}{dt} &= -\gamma i_d + \eta\beta\psi_d + n_p\omega i_q + \eta M\frac{i_q^2}{\psi_d} + \frac{1}{\sigma L_s}u_d\end{aligned} \tag{4.19}$$

These equations are translated into the block diagram of figure 4.10 with the disturbance term $d(t) = n_p\omega i_q + \eta M i_q^2/\psi_d$. In table 4.4 the parameters of the flux system are given. The transfer function of the open loop flux system is:

$$H(s) = \frac{M\eta}{\sigma L_s((s+\gamma)(s+\eta) - \eta^2\beta M)} = B_2\frac{1}{(\tau_1 s+1)(\tau_2 s+1)} \tag{4.20}$$

The time constants of this system are given by τ_1 and τ_2. The time constant of the i_d is much smaller than the time constant of ψ_d. Therefore we can use $d\psi_d/dt = 0$ which simplifies flux

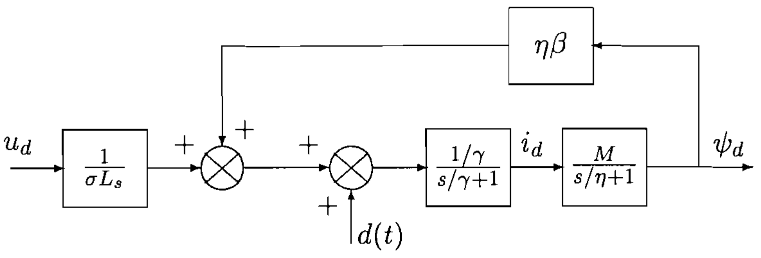


Figure 4.10: Block diagram of the flux dynamics

Table 4.4: Parameters of the flux system

Parameters	Values
η	2.1459
γ	85.8927
β	259.5321
τ_1	11.7e-3
τ_2	476e-3
B_2	38.9298
M	0.0680
$1/\gamma$	11.6e-3
$\eta\beta$	556

system to:

$$\frac{di_d}{dt} = (-\gamma + M\eta\beta)i_d + n_p\omega i_q + \eta\frac{i_q^2}{i_d} + \frac{1}{\sigma L_s}u_d \tag{4.21}$$

This equation shows that the response time of i_d is affected by the flux disturbance term $(\eta M\psi_d)$.

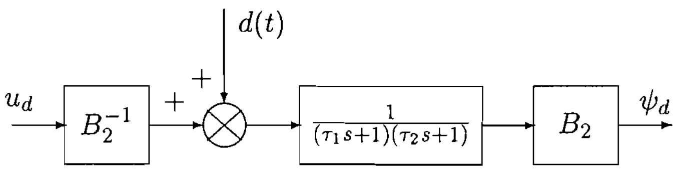


Figure 4.11: Linear approximation of flux system

With $\tau_1 = 1/(\gamma + M\eta\beta)$, $d(t) = n_p\omega i_q + \eta i_q^2/i_d$, $\tau_2 = 1/\eta$ and $B_2 = M/((\gamma - \eta\beta M)\sigma L_s)$. The block B_2^{-1} is used to normalize the gain, and is the new input for the controller. If $i_q = 0$ the disturbance term is zero. In figure 4.12 the second order DCM controller is shown, which is translated to a controller and prefilter.

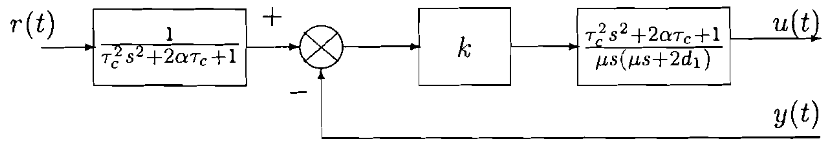

$r(t)$
$\frac{1}{\tau_c^2 s^2+2\alpha\tau_c+1}$
+
−
k
$\frac{\tau_c^2 s^2+2\alpha\tau_c+1}{\mu s(\mu s+2d_1)}$
$u(t)$
$y(t)$

Figure 4.12: DCM flux controller

The flux dynamics and controller are displayed in figure 4.13. The system is analyzed by examining the response time, steady state error and complementary sensivity function.

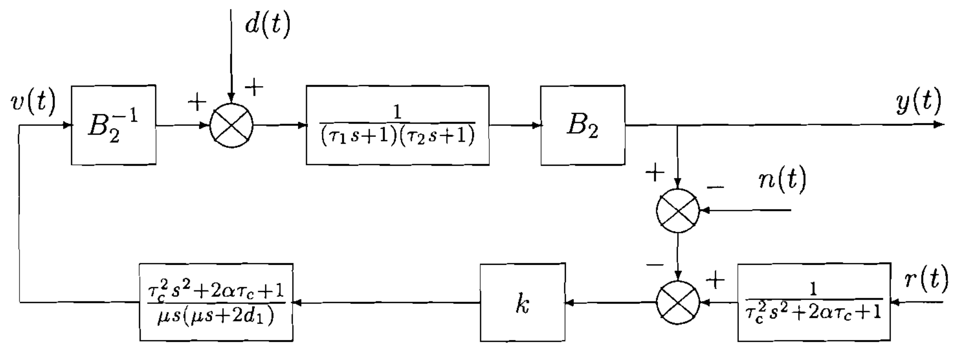

$d(t)$
$v(t)$
B_2^{-1}
+
+
$\frac{1}{(\tau_1 s+1)(\tau_2 s+1)}$
B_2
$y(t)$
+
−
$n(t)$
−
+
$r(t)$
$\frac{\tau_c^2 s^2+2\alpha\tau_c+1}{\mu s(\mu s+2d_1)}$
k
$\frac{1}{\tau_c^2 s^2+2\alpha\tau_c+1}$

Figure 4.13: The flux plant

Response time:
The transfer function from reference input to output is defined by:

$$y = \frac{k}{\tau_c^2\mu s(\mu s + 2d_1)(\tau_1 s + 1)(\tau_2 s + 1) + (\tau_c^2 s^2 + 2\alpha\tau_c s + 1)k} \cdot r(s) \qquad (4.22)$$

If $\lim_{\mu\to 0}$ is used in Eq. (4.22) the slow motion system appears (the desired trajectory). The fast motion part is represented by the part in the denominator which starts with $\tau_c^2\mu\cdots$. k must be large enough to follow the trajectory.

Speed term disturbance:
If we assume that the current i_q is most of the time changing step wise this disturbance is easily compensated by the controller. Again the influence of the speed term is more important. The influence of the ramp wise speed disturbance ($d(s) = n_p i_q/s^2$) on the output is seen from Eq. (4.23).

$$\lim_{s\to 0} s \cdot y(s) = \frac{\mu s(\mu s + 2d_1)B_2}{\mu s(\mu s + 2d_1)(\tau_1 s + 1)(\tau_2 s + 1) + k(\tau_c^2 s + 2\alpha\tau_c + 1)} \cdot \frac{1}{s^2} \qquad (4.23)$$

This results in:

$$y(s) = \frac{2\mu d_1 B_2}{k} \qquad (4.24)$$

k must be large and μ must be small to make the speed term disturbance small. The influence of the ramp disturbance is smaller then in the current loop. This is caused by the μ term which is small.

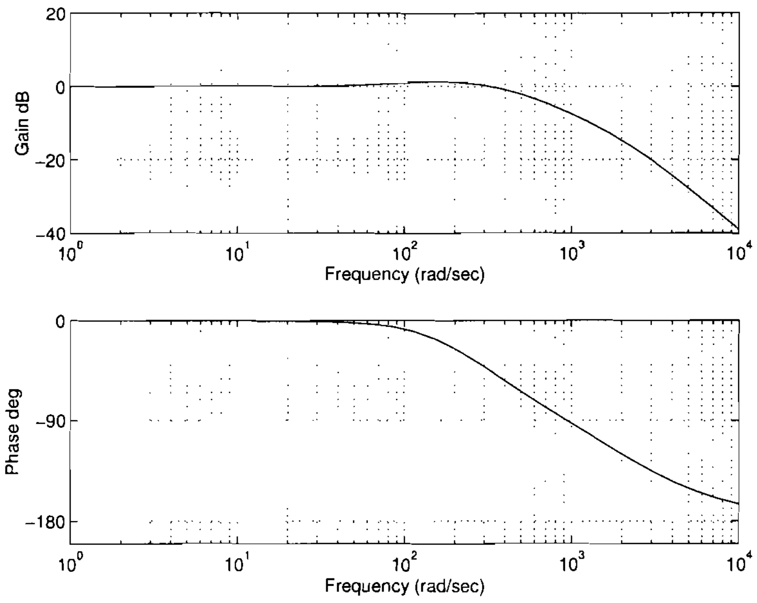


Figure 4.14: Bode plot of $C(s)$ with ($\mu = 0.001$)

Sensor noise:
The complementary sensitivity transfer function is defined by:

$$C(s) = \frac{y(s)}{n(s)} = \frac{k(\tau_c^2 s^2 + 2\alpha\tau_c + 1)}{\mu s(\mu s + 2d_1)(\tau_1 s + 1)(\tau_2 s + 1) + k(\tau_c^2 s^2 + 2\alpha\tau_c + 1)} \tag{4.25}$$

In case the system poles are canceled by the controller zeros the system can be simplified to:

$$C(s) = \frac{k}{\mu s(\mu s + 2d_1) + k} = \frac{1}{\mu_n^2 s^2 + 2\mu_n d_n s + 1}, \tag{4.26}$$

with $\mu_n = \mu/\sqrt{k}$, and $d_n = d_1/\sqrt{k}$.
In figure 4.14 $C(s)$ is drawn for the optimized controller.

To visualize the effects of changing parameters root locus plots are generated in the next part. The most simple case is if the controller zeros exactly cancels the two poles of the flux dynamics. The open loop system is:

$$H(s) = \frac{k}{\mu s(\mu s + 2d_1)} \tag{4.27}$$

The term $2d_1/\mu$ is the fast motion pole. If the gain is high enough, the poles of the closed loop system are on the asymptote. The following formula defines the location of the asymptote.

$$\sigma_c = \frac{\sum_1^n p_i - \sum_1^r z_j}{n - r}, \tag{4.28}$$

with p_i the open loop poles, and z_j the open loop zeros.
From Eq. (4.27) follows:

$$\sigma_c = -\frac{d_1}{\mu} \tag{4.29}$$

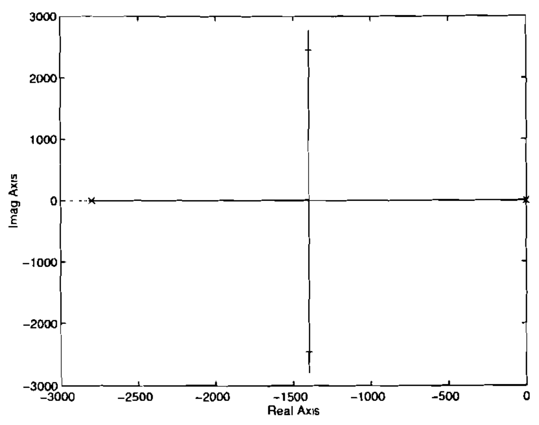


Figure 4.15: Root locus if controller zeros cancel flux poles

For the system to follow the desired trajectory σ_c must be larger then $1/\tau_c$ or in formula form:

$$1/\tau_c < d_1/\mu \tag{4.30}$$

In figure 4.15 the root locus is shown for this case, with o=the zeros, x=the open loop poles, +=the closed loop poles. Increasing k will decrease the damping of the fast motion system. This is an important difference between the first order and second order DCM, as in the first order case this increased the damping. The time scale separation is obvious from this figure. The fast motion poles are much faster than the desired reference.
More difficult is when the poles of the system are not canceled by the controller zeros. If the poles and zeros are close the situation is comparable with the previous case. If not the following is still true:
Using Eq. (4.28) gives:

$$\sigma_c = \frac{-2d_1/\mu - \gamma - \eta + \alpha/\tau}{2} \tag{4.31}$$

The parameter $-2d_1/\mu$ must be chosen in the same way as before. In this case the α/τ term pulls the asymptote extra in direction of the origin. Therefore $-2d_1/\mu$ must be larger to fulfill Eq. (4.30). This is shown in figure 4.16. This last case can be simplified to the first case as seen from the two root locus diagrams. This is handy because the second order case can still be used to choose parameters.

The following design rules can be defined:

- For a smaller steady state error then p percent:
  $k > 2\mu d_1 B_2/(n_p i_q p)$

- Sufficient time scale separation:
  General:
  $1/\tau_c \leq d_1/\mu$
  If controller zero's cancel controller poles:
  $\mu_n = \mu/\sqrt{k} \leq 0.1\tau_1$

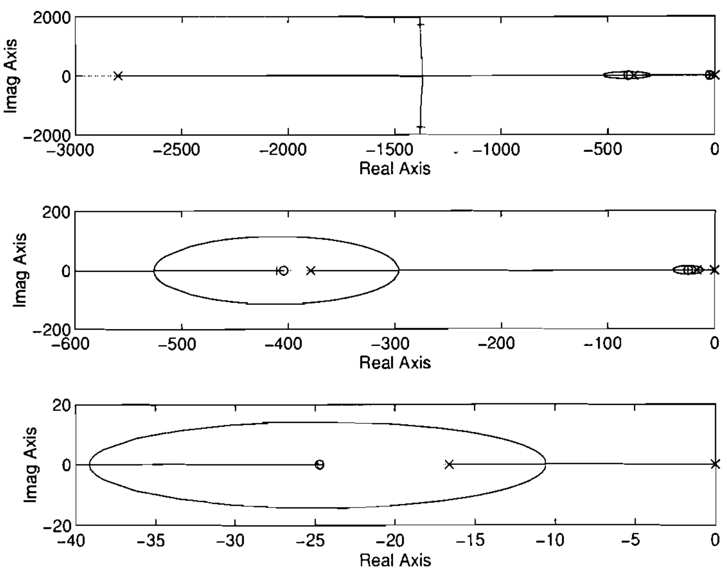


Figure 4.16: Root locus if controller zeros do not cancel flux poles. The upper picture shows the total view of the root locus, the two other pictures are zoomed views of the upper.

