Identification and control of the Rotary Inverted Pendulum

Identification and control of the Rotary Inverted Pendulum

M.B.Arnolds

DCT Report no: 2003.100
october 2003

TU/e Traineeship Report
october 2003

Supervisors:
Prof. I.Mareels
University of Melbourne, Australia
Prof. dr. H.Nijmeijer
Technische Universiteit Eindhoven

Eindhoven University of Technology
Department of Mechanical Engineering
Division Dynamical Systems Design
Dynamics Control Technology Group

In cooperation with:
University of Melbourne, Australia
Department of Electrical and Electronic Engineering
Cooperative Research Centre for Sensor Signal and Information Processing (CSSIP)

Abstract

Research on under-actuated systems becomes more and more popular nowadays. If it is possible to control an under-actuated system, it is possible to design a light construction (i.e. the amount of motors will reduce) which will lead to less inertia. A simple example of an under-actuated system is the Rotary Inverted Pendulum setup (or often called the *Furuta Pendulum*), which is a system with two degrees of freedom and only one actuator. In this report, the identification and the control of this system are discussed in order to be able to balance the pendulum in upright position.

Contents

Chapter 1

Introduction

The vertical positioning of the inverted pendulum requires a continuous correction mechanism to stay upright because this set-point is unstable in an open-loop configuration. This can be compared to a rocket launch. The rocket boosters have to be fired in a controlled matter in order to keep the rocket upright.

The goal of this project is to perform this balancing act with a rotary inverted pendulum setup. The description of the setup will be given in chapter 2. Before designing a balancing controller several steps have to be taken first. Deriving a mathematical model of the inverted pendulum setup is the first step in the design process (chapter 2). It is important to create such a model for the system to be able to describe the response of the actual system as closely as possible. Lagrange's equations of motion will be used for this. The model gives the relationships among all the variables involved except for inherent model uncertainties like flexibilities and slack. With this model it is possible to estimate the unknown parameters of the system. Before the parameter estimation can be performed it is necessary to calibrate the encoders on the setup (chapter 3). For the estimation of the parameters, an Extended Kalman Filter (EKF) will be designed and applied (chapter 4). At the end of the project, once the all parameters of the system are known, the design of a hybride controller will be discussed (chapter 5). This controller will consist of three parts. The combination of these three parts will make the setup able to swing-up and balance the pendulum in upright position which is the main goal of the project.

In between it is necessary to learn how to write a program in `C++`, because the *Kri Inverted Pendulum pp-300* setup that is located at the University of Melbourne has its own processor board which can only read `C++`. This learning process will not be discussed in this report.

The practical part of this traineeship and writing the main part of this report is performed at the University of Melbourne at the department of Electrical and Electronic Engineering, under the supervision of prof. I. Mareels. The finishing of the report is performed back in the Netherlands at the TU/e, under the supervision of prof. dr. H. Nijmeijer.

Chapter 2

The inverted pendulum model

2.1 Setup introduction

A schematic picture of the so-called Furuta pendulum setup is shown in Figure 2.1. The setup has two degrees of freedom: the rotation of the arm and the rotation of the

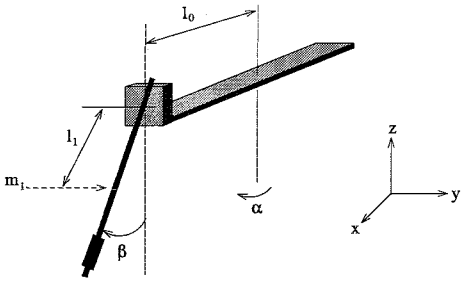

Figure 2.1: Schematic drawing of the setup

pendulum rod. It is a so called under-actuated system so only one of the rotations is forced where the other one has a free response to it. The rotation of the arm, α, is actuated by putting a torque on the motor. The position of α is measured with an optical encoder. The motion, β, of the pendulum rod is a free rotation depending on the actuated arm rotation. It is measured with a servo potentiometer. The calibration and resolution of the encoder and potentiometer will be discussed later on.

2.2 Derivation of the model

A model of the pendulum can be found in [Berg03]. After checking the model some errors were discovered so the model will be derived again in this report. The derivation starts with the Lagrange equations of motion. Because of the two degrees of freedom present in the system, as indicated in Figure 2.1, a set of generalized coordinates is:

$$\underline{q} = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}. \tag{2.1}$$

Here α is the position of the actuated rotating arm and β is the position of the unactuated pendulum rod. See Figure 2.1. The downward position of the pendulum is considered as $\beta = 0$.

To determine the kinetic energy of the system, the position of the considered mass m_1 (located at the center of mass of the pendulum rod, see section 4.2) must be expressed in the generalized coordinates mentioned in (2.1).

$$\underline{r}_{m1} = \begin{pmatrix} l_0 \cos\alpha - l_1 \sin\alpha \sin\beta \\ l_0 \sin\alpha + l_1 \cos\alpha \sin\beta \\ -l_1 \cos\beta \end{pmatrix}. \tag{2.2}$$

Here l_0 and l_1 represents the lengths as shown in Figure 2.1. The velocity of the mass m_1 is then:

$$\dot{\underline{r}}_{m1} = \begin{pmatrix} -\dot{\alpha} l_0 \sin\alpha - \dot{\alpha} l_1 \cos\alpha \sin\beta - \dot{\beta} l_1 \sin\alpha \cos\beta \\ \dot{\alpha} l_0 \cos\alpha - \dot{\alpha} l_1 \sin\alpha \sin\beta + \dot{\beta} l_1 \cos\alpha \cos\beta \\ \dot{\beta} l_1 \sin\beta \end{pmatrix}. \tag{2.3}$$

The total kinetic energy of the systems can be written as:

$$T = \frac{1}{2} \dot{\underline{r}}_{m1}^T m_1 \dot{\underline{r}}_{m1} + \frac{1}{2} J_{z0} \dot{\alpha}^2 + \frac{1}{2} J_{z1} \dot{\beta}^2 \tag{2.4}$$

J_{z0} and J_{z1} represents the moments of inertia for respectively the actuated arm and the pendulum rod. When (2.3) and (2.4) are combined the total kinetic energy become:

$$T = \frac{1}{2} m_1 \dot{\alpha}^2 l_0^2 + \frac{1}{2} m_1 \dot{\alpha}^2 l_1^2 \sin^2\beta + \frac{1}{2} m_1 l_1^2 \dot{\beta}^2 + m_1 \dot{\alpha} \dot{\beta} l_0 l_1 \cos\beta + \frac{1}{2} J_{z1} \dot{\beta}^2 + \frac{1}{2} J_{z0} \dot{\alpha}^2 \tag{2.5}$$

The potential energy of the system in terms of the generalized coordinates is given by:

$$V = -m_1 g l_1 cos\beta. \tag{2.6}$$

In order to bring viscous damping into account, virtual work has to be implemented. First take the work from the virtual displacement $\delta \underline{q}^T = [\delta\alpha, 0]$:

$$\delta W = -[C_0 \dot{\alpha}] \delta\alpha + \tau_m \delta\alpha = Q_\alpha \delta\alpha. \tag{2.7}$$

Next $\delta \underline{q}^T = [0, \delta\beta]$ will be applied:

$$\delta W = -[C_1 \dot{\beta}] \delta\beta = Q_\beta \delta\beta. \tag{2.8}$$

These equations lead to the nonconservative generalized forces

$$\underline{Q}^{nc} = \begin{bmatrix} -C_0 \dot{\alpha} + \tau_m \\ -C_1 \dot{\beta} \end{bmatrix}. \tag{2.9}$$

The Lagrange's equations of motion are then:

$$\frac{d}{dt} \frac{\partial L}{\partial \dot{q}} - \frac{\partial L}{\partial q} = Q^{nc}, \tag{2.10}$$

with $L = T - V$.
This can also be written as:

$$\frac{d}{dt}(T_{,\dot{\underline{q}}}) - T_{,\underline{q}} + V_{,\underline{q}} = (\underline{Q}^{nc}). \tag{2.11}$$

Resulting in the equations of motion of the system:

$$\begin{bmatrix} (J_{z0} + m_1 l_0^2 + m_1 l_1^2 \sin^2\beta)\ddot{\alpha} + m_1 l_1^2 \dot{\alpha}\dot{\beta} \sin\beta\cos\beta + m_1 l_0 l_1 \cos\beta\ddot{\beta} - m_1 l_0 l_1 \sin\beta\dot{\beta}^2 \\ (J_{z1} + m_1 l_1^2)\ddot{\beta} + m_1 l_0 l_1 \ddot{\alpha}\cos\beta - m_1 l_0 l_1 \dot{\alpha}\dot{\beta} sin\beta \end{bmatrix} - \begin{bmatrix} 0 \\ m_1 l_1^2 \dot{\alpha}^2 \sin\beta\cos\beta - m_1 l_0 l_1 \dot{\alpha}\dot{\beta}\sin\beta \end{bmatrix} + \begin{bmatrix} 0 \\ m_1 g l_1 \sin\beta \end{bmatrix} = \begin{bmatrix} -C_0\dot{\alpha} + \tau_m \\ -C_1\dot{\beta} \end{bmatrix} \tag{2.12}$$

Which in a more standard form can be written as:

$$\begin{bmatrix} J_{z0} + m_1 l_0^2 + m_1 l_1^2 \sin^2\beta & m_1 l_0 l_1 cos\beta \\ m_1 l_0 l_1 cos\beta & J_{z1} + m_1 l_1^2 \end{bmatrix} \begin{bmatrix} \ddot{\alpha} \\ \ddot{\beta} \end{bmatrix} + \begin{bmatrix} C_0 + m_1 l_1^2 \dot{\beta}\sin\beta\cos\beta & m_1 l_1^2 \dot{\alpha}\sin\beta\cos\beta - m_1 l_0 l_1 \dot{\beta}\sin\beta \\ -m_1 l_1^2 \dot{\alpha}\sin\beta\cos\beta & C_1 \end{bmatrix} \begin{bmatrix} \dot{\alpha} \\ \dot{\beta} \end{bmatrix} + \begin{bmatrix} 0 \\ m_1 g l_1 \sin\beta \end{bmatrix} = \begin{bmatrix} \tau_m \\ 0 \end{bmatrix} \tag{2.13}$$

Now the equation of motions are defined, the next step is to define τ_m since the input given to the setup is not a real torque. The input given is a real number (either positive or negative) with no units. The input will be called u. The scheme for the motor is shown in Figure 2.2. So it is necessary to determine the relationship between u and

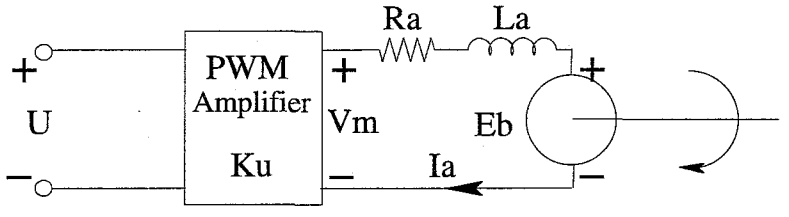

Figure 2.2: Electrical model of the motor

τ_m. The voltage V_m is a result from conversing and amplifying the input u with a factor K_u, which will be called the gain of the PWM (Pulse Width Modulation) amplifier.

$$V_m = K_u u. \tag{2.14}$$

From Kirchhoff's voltage law it is easy to see that:

$$V_m = I_a R_a + L_a \frac{dI_a}{dt} + E_b, \tag{2.15}$$

with:

I_a the armature current,
R_a armature coil resistance,
L_a armature coil inductance,
E_b motor's back EMF.

The motor's back EMF, E_b, is proportional to the rate of change of magnetic flux and hence proportional to the angular velocity of the motor:

$$E_b = K_b \dot{\alpha}. \tag{2.16}$$

For a constant field current, the torque τ exerted by the motor is proportional to the armature current. This leads to:

$$\tau_m = K_t I_a = \frac{K_t(V_m - E_b - L_a \frac{dI_a}{dt})}{R_a}. \tag{2.17}$$

When assuming that the coil inductance had a negligible influence, the torque exerted by the motor becomes:

$$\tau_m = \frac{K_t(V_m - E_b)}{R_a} = \frac{K_t K_u}{R_a} u - \frac{K_t K_b}{R_a} \dot{\alpha}. \tag{2.18}$$

Finally the dry friction that is present in the setup has to be implemented. The friction in the rotation of the pendulum rod (β) is considered to be totally modelled by the viscous damping term in the model. However, the dry friction present in the rotation of the actuated arm (α) is not yet modelled. This is a little bit more complicated because this is so-called Coulomb friction. Coulomb friction is difficult to model but often it is modelled as:

$$F_f = K_f \cdot sign(\dot{\alpha}). \tag{2.19}$$

This means that it is discontinuous if the velocity is equal to zero. See Figure 2.3.

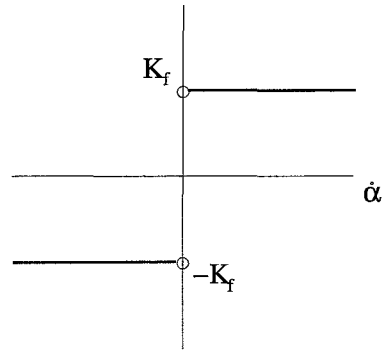

Figure 2.3: $K_f \cdot sign(\dot{\alpha})$

This discontinuity is certainly not desirable because of the use of certain integration methods later on. Because of this an approximation of the *sign*-function will be used to model the dry friction of the actuated arm. A proper approximation will be a *sigmoid* function like:

$$\sigma(\dot{\alpha}) = 1 - \frac{2}{e^{2k\dot{\alpha}} + 1} \tag{2.20}$$

When k is chosen large enough (i.e. $k = 15$) this function will provide a good approximation of the *sign*-function (see Figure 2.4).
The disadvantage of this approximation is that the *stick-phase* in the usually present *stick- and slip-phase* will not be modelled here.

So now the dry friction can be written as:

$$F_f = K_f \cdot \sigma(\dot{\alpha}) \tag{2.21}$$

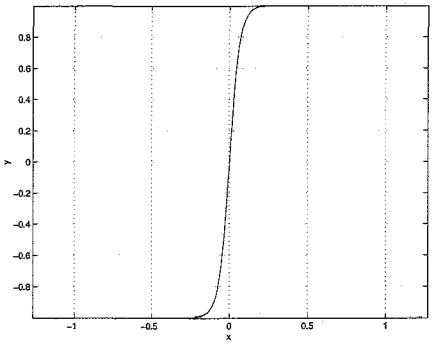

Figure 2.4: Sigmoid function (2.20) with k=15

The two types of friction can be combined so the total friction model for the arm rotation is obtained.

$$F_{total} = K_f \sigma(\dot{\alpha}) + C_0 \dot{\alpha} \tag{2.22}$$

This is graphical presented in Figure 2.5.

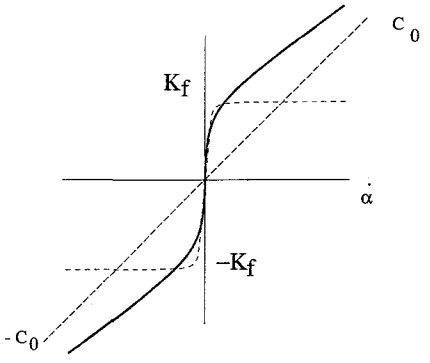

Figure 2.5: Combined friction (2.22)

After combining (2.13), (2.18) and (2.20), the total equations of motion for the inverted pendulum setup are obtained.

$$\begin{bmatrix} J_{z0} + m_1 l_0^2 + m_1 l_1^2 \sin^2\beta & m_1 l_0 l_1 cos\beta \\ m_1 l_0 l_1 cos\beta & J_{z1} + m_1 l_1^2 \end{bmatrix} \begin{bmatrix} \ddot{\alpha} \\ \ddot{\beta} \end{bmatrix} +$$
$$\begin{bmatrix} C_0 + \frac{K_t K_b}{R_a} + m_1 l_1^2 \dot{\beta} \sin\beta\cos\beta & m_1 l_1^2 \dot{\alpha} \sin\beta\cos\beta - m_1 l_0 l_1 \dot{\beta} \sin\beta \\ -m_1 l_1^2 \dot{\alpha} \sin\beta\cos\beta & C_1 \end{bmatrix} \begin{bmatrix} \dot{\alpha} \\ \dot{\beta} \end{bmatrix} +$$
$$\begin{bmatrix} K_f \cdot \sigma(\dot{\alpha}) \\ m_1 g l_1 \sin\beta \end{bmatrix} = \begin{bmatrix} \frac{K_t K_u}{R_a} \\ 0 \end{bmatrix} u \tag{2.23}$$

Chapter 3

Calibration

The angular measurement data recorded from the system, *raw data*, are defined in counts. These counts are a result of the output of the optical encoder and servo potentiometer (See Section 2.1). So to obtain the right input signals for the parameter estimation this raw data has to be converted to useful data, radians in stead of counts. To do so a calibration of both setups has to be made. This calibration is already done in [Berg03] but since one of the setups (setup 2) is replaced this will be done again, but will not be discussed so extensively. For this the same experiments will be used while these have proved to work properly.

3.1 Servo potentiometer

There will be 4 kind of experiments to determine the relation between the amount of counts and the real angle position (in degrees) of the servo potentiometer.

- From 0^o to 180^o in counterclockwise direction,
- From 0^o to 180^o in clockwise direction,
- From 0^o to 270^o in CC and back, 0^o to 90^o in C and back,
- From 0^o to 270^o in CC, 270^o to 180^o in CC, 180^o to 270^o in C.

Every movement within a program is repeated 5 times. The results are tabled in appendix A, tables A.1 to A.5.

After processing the data of the experiments one can see they are, for both setups, almost the same as already obtained in [Berg03]. There may be a difference in the measurements of 1 or 2 counts. The conclusions however will not be the same. For an extended calculation reference is made to [Berg03].