- No high bumps in the $C(s)$ and good damping:
  $d_n = d_1/\sqrt{k} > 0.5$
  $\tau_c > \tau_{noise}$

4.2.3 Simulation

To show the effect of changing μ two simulations are presented, one with too large μ and one with properly chosen μ. Figure 4.17 shows what happens if there is not enough time scale separation (μ not small enough). As $i_q = 0$ no torque is produces, and the motor speed is zero. The system reduces to a simple electrical network with a coil and a resistance. With zero speed in steady state the flux and the current are constant. u_d is DC and therefore $u_d = i_d R_s$. The final value of u_d can be calculated by using $\psi_d = M i_d$.
This gives

$$u_d = \frac{\psi_d R_s}{M} \tag{4.32}$$

Figure 4.18 show the flux response for the optimized controller. From the simulations is seen that a high stator voltage is only needed for transients when an acceleration of the flux response is required compared with the uncontrolled system. An interesting aspect in simulation to study is what happens if the flux is constant and i_q is changing. Is the speed term $n_p \omega i_q$ the most important disturbance as assumed, or is $\eta \frac{i_q^2}{i_d}$ important in some cases? To check this a step is made on i_q with constant flux and zero speed ($t = 0.5$ s). Later the same step is made with speed at a high value ($t = 0.6$ s, and $t = 1.0$ s). This is simulated for two different values of ψ_d, $\psi_d = 1$ Wb and $\psi_d = 0.1$ Wb. From simulation, figures 4.19, 4.20 is seen that decreasing flux will increase the influence of the term $\eta \frac{i_q^2}{i_d}$. Especially the difference in flux error at $t = 0.5$ s and $t = 1$ s makes clear that for $\psi_d = 1$ Wb the speed term

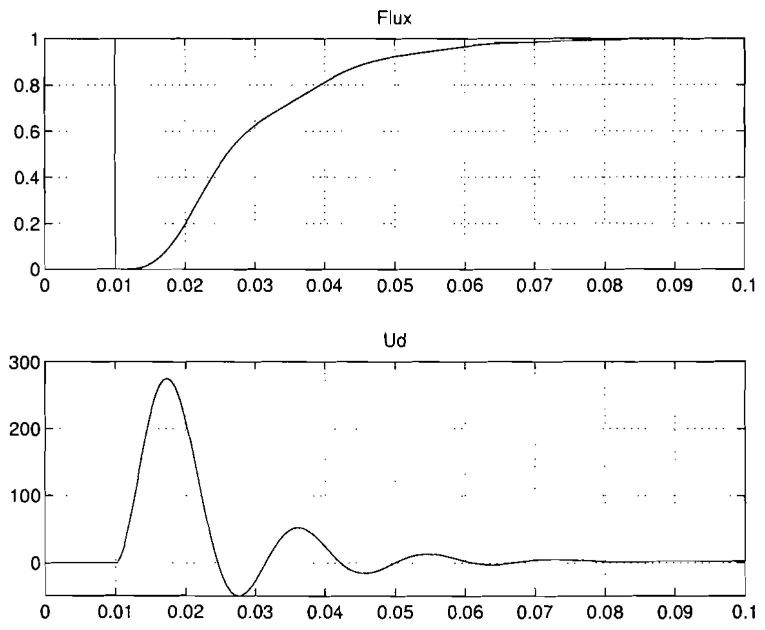


Figure 4.17: Typical flux response with insufficient time scale separation ($\mu = 0.01$)

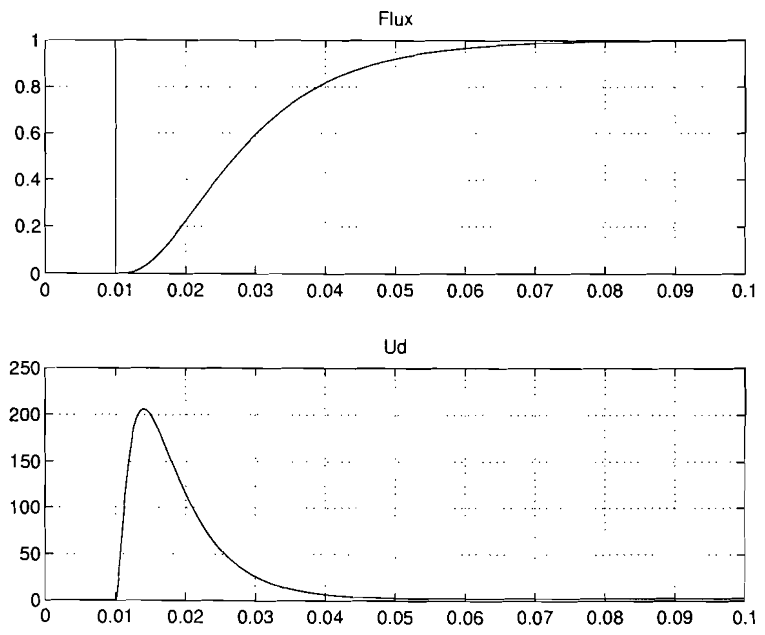


Figure 4.18: Flux response for optimized controller with ($\mu = 0.001$)

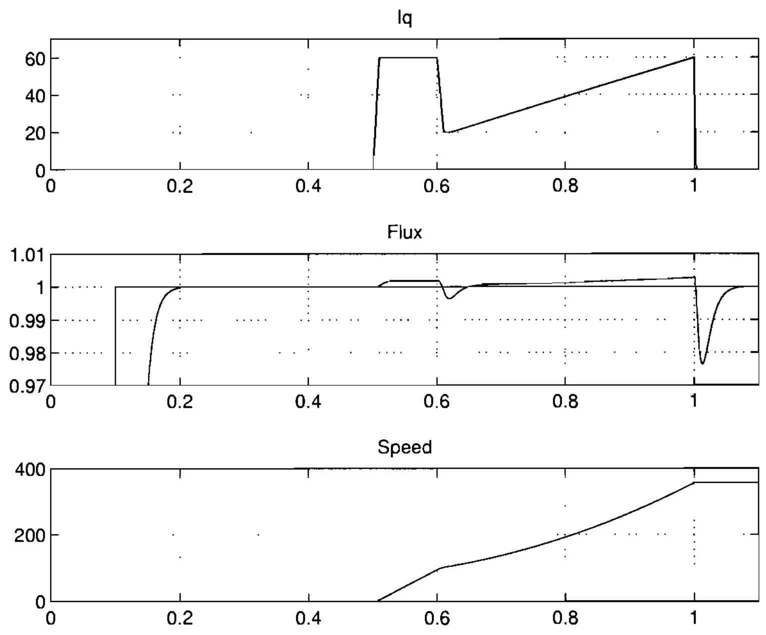


Figure 4.19: Influence of disturbances on ψ_d, with $\psi_d = 1$ Wb

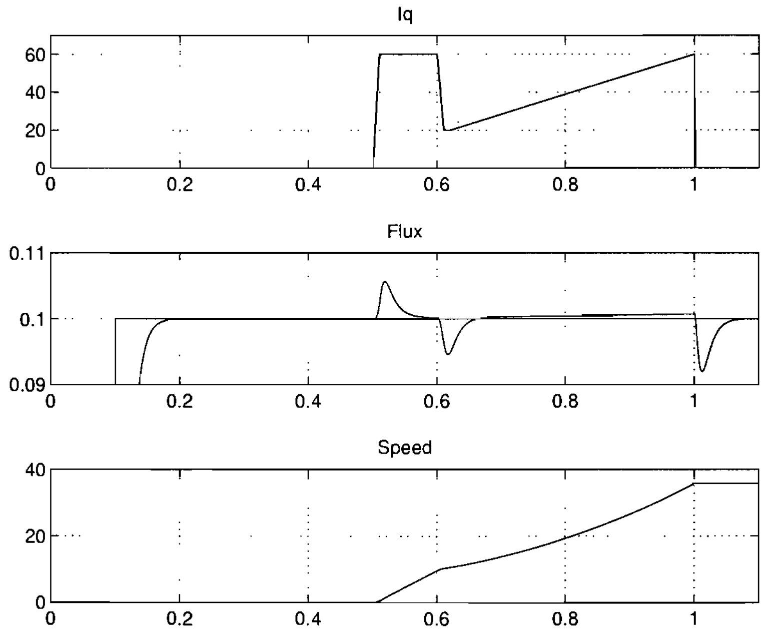


Figure 4.20: Influence of disturbances on ψ_d, with $\psi_d = .1$ Wb

is the most important disturbance. If $\psi_d = 0.1$ Wb the influence of $\eta\frac{i_q^2}{i_d}$ is larger. This can be seen from the flux error at at $t = 0.5$ s and $t = 1$ s. The difference in error is almost the same. This means that the speed term has not large influence on the flux error in this case. The speed is still very low. Practically the speed term disturbance will always be most important. The motor is not accelerated at low speeds with almost zero flux. The flux is reduced only at high speeds (field weakening) to reduce inverter voltage and still increase speed.
The optimized controller fulfills our design goals. It is not sensitive to noise. τ is chosen in such way that the inverter voltage limit ($V = 230V$) is not exceeded [1] .
The controller parameters are presented in table 4.5.

Table 4.5: DCM controller parameters

Parameters	Values
α	1
τ	1e-2
d_0	0
d_1	1.4
μ	0.001
k	1.6

4.3 Comparison of second order DCM with PI

The flux is controlled with two PI controllers, one to control the current and one to control the flux. This is shown in figure 4.21.

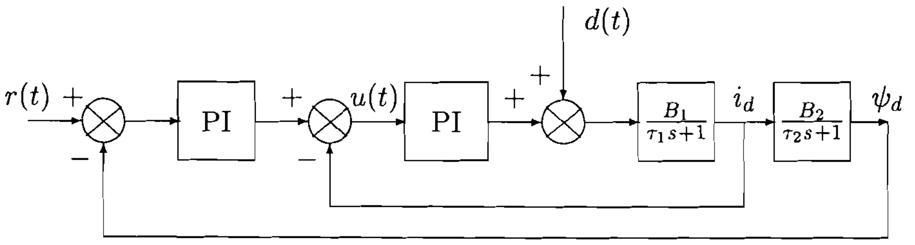


Figure 4.21: PI flux control system

The current controller is much faster than the flux controller. This is the most important design rule in this system. The dynamics of the current loop can be neglected, and the system behaves like first order. If two DCM controllers were used the same design rules can be applied.
If this DCM system is optimally tuned, the system will behave like the second order DCM

[1]The flux controller can be made more robust by increasing the gain and decreasing the controller time constant

controller. Maybe there is a smaller steady state error, which is caused by the extra information from the current sensor. The loops can be analyzed the same way as in section 4.1.1. The same reason can be used as before why this loop will work better with DCM controllers. It is only shown here as an alternative solution.

4.4 Implementation of controllers.

For practical implementation the controller equations must be translated to the state space representation.
The analog state space representation is used to implement the controller in Simulink$^{\mathrm{TM}}$ S-function.

4.4.1 The current controller

Rewriting Eq. (3.31) to the following form and introducing the state x gives:

$$\begin{aligned} u(s)\left[1+\frac{d_0}{\mu}s^{-1}\right] &= \frac{1}{\mu}s^{-1}r(s)-\frac{\tau}{\mu}y(s)-\frac{1}{\mu}s^{-1}y(s) \\ u(s) &= -\frac{\tau}{\mu}y(s)+s^{-1}\left[\frac{1}{\mu}r(s)-\frac{1}{\mu}y(s)-\frac{d_0}{\mu}u(s)\right] \\ u(s) &= -\frac{\tau}{\mu}y(s)+x(s) \\ sx(s) &= \frac{1}{\mu}r(s)-\frac{1}{\mu}y(s)-\frac{d_0}{\mu}u(s) \\ sx(s) &= \frac{1}{\mu}r(s)-\frac{1}{\mu}y(s)+\frac{d_0\tau}{\mu^2}y(s)-\frac{d_0}{\mu}x(s) \end{aligned} \tag{4.33}$$

Using inverse transformation gives:

$$\begin{aligned} \dot{x} &= -\frac{d_0}{\mu}x+\frac{1}{\mu}r+\left[-\frac{1}{\mu}+\frac{d_0\tau}{\mu^2}\right]y \\ v &= k\left[x-\frac{\tau}{\mu}y\right] \end{aligned} \tag{4.34}$$

Introducing the new state $x=\tau/\mu$ gives the desired state-space form:

$$\begin{aligned} \dot{x} &= -\frac{d_0}{\mu}x+\frac{1}{\tau}\left[r-y\right]+\frac{d_0}{\mu}y \\ v &= \frac{k\tau}{\mu}\left[x-y\right] \end{aligned} \tag{4.35}$$

4.4.2 The flux controller

First (3.35) is translated to a transfer function and written in the form:

$$V(s)=k\left[\frac{\frac{s^2}{\mu^2}+\frac{2\alpha}{\tau\mu^2}s+\frac{1}{\tau^2\mu^2}}{s^2+\frac{2d_1}{\mu}s+\frac{d_0}{\mu^2}}y(s)-\frac{\frac{1}{\tau^2\mu^2}}{s^2+\frac{2d_1}{\mu}s+\frac{d0}{\mu^2}}r(s)\right] \tag{4.36}$$

Using appendix C this is written into statespace:

$$\begin{aligned}
\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} &= \begin{bmatrix} \frac{-2d_1}{\mu} & 1 \\ -\frac{d_0}{\mu^2} & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \frac{1}{\mu^2} \begin{bmatrix} \frac{2\alpha}{\tau} - \frac{2d_1}{\mu} \\ \frac{1}{\tau^2} - \frac{d_0}{\mu^2} \end{bmatrix} y + \frac{1}{\mu^2} \begin{bmatrix} 0 \\ \frac{-1}{\tau^2} \end{bmatrix} r \\
v &= x + \frac{1}{\mu^2} y
\end{aligned} \tag{4.37}$$

Multiplying the old states by $\frac{-1}{\mu^2}$ and simplifying gives:

$$\begin{aligned}
\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} &= \begin{bmatrix} \frac{-2d_1}{\mu} & 1 \\ -\frac{d_0}{\mu^2} & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} \frac{2\alpha}{\tau} - \frac{2d_1}{\mu} \\ \frac{1}{\tau^2} - \frac{d_0}{\mu^2} \end{bmatrix} y + \begin{bmatrix} 0 \\ \frac{-1}{\tau^2} \end{bmatrix} r \\
v &= \frac{k}{\mu^2}(y - x)
\end{aligned} \tag{4.38}$$

Chapter 5

Outer loops/time optimal control

5.1 Introduction

The general control problem in this chapter is applied to an elevator system. If the dynamics of the current loop can be neglected the system will reduce to a two integrator system with acceleration as input and the position as output. For given bounds on current, and velocity the system can be solved as a time optimal control problem. The approximation of two integrators for the AC-drive is not completely true, because of the additional time constant in the acceleration. This is still solved as a two integrator problem. The problem with an extra time time constant is difficult to solve and the derivation of the trajectory is based on complex mathematics. Solution to this problem can be found in [12].
In this chapter two solutions to the double integrator are presented: a nonlinear solution and a linear with feed forward signals [13]. Also will be explained in what system this approximation may be used.

5.2 Electrical limits in the system

In table 2.1 the motor parameters are given. The physical constraints are given by [14]:

$$U_a^2 + U_b^2 = U_q^2 + U_d^2 \leq V_{max}^2 \tag{5.1}$$

$$i_{sa}^2 + i_{sb}^2 = i_q^2 + i_d^2 \leq I_{max}^2, \tag{5.2}$$

where V_{max} and I_{max} are respectively the fixed voltage and current limit.

Current Limit
The maximum acceleration can be calculated from Eq. (2.24):

$$\frac{d\omega}{dt} = \mu\psi_d i_q - \frac{T_L}{J}, \tag{5.3}$$

and depends on flux, current and applied torque. ($i_{qmax} = 50$ A and $\psi_d = 1$ Wb). This is different if the lift goes up or down.

Stator voltage limit
Another limit in our system is the stator voltage. To increase torque, i_q has to be increased.

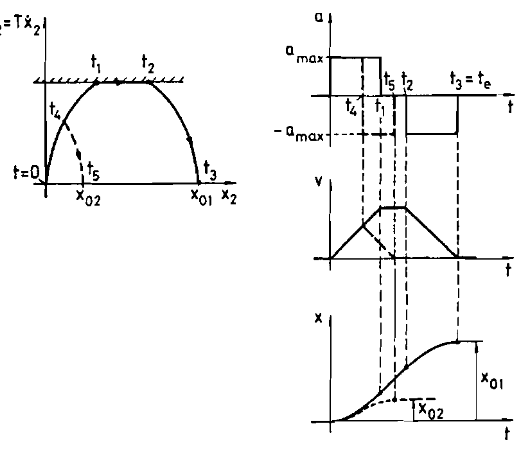


Figure 5.1: Theoretical responses to different position inputs

With constant electrical torque ψ_d and i_q are constant. This means that $di_q/dt = 0$. In steady state the current equation of Eq. (2.24) can therefore be written as:

$$u_q = \sigma L_s(\gamma i_q + \beta n_p \omega \psi_d + n_p \omega i_d + \eta M i_q i_d/\psi_d) \tag{5.4}$$

If the speed increases the dominant term $\omega n_p(\beta\psi_d + i_d)$, the speed term, will get larger and u_q must be increased to keep $di_q/dt = 0$. u_q is limited by the maximum inverter voltage. To avoid a high inverter voltage the flux ψ_d can be decreased, which of course means a decrease in electrical torque. This is the same as a decrease in maximum acceleration. If the flux is decreased too much the motor will eventually decelerate. To find a time optimal trajectory with different maximum speeds, fluxes and torque is difficult. Therefore in our setup the flux is kept constant and a maximum speed limit is used.

5.3 Design of the trajectory

Under assumption of a fast current loop, and neglecting friction the system can be approximated by a double integrator. The time optimal solution consists of two intervals of maximum acceleration/deceleration. If the position error is large enough there is a third interval with maximum speed and zero acceleration. The relation between speed and position error can be calculated as follow:

$$\begin{aligned} x &= \frac{1}{2} a_{max} t^2 \\ \omega &= a_{max} t \end{aligned} \tag{5.5}$$

a_{max} can be calculated from Eq. (5.3) if torque is known. Eliminating the time gives the relation between position and velocity:

$$\omega = \sqrt{2 a_{max} |x_{error}|} sign(x_{error}) \tag{5.6}$$

In figure 5.1 different time optimal responses are shown for limited acceleration. Eq. (5.6) can be implemented in two different ways. The function can be implemented as a nonlinear block (nonl) in the control loop, or it can be used in a circuit which generates feed forward signals for a linear control system.

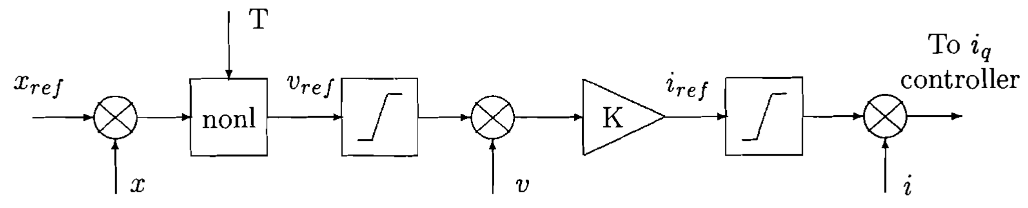


Figure 5.2: Controller with nonlinear function in the control loop

5.4 The non feed forward structure

In figure 5.2 the controller with a nonlinear function in the control loop is shown. The control system works as follows. A step on the input will result in a very high gain from the nonlinear block which will saturate the speed controller (if the position error is large enough). The position error starts decreasing and after some time the velocity controller comes out of saturation and the nonlinear function starts the breaking trajectory. In this setup the maximum acceleration is realized by the current limit. The deceleration curve is realized by the nonlinear block. This function needs to be dependent on torque. The sign of the torque should be changed in this function if the lift goes up or down.
A problem is the high gain around zero. The function has to be linearized for two reasons. First if the system was ideal, the high gain will still lead to oscillation around zero and the non ideal current step will otherwise cause large overshoot. Second, as the current controller will make not an ideal step the region has to be linearized to stop the elevator in time.
In figure 5.4 the nonlinear function is shown. The solid line is the ideal function, and the dashed line is the linearized part of the nonlinear function.

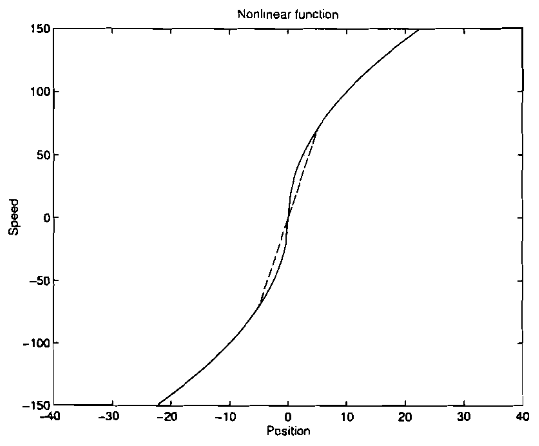


Figure 5.3: The nonlinear function

A PI instead of a P velocity controller can be used to have no steady state error in speed. However a anti-windup part is then required to prevent the controller from saturating. In our setup a P controller is chosen because the simulation results were good enough and no

Table 5.1: Simulation parameters for τ_{i_q}=1e-3, for non-feed forward structure

i_{max}	50 A
ω_{max}	150 rad/s
T	10 Nm
J	2*0.0586
k_{speed}	80
τ_{i_q}	1e-3
k'_{iq}	50
linearizing zone	5

anti-windup part is needed.

5.4.1 Boundaries on operation

At the beginning of the design the control system limits are known, and we can tell if the time optimal control presented here can be used or not. It is clear that this solution can not be applied to all systems.
In section 2.5 was shown that in a practical system the inertia is much larger than only the motor inertia if a stiff system is assumed.
With small inertia and not fast enough current rise time (this is limited by the inverter), the second order time optimal approximation is no longer valid. The current controller will be too late to stop the elevator at the right position during the braking period. This results in overshoot and some extra current peaks. Therefore the system will work not properly.
A solution to this problem is increasing the linearisation zone. With larger inertia the system becomes slower and the time scale separation is getting larger. The system follows better the time optimal trajectory.
The nonlinear function depends on maximum speed, flux, inertia, current, torque, and the linear region. All these parameters have influence on the form of this function. It is interesting to examine the effects on the nonlinear function when these values are changed. For the moment there is assumed that the linearisation zone is not changed. If inertia is increased the slope of the linearisation function will decrease. The function becomes more close to the theoretical function. The same happens if the flux or torque is decreased. If for example the inertia is decreased too much the nonlinear function will be a totally linear function, with still the limit on maximum speed. The same can be shown for torque, flux and speed. These must be in such way that the idea of the time optimal trajectory is not lost.

5.4.2 Simulation

First simulations were done with τ_{i_q} =1e-3. Increasing k of the velocity controller had a positive influence on the the steady state error but it was already very small ($\ll$1%). The overshoot was 0.17 %. In table 5.1 the parameters are shown. Figure 5.4 and 5.6 show the two different cases for a small and a large position error. In case of a small position error the controller is not time optimal and especially the last part is slow. The steady state error is 0.13%. Increasing k has negative influence on the peaks in the control signal. A positive effect of increasing k is the faster time to steady state and less overshoot in position. Figure

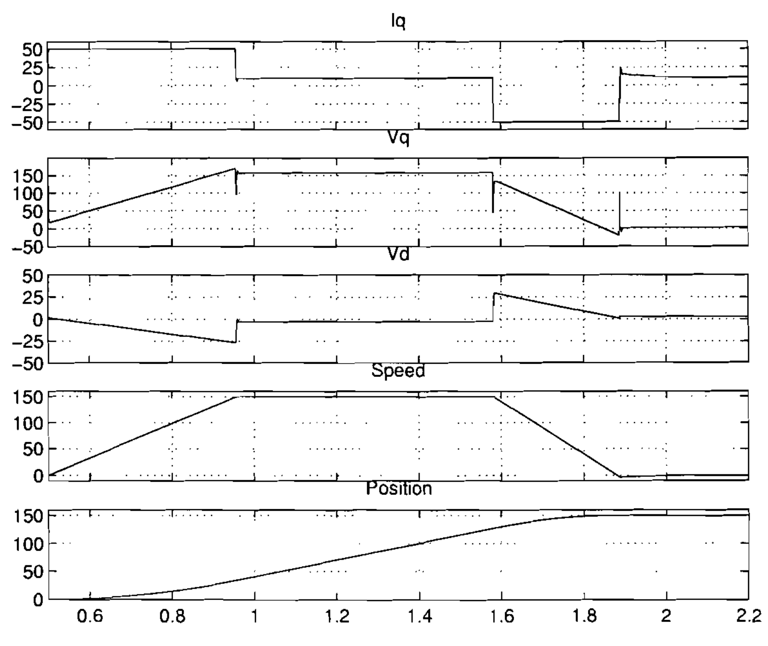


Figure 5.4: Large initial position error with k_{speed}=80

5.7 shows the influence of the linearisation region. This effect can only be observed for small position errors.
Next simulations were done with τ_{i_q}=5e-3 for the current controller, to see what the influence is of a slower current controller. The simulation parameters are given in table 5.2. Figure 5.8 shows the results if the current controller is not fast enough. Increasing the gain of the speed controller resulted in a higher and more oscillatory current peak, but a faster settling time and a smaller steady state error. The overshoot is 0.9% and the steady state error is $\ll$1%. In figure 5.9 the influence on torque shocks in steady state is shown. As the position controller has no integration the steady state error is small but not zero.

Table 5.2: Simulation parameters (slow current controller), for the non feed forward structure

i_{max}	50 A
ω_{max}	150 rad/s
T	10 Nm
J	2*0.0586 kgm^2
k_{speed}	10
τ_{i_q}	5e-3
k'_{iq}	50
linearisation zone	5