Experiment 3 for setup 1 shows that the counts for $\beta = 90^o$ and $\beta = 270^o$ are almost at the perfect 256 counts. This involves an equally distributed range over the total 360^o, meaning that each count stands for $90^o/256 = 0.351^o$. Experiments 1 and 2 both show a mean value of 507 counts in stead of the expected 512 counts. This difference of $5 \cdot 0.351^o = 1.7^o$ from the upright position is quite reasonable for naked eye positioning. Looking at experiment 4 it seems the potentiometer has a blind zone. To be more precise; a direction dependent blind zone.

As can be seen from Figure 3.1 the blind zone is direction dependent. Travelling in clockwise direction, the counts leave the normal count cycle at 717 (= 251.6^o) and

enters it at -306 (= 252.6°). This difference of 1 degree will be neglected. However, travelling in counterclockwise direction, the counts exit the normal count cycle at the expected -306 counts but do not enter again at the also expected 717 but at 640. This means a blind zone of 27° in CC direction. The cross-over point (point where the value jumps 1023 counts up or down) is located at $-107.4°$.

For setup 2, experiment 3 shows a mean value for $\beta = 90°$ and $\beta = 270°$ of about 275 counts. From experiments 1 and 2 the average amount of counts at 0° and 180° is 544 counts. This means that the distribution over 90° is $90°/274 = 0.33°/\text{count}$. As well as setup 1 setup 2 has a blind zone but this one is not direction dependent as can be seen from the results of experiment 4. This blind zone starts (looking at the clockwise direction) at $701 \cdot 0.33 = 231°$ and ends at $-322 \cdot 0.33 = -106° = 253°$. So a blind zone of 22° is present.

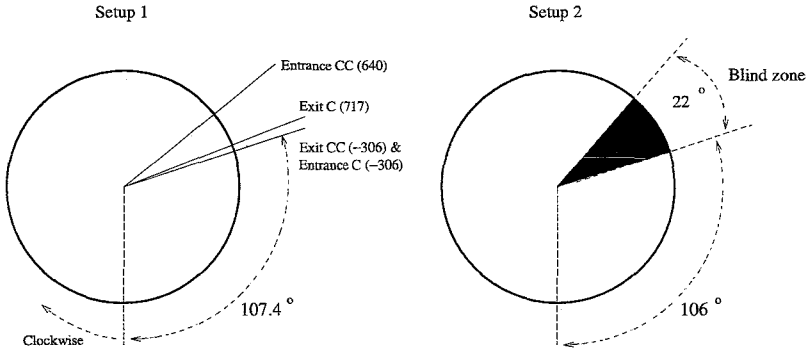

Figure 3.1: Blind zones of servo potentiometers. For setup 1: direction dependent. For setup 2: non-direction dependent. (mention that the drawn angels are a little bit exaggerated)

These blind zones in the measurements will not disturb the estimation of the parameters because the excitation of the pendulum can be limited to avoid entering this blind zones. Of course, during closed loop control it is not possible to avoid these blind zones and these have to be taken into account.

3.2 Optical encoder

For the calibration of the optical encoder the same method is used as with the servo potentiometer: manual movement of the arm while the encoder records the signals. An extended calibration is already done in [Berg03] so here the same experiments will be done just to compare the results.

As said, the arm will be moved manually. To avoid the reset boundary of the optical encoder, first the arm will make 4 rotations in clockwise direction continued with a movement of 8 rotations in counterclockwise direction and after that 4 rotations in clockwise direction again. With the 8 rotations the maximum number of rotations is reached. This experiment is done three times. Afterwards, the directions of movement will be inverted and the experiment will be done another three times.

The experimental results seem to be in agreement with the results of [Berg03]. That is for setup 1 an average of 52330 counts per 8 rotations which is 18.17 counts per degree.

Start Direction		Counts per 8 rotations	
		Setup 1	**Setup 2**
C	exp 1	52254	52512
C	exp 2	52242	52616
C	exp 3	52170	52566
CC	exp 1	52548	52502
CC	exp 2	52288	52542
CC	exp 3	52482	52294
Mean		**52330**	**52505**

Table 3.1: Results of calibration Optical Encoder

For setup 2 it is an average of 52505 counts per 8 rotations which is 18.23 counts per degree.

3.3 Motor dead-zone

Because of the static friction in the arm suspension and the motor the driver input u that is put on the motor has to reach a certain value before the motor will start to move. This will be called the dead-zone of the motor. These boundaries have to be known before doing experiments for the parameter estimation. Two kind of experiments have been performed to measure the transition from static to dynamic friction (the moment when the arm will start to move). The first type (A) will increase the input step by step (1 unit per step) every 2 seconds. The second one (B) will do the same but will jump back to zero between the two steps. This is done because this would happen in controlling the system. The experiments are performed in positive as well as negative direction. The average results are presented in Table 3.3. All the results are presented in Table A.6.

Determining the value for u for which the arm starts to move, a safety zone is taken into account, Figure 3.2. This safety zone is determined by looking at the motion of α. One can see the motion already starts at the beginning of the safety zone but to be sure the motor has totally overcome the static friction the value of u will be determined a short period later.

	Positive input		Negative input	
	Type A	Type B	Type A	Type B
Setup 1	45.4	45.6	-43.8	-44.6
Setup 2	28.4	36.4	-30.4	-38.4

Table 3.2: Average results of motor dead-zone experiment

The motor dead-zone is determined. So the effective value for u should be > 45 for setup 1 and > 40 for setup 2. The fact that the dead-zone for setup 1 is bigger as for setup 2 can be confirmed with the fact that it takes more power to rotate the arm of setup 1 manually then the arm of setup 2. This means there is more static friction present and so the dead-zone will be bigger. The difference between the two different

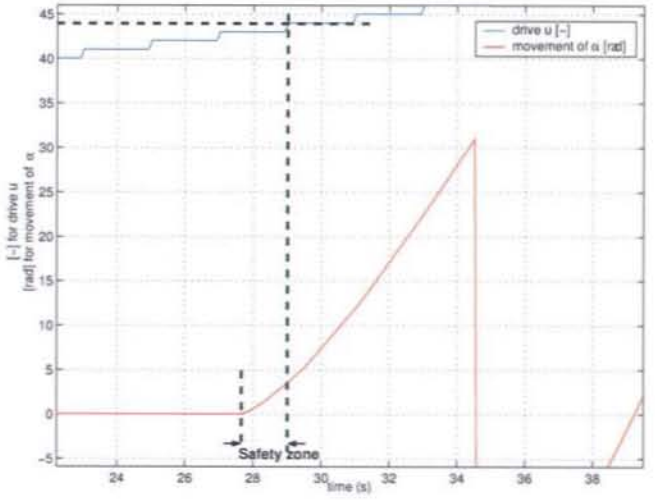

Figure 3.2: Method to determine the value u for the dead-zone

experiments is small. Controlling the system (balancing the pendulum in upright position) will be done by correcting the position of the arm with "shock wise" pulses. So, especially for setup 2, the difference that is present will be very useful.

The dead-zone is schematic presented as:

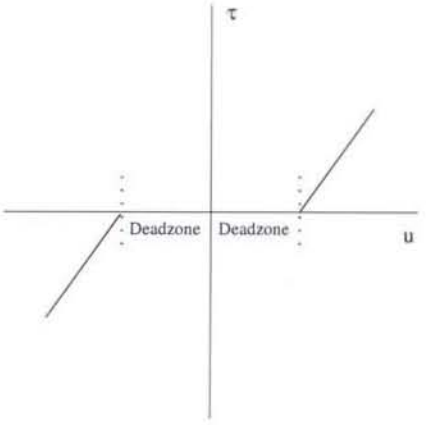

Figure 3.3: Schematic presentation of dead-zone of the motor

Chapter 4

Parameter Estimation

For the design of a good controller for the system all parameters which are involved in the system equation must be known. Since this isn't the case, the unknown parameters have to be derived or estimated.

The estimation of the parameters can be divided in three groups:

- parameters that can be measured directly,
- parameters that can be estimated with an experiment considering only the pendulum,
- parameters that can only be estimated with an experiment considering the total setup.

In table 4.1 the three groups and their parameters are presented:

Directly measured	m_1, l_0, l_1
Experiment involving only pendulum	J_{z1}, C_1
Experiment involving total setup	J_{z0}, C_0, K_t, K_b, K_u, R_a, K_f

Table 4.1: 3 groups and their parameters

It shows 12 parameters that have to be estimated. This can be reduced to 10 parameters since the 4 parameters from the motor only arise in the model in combination with each other ($\frac{K_t K_b}{R_a}$ and $\frac{K_t K_u}{R_a}$).

4.1 Estimators

An optimal estimation algorithm is an algorithm which processes measurements to derive an estimation (with minimum error) of a parameter by using the knowledge of the system and the measurements, assumptions about statistics of system noise and measurement errors and information of initial conditions. The advantage of such an estimation algorithm is that it uses all the previous measurement data and the knowledge of the system.

One can distinguish 3 different types of estimators:

- filtering
- smoothing
- prediction

Filtering is when the moment of estimation is the same as the moment of the last measurement. *Smoothing* is when the moment of estimation is somewhere in the time span of the measurement and it is *prediction* if the moment of estimation will be after the moment of the last measurement. For the next part of this report *filtering* will be used because on every moment an estimation will be made, a measurement is known.

The Least Square Method (LSQM) could be used. This is a discrete time method. It is a numerical optimization routine which selects the best combination of parameters in such a way that it minimizes the fit error. The fit error is defined to be a scalar cost function so that fit errors at all data points are taken into account in determining the best value for the parameters.

$$J = min \sum_{i} (y_{data} - y_{model})^2, \qquad i = datapoints \tag{4.1}$$

This way, the best fitted parameters to the system equations will be obtained.

Another filter is a Kalman filter. The advantage of a Kalman filter is that it uses all the previous knowledge and measurement data to update its estimation. Because the problem is a non-linear problem, an Extended Kalman Filter (EKF) will be designed. The equations for a continuous EKF are [Gelb74]:

$$\dot{\underline{\hat{x}}} = \underline{f}(\underline{\hat{x}}(t), t) + K(\underline{z} - \underline{h}(\hat{x}, t)) \tag{4.2}$$

$$\dot{P} = F(\hat{x}(t), t)P(t) + P(t)F^T(\hat{x}(t), t) + Q(t) - P(t)H^T(\hat{x}(t), t)R^{-1}(t)H(\hat{x}(t), t)P(t) \tag{4.3}$$

$$K = P(t)H^T(\hat{x}(t), t)R^{-1}(t) \tag{4.4}$$

The vectors $\underline{f}(\underline{\hat{x}}(t), t)$, $\underline{h}(\hat{x}, t)$ and the matrices $F(\hat{x}(t), t)$, $H(\hat{x}(t), t)$, $P(t)$, $R(t)$ are defined in Appendix C. Also initial conditions for P (P_0) and $\underline{\hat{x}}$ ($\underline{\hat{x}}_0$) are necessary. These initial conditions are required for solving the Kalman equations (4.2 - 4.4).

Every moment a new value for $\dot{\underline{\hat{x}}}$ will be calculated by using the old estimated value of $\dot{\underline{\hat{x}}}$ and the measurement at that moment. After that a new $\dot{P}$ matrix can be calculated. By integration of $\dot{\underline{\hat{x}}}$ and $\dot{P}$ the values for $\underline{\hat{x}}$ and P are obtained which are used to calculate a new Kalman gain K. The Kalman gain takes care of updating the system. With the new value of K a new $\dot{\underline{\hat{x}}}$ can be calculated again.

The EKF is preferred to use above the Least Square Method to estimate the parameters of this system. The measurement will be in discrete time and the EKF is continuous time but this will not cause any problems since the use of the integration method `ODE45` in Matlab will overcome this difference.

4.2 Directly measured parameters

As stated before the lengths l_0, l_1 and the mass m_1 can be directly measured and calculated.

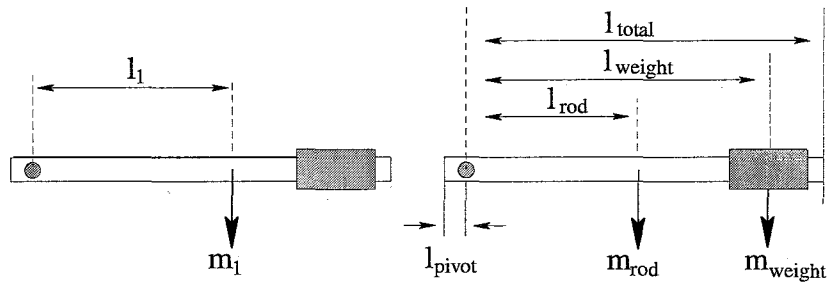

Figure 4.1: Pendulum rod parameters

Figure 4.1 shows the pendulum rod with its masses. With this information l_1 and m_1 can be obtained. Assuming that the mass is equally distributed along the pendulum rod,

$$m_{rod} = \frac{l_{total}}{l_{total} + l_{pivot}} m_{rod} = 20.48\ gr \tag{4.5}$$

so,

$$m_1 = m_{rod} + m_{weight} = 27.6\ gr \tag{4.6}$$

Using the law of moment the effective length l_1 can be obtained.

$$l_1 = \frac{1}{m_1}(m_{rod} l_{rod} + m_{weight} l_{weight}) = 154.7\ mm \tag{4.7}$$

The results:

Parameter	Result	Comment
l_0	154.7 mm	Measured directly
l_1	145 mm	Calculated
m_1	27.6 gram	Calculated

Table 4.2: Results of group 1

4.3 Parameters concerning the pendulum only

The first experiment considers only the pendulum rod. α and its derivatives will be stated zero and the free vibration of the pendulum is recorded. For this experiment the equation (2.23) reduces to:

$$(J_{z1} + m_1 l_1^2)\ddot{\beta} + C_1\dot{\beta} + m_1 g l_1 \sin(\beta) = 0 \quad ; y = \beta \tag{4.8}$$

The first term $(J_{z1} + m_1 l_1^2)$ is the moment of inertia concerning the rotating point of the pendulum constructed in relation to the normal moment of inertia with Steiner's law. Hence this term will be referred to as J_{r1}. Observe that it is essential to know the constant parameters m_1, g and l_1 because otherwise it is not possible to estimate the other parameters.

Since we want to estimate the parameters C_1 and J_1 a suitable augmented state-space vector will be:

$$\underline{x} = \begin{bmatrix} \beta \\ \dot{\beta} \\ C_1 \\ J_{r1} \end{bmatrix} \quad ; y = \beta.$$

4.3.1 Off-line simulations

As stated in Section 4.1 an EKF will be designed and used to estimate the parameters. Before estimating the parameters an off-line simulation will investigate if the EKF works properly. To achieve this a Matlab Simulink model is produced. In this model the system equation and the EKF are implemented. For this off-line simulations a certain input u has to be defined to give the system the required excitation in order to calculate a proper response. It is not really important what kind of input this is as long as the output of the system model will look like the real output that will be obtained during the real experiments. For the design of the entire EKF reference is made to appendix C.
In order to approximate the reality of the free vibration of the pendulum the input u is defined as shown in Figure 4.2:

In the simulations the parameters C_1 and J_{r1} are supposed to converge to a defined value. These values are:

$$\begin{bmatrix} C_1 \\ J_{r1} \end{bmatrix} = \begin{bmatrix} 0.7 \times 10^{-3} \\ 0.08 \times 10^{-3} \end{bmatrix}$$

In Figure 4.2 the results of the simulations are presented. The parameters converge to their defined values quite fast, within 2 seconds. Also the estimated β (β_e) follows the output (z) of the system very well. The difference between z and β_e is also presented in Figure 4.2.

To be certain that the EKF is finished estimating the parameters, the Kalman gains should be zero. This means that there is no update on the system and the final parameters are determined. The last four figures of Figure 4.2 present the Kalman gains. One can see they all are nicely converged to zero.

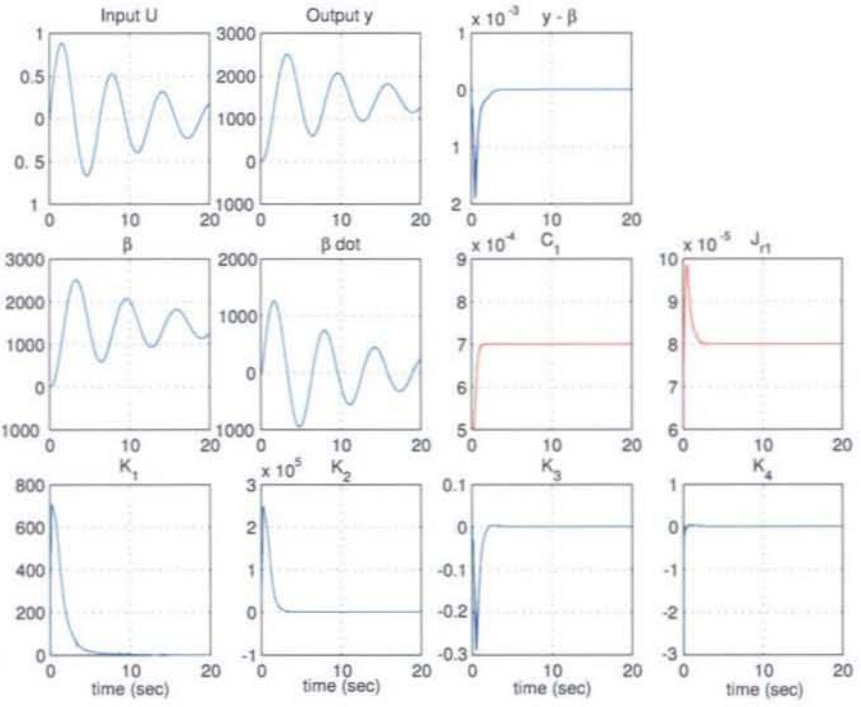

Figure 4.2: Results of the simulation for pendulum

4.3.2 Experiments

Now the Kalman filter is working properly some experiments can be done. In the simulation part an input u was set on the system to obtain a response. To obtain a free response during the experiments a very short and quick input will be set on the system so the pendulum will get an excitation. The first concern is to get familiar with programming the system. This turns out to be quite tricky. But after several experiments useful data is recorded successfully. With this data, the parameter estimation is performed off-line by loading it into the workspace of Matlab and use it as the input of the EKF.