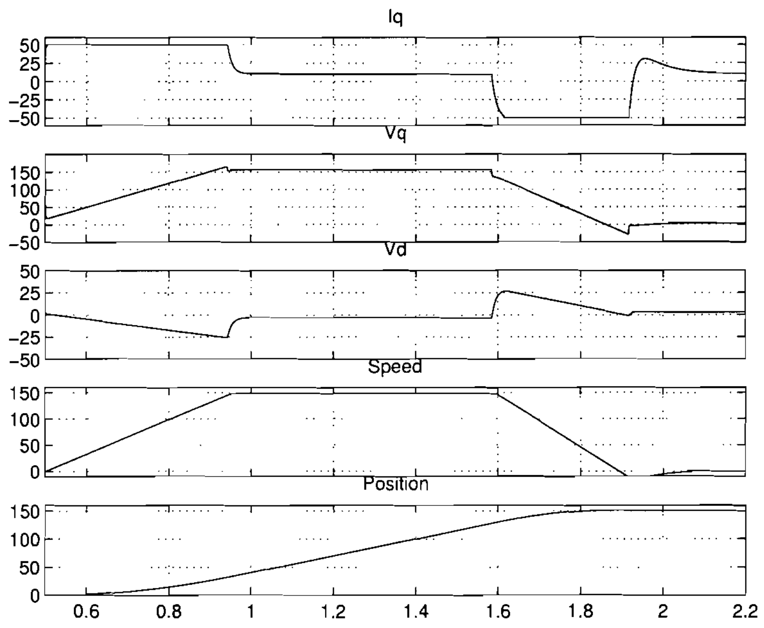


Figure 5.5: Large initial position error with smaller k_{speed}, k_{speed}=10

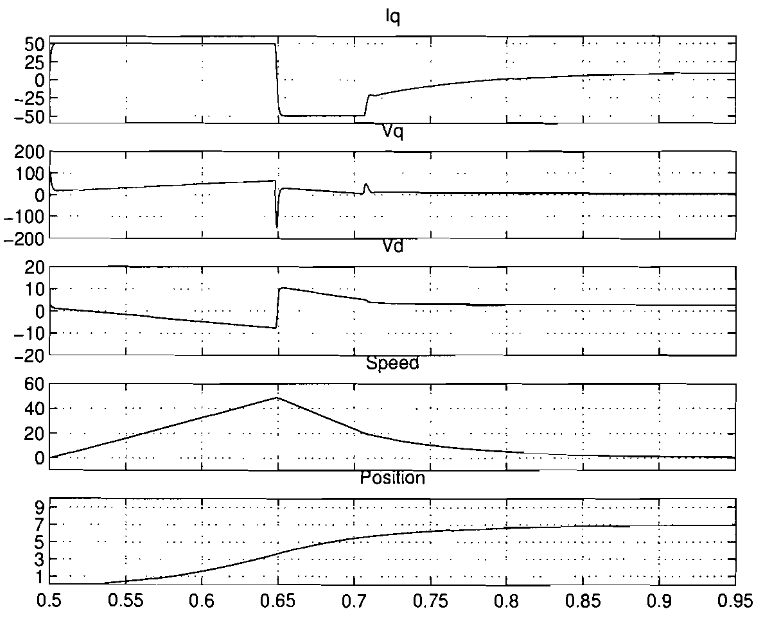


Figure 5.6: Small initial position error

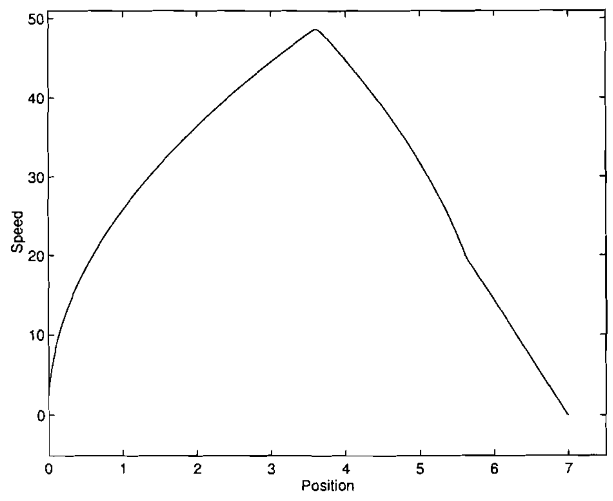


Figure 5.7: Simulated state-space trajectory

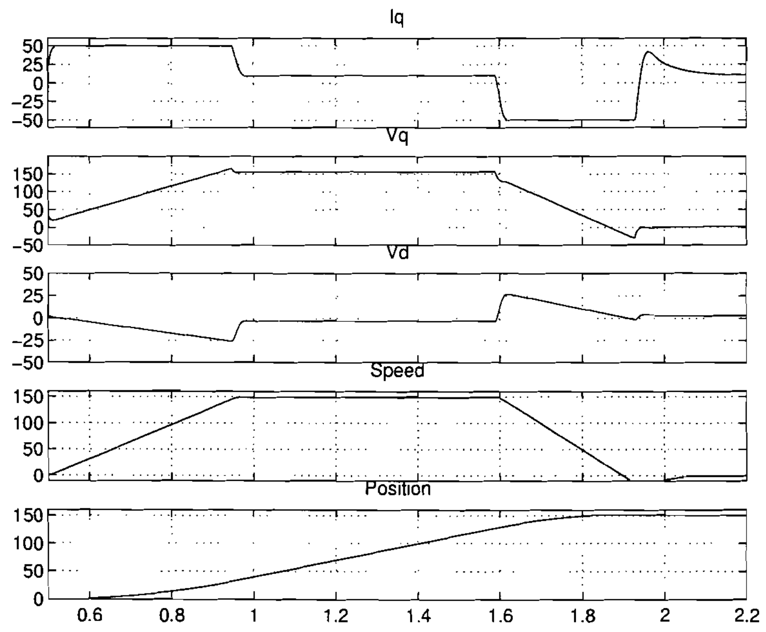


Figure 5.8: Influence of slow current controller

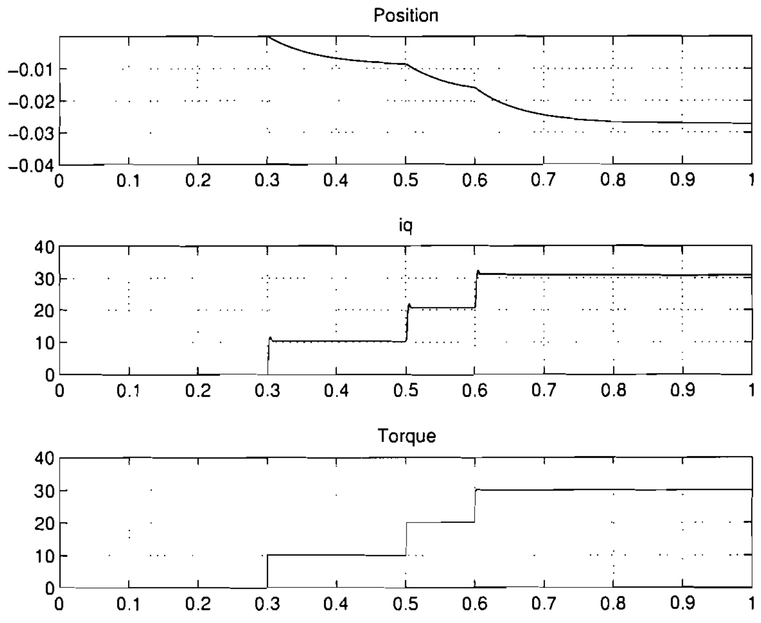


Figure 5.9: The effect of torque shock on the steady state error

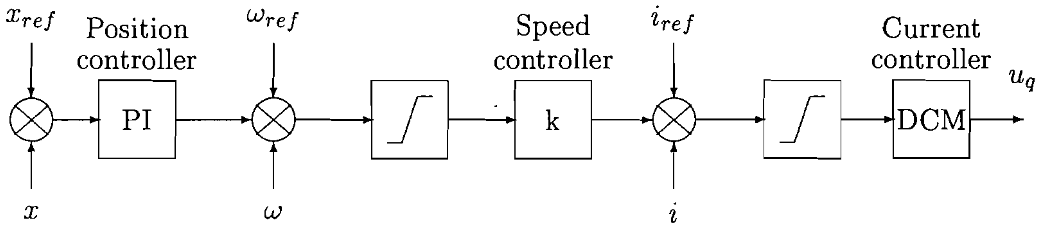


Figure 5.10: Control structure block

5.4.3 Conclusion

The design of the trajectory is very simple, but the linearized function could be changed in such way that the idea of time optimal control is lost.

To successfully apply our control structure the system error and speed must be large enough, otherwise it makes no sense to use time optimal control and the nonlinear function will then be used in its linear area.

Increasing the gain of the velocity controller has positive effect on steady state and settling time of the position. The peaks in the v_q response increase with increasing gain.

The torque shocks caused a small position error in steady state, which can not be avoided in this control structure.

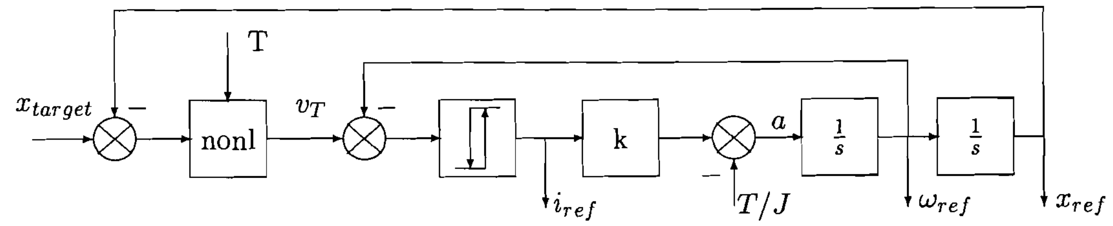


Figure 5.11: Feed-forward control block

Table 5.3: Simulation parameters for the feed forward structure

i_{max}	50 A
ω_{max}	150 rad/s
T	0 Nm
J	2*0.0586 kgm^2
$k_{position}$	10
$\tau_{position}$	2
k_{speed}	10
τ_{i_q}	10e-4
k'_{i_q}	50
Linearisation zones	
Sign function	0.1
Trajectory function	0.1

5.5 The feed forward structure

In figure 5.10 the linear control loop is shown. Eq. (5.6) is used in the feed forward structure in figure 5.11 to obtain the desired reference trajectory. The feed forward part is a model of a ideal time optimal system.
This loop has the advantage of behaving time optimal for large position errors. If the position error is close to steady state the controller will behave linear. As the position controller is a PI controller the steady state error will be zero. A step on the input results in high gain from the nonlinear block. The hysteresis block switches to maximum positive gain. The electrical torque is $i_q\psi_d\mu$ (See Eq. 5.3). The gain block multiplies i_q by $k = \psi_d\mu$. The torque must be subtracted or added to get the right acceleration. This depends on the sign of the current. Integrating acceleration twice gives the feed forward signal for position, which is fed back to the input. The sign function is linearized to prevent slow simulation times. If the feed forward signal for the current controller is zero, the sign function switches fast from minus to maximum current. Simulink^{TM} will decrease the step size and the simulation is slowed down.

5.5.1 Simulations

The simulation parameters are shown in table 5.3. The simulation results of the control loop for different step sizes are presented in figures 5.12, and 5.13. In figure 5.14 the time optimal behavior for a small position error shown. The gain of the position controller can not be too large, this will cause overshoot. The linearisation zone of the nonlinear function can be made

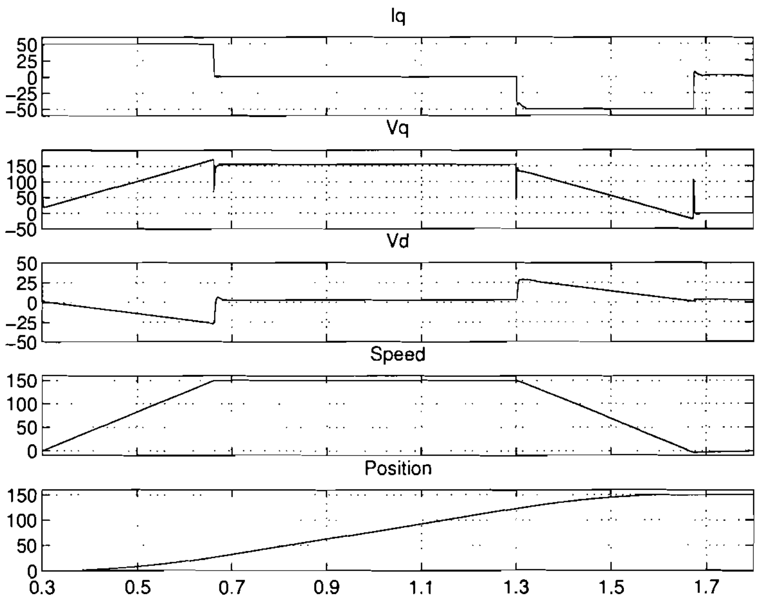


Figure 5.12: Large initial position error

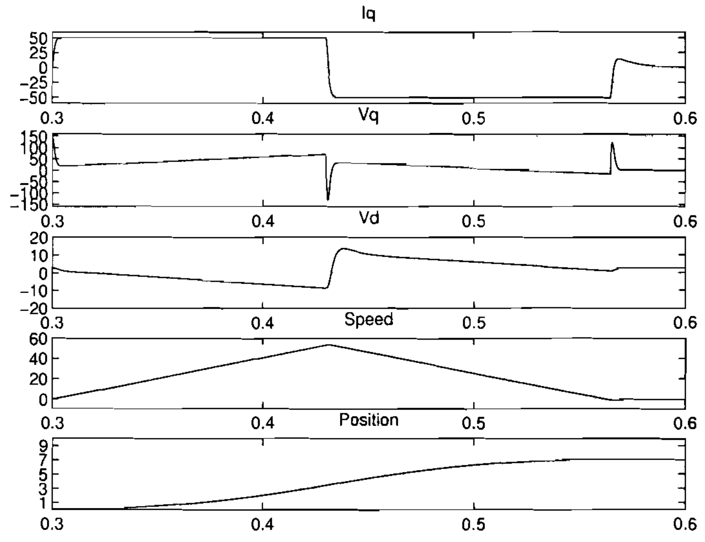


Figure 5.13: Small initial position error

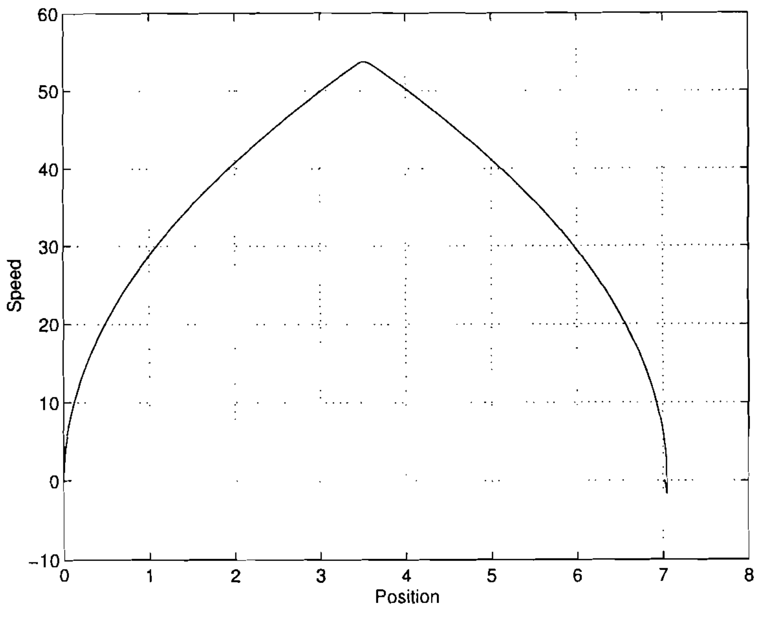


Figure 5.14: State-space trajectory

considerably smaller than in the previous structure. Therefore the system will behave also time optimal for smaller position errors. In figure 5.15, 5.16 is shown what happens if no torque is assumed in the feed forward structure and a torque of $T = 10$ Nm is applied to the system. Large errors between reference and real position for long time saturate the position controller. An anti windup part is needed to prevent this. In simulation this is avoided by using a larger integration time constant.

5.6 Comparison with non feed forward structure

The feed forward (FF) structure has the advantage of behaving linear for small errors. Also for smaller steps than in the non feed forward structure the system is time optimal. As the position error in the loop is a PI controller the steady state error is zero. This is not true with the non feed forward (NFF) structure the position error depends on the gains in the loop. The steady state error can be decreased by increasing the gains. A big advantage of the NFF structure is that torque can be unknown at start. This is due to the fact that the first part is not realized by the nonlinear function. After some time the torque is observed and it can be used in the nonlinear braking curve. In the feed forward structure the torque has to be known at the beginning otherwise wrong feed forward signals for current, speed and position controller are generated. The influence of this wrong signals is best seen if the position error is large. For small errors with torque the response is still better than in the NFF structure. Another important advantage of the NFF structure is it robustness against torque shocks. The nonlinear function will still use the right position error information and decelerate following the trajectory. The FF structure is also sensitive to inverter delays. If the inverter reacts with a time delay, wrong and too late feed forward signals are generated.

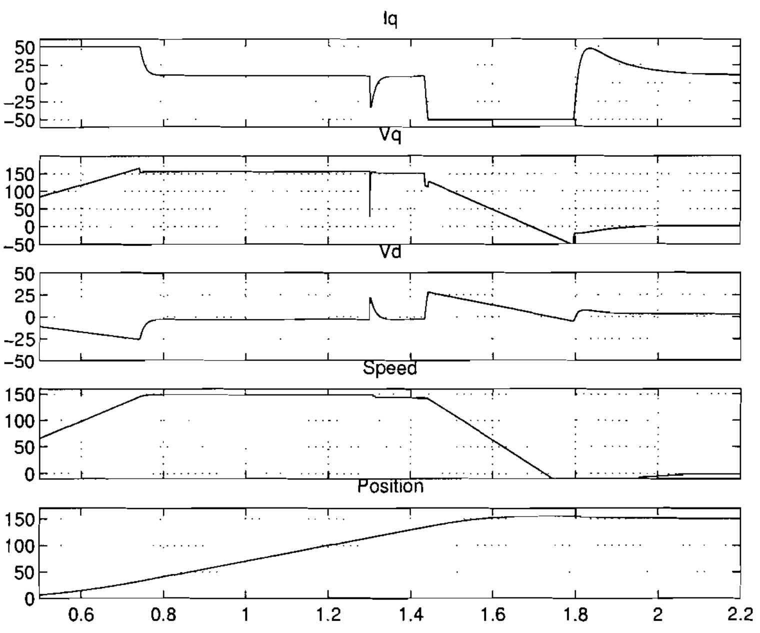


Figure 5.15: Large initial position error with torque error, with $tau_{position} = 10$

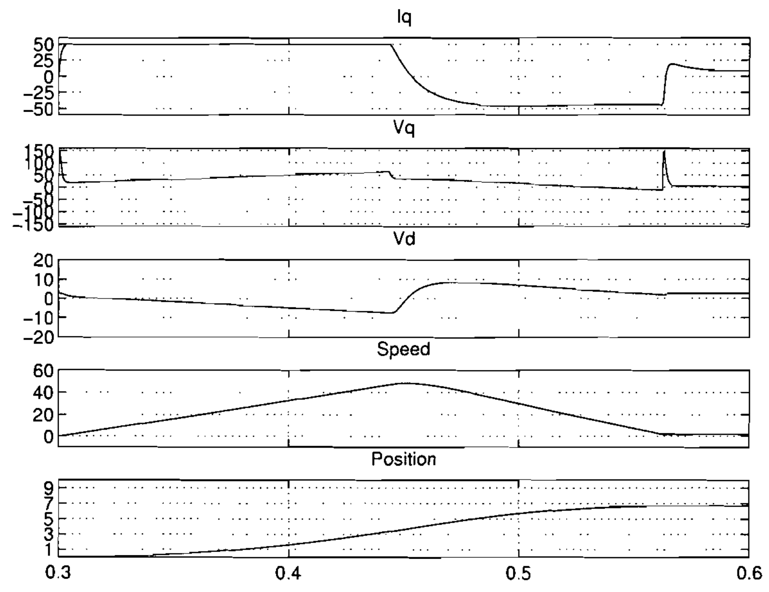


Figure 5.16: Small initial position error with torque error

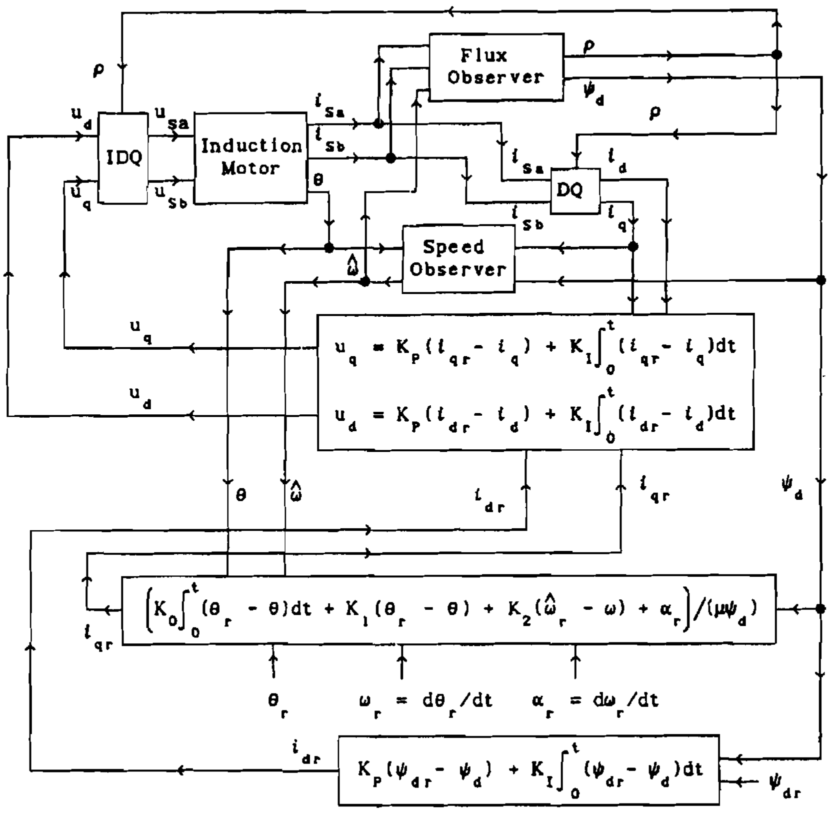

Flux Observer
Induction Motor
IDQ
DQ
Speed Observer
$u_q = K_p(i_{qr} - i_q) + K_I\int_0^t (i_{qr} - i_q)dt$
$u_d = K_p(i_{dr} - i_d) + K_I\int_0^t (i_{dr} - i_d)dt$
$\left[K_0\int_0^t (\theta_r - \theta)dt + K_1(\theta_r - \theta) + K_2(\hat{\omega}_r - \omega) + \alpha_r\right]/(\mu\psi_d)$
$\omega_r = d\theta_r/dt$
$\alpha_r = d\omega_r/dt$
$K_p(\psi_{dr} - \psi_d) + K_I\int_0^t (\psi_{dr} - \psi_d)dt$
ψ_{dr}
i_{dr}
i_{qr}

Figure 5.17: Control structure from article

Another effect which is not simulated here is the difference between real and estimated inertia. Smaller inertia is no problem, as the system will be faster then estimated. A much larger inertia than estimated will cause extra bangs in the NFF structure. The FF structure will generate wrong signals, but the linear loop will still work.
It is difficult to tell what structure is better. Each structure has its own advantages and disadvantages. The main difference is for small position errors. The FF structure is better in this case.

5.7 Comparison with article structure

The structure in figure 5.17 is presented in [1] consists of only PI controllers. The inner loops are different in this setup. Two current controllers are used to force the motor in the so called *current-command mode.* These controllers are fast enough to follow the reference currents i_{dr} and i_{qr}. One controller is used to control the position, and the other to control the flux. The advantage of our control structure is simplicity. Our current and flux controller have higher gains then these controllers and therefore have a smaller steady state error and a higher robustness. In this article a flux reference is generated which maximizes the torque at each speed without violating the voltage and current limits, which is explained in [14].

Unfortunately, the analysis in [14] was performed using steady-state values and is only useful when electrical transients are much faster than the mechanical ones. This is only true for micro motors (low power) and useless for larger motors. Another difference is the small flux values used in this article. No larger values for flux $\psi_d \geq 0.05$ Wb can be used. This is due to the low current limits and smaller value of M. The problem is how to create a time optimal trajectory for different position errors in this case.

Chapter 6

Adding observers

6.1 A flux observer without a speed measurement

The following observers are obtained from [15]. The second and the fifth equation of system 2.24 describe the dynamics of ρ and ψ_d.

$$\begin{aligned} \dot{\rho} &= n_p\omega + \eta M \frac{i_q}{\psi_d} \\ \dot{\psi}_d &= -\eta\psi_d + \eta M i_d \end{aligned} \tag{6.1}$$

$\eta \triangleq R_r/L_r$
To eliminate the dependency on speed, the slip angle S is introduced.

$$\dot{S} = \eta M i_q/\psi_d \tag{6.2}$$

Integrating the first equation of (6.1) gives:

$$\rho(t) = n_p\theta(t) - n_p\theta(0) + S(t) \tag{6.3}$$

The observer equations are given by:

$$\begin{aligned} \dot{\hat{S}} &= \eta M i_q/\hat{\psi}_d \\ \dot{\hat{\psi}}_d &= -\eta\hat{\psi}_d + \eta M \hat{i}_d \\ \hat{\rho}(t) &= n_p\theta(t) - n_p\theta(0) + \hat{S}(t) \end{aligned} \tag{6.4}$$

Where $\hat{\rho}$ and $\hat{\psi}_d$ are the estimated values. The advantage of this approach is that the accuracy no longer depends on the estimate of the speed but is limited by the encoder resolution. The convergence rate of the error dynamics of this observer is bounded by the rotor time constant $1/\eta$. The rotor time constant in our case is: $1/\eta = 0.466s$. In figure 6.1 the practical realization of the flux observer is shown.

6.2 A speed observer

The most simple way to construct a speed observer is differentiating the output:

$$\hat{w}(k) = \frac{\theta(k) - \theta(k-1)}{T}, \tag{6.5}$$

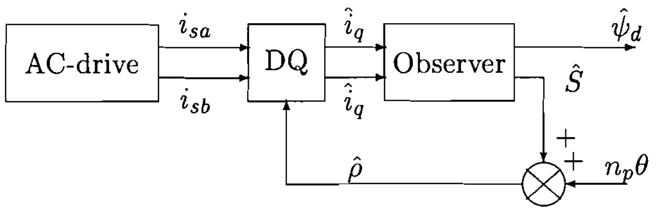


Figure 6.1: Observer implementation

where T is the sample time and k is the sample number. However this leads to large errors at low speeds and high sample rates. In this case the encoder counts per sample period are less than at higher speeds. An alternative way is to use the measured position and estimate the applied torque. The relevant motor equations are:

$$\begin{aligned}\frac{d\theta}{dt} &= \omega \\ \frac{d\omega}{dt} &= \mu\psi_d i_q - T/J \\ \frac{dT}{dt} &= 0\end{aligned} \tag{6.6}$$

$\mu \triangleq n_p M/(JL_r)$.
A speed observer is then defined by:

$$\begin{aligned}\frac{d\hat{\theta}}{dt} &= \hat{\omega} + l_1(\theta - \hat{\theta}) \\ \frac{d\hat{\omega}}{dt} &= \mu\hat{\psi}_d\hat{i}_q - \hat{T}/J + l_2(\theta - \hat{\theta}) \\ \frac{d\hat{T}}{dt} &= -l_3(\theta - \hat{\theta})\end{aligned} \tag{6.7}$$

Even friction can be applied in the torque if this is desired.
The convergence of this system is studied by subtracting (6.7) from (6.6):

$$\begin{aligned}\frac{\epsilon_1}{dt} &= \epsilon_2 - l_1\epsilon_1 \\ \frac{\epsilon_2}{dt} &= -\epsilon_3/J - l_2\epsilon_1 \\ \frac{\epsilon_3}{dt} &= +l_3\epsilon_1\end{aligned} \tag{6.8}$$

Where $\epsilon_1 \triangleq \theta - \hat{\theta}$, $\epsilon_2 \triangleq \omega - \hat{\omega}$, and $\epsilon_3 \triangleq T - \hat{T}$. The observer $\hat{\psi}_d$ converges fast enough to ψ_d. Therefore the assumption $\mu\hat{\psi}_d\hat{i}_q \rightarrow \mu\psi_d i_q$ is used. Transforming system to:

$$\begin{aligned}(s + l_1)\epsilon_1 &= \epsilon_2 \\ s\epsilon_2 &= -\epsilon_3/J - l_2\epsilon_1 \\ s\epsilon_3 &= l_3\epsilon_1\end{aligned} \tag{6.9}$$

Differentiating the second equation of (6.9) and substituting ϵ_2 and ϵ_3 gives the characteristic polynomial of the error system:

$$s^3 + s^2 l_1 + l_2 s + l_3/J \tag{6.10}$$

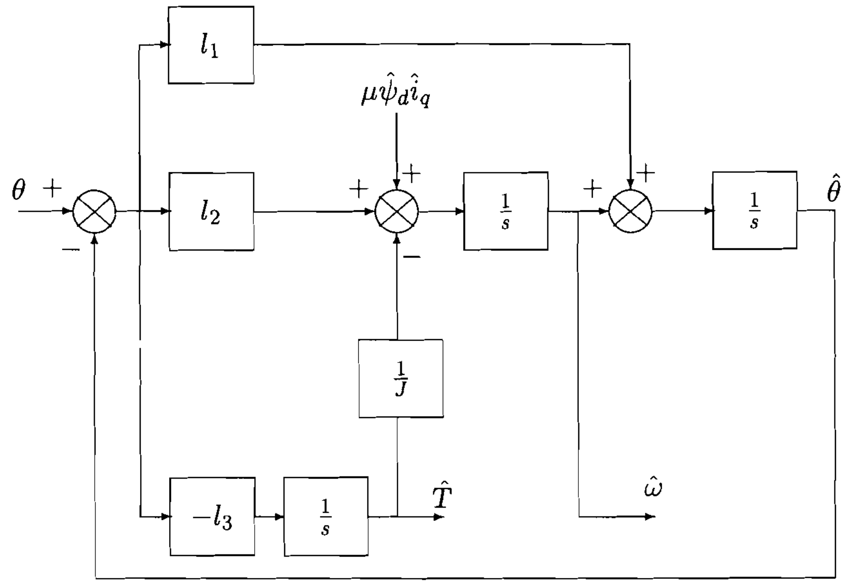


Figure 6.2: Block schema of speed and torque observer

The gains l_1, l_2 and l_3 must be chosen in such way that the poles give a small enough time constant.

Observer gain	Value
l_1	1e4
l_2	1e6
l_3	1e8

The convergence speed can be increased by increasing the gains. Choosing too large gains in practice causes high quantization noise. The torque estimate can now be used to optimally tune the trajectory. The motor is accelerated at maximum current as before and the torque is know after some time. The trajectory can now be changed for the deceleration. An advantage of these observers are that it is not needed to know R_r exactly. If R_r changes this has only influence on the convergation time and this will not introduce an error in the estimation.

6.3 Simulation results

Figure 6.3 shows the simulation results for the observers. Increasing the gain of the torque observer will make it better.
The observers work good, if the rotor resistance is constant. The torque is observed after some time and can be used as input for the nonlinear function in the NFF structure.

6.4 Simulations in $a - b$ coordinate system

In this section the motor is simulated in the $a - b$ coordinate system. ρ is observed and used for the IDQ and DQ transformation. The rest of the observers are not used as inputs for the system in the next simulations. In figure 6.4 a simulation in $a - b$ coordinates is shown. Figure 6.5 and 6.6 shows what happens if the rotor resistance is wrong estimated, and used

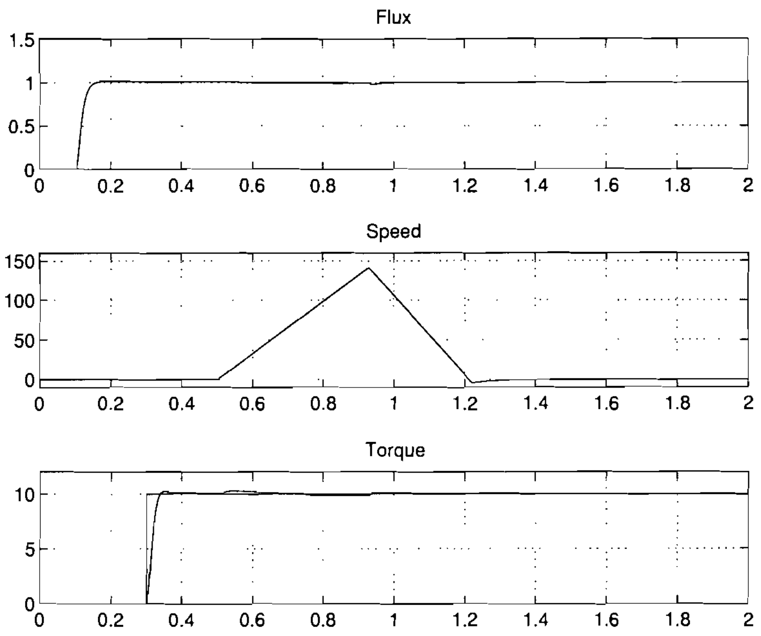


Figure 6.3: Observer responses

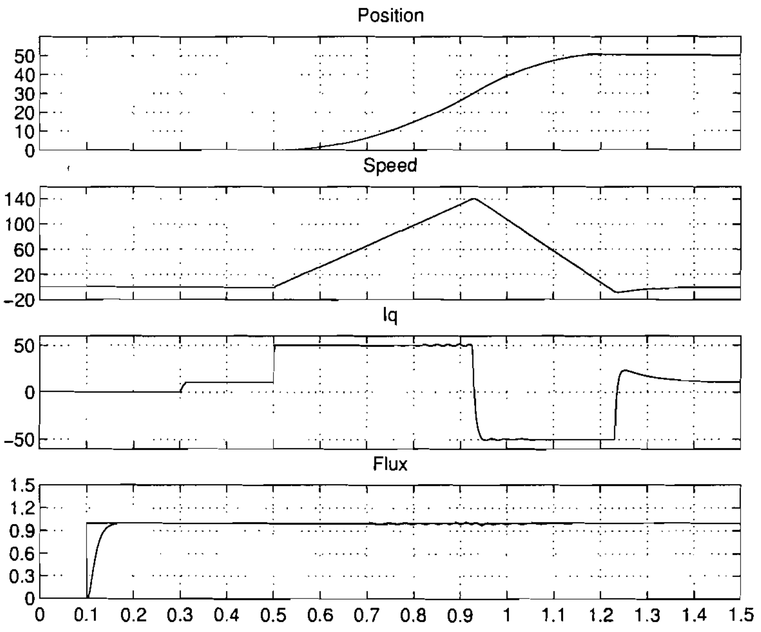


Figure 6.4: Simulation in $a - b$ coordinates, with $R_{r0} = R_r$

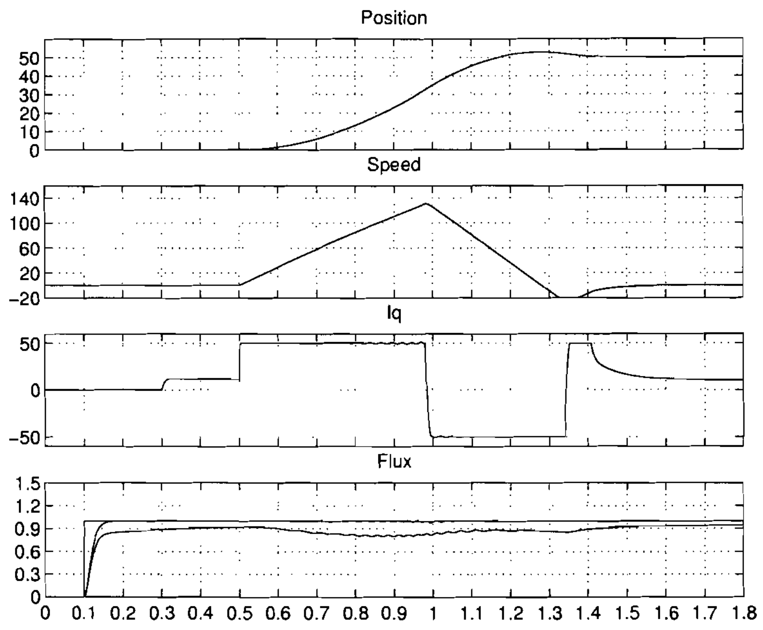


Figure 6.5: Simulation with $R_{r0} = 1.2R_r$

in the observer equation for ρ. R_r is the real resistance, and R_{r0} is the resistance used in the observer equations. The flux figures show the the flux input (step), the flux output, and the observer for flux. The simulation in $a - b$ coordinates shows extra transients in the current and flux responses.