The signal recorded for β is very smooth and almost free of noise. Because of this the data will not be filtered. This will save the data from losing magnitude.

For each setup 5 experiments were done. All the numerical results are shown in Table B.1 of Appendix B. The graphs in Figure 4.3 and Figure 4.4 summarize the results. It is clear that the Kalman filter has converged as well as the parameters. The error between the measured β (β_m) and the estimated β (β_e) is around 2 % for both setups, which is acceptable.

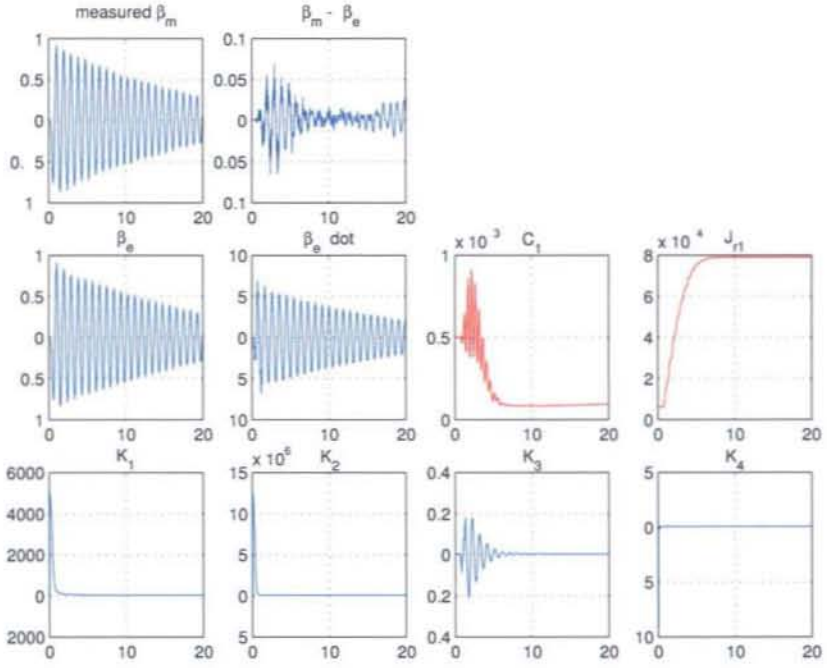

Figure 4.3: Results of parameters estimation C_1 and J_1 for Setup 1

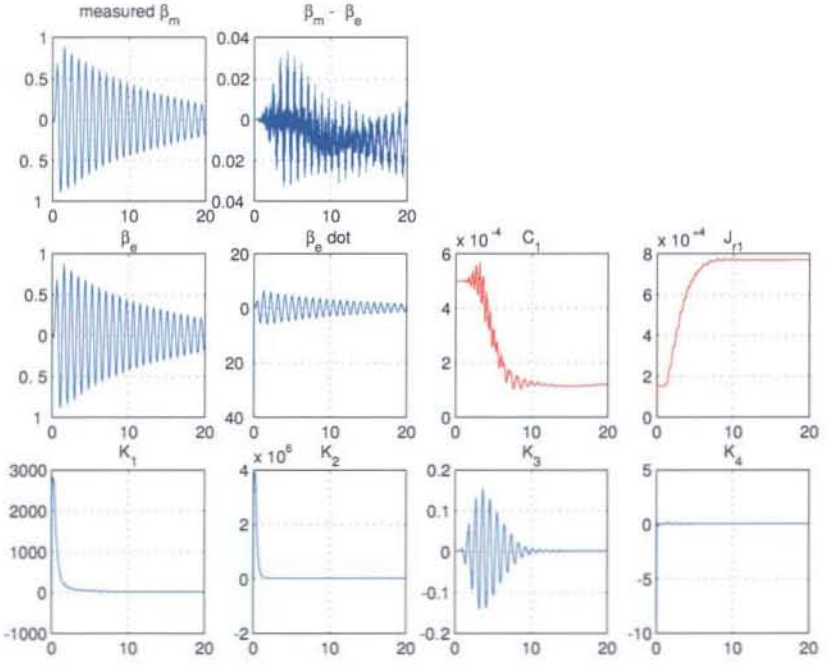

Figure 4.4: Results of parameters estimation C_1 and J_1 for Setup 2

After these experiments we successfully obtained the values for the parameters C_1 and J_{r1}.

Parameter	Setup 1	Setup 2
C_1 $[kgm^2/s]$	0.1×10^{-3} $\pm 0.013 \times 10^{-3}$	0.12×10^{-3} $\pm 0.0025 \times 10^{-3}$
J_{r1} $[kgm^2]$	0.79×10^{-3} $\pm 0.0009 \times 10^{-3}$	0.77×10^{-3} $\pm 0.0002 \times 10^{-3}$

Table 4.3: Results of parameters estimation Pendulum rod

4.4 Parameters concerning total setup

After the estimation of the parameters of the pendulum only, the parameters of the entire system have to be estimated. For this the total model equation has to be used. After rewriting equation (2.23) we obtain:

$$\begin{bmatrix} m_1 l_0^2 + m_1 l_1^2 \sin^2\beta & m_1 l_0 l_1 \cos\beta \\ m_1 l_0 l_1 \cos\beta & J_{r1} \end{bmatrix} \begin{bmatrix} \ddot{\alpha} \\ \ddot{\beta} \end{bmatrix} + \begin{bmatrix} m_1 l_1^2 \dot{\beta} \sin\beta\cos\beta & m_1 l_1^2 \dot{\alpha} \sin\beta\cos\beta - m_1 l_0 l_1 \dot{\beta}\sin\beta \\ -m_1 l_1^2 \dot{\alpha}\sin\beta\cos\beta & C_1 \end{bmatrix} \begin{bmatrix} \dot{\alpha} \\ \dot{\beta} \end{bmatrix} + \begin{bmatrix} 0 \\ m_1 g l_1 \sin\beta \end{bmatrix} = \begin{bmatrix} -J_{z0} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \ddot{\alpha} \\ \ddot{\beta} \end{bmatrix} - \begin{bmatrix} C_0 + \frac{K_t K_b}{R_a} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{\alpha} \\ \dot{\beta} \end{bmatrix} - \begin{bmatrix} K_f \sigma(\dot{\alpha}) \\ 0 \end{bmatrix} + \begin{bmatrix} \frac{K_t K_u}{R_a} \\ 0 \end{bmatrix} u \tag{4.9}$$

Only the first equation will be used as the system equation. But to simplify the model, $\ddot{\beta}$ will be extracted from the second equation and substituted in the first one. It is the same as taking the inverse of the first matrix and multiply all the other matrices with this inverse. β will be defined as an input of the system (with $\dot{\beta}$ as its time-derivative), so $\ddot{\alpha}$ only depends on $\dot{\alpha}$, β and $\dot{\beta}$.

From the second equation we obtain:

$$\ddot{\beta} = -\frac{m_1 l_0 l_1 \cos\beta}{J_{r1}}\ddot{\alpha} + \frac{m_1 l_1^2 \sin\beta\cos\beta}{J_{r1}}\dot{\alpha}^2 - \frac{C_1}{J_{r1}}\dot{\beta} - \frac{m_1 g l_1 \sin\beta}{J_{r1}}. \tag{4.10}$$

Substituting equation (4.10) into the first equation of (4.9) and extracting $\ddot{\alpha}$ the system equation for the entire system is obtained:

$$\ddot{\alpha} = f_1(t)\dot{\alpha} - f_2(t)\dot{\alpha}^2 - f_3\sigma(\dot{\alpha}) + f_4 + f_5 u, \tag{4.11}$$

with:

$$f_1 = \frac{\left(-(C_0 + \frac{K_t K_b}{R_a}) - 2m_1 l_1^2 \dot{\beta} \sin\beta \cos\beta\right)}{J_{z0} + m_1 l_0^2 + m_1 l_1^2 \sin^2\beta - \frac{m_1^2 l_0^2 l_1^2 \cos^2\beta}{J_{r1}}},$$

$$f_2 = \frac{\frac{m_1^2 l_0 l_1^3 \sin\beta \cos^2\beta}{J_{r1}}}{J_{z0} + m_1 l_0^2 + m_1 l_1^2 \sin^2\beta - \frac{m_1^2 l_0^2 l_1^2 \cos^2\beta}{J_{r1}}},$$

$$f_3 = \frac{K_f}{J_{z0} + m_1 l_0^2 + m_1 l_1^2 \sin^2\beta - \frac{m_1^2 l_0^2 l_1^2 \cos^2\beta}{J_{r1}}},$$

$$f_4 = \frac{\frac{C_1 m_1 l_0 l_1 \dot{\beta} \cos\beta}{J_{r1}} + \frac{m_1^2 g l_0 l_1^2 \sin\beta \cos\beta}{J_{r1}} + m_1 l_0 l_1 \dot{\beta}^2 \sin\beta}{J_{z0} + m_1 l_0^2 + m_1 l_1^2 \sin^2\beta - \frac{m_1^2 l_0^2 l_1^2 \cos^2\beta}{J_{r1}}},$$

$$f_5 = \frac{\frac{K_t K_u}{R_a}}{J_{z0} + m_1 l_0^2 + m_1 l_1^2 \sin^2\beta - \frac{m_1^2 l_0^2 l_1^2 \cos^2\beta}{J_{r1}}}.$$

So now a vector $\underline{x}$ can be specified for only 2 states and 4 parameters instead of 4 states and 4 parameters. Since it is impossible to distinguish the contribution of C_0 and $\frac{K_t K_b}{R_a}$ separately, these will be estimated together as one parameter.

$$\underline{x} = \begin{bmatrix} \alpha \\ \dot{\alpha} \\ \frac{K_t K_u}{R_a} \\ J_{z0} \\ C_0 + \frac{K_t K_b}{R_a} \\ K_f \end{bmatrix}$$

These equations are used for the design of the EKF (appendix D) and implemented in Simulink.

4.4.1 Off-line simulations

The approach will be the same as used for the estimation of the parameters for the pendulum rod only (section 4.3). This time an input u as well as β have to be defined. Again it does not really matter what kind of input this is, but since we will use a step-function (the reason for this will be a point of discussion in section 4.4.2) for α in the real model it will also be used in the simulation. For β a simple sine-response is taken. This may not be a really good approximation of the reality but for the simulations it will turn out to be good enough. In the Simulink model $\ddot{\alpha}$ is calculated and twice integrated to obtain α. To obtain the needed $\dot{\beta}$, the time-derivative of β is taken. This may cause some problems for the real experiments because of the noise but if a low-pass filter is used, these problems are probably solved.

The input u that is used for the simulation is shown in Figure 4.5 Again the parameters are supposed to converge to certain defined values.

$$\begin{bmatrix} \frac{K_t K_u}{R_a} \\ J_{z0} \\ C_0 + \frac{K_t K_b}{R_a} \\ K_f \end{bmatrix} = \begin{bmatrix} 3.5 \times 10^{-3} \\ 0.9 \times 10^{-3} \\ 8.8 \times 10^{-3} \\ 1.0 \times 10^{-3} \end{bmatrix}$$

The results of the simulations are presented in Figure 4.5. To check if the EKF has

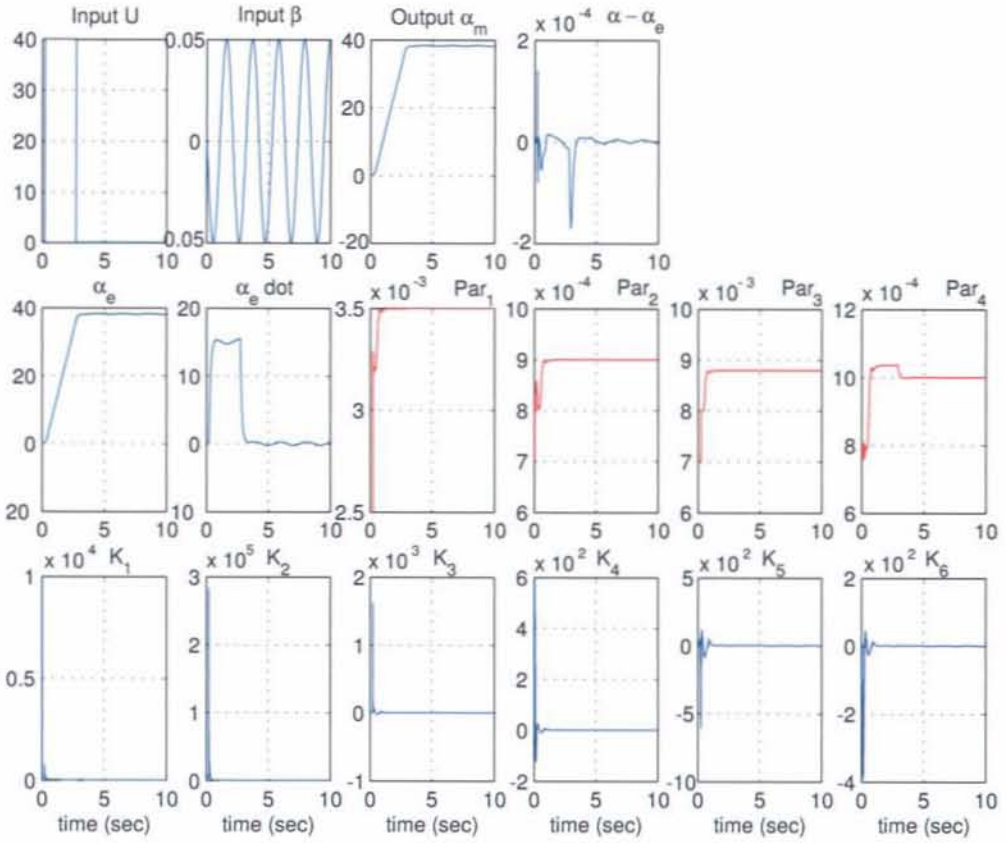

Figure 4.5: Results of the simulation for the entire system

finished to estimate the parameters the Kalman gains are examined. They are also presented in Figure 4.5. From this figure we can conclude that the filter works properly. Although there are some very small fluctuations around zero, it is reasonable to assume that no large updates will occur anymore and so the final parameters values are reached.

4.4.2 Experiments

In this section the experiments and the results of the experiments for the entire setup will be discussed. During the experiments an input u was set on the system and the *time*, u, α and β were recorded. Because of the limited data points that could be stored on the processor board it is only possible to record experiments for 10 seconds (4000 datapoints at 100 Hz = 40 sec.; 40 sec./4 = 10 seconds per signal).

First of all, because it is measured data there could be a lot noise on the signal. This could cause trouble if the time-derivative of β is calculated. Looking at the data it seems very smooth with hardly any noise on it. In spite of this, to be sure, it will be filtered using a first order low-pass filter. A low-pass filter is one which does not affect low frequencies and rejects high frequencies. The function giving the gain of a filter at every frequency is called the amplitude response (or magnitude frequency response). This amplitude response, for an ideal low-pass filter, is 1 for frequencies between 0 Hz

and the cut-off frequency, and it is 0 for all higher frequencies. The output spectrum is obtained by multiplying the input spectrum by the low-pass filter function.

The transfer function of the filter is:

$$H(z) = \frac{0.15}{1 - 0.85z^{-1}} \tag{4.12}$$

The magnitude response $|H(e^{j\omega})|$ of the filter is:

$$|H(e^{j\omega})| = \frac{0.15}{\sqrt{1 - 1.7cos\omega + 0.85^2}} \tag{4.13}$$

Hence the cut-off frequency can be found:

$$|H(e^{j\omega})|^2 = \frac{1}{\sqrt{2}} \tag{4.14}$$

$$\omega = 0.1048\ rad/sample \tag{4.15}$$

$$f = 0.01667\ cycle/sample \tag{4.16}$$

So this function has a cut-off frequency of 1.67 Hz at a sample frequency of 100 Hz. The Matlab function `filtfilt` is used for this filter. The data will first be filtered in forward direction. This will lead to a phase-disturbance. To avoid this it will pass the filter twice. The second time in negative direction so zero phase-disturbance is obtained. All the data recorded is filtered. A small disadvantage is the fact that the magnitude of β will decrease a little bit. This may lead to a small error in the parameter estimation. But this error can be overcome and will hardly influence the final values. Not using this filter has shown that the EKF was not able to work. This means the little noise that is present in the signal was too much for these experiments.

The first data obtained from the experiments, in which a sine-function of 10 Hz was put on the system, was not useful at all. The Kalman gains and the parameters did not converge totally. It was even so that end values were depending on the given initial values for the covariance error matrix P and the noise intensity matrix R, which in fact should not make any difference and only affect the update speed of the parameters.

Another input is tried; just a step function in positive as well as negative direction within between a little break. Still depending on the initial conditions, the EKF even estimated negative values for the friction this time, which is physical impossible.

This could be explained by the fact that the EKF method is based on linearisation. When a problem contains strong non-linear characteristics there may exist many local optima. So when an experiment is started with different initial values it could be possible that these several local optima causes different solutions.