If the rotor resistance R_r changes, the slip changes, and the variable for ρ changes. This has influence on the IDQ en DQ transformations. This causes a steady state error in the flux. Due to the error in flux the torque estimated is also not correct. This make the system performance worse, and can introduce some extra bangs as in figure 6.5. A solution would be to construct an observer for the rotor resistance, which is not presented here.

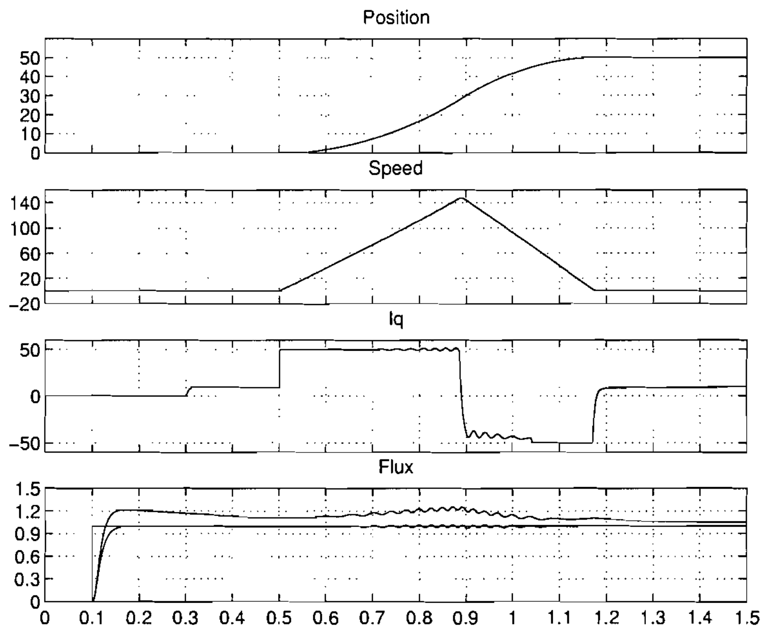


Figure 6.6: Simulation with $R_{r0} = .8R_r$

Chapter 7

Conclusions

The inner loops are controlled by Dynamic Contraction Method (DCM) controllers which have advantages above PI controllers. A comparison between PI and DCM was made if the input was stepwise. In this case the DCM controller was superior. In PI case the gain could not be as high as in the DCM case, because this resulted in oversteered stator voltages. Due to the high gain the DCM controller is less sensitive to disturbances and has a smaller steady state error.
After controlling the current and flux with DCM controllers, the motor is described by a two integrator problem with an extra time constant. If this system can be approximated by a two integrator problem depends on the motor parameters as inertia, maximum speed, rotor time constant, maximum rotor current, position errors and torque.
As shown our motor could be approximated by the two integrator problem. Two control structures which solve the two integrator problem were presented. The non-feed forward structure was superior if torque was not known. It was not time optimal for small position errors. If small position errors are used the feed forward structure is preferred. In case of large position errors and unknown torque the non feed forward structure is better.
A comparison was made between our non-feed forward structure and a control structure from [1]. The article control structure had some disadvantages. The controllers had lower gains and are therefore more sensitive to disturbances, and have a larger steady state error. In this article a flux reference is generated which maximizes the torque at each speed without violating the voltage and current limits [14]. The problem is how to use this flux reference and construct time optimal trajectories for different position errors. Unfortunately the analysis in [14] was performed assuming steady-state values and is only useful when electrical transients are much faster than the mechanical ones. This is only useful for micro motors and useless for larger motors.
Observers were added to the motor control structure. These work well if there was a good observation of the flux angle. If the rotor resistance varies the flux angle is not observed correctly. A solution would be to construct an observer for the rotor resistance, which is not shown in this report.

Bibliography

[1] Marc Bodson, John Chiasson, and Robert Novotnak. High-performance induction motor control via input-output linearization. *IEEE Control Systems*, Aug. 1994.

[2] G. Bartolini, P. Pisu M. Marchesoni, and E. Usai. Chattering reduction and robust position control in induction motor with second-order vss. *Int. Journal of System Science*, 29(1):11–12, 1998.

[3] Riccardo Marino, Sergei Peresada, and Paolo Valigi. Adaptive input-output linearizing control of induction motors. *IEEE Trans. Automatic Control*, 38(2), Feb. 1993.

[4] Marc Bodson. A systematic approach to selecting flux references for torque maximization in induction motors. *IEEE Trans. Control Systems*, 3(4):388–397, Dec. 1995.

[5] Ronald L. Racicot. Limiting servo motor torque gradients with near minimum time repositioning. *IEEE Trans. Control Systems*, 1(11):284–289, Aug. 1993.

[6] V.D. Yurkevich. Control of uncertain systems: dynamic compaction (contraction) method. *Proc. of the 9-th Int. Conf on Systems Engineering*, pages 636–640, 1993.

[7] Marian J. Błachuta, Valery D. Yurkevich, and Konrad Wojciechowski. Aircraft motion control by dynamic contraction method. *ICRAM'95*.

[8] Robert Jan Gorter. *Grey-box identification of induction machines.* PhD thesis, Technische Universiteit Eindhoven, 1997.

[9] P. C. Krause and C.H. Thomas. Simulation of symetrical induction machinery. *IEEE Trans. Power Apparatus Syst.*, PAS-84(11):1038–1053, 1965.

[10] Felix Blaschke. Das Prinzip der Feldorientierung, die Grundlage für die TRANSVEKTOR-Regelung von Drehfeldmaschinen. *Siemens-Zeitschrift*, (45):757–760, 1971.

[11] Marc Bodson, John Chaisson, and Robert Novotnak. Nonlinear servo control of an induction motor with saturation. *Proc. of the 33rd conference on decesion and Control*, Dec. 1994.

[12] E. P. Ryan. *Optimal relay and saturating control system synthesis.* The institution of electrical engineers, 1982.

[13] W. Leonard. *Control of Electrical Drives.* Springer-Verlag Berlin, Heidelberg, second edition, 1990.

[14] Marc Bodson, John N. Chaisson, and Robert T. Novotnak. A systematic approach to selecting flux references for torque maximization in induction motors. *IEEE Trans. Control Systems Technology*, 3(4), Dec. 1995.

[15] Robert T. Novotnak Marc Bodson, John Chaisson. Nonlinear speed observer for high-performance induction motor control. *IEEE Trans. Industry Applications*, 42(4):337–343, Aug. 1995.

Appendix A

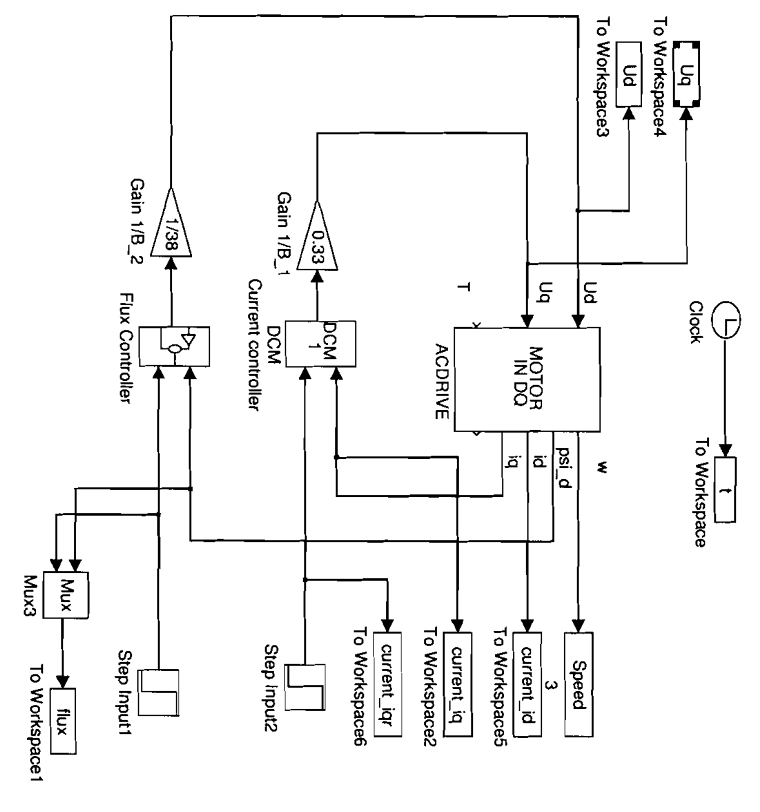
Tips for using Simulink$^{\mathrm{TM}}$

- Make blocks and test them separately
- Re-use blocks, and put them in a library. This way old and new version of files do not mix up
- Use a separate directory for the library and m block files.
- Make a constant file.
- Make controller parameters accessible within the simulation program.
- Be very careful with using the Adams/Gear method. Spikes due to discontinuities are filtered out with this method !!
- Save important simulations in mat files. Fast changes can be made to graphs this way.