The excitation of the pendulum is in all case larger than 1 rad, which is approximately 58 degrees. This is a considerable angle in which it is to be expected that non-linear effects will play a major role. We could avoid this by putting a short pulse (either positive or negative) which magnitude is just above the dead-zone of the motor. Initially, this pulse will hit all the dynamics of the system before the free dynamical response takes over. Now the β becomes smaller than 0.6 rad and the Kalman filter wins more time to converge to its right values. The results of these experiment are better but not good yet. Although all the parameters are positive right now and the EKF gains converged to zero so now there is no physical update on the estimation anymore, the parameters have not converged to a stable value and still depend on the initial values. The length of the pulses varied from 0.5 to 1.0 to 1.5 seconds. The results of the pulses

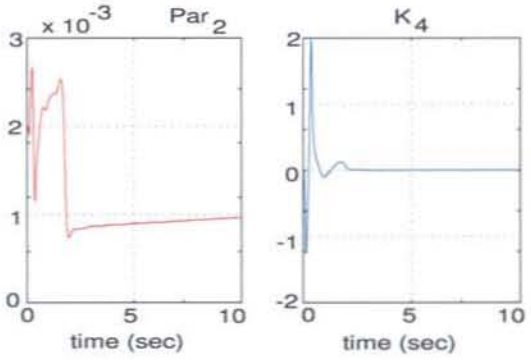

Figure 4.6: Example of a parameter that has not converged while the corresponding Kalman gain already did

of 1.0 and 1.5 seconds were far much better as the results of 0.5 seconds. This could be explained with the fact that 0.5 seconds is much too short to excite all the dynamics. For example, it would probably just reach the border of the Coulomb friction so the viscous friction will not even play part in the dynamics.
The last results were at least in the right direction. After observing the literature again it seems that the motor constants $\frac{K_t K_u}{R_a}$ and $\frac{K_t K_b}{R_a}$ can be obtained in another way as well. This way, only three parameters have to be estimated (in-stead of four) which gives the EKF more possibilities to estimate the dynamics of the system as well as the parameters in such a short time. The characteristics K_t, K_b and R_a are given in the specifications of the setup. Assuming they are the same for both the motors these are:

K_t [Nm/A]	0.0706
K_b [Vs/rad]	0.0707
R_a [ohm]	0.9

Table 4.4: Known motor characteristics

By measuring the voltage V_m that is send to the motor at a certain PWM driving command u the motor driver amplifiers gain K_u can be determined. The results of these experiments are shown in table B.2 of appendix B. Here, just the average values are given.

	Setup 1	Setup 2
K_u [V/count]	0.0618	0.0682

Table 4.5: Measured motor drive amplifier gain K_u

Now that the motor characteristics are known, new experiments are done but now with pulse lengths of only 1.0 and 1.5 seconds. Again the magnitude of u is set just above the dead-zone of the motor. 10 experiments are done per time span, 5 in clockwise direction (C) and 5 in counterclockwise direction (CC). The EKF seems to work far more better now, as shown in Figure 4.7.

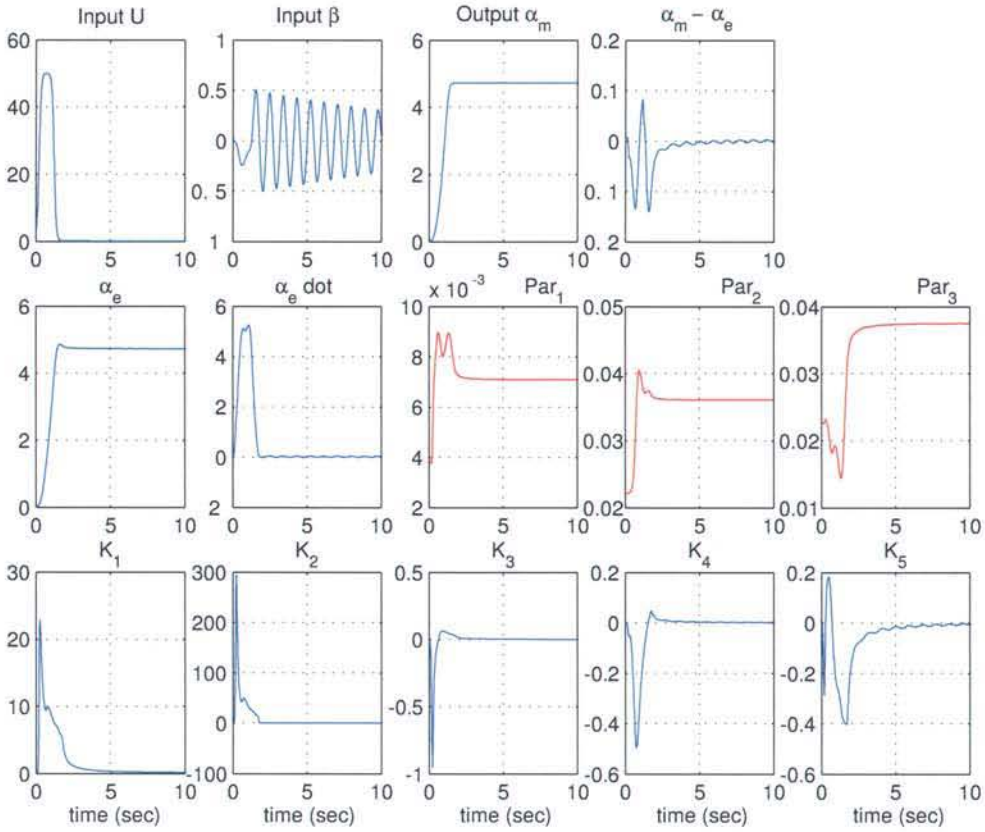

Figure 4.7: Results of the EKF after reducing the number of parameters from 4 to 3

The results seems to be quite reasonable. To check these results the Least Square Method is used. The advantage of this method is that, given the model equations and measurement data, it will give the most suitable solution there is to fit the parameters. This is applied by using:

$$x = (A^T A)^{-1} A^T y \tag{4.17}$$

Where:

$$A = \begin{bmatrix} \ddot{\alpha} \\ \dot{\alpha} \\ 1 - \frac{2}{e^{2k\dot{\alpha}}+1} \end{bmatrix},$$

and,

$$y = -m_1 l_0^2 \ddot{\alpha}_i - m_1 l_1^2 \ddot{\alpha}_i sin^2\beta_i - m_1 l_0 l_1 \ddot{\beta}_i cos\beta_i - 2m_1 l_1^2 \dot{\alpha}_i \dot{\beta}_1 sin\beta_i cos\beta_i + m_1 l_0 l_1 \dot{\beta}_i^2 sin\beta_i + \frac{K_t K_u}{R_a} u \tag{4.18}$$

with $i = datapoints$. The derivatives of α and β will be calculated using the central difference method.

Both the solutions for all the experiments are tabled in tables B.4 and B.6 of appendix B.

LSQM			EKF		
$J_{z0}[kgm^2]$	$C_0[Ns/m]$	$K_f[-]$	$J_{z0}[kgm^2]$	$C_0[Ns/m]$	$K_f[-]$
0.0035	0.0169	0.0351	0.0070	0.0268	0.0486
$\pm$0.00021	$\pm$0.0040	$\pm$0.0166	$\pm$0.00068	$\pm$0.00637	$\pm$0.012

Table 4.6: Results for LSQM and EKF for setup 1

LSQM			EKF		
$J_{z0}[kgm^2]$	$C_0[Ns/m]$	$K_f[-]$	$J_{z0}[kgm^2]$	$C_0[Ns/m]$	$K_f[-]$
0.0059	0.0140	0.0665	0.0052	0.0160	0.0569
$\pm$0.00092	$\pm$0.0106	$\pm$0.0143	$\pm$0.00237	$\pm$0.0165	$\pm$0.022

Table 4.7: Results for LSQM and EKF for setup 2

4.5 Summary

Looking at the results of the experiments in Table B.4 and B.6 one can say that they have reasonable and useful values. Looking at the values for C_0 and K_f we see a kind of connection. If the value of C_0 increases, the value of K_f will decrease and vice verse. This is probably caused because they both depend on $\dot{\alpha}$ in the relation mentioned before in section 2.2:

$$F_{total} = K_f \sigma(\dot{\alpha}) + C_0 \dot{\alpha} \tag{4.19}$$

So in order to estimate the right values for these parameters as well as the EKF as the LSQM have to deal with this problem.

But there is more. It seems that the friction in both directions is not the same. This is very clear with Setup 2, since here the difference for the viscous friction between clockwise experiments and counterclockwise experiments are rather big. For Setup 1 these are nearly the same. So now the friction present in the setups is not symmetric anymore and can be illustrated as shown in Figure 4.8. This phenomenon is hard to explain but will be very useful while designing a controller.

	$C_0[Ns/m]$	$K_f[-]$
Clockwise (C)	0.0161 $\pm$0.0036	0.0374 $\pm$0.0134
Counterclockwise (CC)	0.0176 $\pm$0.0046	0.0327 $\pm$0.0205

Table 4.8: Difference friction for C- and CC-direction Setup 1

	$C_0[Ns/m]$	$K_f[-]$
Clockwise (C)	0.0074 $\pm$0.0011	0.0695 $\pm$0.0133
Counterclockwise (CC)	0.0207 $\pm$0.00123	0.0634 $\pm$0.0175

Table 4.9: Difference friction for C- and CC-direction Setup 2

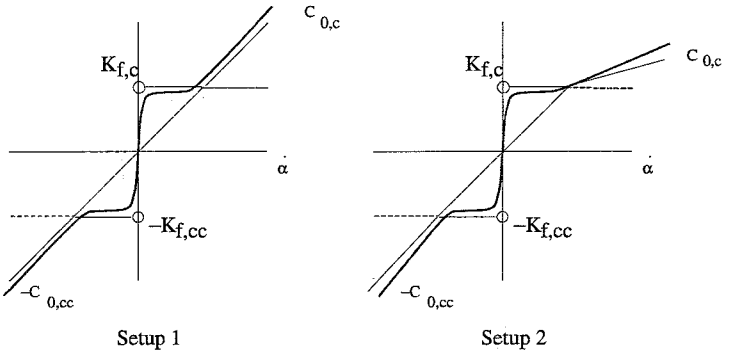

Figure 4.8: Different friction coefficients

This fact, possible small model errors and the time limit will be the reason the EKF is not working properly. It is difficult for the EKF to estimate the dynamics as well as the parameters within a short time where the parameters depend on the same state ($\dot{\alpha}$). It certainly works, and is more accurate than the LSQM, if the time length of the experiments is increased to a much more longer period (in the order of minutes). The Kalman filter will have time to distinguish the contribution of C_0 and K_f to the dynamics separately (and so, not depending on each other). For now the results presented above will be used to design a controller.

Chapter 5

Control of the pendulum

The goal of this project is to be able to balance the pendulum in upright position ($\beta = \pi$). After studying some literature it seems a model-based linear controller will be sufficient enough to control and balance the pendulum in upright position so no attention will be given to fuzzy controllers.

Since the pendulum is always in downward position when a balancing act is to be performed it is obvious necessary to get the pendulum in upright position first. This will be done with a *swing up controller*. This controller will swing up the pendulum to the upright position, before a *balancing controller* will take over to keep the pendulum there. In between a smart switching strategy has to be implemented so the balancing controller takes over at the right moment. Hence, the process of designing a controller is divided in 3 parts:

- Swing up controller
- Switching strategy
- Balancing controller

5.1 Swing up controller

The swing up of the pendulum can be realized in different ways. One of these is to swing up the pendulum by energy control. This was first done by K.Furuta and K.J.Åström. The entire theory behind this controller is already explained in [Åst00] and [Berg03] so this will not be mentioned again in this report. It is proved to work properly.

In this project Simulink is only used to perform simulations. In a dSPACE environment these files can be used to control the pendulum. Since no dSPACE environment is available for these setups, the programs have to be written in C++. To write the energy based controller in C++ is rather difficult and not really necessary. For this reason (and the reason that the objective of project is to balance the pendulum) just a simple swing up controller (which is still a kind of derivative of the energy based controller) is implemented.

With this controller the pendulum starts in zero position ($\beta = 0$). The arm will give the pendulum a small excitation. To give the pendulum a larger excitation, energy has to be put into the system. This energy is inserted by movements of the arm. But it must be well timed and in the right direction because otherwise it will work against the swing up and will extinguish the excitation instead of increasing it. So the

arm has to move and stop at the right moment. It is not so difficult to see that this has to be when the pendulum is within a certain defined swing-zone, Figure 5.1. It is possible to introduce and work with this swing-zone since the position of the pendulum is measured and therefore is known at every moment. The moment of putting energy

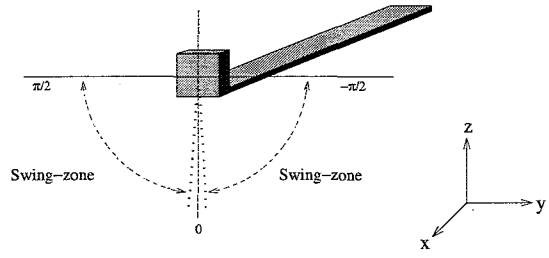

Figure 5.1: Swing-zone of the swing up controller

into the system has to be when the pendulum is within either of the swing-zones and the pendulum is moving in the direction of $\beta = 0$. At that moment the arm has to move in opposite direction as the pendulum is heading to, to give it more energy but has to stop again before the pendulum is just before $\beta = 0$. This is illustrated in Figure 5.2.

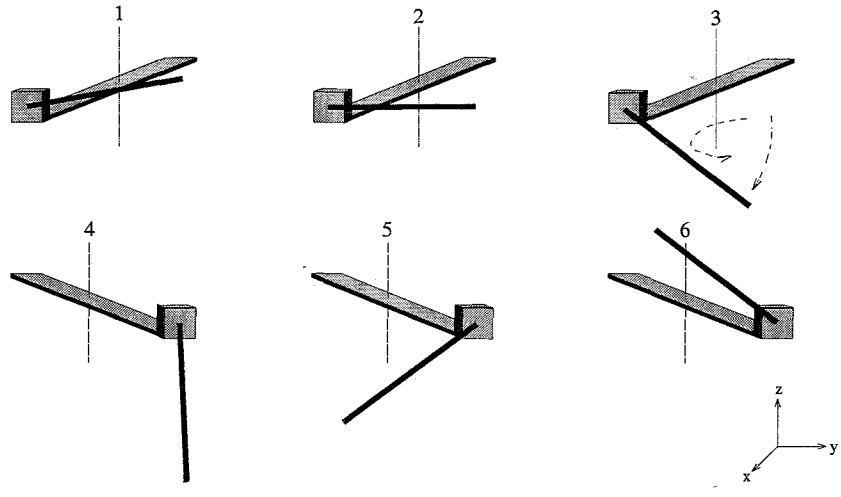

Figure 5.2: Example of swinging up the pendulum

Here the pendulum is at a certain position (1). The arm is not moving at that time. As soon as the pendulum reaches the beginning of the swing-zone (2) the arm starts to move and thus give extra impulse to the pendulum (3). Just before the pendulum reaches the zero position (4) the arm will stop moving. The pendulum swings up the other direction (5) and with the given extra impulse it will reach a higher position (6).

The drive that is put on the motor to move the arm is just a value obtained by trial and error. A drive of $u = 85$ is enough to swing up the pendulum in a reasonable number of swings ($\pm$ 8). Of course, it is obvious that smaller values will lead to more swings and so a longer swing up time and vice versa.

In Figure 5.3 the result of an experiment with the swing up controller for setup 1 is shown.

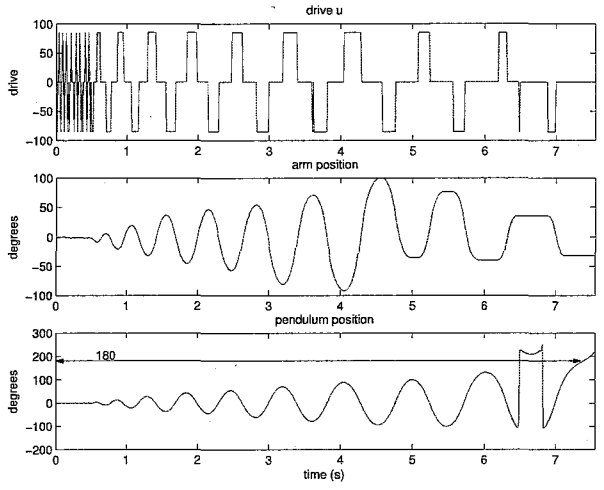

Figure 5.3: Result of an experiment with the swing up controller

From these results it can be concluded that the controller works. Because no balancing controller was there yet to take over, the pendulum will continue its way, passes the upright position and returns to 0^o. Later on, when a switching strategy and balancing controller are implemented this will not happen anymore. Notice two important aspects of this plot. One, the discontinuity that is present in the graph of the pendulum position is caused by the cross-over point. It is, as expected to be, located at -107^o. The peak present in the drive u is caused by this cross-over point but will not make any difference in swing-up. Second, the angle after 180^o continues increasing instead of decreasing. This is because the angle in the swing up controller is not normalized yet. In fact, this is not really necessary because the controller will work with the number of counts from the potentiometer instead of the angle.

5.2 Switching Strategy

Now that it is possible to get the pendulum close to upright position the balancing controller has to keep it there. To activate the balancing controller a switching strategy has to be implemented to switch from the swing up mode to balancing mode. So this strategy has to say when the balancing controller can take over form the swing up controller. It is almost impossible to switch when the pendulum is at exactly $\beta = \pi$ and since the balancing controller will have some range of attraction as well it is smart to choose a range of ± 10 degrees from the vertical position to switch over from one to another. So if the pendulum position is within 10 degrees ($\theta = 10^o$) from the vertical, the program will switch controller. In case the balancing controller "looses" the pendulum the swing up controller will be activated again. To prevent the controller from chattering when the pendulum position is close to 10 degrees from the vertical the switch-out point is placed lower than the switch-in point (i.e. $\gamma = 20^o$). See Figure 5.4.

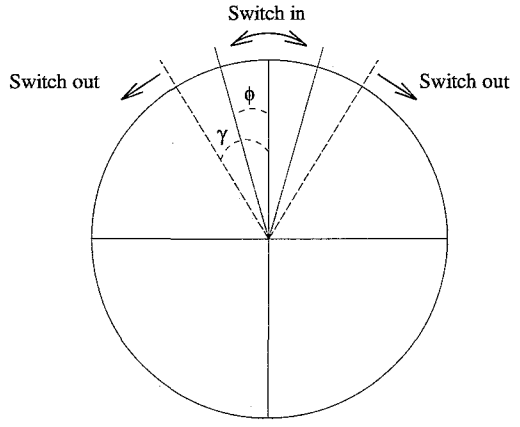

Figure 5.4: Switching areas for the switching strategy

5.3 Balancing controller

The balancing controller is the most important part of this chapter. The controller will be based on the linearized model (around the equilibrium) and is a state feedback controller. The disadvantage of a linearized controller is the limited region of attraction, but with the previous described switching strategy it should be good enough. The control law, $u = -Kx$, will be obtained using pole placement techniques.

5.3.1 Linearized model

The first step is to linearize the model around the unstable equilibrium $[\alpha\ \dot{\alpha}\ \beta\ \dot{\beta}]^T = [0\ 0\ \pi\ 0]^T$. This is done in appendix E and hence the linearized model is:

$$\begin{bmatrix} J_{z0}+m_1l_0^2 & -m_1l_0l_1 \\ -m_1l_0l_1 & J_{r1} \end{bmatrix}\begin{bmatrix} \ddot{\alpha} \\ \ddot{\beta} \end{bmatrix} + \begin{bmatrix} C_0+\frac{K_tK_b}{R_a} & 0 \\ 0 & C_1 \end{bmatrix}\begin{bmatrix} \dot{\alpha} \\ \dot{\beta} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & -m_1gl_1 \end{bmatrix}\begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \frac{K_tK_u}{R_a} \\ 0 \end{bmatrix} u \tag{5.1}$$

After rearranging and introducing some variables the state-space form is obtained:

$$\begin{bmatrix} \dot{\alpha} \\ \ddot{\alpha} \\ \dot{\beta} \\ \ddot{\beta} \end{bmatrix} = \frac{1}{ac-b^2}\begin{bmatrix} 0 & ac-b^2 & 0 & 0 \\ 0 & -cd & be & -bC_1 \\ 0 & 0 & 0 & ac-b^2 \\ 0 & -bd & ae & -aC_1 \end{bmatrix}\begin{bmatrix} \alpha \\ \dot{\alpha} \\ \beta \\ \dot{\beta} \end{bmatrix} + \frac{1}{ac-b^2}\begin{bmatrix} 0 \\ cf \\ 0 \\ bf \end{bmatrix} u \tag{5.2}$$

with:

$$a = J_{z0}+m_1l_0^2 \qquad b = m_1l_0l_1 \qquad c = J_{r1}$$
$$d = C_0+\frac{K_tK_b}{R_a} \qquad e = m_1gl_1 \qquad f = \frac{K_tK_u}{R_a}$$

and:

$$y = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \alpha \\ \dot{\alpha} \\ \beta \\ \dot{\beta} \end{bmatrix} \tag{5.3}$$

The matrix C is of the dimension $[2 \times 4]$ because both the arm and the pendulum position are measured. From now on, all the used and defined parameters will only concern setup 1. The design of the controller for setup 2 will be discussed later in this chapter. Substituting all known parameters into (5.2) the linearized state space model becomes:

$$\begin{bmatrix} \dot{\alpha} \\ \ddot{\alpha} \\ \dot{\beta} \\ \ddot{\beta} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -6.1 & 8.1 & -0.021 \\ 0 & 0 & 0 & 1 \\ 0 & -4.7 & 55.5 & -0.14 \end{bmatrix} \begin{bmatrix} \alpha \\ \dot{\alpha} \\ \beta \\ \dot{\beta} \end{bmatrix} + \begin{bmatrix} 0 \\ 1.32 \\ 0 \\ 1.01 \end{bmatrix} u \tag{5.4}$$

$\triangle A$ and $\triangle B$ are matrices that indicate how much deviation there is on these values. They are calculated with the values for the standard deviation on the parameters (chapter 4) using the chain and quotient rule.

$$\triangle A = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0.7 & 0.5 & 0.002 \\ 0 & 0 & 0 & 0 \\ 0 & 0.6 & 0.4 & 0.05 \end{bmatrix}, \quad \triangle B = \begin{bmatrix} 0 \\ 0.08 \\ 0 \\ 0.06 \end{bmatrix}$$

The deviation, as can be seen in $\triangle A$ and $\triangle B$, on the used system matrices is rather small. This means there will probably be no large influence from the deviation of the estimated parameters on the model.

5.3.2 Discrete time model

Since the setups work in discrete time a discrete state feedback controller will be designed. The first thing to do is to convert the above described continuous time model to a discrete time model. To do so, the Matlab function `c2dm` will be used. The sampling time and the 'method' have to be defined. The sampling time, $T_s = 0.01$ seconds, is equal to the sampling frequency. The method used is the so-called 'zero-order hold' (`zoh`).

$$x(k+1) = A_d x(k) + B_d u(k) \tag{5.5}$$
$$y(k) = C_d x(k) \tag{5.6}$$

$$\begin{bmatrix} \alpha(k+1) \\ \dot{\alpha}(k+1) \\ \beta(k+1) \\ \dot{\beta}(k+1) \end{bmatrix} = \begin{bmatrix} 1 & 0.01 & 0.0004 & 0 \\ 0 & 0.94 & 0.08 & 0.0002 \\ 0 & -0.0002 & 1.003 & 0.01 \\ 0 & -0.045 & 0.553 & 1.001 \end{bmatrix} \begin{bmatrix} \alpha(k) \\ \dot{\alpha}(k) \\ \beta(k) \\ \dot{\beta}(k) \end{bmatrix} + \begin{bmatrix} 0.0001 \\ 0.013 \\ 0 \\ 0.01 \end{bmatrix} u(k) \tag{5.7}$$

$\triangle A$ and $\triangle B$ will also be calculated for the discrete time model. Here the following equation can be written:

$$\begin{aligned} e^{(A+\triangle A)T_s} &= I + (A + \triangle A)T_s + h.o.t. \\ &= I + AT_s + \triangle AT_s + h.o.t. \end{aligned} \tag{5.8}$$

These higher order terms (h.o.t.) contain T_s^n with $n \geq 2$ which will be equal to 10^{-4} or smaller. Since the term $\triangle AT_s$ will be in the order of 10^{-3} these higher order terms can be neglected which will lead to the statement that $\triangle A_d = \triangle AT_s$

$$\triangle A_d = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0.008 & 0.005 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0.006 & 0.004 & 0.0005 \end{bmatrix}$$

For $\triangle B_d$ can be written:

$$\triangle B_d = \triangle ABT_s + A\triangle BT_s \tag{5.9}$$

This leads to:

$$\triangle B_d = \begin{bmatrix} 0.0008 \\ 0.005 \\ 0 \\ 0.005 \end{bmatrix}$$

Again for A the deviation is rather small. This is not the case for B but since the values are pretty small (order of h and h^2) it is reasonable to assume that also the discrete time model will be representative for the system.

5.3.3 Incremental discrete time model

In addition to the present system an extra state variable $u(k-1)$ will be added to the model resulting in an incremental model. This extra state variable implies that it is effectively performing an integral action over all the other state variables. This gives a positive effect on the control law. The integrator will work as a kind of low-pass filter so the control signal will be smoother and constant errors will be filtered.

So now the state vector becomes:

$$x_i(k) = \begin{bmatrix} x(k) \\ u(k-1) \end{bmatrix} = \begin{bmatrix} \alpha(k) \\ \dot{\alpha}(k) \\ \beta(k) \\ \dot{\beta}(k) \\ u(k-1) \end{bmatrix}$$

The model can be written as:

$$x_i(k+1) = A_{di}x_i(k) + B_{di}\triangle u(k) \tag{5.10}$$

$$y(k) = C_{di}x_i(k) \tag{5.11}$$

$$A_{di} = \begin{bmatrix} 1 & 0.01 & 0.0004 & 0 & 0.0001 \\ 0 & 0.94 & 0.08 & 0.0002 & 0.013 \\ 0 & -0.0002 & 1.003 & 0.01 & 0 \\ 0 & -0.045 & 0.553 & 1.001 & 0.01 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad B_{di} = \begin{bmatrix} 0.0001 \\ 0.013 \\ 0 \\ 0.01 \\ 1 \end{bmatrix}$$

To see if the obtained system is totally controllable and observable this will be checked with the controllability and observability matrices.

$$\vartheta = \begin{bmatrix} 0.0001 & 0.0003 & 0.0006 & 0.001 & 0.0015 \\ 0.013 & 0.025 & 0.036 & 0.047 & 0.057 \\ 0 & 0.0002 & 0.0004 & 0.001 & 0.001 \\ 0.01 & 0.02 & 0.03 & 0.036 & 0.04 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \tag{5.12}$$

$$\Xi = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0.01 & 0.0004 & 0 & 0.0001 \\ 0 & -0.0002 & 1.03 & 0.01 & 0 \\ 1 & 0.02 & 0.002 & 0 & 0.0003 \\ 0 & -0.001 & 1.01 & 0.02 & 0.0002 \\ 1 & 0.027 & 0.003 & 0 & 0.0006 \\ 0 & -0.002 & 1.025 & 0.03 & 0.0004 \\ 1 & 0.035 & 0.006 & 0.0001 & 0.001 \\ 0 & -0.004 & 1.04 & 0.04 & 0.001 \end{bmatrix} \tag{5.13}$$

Both these matrices are *full of rank* which proves the discrete time system is totally state controllable and state observable.

To check the stability of the open loop response of the system, which is most likely unstable, the poles of A_{di} are calculated. These are:

$$\begin{aligned} 1 &\leftarrow \alpha - integrator \\ 1.0746 &\leftarrow \beta - pole \\ 0.9520 &\leftarrow \alpha - friction \\ 0.9183 &\leftarrow \beta - pole \\ 1 &\leftarrow u - integrator \end{aligned}$$

Two of these poles are located at 1 and one at 1.0746. This confirms the intuition that the system is unstable in open-loop

5.3.4 Control law

The next step in designing a balancing controller is to find a control gains K. The first assumption to be made here is that all five states are measurable. However, during the experiments this is not the case and only α and β can be measured and $u(k-1)$ is known. To obtain the needed $\dot{\alpha}$ and $\dot{\beta}$ an observer could be implemented. This not easily done in C++ and with the knowledge that the signals obtained from

the measurements are pretty free of noise $\dot{\alpha}$ and $\dot{\beta}$ will be obtained using the "dirty derivative" during the experiments.

$$\dot{\alpha}(k) = \frac{\alpha(k) - \alpha(k-1)}{T_s} \tag{5.14}$$

$$\dot{\beta}(k) = \frac{\beta(k) - \beta(k-1)}{T_s} \tag{5.15}$$

The closed loop state feedback system in discrete time is shown in Figure 5.5.

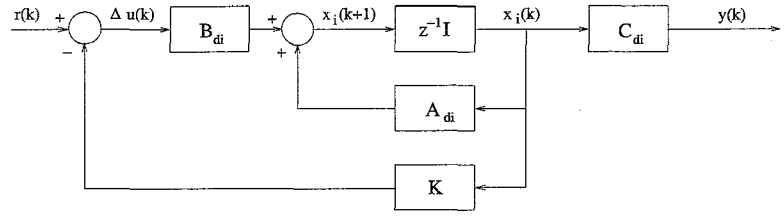

Figure 5.5: Schematic representation of the state feedback system

The controlled system becomes:

$$x_i(k+1) = (A_{di} - B_{di}K)x_i(k) + B_{di}r \tag{5.16}$$

with K the controller gains and r the desired state.

Looking at Figure 5.5 the control law can then be found by:

$$\begin{aligned}
\triangle u(k) &= r(k) - Kx_i(k) \\
u(k) &= u(k-1) + \triangle u(k) \\
&= u(k-1) + r(k) - Kx_i(k) \\
&= u(k-1) + r(k) - [K_1 \;\; K_2] \begin{bmatrix} x(k) \\ u(k-1) \end{bmatrix} \\
&= u(k-1) + r(k) - K_1 x(k) - K_2 u(k-1) \\
&= r(k) + (1 - K_2)u(k-1) - K_1 x(k)
\end{aligned} \tag{5.17}$$

Because initially the reference trajectory vanishes (i.e. $r(k) = 0$), the control law becomes:

$$u(k) = (1 - K_2)u(k-1) - K_1 x(k) \tag{5.18}$$

At first, pole placement techniques will be used to design a control law. There are several ways this can be done. For example the `place` or `acker` commands can be used. These methods allow you to place your (desired) closed-loop poles where you want to and calculate the matching control gains K. Another method is to use the Linear Quadratic Regulator (LQR) method. The basic theory behind this is that this method allows to find optimal controller gains. For the system $\dot{x} = Ax + Bu$ optimal controller gains is found in such a way that

$$J = \int_0^\infty (x^T Q x + u^T R u) dt \tag{5.19}$$

is minimized. Here, R is the *performance index matrix* and Q the *state-cost matrix*. The control law that minimizes J is given by linear state feedback $u = -Ku$. One can also say that the optimal value of K is that which places the closed-loop poles at a stable location.

To find suitable control gains, the parameters Q and R have to be specified. For simplicity R will be equal to $R = 1$. Q can be taken as $Q = C_{di}^T C_{di}$. The controller can be tuned by changing these matrices to get a desirable response.

Several controller gains (K) have been calculated, tested (with the help of a step function), and implemented into the setup but unfortunately none of them worked properly. Once swung up in upright position the controller is not able to keep it there for longer then 1 second caused by too large corrections on the actuated arm.

However, using pole placement techniques have helped to prepare a way in the right direction in order to find the gains of the control law which are able to balance the pendulum. A trial and error method has been used to find them. After a lot of experiments with "random" (but tactical) chosen values, the values of the gains to keep the pendulum in upright position have been found.

$$K = [-6 \quad -5 \quad 137 \quad 20 \quad 0.16]$$

The results are plotted in Figure 5.6.

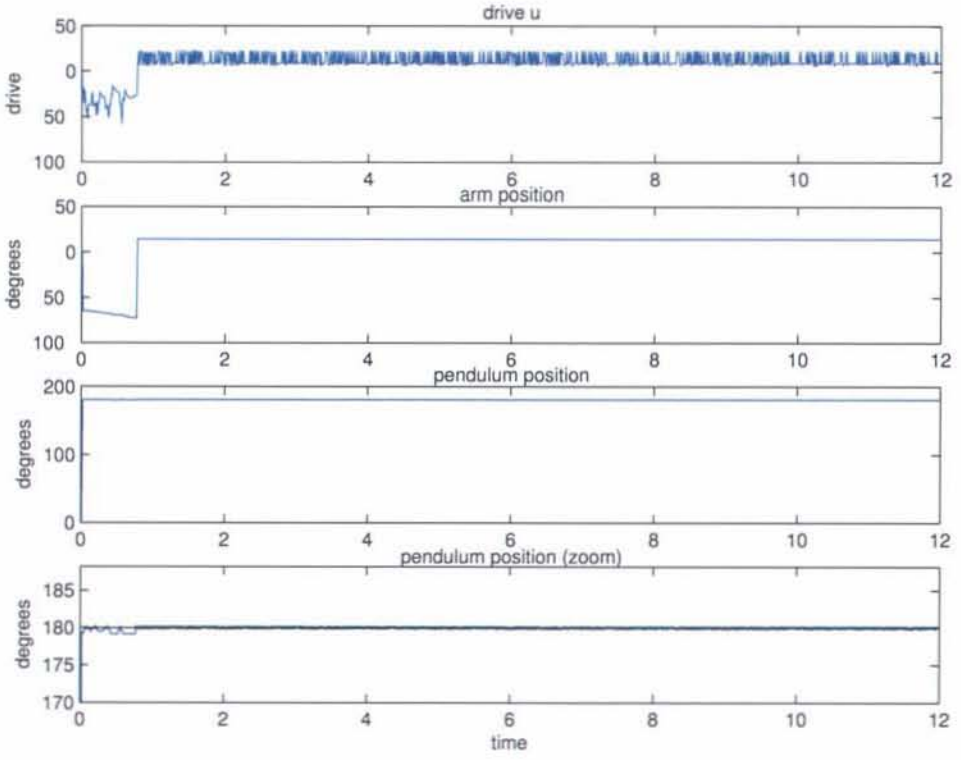

Figure 5.6: Results of experiment controller for setup 1: Perfect controlled

The plot starts at the moment the balancing controller took over from the swing up controller. The pendulum is immediately in upright position and stays there. Even after $\pm$ 1 second it is totally balanced and the controller does not have to correct the arm position (α). So the arm stands still. The fact that the drive u is showing excitation in the plot is caused by a sort of bias offset. It is within the dead zone of the motor so this won't have any effect on the arm position. Unfortunately this is the perfect case. Most of the times the results are as plotted in 5.7.