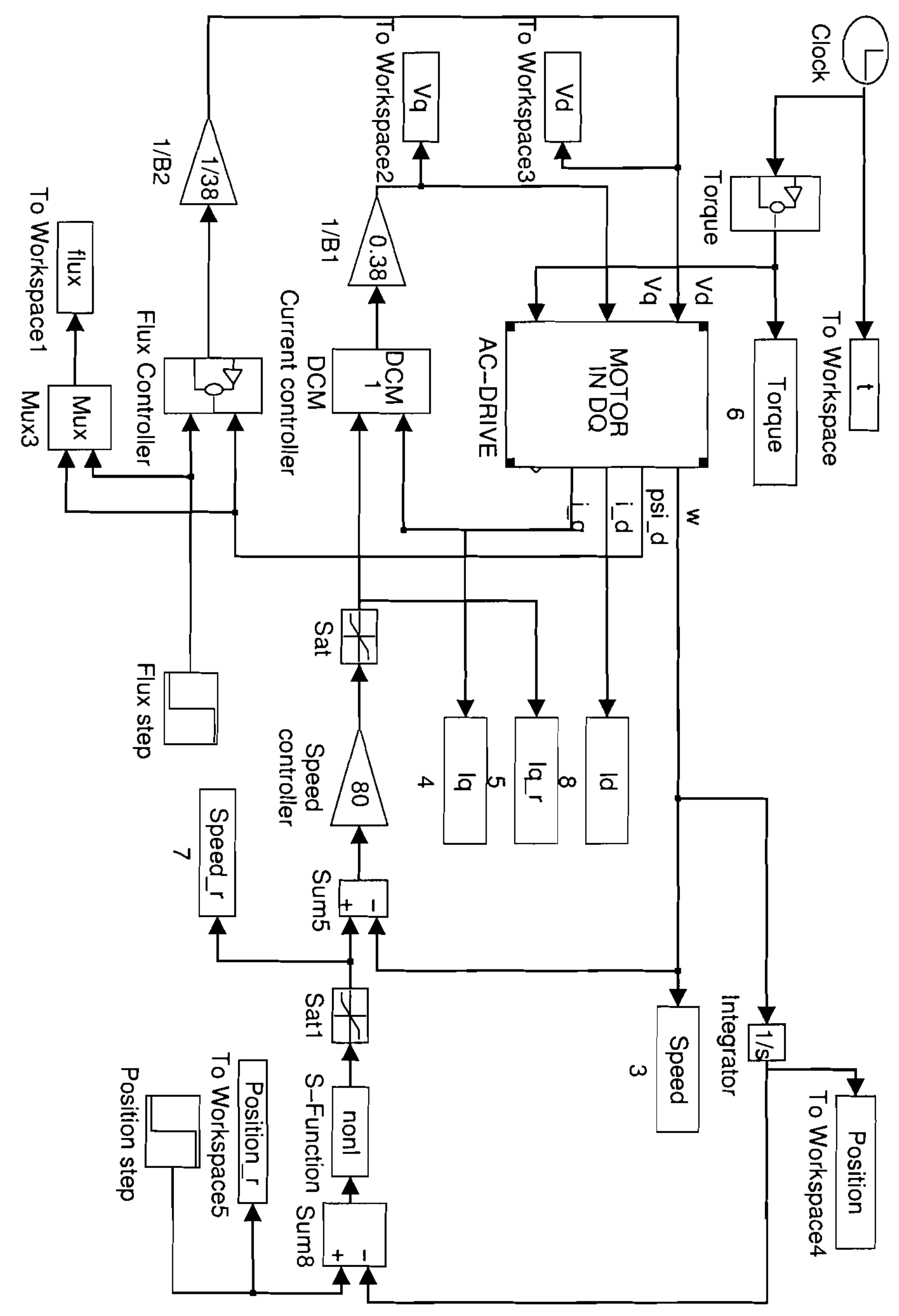
Appendix B

Simulation structures in Simulink$^{\mathrm{TM}}$

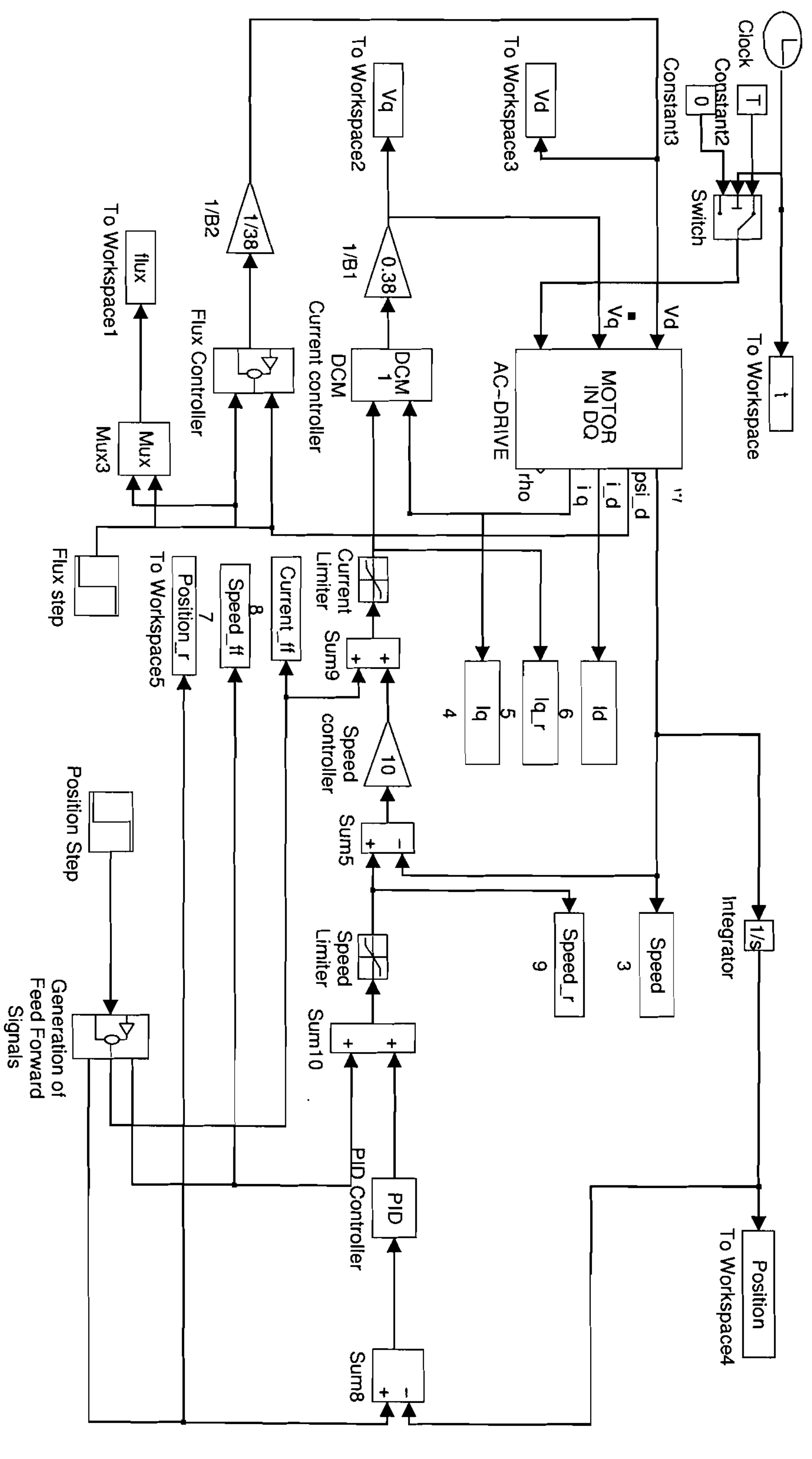
B.1 Flux and current structure



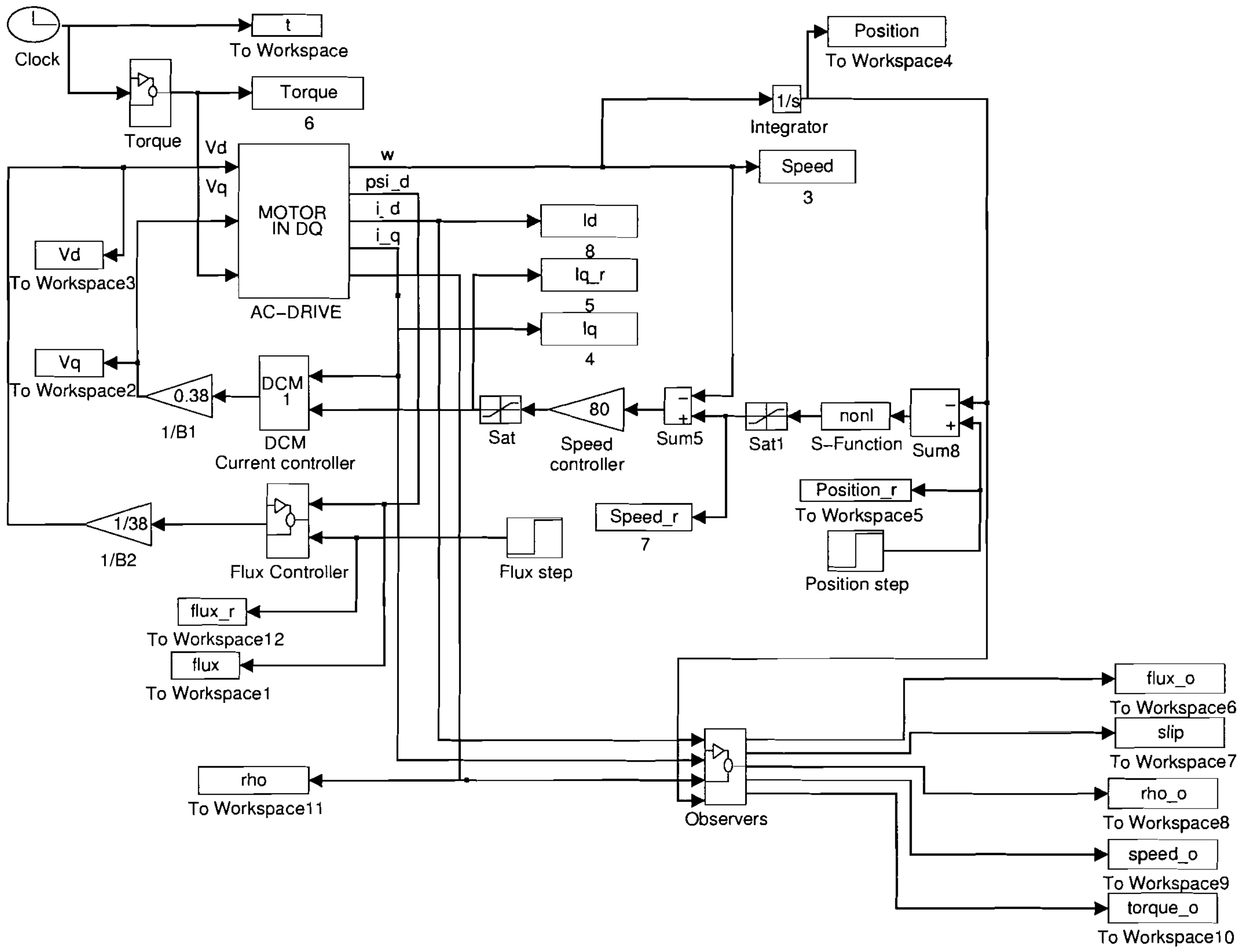
B.2 Non feedforward structure



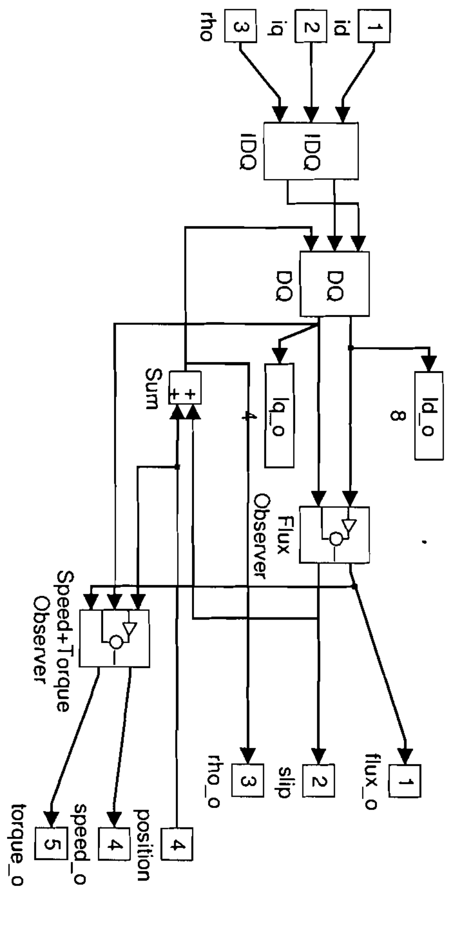
B.3 Feedforward structure



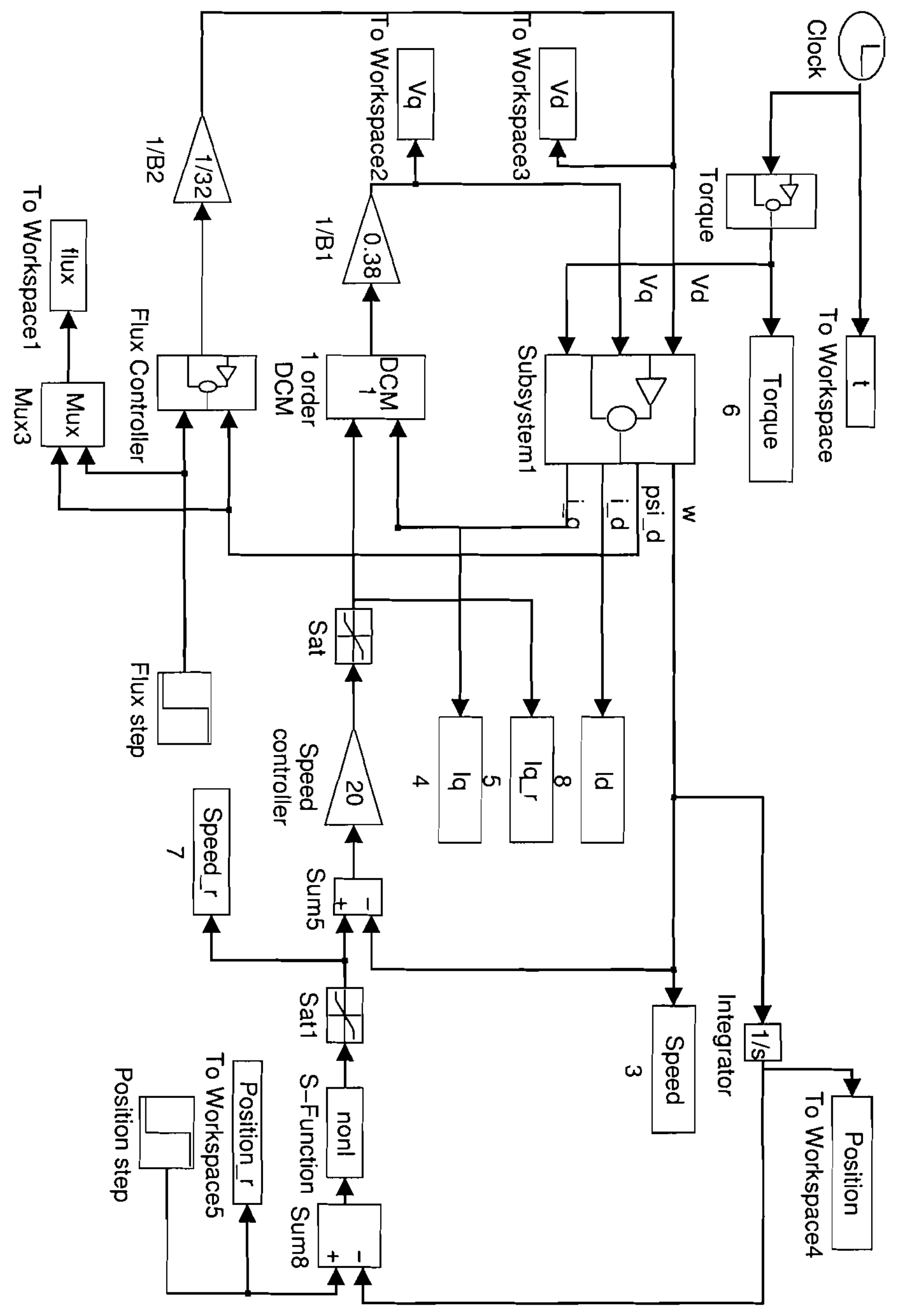
B.4 Non feedforward structure with observers



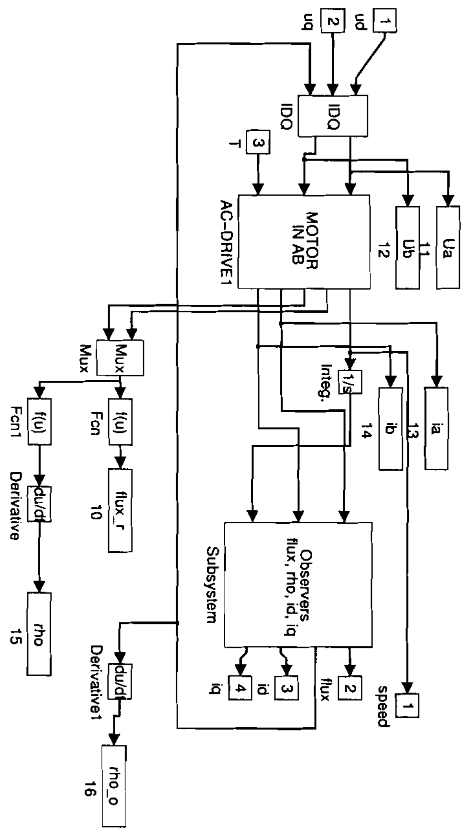
B.4.1 Observer block



B.5 Non feedforward structure in $a - b$ coordinate system



B.5.1 $a - b$ motor block



Appendix C

General implementation of a DCM controller in Simulink$^{\text{TM}}$

$$V(s) = \frac{b_0 + b_1 s^{-1} + \dots b_n s^{-n}}{1 + a_1 s^{-1} + \dots a_n s^{-n}} Y(s) + \frac{c_0 + c_1 s^{-1} + \dots c_n s^{-n}}{1 + a_1 s^{-1} + \dots a_n s^{-n}} R(s) \tag{C.1}$$

This equation can be written in the form:

$$\begin{aligned} V(s) = b_0 Y(s) + c_0 R(s) &+ s^{-1}(b_1 V(s) + c_1 R(s) - a_1 Y(s) \\ &+ s^{-1}(b_2 V(s) + c_2 R(s) - a_2 Y(s) \\ &+ s^{-1}(b_3 V(s) + c_3 R(s) - a_3 Y(s) \\ &+ \cdots))) \end{aligned} \tag{C.2}$$

The state variables can be defined as:

$$\begin{aligned} X_1(s) &= s^{-1}\left[b_1 Y(s) + c_1 R(s) - a_1 V(s) + X_2(s)\right] \\ X_2(s) &= s^{-1}\left[b_2 Y(s) + c_2 R(s) - a_2 V(s) + X_3(s)\right] \\ &\vdots \\ X_{n-1}(s) &= s^{-1}\left[b_{n-1} Y(s) + c_{n-1} - a_{n-1} V(s) + X_n(s)\right] \\ X_n(s) &= s^{-1}\left[b_n Y(s) + c_n R(s) - a_n V(s)\right] \end{aligned} \tag{C.3}$$