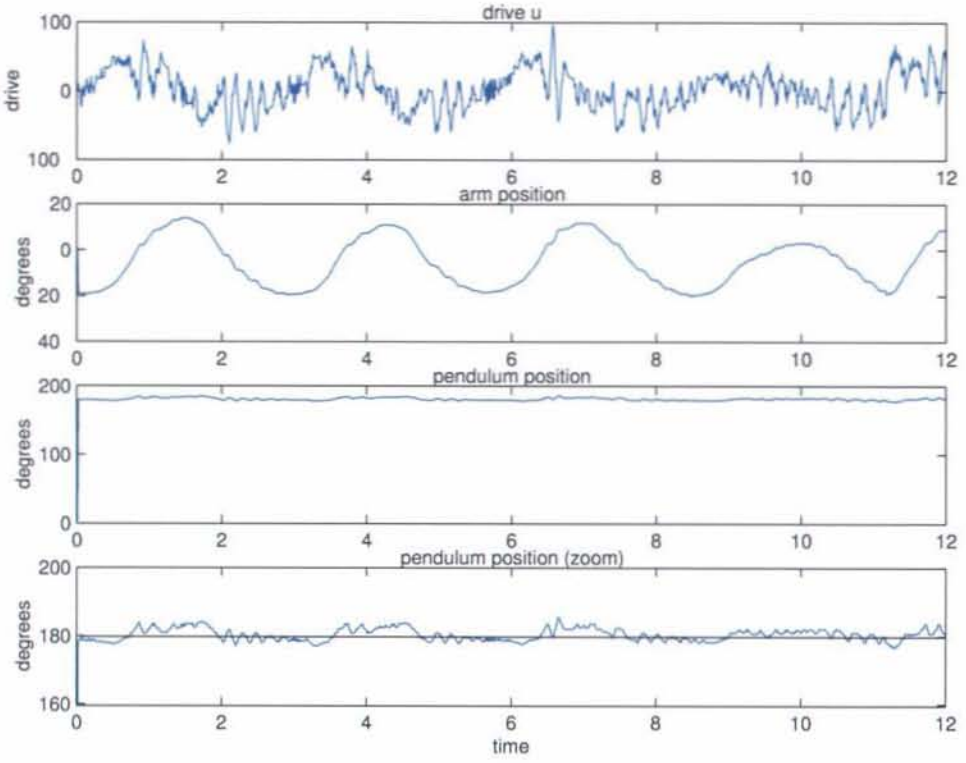

Figure 5.7: Results of experiment controller for setup 1: Good but less perfect controlled

The results are reasonable. The fact that the controller is not able to keep the pendulum in perfect upright position is caused by the fact that in the beginning of every experiment the zero and upright position are defined. This upright position will always be the same and is obtained during the calibration. But for the zero position this is different. Before starting an experiment the pendulum is at rest in downward position. This will be defined as the zero position. Due to some friction, this position doesn't have to be exactly at zero but for instance at -2 counts (virtual) but this is unknown. So now "-2 = 0" and the new upright position is "-2 + upright position" which is not exactly straight up but 0.7^o from the vertical. So now the controller thinks 179.3^o is upright. Trying to balance it around this position it will always keep on moving due to gravity. This is visualized is Figure 5.8.

This phenomena could be solved by letting the pendulum swing out after a "perfect" experiment. The end position is then known and the "new" upright position could be

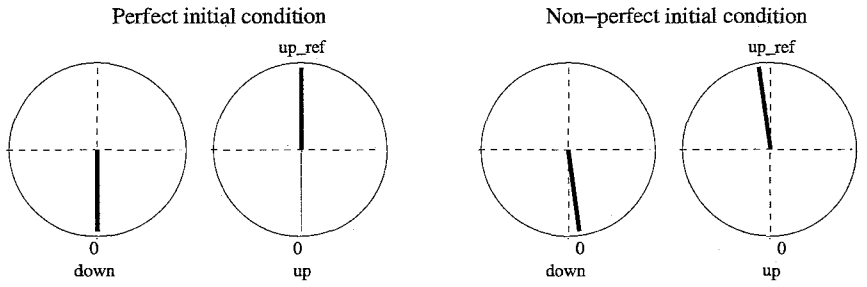

Figure 5.8: Perfect initial condition

calculated easily. Unfortunately this is not possible at this moment since the control program has to be loaded onto the setup every time you want to start an experiment and so the pendulum position will be reset every time.

Another thing that is remarkable is that the arm position moves around a certain point and tries to stay within a reasonable angle from this point. This is caused by the linearisation of the arm position around 0. Because of this the controller will always aspire to bring the arm position back to 0.

The values of the control gains that have been found by trial and error were implemented again in the model to see what the step response and the locations of the closed loop poles would be.

$$\begin{aligned} 0.8731 + 0.3352i \\ 0.8731 - 0.3352i \\ 0.9372 \\ 0.9868 + 0.0172i \\ 0.9868 - 0.0172i \end{aligned}$$

As we can see, the controller has to be smooth and robust. Perhaps, this can be explained by the fact that the parameters that were estimated are not good enough to describe the system.

Analyzing the stability radius of the continuous time system is an extra check wether the system is stable or not. The stability radius can be seen as the margin on ΔA and can be defined as the largest singular value of the system:

$$inv(max(svd(E_2(\omega I - A + BK)^{-1}E_1))) \tag{5.20}$$

Here E_1 and E_2 are the structure matrices and ω is the eigenfrequency of the system and will be investigated from 0 to 100 Hz. The stability radius has to be larger than the largest value in ΔA. Analyzing 2 different controllers (the first one is obtained using pole placement techniques, the second one by trial and error) the following radii were found:

K = [-12 -7 99 14 0.3] St.rad. = 0.81
K = [-6 -5 137 20 0.15] St.rad. = 1.35

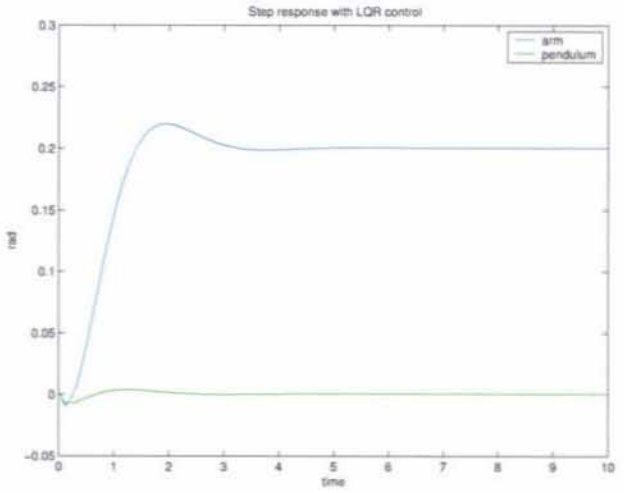

Figure 5.9: Simulation results of controller for setup 1 with K found by trial and error

The first one should be working fine. The step response on the system was reasonable and the stability radius is larger as the largest value in ΔA. An explanation for the fact it does not work could be that the margin on the stability radius is too small to be stable. Also the presence of the dead zone could cause a problem. The gains for α are rather large as well which makes the controller a very aggressive controller. The second combination of gains works very well. It is more smooth and robust. α has decreased a little bit and β is increased. The stability radius has a nice margin regarding ΔA.

Although there is difference between the "theoretical" controller and the controller which really performs the balancing act, this is not enormous. Not accurate enough estimated parameters can be a reason to make this difference.

5.3.5 Controller for setup 2

The procedure for setup 2 is similar to the one of setup 1, so this will not be discussed, except for the experiments.

Again, the LQR method is used to prepare a way in the right direction to find the controller gains. As expected, these theoretical values does not work so the controller is "tuned" using trial and error again. The obtained gain values for which the controller is able to balance the pendulum in upright position are:

$$K = [-5.5 \quad -4.5 \quad 140 \quad 18 \quad 0.17]$$

The results are shown in Figure 5.10.

The controller behaves pretty well. Even after $\pm$ 9.5 seconds it manages to get and keep the pendulum in perfect upright position for a moment. After that it looses this position again. But overall, the pendulum is well controlled and it never exceeds an angle more than $\pm 1.5^o$ from the vertical where the arm position is kept within a 10^o movement.

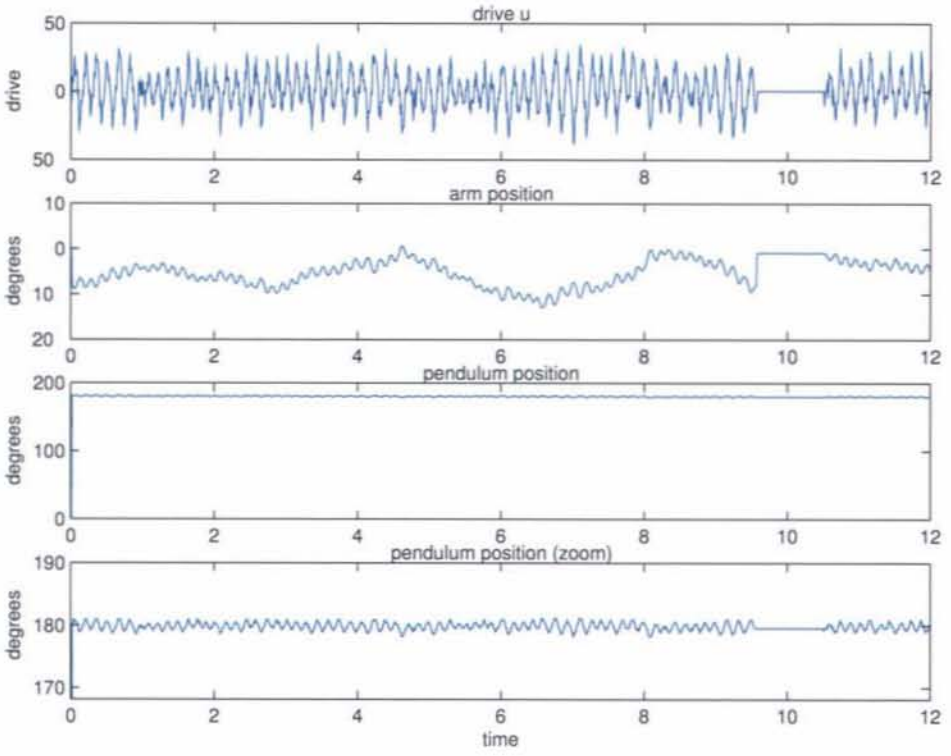

Figure 5.10: Results of the experiments controller for setup 2

The closed loop poles belonging to these values of the control matrix are:

$$\begin{aligned} 0.9093 + 0.2585i \\ 0.9093 - 0.2585i \\ 0.9186 \\ 0.9885 + 0.0162i \\ 0.9885 - 0.0162i \end{aligned}$$

The stability radius for this last controller is calculated to be 1.46. Although the margin here is not as large as the margin of the stability radius with setup 1, it is stable and pretty robust. Probably this can be explained with the fact that the dead-zone here is not as big as in setup 1 and the parameters were better estimated.

5.4 Summary

Now, two working controllers are designed. A controller which is able to swing up the pendulum into upright position in a reasonable number of swings, a switching strategy which switches from one controller to another at the right moment and a controller which is able to keep the pendulum in upright position within a very close angle from the vertical (or sometimes even keeps it steady depending on the initial condition of β)

The previous described differences of the blind and dead-zones in clockwise and counter-clockwise direction have been taken into account while writing the controller program. The differences in C_0 and K_f in C and CC direction is hard to implement in C++. This will certainly be possible in Simulink.

Chapter 6

Conclusions and Recommendations

6.1 Conclusions

Three months after the start of the project a lot of experience about the setup is gained. Improving the model, calibrating the encoders, estimating the parameters and designing and building a controller have been performed. Running against problems with the parameter estimation has been very useful to understand how the system works and how they have to be solved. For the moment they can not be solved but once a dSPACE environment is build the method and files are ready to perform a new estimation. Concerning the design of a controller, a hybrid controller is build consisting of three parts; a swing-up, a switching and a balancing part. All these parts communicate so they can take over from each other at the right moment. Pole placement techniques are used to design the balancing controller but without succes. However, they have been very useful to search for those gains that are able to balance the pendulum. Although the controllers work properly and are able to keep the pendulum in upright position they can be optimized once the correct parameters are known.

It can be concluded that it is possible to build a proper controller for a simple under-actuated system like the inverted pendulum. The experiences and files of this project will be useful for future projects with this setup.

6.2 Recommendations

Because of several reasons it is necessary to build a dSPACE environment for these setup. First of all, one can build a model in Matlab Simulink and convert it to dSPACE. This is more convenient than writing in C++. Second, it is possible to do longer experiments which seems to be necessary. And third, with dSPACE it is possible to control both setups "online" and at the same time.

Once dSPACE environment is build the parameters have to be estimated again and the gains of the controller have to be adapted. Once this is done it is possible to introduce a reference trajectory. This reference trajectory defines a certain α or β the system has to follow. This could be a first step to future research on synchronization of this under-actuated system.

Bibliography

[Berg03] H.W.J.v.d.Berg, Technical Report Traineeship University of Melbourne, 2003, DCT report 2003.56

[Fran94] G.F.Franklin, "Feedback control of dynamic systems", 3rd editon, Addison-Wesley Publishing Company, 1994, ISBN: 0–201–53487–8

[Gelb74] A.Gelb, "Applied Optimal Estimation", MIT Press, Cambridge (MA), 1974

[Chye99] T.K.Chye and T.C.Sang, "Rotary Inverted Pendulum", Technical Report, School of Electrical and Electronic Engineering, Nanyang Technological University, 1999

[Åst00] K.J.Åström and K.Furuta, "Swinging up a pendulum by energy control", *Automatica*, volume 36 pp 287–295, 2000

[Kuc79] V.Kucera, "Discrete Linear Control; the polynomial equation approach", Wiley, Chichester, 1979, ISBN: 0–471–99726–9

[web1] http://wolfman.eos.uoguelph.ca/ jzelek/matlab/ctms

Appendix A

Calibration Results

	Setup 1		Setup 2	
	$\beta = 0^o$	$\beta = 180^o$	$\beta = 0^o$	$\beta = 180^o$
mov. 1	0	508	0	546
mov. 2	0	505	2	547
mov. 3	0	506	2	543
mov. 4	-1	507	-2	545
mov. 5	0	506	0	545

Table A.1: Servo potentiometer calibration results Experiment 1

	Setup 1		Setup 2	
	$\beta = 0^o$	$\beta = 180^o$	$\beta = 0^o$	$\beta = 180^o$
mov. 1	0	507	0	540
mov. 2	0	508	1	541
mov. 3	0	507	0	540
mov. 4	0	507	0	545
mov. 5	0	505	0	545

Table A.2: Servo potentiometer calibration results Experiment 2

	Setup 1		Setup 2	
	$\beta = 90^o$	$\beta = 270^o$	$\beta = 90^o$	$\beta = 270^o$
mov. 1	257	-255	274	-279
mov. 2	255	-258	274	-277
mov. 3	260	-256	275	-276
mov. 4	257	-255	274	-275
mov. 5	257	-255	274	-275

Table A.3: Servo potentiometer calibration results Experiment 3

Setup 1				
	exit cc	entrance cc	exit c	entrance c
mov. 1	-306	640	717	-306
mov. 2	-306	640	717	-306
mov. 3	-306	640	717	-306
mov. 4	-306	640	717	-306
mov. 5	-306	640	717	-306

Table A.4: Servo potentiometer calibration results Experiment 4 setup 1

Setup 2				
	exit cc	entrance cc	exit c	entrance c
mov. 1	-322	701	701	-322
mov. 2	-322	701	701	-322
mov. 3	-322	701	701	-322
mov. 4	-322	701	701	-322
mov. 5	-322	701	701	-322

Table A.5: Servo potentiometer calibration results Experiment 4 setup 2

Setup	Type	exp 1	exp 2	exp 3	exp 4	exp 5	Average
1	A-C	44	43	44	48	48	45.4
1	A-CC	-46	-41	-42	-44	-46	-43.8
1	B-C	45	46	45	46	46	45.6
1	B-CC	-45	-45	-44	-44	-45	-44.6
2	A-C	27	29	28	28	30	28.4
2	A-CC	-31	-32	-31	-29	-29	-30.4
2	B-C	36	35	37	38	38	36.4
2	B-CC	-38	-39	-39	-38	-38	-38.4

Table A.6: Results of motor dead zone experiment

Appendix B

Numerical Results of PE experiments

	Setup 1		Setup 2	
	C_1	J_{r1}	C_1	J_{r1}
exp 1	0.0959×10^{-3}	0.7877×10^{-3}	0.1176×10^{-3}	0.7727×10^{-3}
exp 2	0.1112×10^{-3}	0.7867×10^{-3}	0.1199×10^{-3}	0.7700×10^{-3}
exp 3	0.0905×10^{-3}	0.7883×10^{-3}	0.1204×10^{-3}	0.7700×10^{-3}
exp 4	0.0913×10^{-3}	0.7886×10^{-3}	0.1239×10^{-3}	0.7705×10^{-3}
exp 5	0.0888×10^{-3}	0.7888×10^{-3}	0.1216×10^{-3}	0.7701×10^{-3}
Average	0.0955×10^{-3}	0.7879×10^{-3}	0.1207×10^{-3}	0.7707×10^{-3}

Table B.1: Results of parameters estimation Pendulum rod

	Setup 1		Setup 2	
Drive	V_m	K_u	V_m	K_u
40	2.10	0.0525	2.59	0.0648
-40	-2.20	0.0550	-2.60	0.0650
50	2.90	0.0580	3.35	0.0670
-50	-3.00	0.0600	-3.35	0.0670
60	3.75	0.0625	4.14	0.0690
-60	-3.80	0.0633	-4.15	0.0692
70	4.50	0.0643	4.82	0.0689
-70	-4.60	0.0657	-4.86	0.0694
Average		**0.0618**		**0.0682**

Table B.2: Results of voltage measurement

Experiments	Direction	Time (s)
1-5	Clockwise (C)	1.0
6-10	Counterclockwise (CC)	1.0
11-15	Clockwise (C)	1.5
16-20	Counterclockwise (CC)	1.5

Table B.3: Experiments setup

	LSQM			**EKF**		
Exp	J_{z0}	C_0	K_f	J_{z0}	C_0	K_f
1	0.0038	0.0221	0.0226	0.0071	0.0347	0.0401
2	0.0037	0.0198	0.0311	0.0076	0.0337	0.0377
3	0.0035	0.0173	0.0268	0.0070	0.0283	0.0415
4	0.0036	0.0188	0.0252	0.0071	0.0305	0.0375
5	0.0034	0.0188	0.0216	0.0066	0.0292	0.0389
6	0.0036	0.0268	0.0185	0.0063	0.0382	0.0397
7	0.0036	0.0203	0.0154	0.0066	0.0304	0.0391
8	0.0033	0.0177	0.0257	0.0064	0.0270	0.0432
9	0.0033	0.0195	0.0135	0.0062	0.0289	0.0393
10	0.0036	0.0180	0.0198	0.0067	0.0279	0.0389
11	0.0035	0.0134	0.0516	0.0081	0.0237	0.0531
12	0.0033	0.0125	0.0403	0.0075	0.0205	0.0594
13	0.0032	0.0126	0.0568	0.0065	0.0197	0.0621
14	0.0033	0.0129	0.0499	0.0075	0.0221	0.0520
15	0.0033	0.0128	0.0481	0.0076	0.0213	0.0586
16	0.0040	0.0203	0.0724	0.0085	0.0369	0.0390
17	0.0037	0.0174	0.0617	0.0079	0.0302	0.0430
18	0.0035	0.0115	0.0359	0.0069	0.0174	0.0709
19	0.0033	0.0127	0.0209	0.0064	0.0182	0.0680
20	0.0032	0.0121	0.0436	0.0059	0.0166	0.0708
Average	**0.0035** ±0.0002	**0.0169** ±0.0040	**0.0351** ±0.016	**0.0070** ±0.0007	**0.0268** ±0.0064	**0.0486** ±0.012

Table B.4: Results of LSQM and EKF for setup 1

	$C_0[Ns/m]$	$K_f[-]$
Clockwise (C)	**0.0161** ±0.0036	**0.0374** ±0.0134
Counterclockwise (CC)	**0.0176** ±0.0046	**0.0327** ±0.0205

Table B.5: Difference friction for C- and CC-direction setup 1

	LSQM			EKF		
Exp	J_{z0}	C_0	K_f	J_{z0}	C_0	K_f
1	0.0051	0.0085	0.0517	0.0035	0.0034	0.0728
2	0.0052	0.0085	0.0605	0.0025	0.0039	0.0836
3	0.0051	0.0084	0.0582	0.0025	0.0035	0.0823
4	0.0052	0.0083	0.0610	0.0020	-0.0023	0.0896
5	0.0053	0.0082	0.0678	0.0001	-0.0069	0.1097
6	0.0068	0.0250	0.0530	0.0056	0.0231	0.0554
7	0.0070	0.0266	0.0557	0.0053	0.0235	0.0602
8	0.0071	0.0219	0.0640	0.0042	0.010	0.0771
9	0.0078	0.0387	0.0415	0.0080	0.0567	0.0233
10	0.0076	0.0419	0.0369	0.0077	0.0627	0.0223
11	0.0052	0.0067	0.0780	0.0037	0.0140	0.0515
12	0.0049	0.0062	0.0846	0.0061	0.0091	0.0479
13	0.0051	0.0058	0.0828	0.0042	0.0111	0.0542
14	0.0051	0.0069	0.0741	0.0044	0.0126	0.0521
15	0.0051	0.0067	0.0764	0.0042	0.0139	0.0492
16	0.0065	0.0131	0.0609	0.0080	0.0210	0.0359
17	0.0065	0.0120	0.0685	0.0082	0.0158	0.0439
18	0.0062	0.0094	0.0809	0.0079	0.0164	0.0384
19	0.0059	0.0091	0.0867	0.0080	0.0150	0.0406
20	0.0057	0.0090	0.0861	0.0076	0.0131	0.0470
Average	**0.0059** ±0.0009	**0.0140** ±0.011	**0.0665** ±0.014	**0.0052** ±0.0024	**0.0160** ±0.016	**0.0569** ±0.022

Table B.6: Results of LSQM and EKF for setup 2

	$C_0[Ns/m]$	$K_f[-]$
Clockwise (C)	**0.0074** ±0.0011	**0.0695** ±0.0133
Counterclockwise (CC)	**0.0207** ±0.00123	**0.0634** ±0.0175

Table B.7: Difference friction for C- and CC-direction setup 2

Appendix C

Design of an EKF for the pendulum rod

The design of an Extended Kalman Filter (EKF) starts with the equations of motion. These are already defined in section (4.3).

$$J_{r1}\ddot{\beta} + C_1\dot{\beta} + m_1 g l_1 \sin(\beta) = 0 \tag{C.1}$$

Only for the simulation an input u will be added so equation (C.1) will become:

$$J_{r1}\ddot{\beta} + C_1\dot{\beta} + m_1 g l_1 \sin(\beta) = u \tag{C.2}$$

but for the derivation of the EKF this will be ignored.

Define a vector $\underline{x}$ which represents the states and the parameters to be estimated.

$$\underline{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} \beta \\ \dot{\beta} \\ C_1 \\ J_{r1} \end{bmatrix}$$

and $\dot{\underline{x}}$ becomes:

$$\dot{\underline{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} \dot{\beta} \\ \ddot{\beta} \\ 0 \\ 0 \end{bmatrix}$$

Now the equation of motion can be written into state-space form.

$$\dot{\underline{x}} = \underline{f}(\underline{x}(t), t) = \begin{bmatrix} \dot{\beta} \\ \ddot{\beta} \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} x_2 \\ -\frac{x_2 x_3}{x_4} - \frac{m_1 g l_1}{x_4} \sin x_1 \\ 0 \\ 0 \end{bmatrix}$$

Since we will only measure the position β of the system in this part of the estimation $\underline{h}(\underline{x}(t), t)$ will be

$$\underline{h}(\underline{x}(t), t) = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}$$

so

$$z(t) = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} x(t)$$

The Extended Kalman Filter is defined by:

$$\dot{\underline{\hat{x}}} = \underline{f}(\underline{\hat{x}}(t), t) + K(\underline{z} - \underline{h}(\hat{x}, t)) \tag{C.3}$$

$$\dot{P} = F(\hat{x}(t), t)P(t) + P(t)F^T(\hat{x}(t), t) + Q(t) - P(t)H^T(\hat{x}(t), t)R^{-1}(t)H(\hat{x}(t), t)P(t) \tag{C.4}$$

$$K(t) = P(t)H^T(\hat{x}(t), t)R^{-1}(t) \tag{C.5}$$

Now equation (C.3) can be written as:

$$\dot{\underline{\hat{x}}} = \begin{bmatrix} \hat{x}_2 + K_1(z - \hat{x}_1) \\ -\frac{\hat{x}_2\hat{x}_3}{\hat{x}_4} - \frac{m_1 g l_1}{\hat{x}_4}\sin\hat{x}_1 + K_2(z - \hat{x}_1) \\ K_3(z - \hat{x}_1) \\ K_4(z - \hat{x}_1) \end{bmatrix}$$

In order to solve this K_i has to be determined. K_i can be solved from equation (C.5). But in order to solve this one equation (C.4) has to be solved first.

$F(\underline{x}, t)$ is the Jacobian of $f(t)$ and so is $H(\underline{x}, t)$ the Jacobian of $h(t)$. So:

$$F(\underline{x}(t), t) = \frac{\partial f_i(x, t)}{\partial x_j} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ F_1 & F_2 & F_3 & F_4 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

with:

$$F_1 = -\frac{m_1 g l_1}{x_4}\cos x_1 \tag{C.6}$$

$$F_2 = -\frac{x_3}{x_4} \tag{C.7}$$

$$F_3 = -\frac{x_2}{x_4} \tag{C.8}$$

$$F_4 = \frac{x_2 x_3}{x_4^2} + \frac{m_1 g l_1}{x_4^2}\sin x_1 \tag{C.9}$$

$$H(\underline{x}(t), t) = \frac{\partial h}{\partial x_j} = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}$$

R represents the matrix with measurement noise and its dimension is [1x1] because only β is measured. So:

$$R = \begin{bmatrix} R \end{bmatrix}$$

Q represents the model uncertainty and is defined as:

$$Q = \begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{21} & q_{22} & q_{23} & q_{24} \\ q_{31} & q_{32} & q_{33} & q_{34} \\ q_{41} & q_{42} & q_{43} & q_{44} \end{bmatrix}$$

P is the covariance error matrix and is symmetric.

$$P = \begin{bmatrix} P_{11} & P_{12} & P_{13} & P_{14} \\ P_{21} & P_{22} & P_{23} & P_{24} \\ P_{31} & P_{32} & P_{33} & P_{34} \\ P_{41} & P_{42} & P_{43} & P_{44} \end{bmatrix} = \begin{bmatrix} P_{11} & P_{12} & P_{13} & P_{14} \\ P_{12} & P_{22} & P_{23} & P_{24} \\ P_{13} & P_{23} & P_{33} & P_{34} \\ P_{14} & P_{24} & P_{34} & P_{44} \end{bmatrix}$$

To calculate P we have to calculate and integrate $\dot{P}$. If (C.4) is calculated a set of so called *Ricatti equations* are obtained.

$$\dot{P}_{11} = 2P_{12} + q_{11} - R^{-1}P_{11}^2 \tag{C.10}$$
$$\dot{P}_{12} = P_{22} + F_1P_{11} + F_2P_{12} + F_3P_{13} + F_4P_{14} + q_{12} - R^{-1}P_{11}P_{12} \tag{C.11}$$
$$\dot{P}_{13} = P_{23} + q_{13} - R^{-1}P_{11}P_{13} \tag{C.12}$$
$$\dot{P}_{14} = P_{24} + q_{14} - R^{-1}P_{11}P_{14} \tag{C.13}$$
$$\dot{P}_{22} = 2F_1P_{12} + 2F_2P_{22} + 2F_3P_{23} + 2F_4P_{24} + q_{22} - R^{-1}2P_{12} \tag{C.14}$$
$$\dot{P}_{23} = F_1P_{13} + F_2P_{23} + F_3P_{33} + F_4P_{34} + q_{23} - R^{-1}P_{12}P_{13} \tag{C.15}$$
$$\dot{P}_{24} = F_1P_{14} + F_2P_{24} + F_3P_{34} + F_4P_{44} + q_{24} - R^{-1}P_{12}P_{14} \tag{C.16}$$
$$\dot{P}_{33} = q_{33} - R^{-1}2P_{13} \tag{C.17}$$
$$\dot{P}_{34} = q_{34} - R^{-1}P_{13}P_{14} \tag{C.18}$$
$$\dot{P}_{44} = q_{44} - R^{-1}2P_{14} \tag{C.19}$$

Now it is possible to calculate the Kalman gains with equation (C.5).

$$\begin{bmatrix} K_1 \\ K_2 \\ K_3 \\ K_4 \end{bmatrix}(t) = \begin{bmatrix} P_{11} & P_{12} & P_{13} & P_{14} \\ P_{12} & P_{22} & P_{23} & P_{24} \\ P_{13} & P_{23} & P_{33} & P_{34} \\ P_{14} & P_{24} & P_{34} & P_{44} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} R \end{bmatrix}^{-1} = \begin{bmatrix} \frac{P_{11}}{R} \\ \frac{P_{12}}{R} \\ \frac{P_{13}}{R} \\ \frac{P_{14}}{R} \end{bmatrix}(t)$$

Now all equations are defined and ready to implement into Matlab Simulink. This is analog to the pictures that will be presented in appendix D. This is done because these are more interesting to see.

Appendix D

Design of an EKF for the total system

D.1 2 states and 4 parameters

This part will in fact be the same as Appendix C but of course with other equations. Here the equation of motions are a little bit more difficult. They are already defined in section (4.4).

$$\ddot{\alpha} = f_1(t)\dot{\alpha} - f_2(t)\dot{\alpha}^2 - f_3\sigma(\dot{\alpha}) + f_4 + f_5 u \tag{D.1}$$

with:

$$f_1 = \frac{\left(-(C_0 + \frac{K_t K_b}{R_a}) - 2m_1 l_1^2 \dot{\beta} \sin\beta\cos\beta\right)}{J_{z0} + m_1 l_0^2 + m_1 l_1^2 \sin^2\beta - \frac{m_1^2 l_0^2 l_1^2 \cos^2\beta}{J_{r1}}} \tag{D.2}$$

$$f_2 = \frac{\frac{m_1^2 l_0 l_1^3 \sin\beta\cos^2\beta}{J_{r1}}}{J_{z0} + m_1 l_0^2 + m_1 l_1^2 \sin^2\beta - \frac{m_1^2 l_0^2 l_1^2 \cos^2\beta}{J_{r1}}} \tag{D.3}$$

$$f_3 = \frac{K_f}{J_{z0} + m_1 l_0^2 + m_1 l_1^2 \sin^2\beta - \frac{m_1^2 l_0^2 l_1^2 \cos^2\beta}{J_{r1}}} \tag{D.4}$$

$$f_4 = \frac{\frac{C_1 m_1 l_0 l_1 \dot{\beta}\cos\beta}{J_{r1}} + \frac{m_1^2 g l_0 l_1^2 \sin\beta\cos\beta}{J_{r1}} + m_1 l_0 l_1 \dot{\beta}^2 \sin\beta}{J_{z0} + m_1 l_0^2 + m_1 l_1^2 \sin^2\beta - \frac{m_1^2 l_0^2 l_1^2 \cos^2\beta}{J_{r1}}} \tag{D.5}$$

$$f_5 = \frac{\frac{K_t K_u}{R_a}}{J_{z0} + m_1 l_0^2 + m_1 l_1^2 \sin^2\beta - \frac{m_1^2 l_0^2 l_1^2 \cos^2\beta}{J_{r1}}} \tag{D.6}$$

Define a vector $\underline{x}$ which represents the states and the parameters to be estimated.

$$\underline{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} = \begin{bmatrix} \alpha \\ \dot{\alpha} \\ \frac{K_t K_u}{R_a} \\ J_{z0} \\ (C_0 + \frac{K_t K_b}{R_a}) \\ K_f \end{bmatrix}$$

and $\dot{\underline{x}}$ becomes:

$$\dot{\underline{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \\ \dot{x}_5 \\ \dot{x}_6 \end{bmatrix} = \begin{bmatrix} \dot{\alpha} \\ \ddot{\alpha} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Now the equation of motion can be written into state-space configuration.

$$\dot{\underline{x}} = \underline{f}(\underline{x}(t), t) = \begin{bmatrix} \dot{\alpha} \\ \ddot{\alpha} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} x_2 \\ f_1 x_2 - f_2 x_2^2 - f_3 \sigma(x_2) + f_4 + f_5 u \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

With:

$$f_1 = \frac{\left(-(x_5) - 2m_1 l_1^2 \dot{\beta} \sin\beta \cos\beta\right)}{x_4 + m_1 l_0^2 + m_1 l_1^2 \sin^2\beta - \frac{m_1^2 l_0^2 l_1^2 \cos^2\beta}{J_{r1}}} \tag{D.7}$$

$$f_2 = \frac{\frac{m_1^2 l_0 l_1^3 \sin\beta \cos^2\beta}{J_{r1}}}{x_4 + m_1 l_0^2 + m_1 l_1^2 \sin^2\beta - \frac{m_1^2 l_0^2 l_1^2 \cos^2\beta}{J_{r1}}} \tag{D.8}$$

$$f_3 = \frac{x_6}{x_4 + m_1 l_0^2 + m_1 l_1^2 \sin^2\beta - \frac{m_1^2 l_0^2 l_1^2 \cos^2\beta}{J_{r1}}} \tag{D.9}$$

$$f_4 = \frac{\frac{C_1 m_1 l_0 l_1 \dot{\beta} \cos\beta}{J_{r1}} + \frac{m_1^2 g l_0 l_1^2 \sin\beta \cos\beta}{J_{r1}} + m_1 l_0 l_1 \dot{\beta}^2 \sin\beta}{x_4 + m_1 l_0^2 + m_1 l_1^2 \sin^2\beta - \frac{m_1^2 l_0^2 l_1^2 \cos^2\beta}{J_{r1}}} \tag{D.10}$$

$$f_5 = \frac{x_3}{x_4 + m_1 l_0^2 + m_1 l_1^2 \sin^2\beta - \frac{m_1^2 l_0^2 l_1^2 \cos^2\beta}{J_{r1}}} \tag{D.11}$$

This time only the position α will be measured as an output of the system. β will be treated as an input of the system. It will be measured but it is not necessary to include this as a measured input in h. So:

$\underline{h}(\underline{x}(t),t)$ will be

$$\underline{h}(\underline{x}(t),t) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

so

$$z(t) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} x(t)$$

Now equation (C.3) can be written as:

$$\dot{\underline{\hat{x}}} = \begin{bmatrix} \hat{x}_2 + K_1(z-\hat{x}_1) \\ f_1 x_2 - f_2 x_2^2 - f_3\sigma(x_2) + f_4 + f_5 u + K_2(z-\hat{x}_1) \\ K_3(z-\hat{x}_1) \\ K_4(z-\hat{x}_1) \\ K_5(z-\hat{x}_1) \\ K_6(z-\hat{x}_1) \end{bmatrix}$$

and

$$F(\underline{x}(t),t) = \frac{\partial f_i}{\partial x_j} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ F_1 & F_2 & F_3 & F_4 & F_5 & F_6 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

with:

$$F_1 = 0 \tag{D.12}$$

$$F_2 = \frac{-x_5 - 2m_1 l_0 l_1^2 \dot{\beta} sin\beta \cos\beta}{S_1} - 2\frac{m_1^2 l_0 l_1^3 \sin\beta cos^2\beta x_2}{J_{r1} S_1} - 4\frac{x_6 15 e^{30x_2}}{(e^{30x_2}+1)^2 S_1} \tag{D.13}$$

$$F_3 = \frac{u}{S_1} \tag{D.14}$$

$$F_4 = -\frac{(-x_5 - 2m_1 l_0 l_1^2 \dot{\beta} \sin\beta \cos\beta)x_2}{S_1^2} + \frac{m_1^2 l_0 l_1^3 \sin\beta cos^2\beta x_2^2}{J_{r1} S_1^2} + \frac{x_6(1-\frac{2}{e^{30x_2}+1})}{S_1^2} - \frac{m_1 l_0 l_1 \dot{\beta}^2 sin\beta + \frac{C_1 m_1 l_0 l_1 \dot{\beta}\cos\beta}{J_{r1}} + \frac{m_1^2 g l_0 l_1^2 \sin\beta\cos\beta}{J_{r1}}}{S_1^2} - \frac{x_3 u}{S_1^2} \tag{D.15}$$

$$F_5 = -\frac{x_2}{S_1} \tag{D.16}$$

$$F_6 = -\frac{1-\frac{2}{e^{30x_2}+1}}{S_1} \tag{D.17}$$

$$S_1 = x_4 + m_1 l_0^2 + m_1 l_1^2 sin^2\beta - \frac{m_1^2 l_0^2 l_1^2 cos^2\beta}{J_{r1}} \tag{D.18}$$

$$H(\underline{x}(t),t) = \frac{\partial h}{\partial x_j} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