Equation (C.2) can be written in the form:

$$V(s) = b_0 Y(s) + c_0 R(s) + X_1(s) \tag{C.4}$$

After substitution of equation (C.4) in (C.2) we obtain:

$$\begin{aligned} sX_1(s) &= X_2(s) - a_1 X_1(s) + (b_1 - a_1 b_0) Y(s) + (c_1 - a_1 c_0) R(s) \\ sX_2(s) &= X_3(s) - a_2 X_1(s) + (b_2 - a_2 b_0) Y(s) + (c_2 - a_2 c_0) R(s) \\ &\vdots \\ sX_{n-1}(s) &= X_n(s) - a_{n-1} X_1(s) + (b_{n-1} - a_{n-1} b_0) Y(s) \\ &\quad + (c_{n-1} - a_{n-1} c_0) R(s) \\ sX_n(s) &= -a_n X_1(s) + (b_n - a_n b_0) Y(s) + (c_n - a_n c_0) R(s) \end{aligned} \tag{C.5}$$

Using the inverse s-transform on (C.5) gives:

$$\begin{aligned}
\dot{x}_1 &= -a_1 x_1(k) + x_2(k) + (b_1 - a_1 b_0) y(k) + (b_1 - a_1 b_0) r(k) \\
\dot{x}_2 &= -a_2 x_1(k) + x_3(k) + (b_2 - a_2 b_0) y(k) + (b_2 - a_2 b_0) r(k) \\
&\vdots \\
\dot{x}_{n-1} &= -a_{n-1} x_1(k) + x_n(k) + (b_{n-1} - a_{n-1} b_0) u(k) \\
&\quad + (c_{n-1} - a_{n-1} c_0) r(k) \\
\dot{x}_n &= -a_n x_1(k) + (b_n - a_n b_0) y(k) + (c_n - a_n c_0) y(k)
\end{aligned} \tag{C.6}$$

Writing this in matrix form gives:

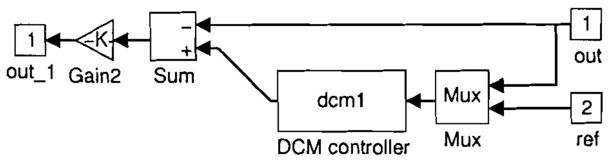
$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \vdots \\ \dot{x}_{n-1} \\ \dot{x}_n \end{bmatrix} = \begin{bmatrix} -a_1 & 1 & 0 & \dots & 0 & 0 \\ -a_2 & 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ -a_{n-1} & 0 & 0 & \dots & 0 & 1 \\ -a_n & 0 & 0 & \dots & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \\ \vdots \\ x_{n-1}(k) \\ x_n(k) \end{bmatrix} + \begin{bmatrix} b_1 - a_1 b_0 \\ b_2 - a_2 b_0 \\ \vdots \\ b_{n-1} - a_{n-1} b_0 \\ b_n - a_n b_0 \end{bmatrix} y(k) + \begin{bmatrix} c_1 - a_1 c_0 \\ c_2 - a_2 c_0 \\ \vdots \\ c_{n-1} - a_{n-1} c_0 \\ c_n - a_n c_0 \end{bmatrix} r(k) \tag{C.7}$$

$$y(k) = \begin{bmatrix} 1 & 0 & \dots & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \\ \vdots \\ x_{n-1}(k) \\ x_n(k) \end{bmatrix} + b_0 y(k) + c_0 r(k) \tag{C.8}$$

Appendix D

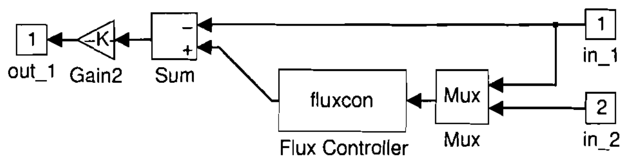
Simulink$^{\mathrm{TM}}$ files

D.1 DCM current controller



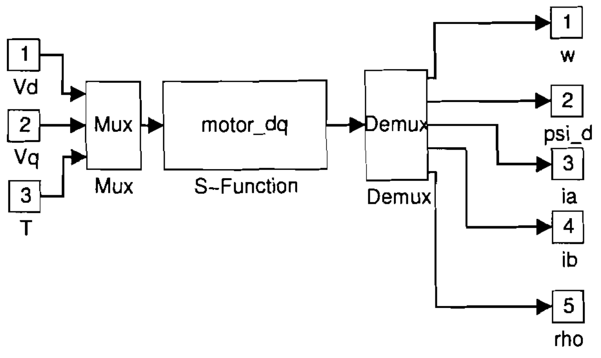
```
% First order dcm controller controller
% Modified 31/9/98
function[sys,x0]=dcm1(t,x,u,flag,d0,tau)
% u(1)=output of system
% u(2)=reference
if nargin==0, sys=[1,0,1,2,0,0];        x0=[0;0]; return, end
if abs(flag)==1
sys=-d0+(u(2)-u(1))/tau+d0*u(1);
elseif flag==3
sys=[x];
elseif flag==0
sys=[1,0,1,2,0,0];
x0=[0];
else
sys=[];
end
```

D.2 DCM flux controller



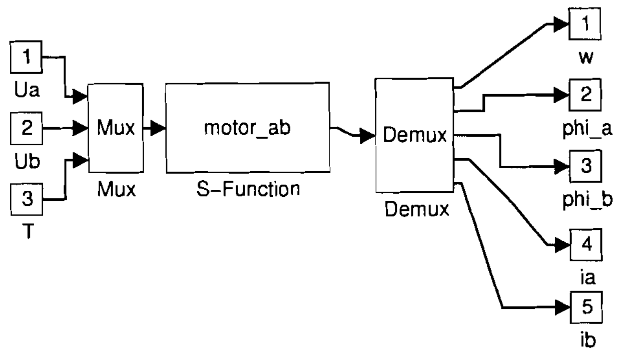
```
% Second order flux controller
function[sys,x0]=fluxcon(t,x,u,flag)
global kp_flux d0_flux d1_flux tau_flux mu_flux alfa_flux
% u(1)=output of system
% u(2)=reference
if nargin==0, sys=[2,0,1,2,0,0];        x0=[0;0]; return, end
if abs(flag)==1
% return derative of x
sys(1)=x(2)-2*(alfa_flux/tau_flux)*u(1)-(2*d1_flux/mu_flux)*(x(1)-u(1));
sys(2)=(1/(tau_flux)^2)*(u(2)-u(1))-(d0_flux/(mu_flux)^2)*(x(1)-u(1));
elseif flag==3
sys=[x(1)];
elseif flag==0
sys=[2,0,1,2,0,0];
x0=[0;0];
else
sys=[];
end
```

D.3 AC-motor in $d-q$ coordinates



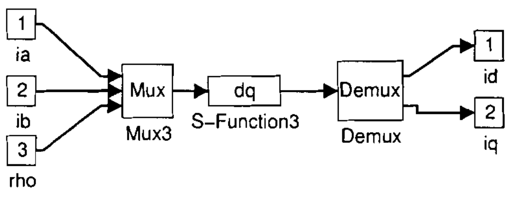
```
function [sys,x0] = motor_dq (t,x,u,flag)
% Model of an induction motor with field oriented control
global M L_r L_s R_r R_s J n_p alfa beta gamma sigma mu
% ********************** STATE VARIABLES ************************
% x(1)=w;
% x(2)=psi_d;
% x(3)=i_d;
% x(4)=i_q;
% x(5)=ro;
% ********************** INPUTS ******************************
% u(1)=u_d;
% u(2)=u_q;
% u(3)=T_0;
if nargin==0, sys=[5,0,4,3,0,0];x0=[0;1e-20;0;0;0]; return, end
if abs(flag) == 1
% Motor model after nonlinear state-feedback (=Field oriented control)
sys(1) = mu*x(2)*x(4)-(u(3)/J);
sys(2) = -alfa*x(2)+alfa*M*x(3);
sys(3) = -gamma*x(3)+alfa*beta*x(2)+n_p*x(1)*x(4)
         +alfa*M*x(4)^2/x(2)+u(1)/(sigma*L_s);
sys(4) = -gamma*x(4)-beta*n_p*x(1)*x(2)-n_p*x(1)*x(3)
         -alfa*M*x(3)*x(4)/x(2)+u(2)/(sigma*L_s);
sys(5) = n_p*x(1)+alfa*M*x(4)/x(2);
elseif flag == 3
sys = [x(1);x(2);x(3);x(4);x(5)];
elseif abs(flag) == 0
sys = [5,0,5,3,0,0];
```

D.4 AC-motor in $a - b$ coordinates



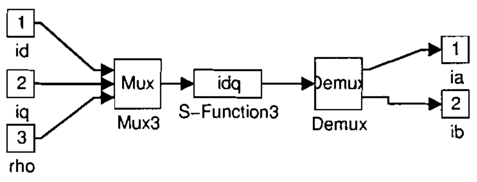
```
function [sys,x0] = motor_ab(t,x,u,flag)
% Model of an induction motor in a-b coordinates
global M L_r L_s R_r R_s J n_p alfa beta gamma sigma mu
% ********************** STATE VARIABLES ************************
% x(1)=w;
% x(2)=psi_ra;
% x(3)=psi_rb;
% x(4)=i_sa;
% x(5)=i_sb;
% x(6)= psi_d
% x(7)= rho
% ********************** INPUTS ******************************
% u(1)=u_sa;
% u(2)=u_sb;
% u(3)=T_0;
if nargin==0, sys=[5,0,5,3,0,0];x0=[0;1e-20;0;0;0]; return, end
if abs(flag) == 1
sys(1) =  mu*(x(2)*x(5)-x(3)*x(4))-u(3)/J;
sys(2) = -alfa*x(2)-n_p*x(1)*x(3)+alfa*M*x(4);
sys(3) = n_p*x(1)*x(2)-alfa*x(3)+alfa*M*x(5);
sys(4) = alfa*beta*x(2)+beta*n_p*x(1)*x(3)-gamma*x(4)+u(1)/(sigma*L_s);
sys(5)= -alfa*beta*n_p*x(1)*x(2)+alfa*beta*x(3)-gamma*x(5)+u(2)/(sigma*L_s);
elseif flag == 3
sys = [x(1);x(2);x(3);x(4);x(5)];
elseif abs(flag) == 0
sys = [5,0,5,3,0,0];
        % 1e-20 to have no numerical problems
```

D.5 DQ transformation



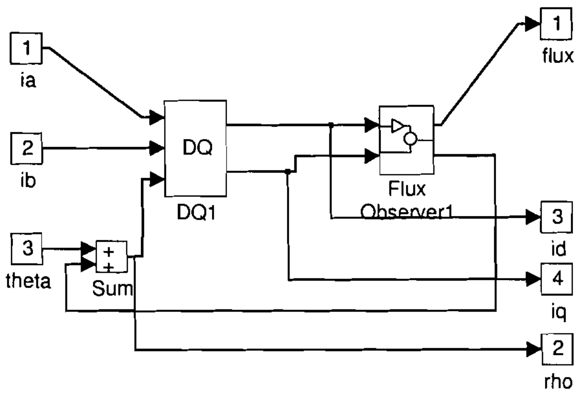
```
% DQ- Transformation
function[sys,x0]=dq(t,x,u,flag)
% transformation to DQ-coordinates
% u(1)=u_sa
% u(2)=u_sb
% u(3)=rho
if abs(flag) == 1
sys=0;
elseif flag == 3
A=[cos(u(3)) sin(u(3));-sin(u(3)) cos(u(3))];
sys=A*[u(1);u(2)];
elseif abs(flag) == 0
sys = [0,0,2,3,0,0];
x0 = [0];
else
sys =[];
end
```

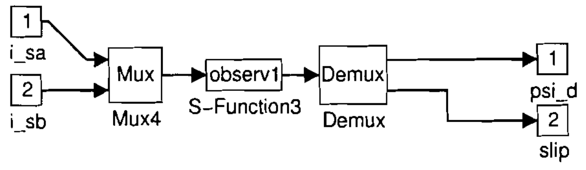
D.6 IDQ transformation



```
% IDQ- Transformation
function[sys,x0]=idq(t,x,u,flag)
% Transformation to a-b coordinates
% u(1)=u_sd
% u(2)=u_sq
% u(3)=rho
if abs(flag) == 1
sys=0;
elseif flag == 3
A=[cos(u(3)) -sin(u(3));sin(u(3)) cos(u(3))];
sys=A*[u(1);u(2)];
elseif abs(flag) == 0
sys = [0,0,2,3,0,0];
x0 = [0];
else
sys =[];
end
```

D.7 Flux and rho observer

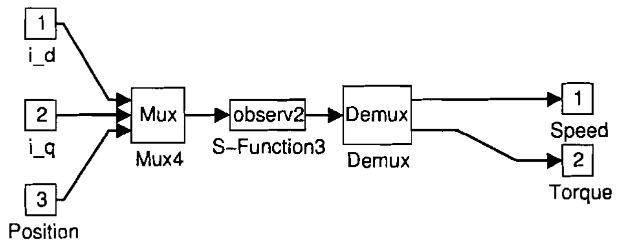




```
% Flux observer
function [sys,x0] = observ1(t,x,u,flag)
global M L_r L_s R_r R_s J n_p
global alfa beta gamma sigma mu
global l_1 l_2 l_3
% ******************** STATE VARIABLES ************************
% x(1)=psi_d;
% x(2)=Slip;
% ********************** INPUTS ******************************
% u(1)=i_sa;
% u(2)=i_sb;
if nargin==0, sys=[2,0,2,2,0,0];x0=[1e-10;0]; return, end
if abs(flag) == 1
```

```
sys(1) = -0.9*alfa*x(1)+0.9*alfa*M*( u(1));
sys(2) = 0.9*alfa*M*u(2)/x(1);
elseif flag == 3
sys = [x(1);x(2)];
elseif abs(flag) == 0
sys = [2,0,2,2,0,0];
x0 = [1e-10;0];
else
sys = [];
end
```

D.8 Torque/speed observer



```
% Speed and torque observer
function [sys,x0] = observ2(t,x,u,flag)
global M L_r L_s R_r R_s J n_p
global alfa beta gamma sigma mu
global l_1 l_2 l_3
% ******************** STATE VARIABLES ************************
% x(1)=theta;
% x(2)=w;
% x(3)=torque;
% ********************** INPUTS *******************************
% u(1)=x;
% u(2)=i_q;
% u(3)=psi_d;
if nargin==0, sys=[3,0,2,3,0,0];x0=[0;0;0]; return, end
if abs(flag) == 1
sys(1) = x(2)+l_1*(u(1)-x(1));
sys(2) = mu*u(2)*u(3)-x(3)/J+l_2*(u(1)-x(1));
sys(3) = (-l_3*(u(1)-x(1)))*J;
elseif flag == 3
sys = [x(2);x(3)];
elseif abs(flag) == 0
sys = [3,0,2,3,0,0];
x0 = [0;0;0];
else
sys = [];
end
```