R represents the matrix with measurement noise and its dimensions are [1x1] because only α is used as a measured parameter. So:

$$R = \begin{bmatrix} R \end{bmatrix}$$

Q represents the model uncertainty and is defined as:

$$Q = \begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} & q_{15} & q_{16} \\ q_{21} & q_{22} & q_{23} & q_{24} & q_{25} & q_{26} \\ q_{31} & q_{32} & q_{33} & q_{34} & q_{35} & q_{36} \\ q_{41} & q_{42} & q_{43} & q_{44} & q_{45} & q_{46} \\ q_{51} & q_{52} & q_{53} & q_{54} & q_{55} & q_{56} \\ q_{61} & q_{62} & q_{63} & q_{64} & q_{65} & q_{66} \end{bmatrix}$$

P is the covariance error matrix and is symmetric.

$$P = \begin{bmatrix} P_{11} & P_{12} & P_{13} & P_{14} & P_{15} & P_{16} \\ P_{21} & P_{22} & P_{23} & P_{24} & P_{25} & P_{26} \\ P_{31} & P_{32} & P_{33} & P_{34} & P_{35} & P_{36} \\ P_{41} & P_{42} & P_{43} & P_{44} & P_{45} & P_{46} \\ P_{51} & P_{52} & P_{53} & P_{54} & P_{55} & P_{56} \\ P_{61} & P_{62} & P_{63} & P_{64} & P_{65} & P_{66} \end{bmatrix} = \begin{bmatrix} P_{11} & P_{12} & P_{13} & P_{14} & P_{15} & P_{16} \\ P_{12} & P_{22} & P_{23} & P_{24} & P_{25} & P_{26} \\ P_{13} & P_{23} & P_{33} & P_{34} & P_{35} & P_{36} \\ P_{14} & P_{24} & P_{34} & P_{44} & P_{45} & P_{46} \\ P_{15} & P_{25} & P_{35} & P_{45} & P_{55} & P_{56} \\ P_{16} & P_{26} & P_{36} & P_{46} & P_{56} & P_{66} \end{bmatrix}$$

To calculate P we have to calculate and integrate $\dot{P}$. If (C.4) is calculated a set of so called *Ricatti equations* are obtained.

$$\dot{P}_{11} = 2P_{12} + q_{11} - R^{-1}P_{11}^2 \tag{D.19}$$

$$\dot{P}_{12} = P_{22} + F_1P_{11} + F_2P_{12} + F_3P_{13} + F_4P_{14} + F_5P_{15} + F_6P_{16} + q_{12} - R^{-1}P_{11}P_{12} \tag{D.20}$$

$$\dot{P}_{13} = P_{23} + q_{13} - R^{-1}P_{11}P_{13} \tag{D.21}$$

$$\dot{P}_{14} = P_{24} + q_{14} - R^{-1}P_{11}P_{14} \tag{D.22}$$

$$\dot{P}_{15} = P_{25} + q_{15} - R^{-1}P_{11}P_{15} \tag{D.23}$$

$$\dot{P}_{16} = P_{26} + q_{16} - R^{-1}P_{11}P_{16} \tag{D.24}$$

$$\dot{P}_{22} = 2F_1P_{12} + 2F_2P_{22} + 2F_3P_{23} + 2F_4P_{24} + 2F_5P_{25} + 2F_6P_{26} + q_{22} - R^{-1}2P_{12} \tag{D.25}$$

$$\dot{P}_{23} = F_1P_{13} + F_2P_{23} + F_3P_{33} + F_4P_{34} + F_5P_{35} + F_6P_{36} + q_{23} - R^{-1}P_{12}P_{13} \tag{D.26}$$

$$\dot{P}_{24} = F_1P_{14} + F_2P_{24} + F_3P_{34} + F_4P_{44} + F_5P_{45} + F_6P_{46} + q_{24} - R^{-1}P_{12}P_{14} \tag{D.27}$$

$$\dot{P}_{25} = F_1P_{15} + F_2P_{25} + F_3P_{35} + F_4P_{45} + F_5P_{55} + F_6P_{56} + q_{25} - R^{-1}P_{12}P_{15} \tag{D.28}$$

$$\dot{P}_{26} = F_1P_{16} + F_2P_{26} + F_3P_{36} + F_4P_{46} + F_5P_{56} + F_6P_{66} + q_{26} - R^{-1}P_{12}P_{15} \tag{D.29}$$

$$\dot{P}_{33} = q_{33} - R^{-1}2P_{13} \quad \text{(D.30)}$$
$$\dot{P}_{34} = q_{34} - R^{-1}P_{13}P_{14} \quad \text{(D.31)}$$
$$\dot{P}_{35} = q_{35} - R^{-1}P_{13}P_{15} \quad \text{(D.32)}$$
$$\dot{P}_{36} = q_{36} - R^{-1}P_{13}P_{16} \quad \text{(D.33)}$$
$$\dot{P}_{44} = q_{44} - R^{-1}2P_{14} \quad \text{(D.34)}$$
$$\dot{P}_{45} = q_{45} - R^{-1}P_{14}P_{15} \quad \text{(D.35)}$$
$$\dot{P}_{46} = q_{46} - R^{-1}P_{14}P_{16} \quad \text{(D.36)}$$
$$\dot{P}_{55} = q_{55} - R^{-1}2P_{15} \quad \text{(D.37)}$$
$$\dot{P}_{56} = q_{56} - R^{-1}P_{15}P_{16} \quad \text{(D.38)}$$
$$\dot{P}_{66} = q_{66} - R^{-1}2P_{13} \quad \text{(D.39)}$$

Now it is possible to calculate the Kalman gains with equation (C.5).

$$\begin{bmatrix} K_1 \\ K_2 \\ K_3 \\ K_4 \\ K_5 \\ K_6 \end{bmatrix}(t) = \begin{bmatrix} \frac{P_{11}}{R} \\ \frac{P_{12}}{R} \\ \frac{P_{13}}{R} \\ \frac{P_{14}}{R} \\ \frac{P_{15}}{R} \\ \frac{P_{16}}{R} \end{bmatrix}(t)$$

Now all equations are defined and ready to implement into Matlab Simulink. This is showed in figures D.1, D.3 and D.3.

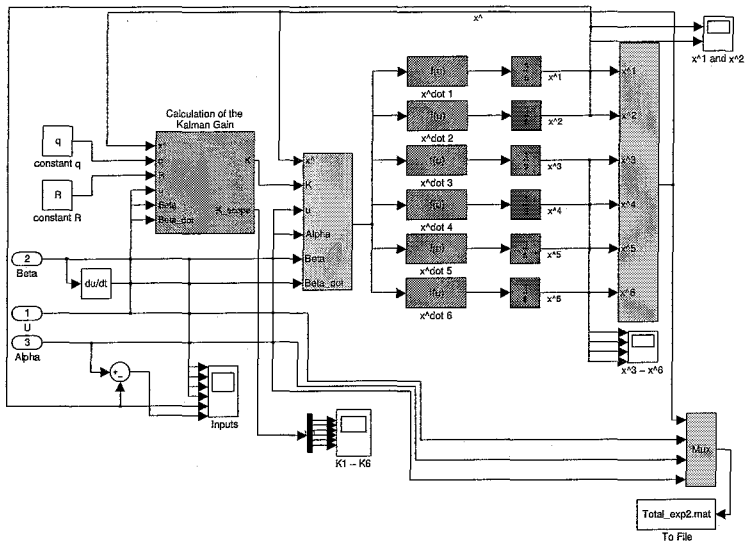

Figure D.1: Main file of Kalman filter

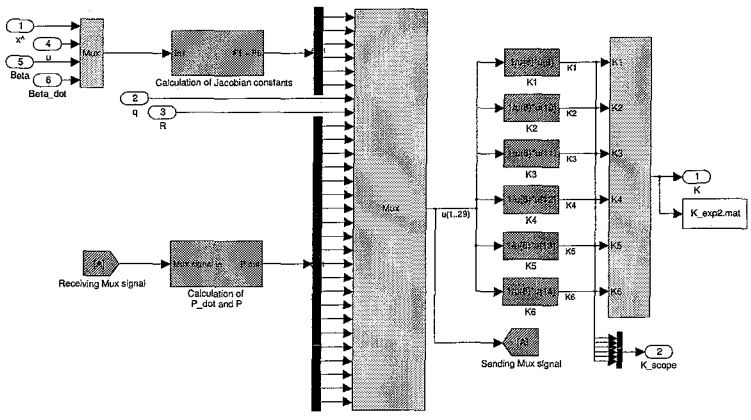

Figure D.2: Calculation of Kalman gains

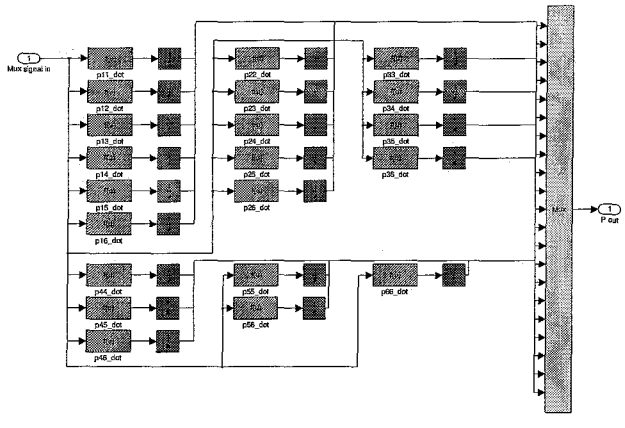

Figure D.3: Calculation of $\dot{P}$ and P

D.2 2 states and 3 parameters

The design of an EKF for the 2 states and only 3 parameters to estimate will be analog as described above. Here the vector $\underline{x}$ becomes:

$$\underline{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} \alpha \\ \dot{\alpha} \\ J_{z0} \\ (C_0 + \frac{K_t K_b}{R_a}) \\ K_f \end{bmatrix}$$

The continuous part of the design is assumed to be clear now so will not be described.

Appendix E

Linearisation of the non-linear model

Here the linearisation of the non-linear model will be shown. The linearisation is based on the expression of the non-linear function into a Taylor series about the operating point and the retention of only the linear terms. For n variables, $x_1, x_2, ..., x_n$ it can be briefly stated as:

$$y = f(x_1, x_2, ..., x_n)$$
$$\bar{y} = \bar{f}(\bar{x}_1, \bar{x}_2, ..., \bar{x}_n)$$

$$y - \bar{y} \approx (x_1 - \bar{x}_1)\frac{\partial f}{\partial x_1}\Big|_{\substack{x_1=\bar{x}_1 \\ x_2=\bar{x}_2 \\ \cdots \\ x_n=\bar{x}_n}} + (x_2 - \bar{x}_2)\frac{\partial f}{\partial x_2}\Big|_{\substack{x_1=\bar{x}_1 \\ x_2=\bar{x}_2 \\ \cdots \\ x_n=\bar{x}_n}} + ... + (x_n - \bar{x}_n)\frac{\partial f}{\partial x_n}\Big|_{\substack{x_1=\bar{x}_1 \\ x_2=\bar{x}_2 \\ \cdots \\ x_n=\bar{x}_n}}$$

This looks difficult but in practice it is very easy to apply. Beginning with the non-linear model, this is defined as (2.23):

$$\begin{bmatrix} J_{z0} + m_1 l_0^2 + m_1 l_1^2 \sin^2\beta & m_1 l_0 l_1 cos\beta \\ m_1 l_0 l_1 cos\beta & J_{z1} + m_1 l_1^2 \end{bmatrix} \begin{bmatrix} \ddot{\alpha} \\ \ddot{\beta} \end{bmatrix} +$$
$$\begin{bmatrix} C_0 + \frac{K_t K_b}{R_a} + m_1 l_1^2 \dot{\beta} \sin\beta\cos\beta & m_1 l_1^2 \dot{\alpha} \sin\beta\cos\beta - m_1 l_0 l_1 \dot{\beta}\sin\beta \\ -m_1 l_1^2 \dot{\alpha}\sin\beta\cos\beta & C_1 \end{bmatrix} \begin{bmatrix} \dot{\alpha} \\ \dot{\beta} \end{bmatrix} +$$
$$\begin{bmatrix} K_f \cdot \sigma(\dot{\alpha}) \\ m_1 g l_1 \sin\beta \end{bmatrix} = \begin{bmatrix} \frac{K_t K_u}{R_a} \\ 0 \end{bmatrix} u \tag{E.1}$$

This can be written as:

$$f_1(\alpha, \dot{\alpha}, \beta, \dot{\beta}) = (J_{z0} + m_1 l_0^2)\ddot{\alpha} + m_1 l_1^2 \ddot{\alpha} sin^2\beta + m_1 l_0 l_1 \ddot{\beta} cos\beta + (C_0 + \frac{K_t K_b}{R_a})\dot{\alpha} +$$
$$2m_1 l_1^2 \dot{\alpha}\dot{\beta} sin\beta cos\beta - m_1 l_0 l_1 \dot{\beta}^2 sin\beta + K_f \sigma(\dot{\alpha}) - \frac{K_t K_u}{R_a} u \tag{E.2}$$

$$f_2(\alpha, \dot{\alpha}, \beta, \dot{\beta}) = m_1 l_0 l_1 \ddot{\alpha} cos\beta + J_{r1}\ddot{\beta} - m_1 l_1^2 \dot{\alpha}^2 cos\beta sin\beta + C_1\dot{\beta} + m_1 g l_1 sin\beta \quad \text{(E.3)}$$

Linearisation around the unstable equilibrium $[0, 0, \pi, 0]^T$ for equation (E.2):

$$\bar{f}_1(0, 0, \pi, 0) = (J_{z0} + m_1 l_0^2)\ddot{\alpha} - m_1 l_0 l_1 \ddot{\beta} - \frac{K_t K_u}{R_a} u$$

and:

$$\frac{\partial f_1}{\partial \alpha}\Big|_{\alpha=0,\dot{\alpha}=0,\beta=\pi,\dot{\beta}=0} = 0$$

$$\frac{\partial f_1}{\partial \dot{\alpha}}\Big|_{\alpha=0,\dot{\alpha}=0,\beta=\pi,\dot{\beta}=0} = C_0 + \frac{K_t K_b}{R_a}$$

$$\frac{\partial f_1}{\partial \beta}\Big|_{\alpha=0,\dot{\alpha}=0,\beta=\pi,\dot{\beta}=0} = 0$$

$$\frac{\partial f_1}{\partial \dot{\beta}}\Big|_{\alpha=0,\dot{\alpha}=0,\beta=\pi,\dot{\beta}=0} = 0$$

Linearisation of (E.2) yields:

$$(J_{z0} + m_1 l_1^2)\ddot{\alpha} - m_1 l_0 l_1 \ddot{\beta} + (C_0 + \frac{K_t K_b}{R_a})\dot{\alpha} - \frac{K_t K_b}{R_a} u = 0 \quad \text{(E.4)}$$

Linearisation around the unstable equilibrium $[0, 0, \pi, 0]^T$ for equation (E.3):

$$\bar{f}_2(0, 0, \pi, 0) = -m_1 l_0 l_1 \ddot{\alpha} + J_{r1}\ddot{\beta}$$

and:

$$\frac{\partial f_2}{\partial \alpha}\Big|_{\alpha=0,\dot{\alpha}=0,\beta=\pi,\dot{\beta}=0} = 0$$

$$\frac{\partial f_2}{\partial \dot{\alpha}}\Big|_{\alpha=0,\dot{\alpha}=0,\beta=\pi,\dot{\beta}=0} = 0$$

$$\frac{\partial f_2}{\partial \beta}\Big|_{\alpha=0,\dot{\alpha}=0,\beta=\pi,\dot{\beta}=0} = -m_1 g l_1$$

$$\frac{\partial f_2}{\partial \dot{\beta}}\Big|_{\alpha=0,\dot{\alpha}=0,\beta=\pi,\dot{\beta}=0} = C_1$$

Linearisation of (E.3) yields:

$$-m_1 l_0 l_1 \ddot{\alpha} + J_{r1}\ddot{\beta} - m_1 g l_1 \beta + C_1\dot{\beta} = 0 \quad \text{(E.5)}$$

Hence, the linearized model is:

$$\begin{bmatrix} J_{z0} + m_1 l_0^2 & -m_1 l_0 l_1 \\ -m_1 l_0 l_1 & J_{r1} \end{bmatrix} \begin{bmatrix} \ddot{\alpha} \\ \ddot{\beta} \end{bmatrix} + \begin{bmatrix} C_0 + \frac{K_t K_b}{R_a} & 0 \\ 0 & C_1 \end{bmatrix} \begin{bmatrix} \dot{\alpha} \\ \dot{\beta} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & -m_1 g l_1 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \frac{K_t K_u}{R_a} \\ 0 \end{bmatrix} u \quad \text{(E.6)}$$