Educational Manual

Table of Contents

List of Figures

Preface

The CE 152 **Magnetic Levitation Model** is one of the range of educational scale models offered by Humusoft for teaching system dynamics and control engineering principles. The **Magnetic Levitation Model** and the associated manual are teaching aid for control engineering students at all levels and the experiments cover wide range of problems which appear in the industry.

The model belongs to the range of teaching systems directly controllable by an IBM PC compatible computer in real time. There is no need for using any standard laboratory instrumentation such as oscilloscope, plotter, signal generator, voltmeter etc. All functions of the above mentioned laboratory instrumentation are done by the software running on the PC. The student has no access to the signals coming from/to the model. Thus, there is no danger of damaging the model, amplifiers or the data acquisition card by manipulation with the cables. Instead of the direct physical access, software access to all signals measured and manipulated is available to the user.

There are four types of the software environment available to the user:

1. **DEMO program environment,** directly executable program written in C language, demonstrating closed loop PID control of the CE152 model. Friendly man-machine interface and good graphics enable the user to change the controller parameters in real time noticing the influence of the change on the system behavior.

2. **Simulink environment** - more complex demonstration model with graphical user interface running under MATLAB[1] and Simulink. This program allows to export recorded data directly into the MATLAB environment which facilitates tasks like simulation, data analysis or model tuning.

MATLAB is a high performance software package for scientific and numeric computation, signal processing and graphics in an environment where problems and solutions are expressed just as they are written mathematically - without traditional programming. The power of MATLAB environment is further extended by Simulink - a block oriented environment for simulation of dynamic systems and numerous toolboxes. Some of them are highly recommended for the experimentation with the CE152 model: Control System Toolbox, System Identification Toolbox, Optimisation Toolbox, Real-Time Windows Target and Virtual Reality Toolbox. An absolute minimum to conduct experiments from this manual is the Control Systems Toolbox and Simulink.

3. **User developed environment**, when just software drivers for the data acquisition cards supplied with the model are used in user written programs. This might be useful in solving digital image processing and pattern recognition tasks or special tasks where the following possibilities are not sufficient.

---

[1]MATLAB is a registered trademark of The MathWorks, Inc.

4. **Real-Time Windows Target** - this environment offers best control performance. Real-Time Windows Target is not included, but offered separately by The MathWorks. Simulink controller examples are included.

1. Getting Started

1.1. Introduction

The CE 152 Magnetic Levitation Model is one of a unique range of products designed for the theoretical study and practical investigation of basic and advanced control engineering principles. This includes system dynamics modelling, identification, analysis and various controllers design by classical and modern methods.

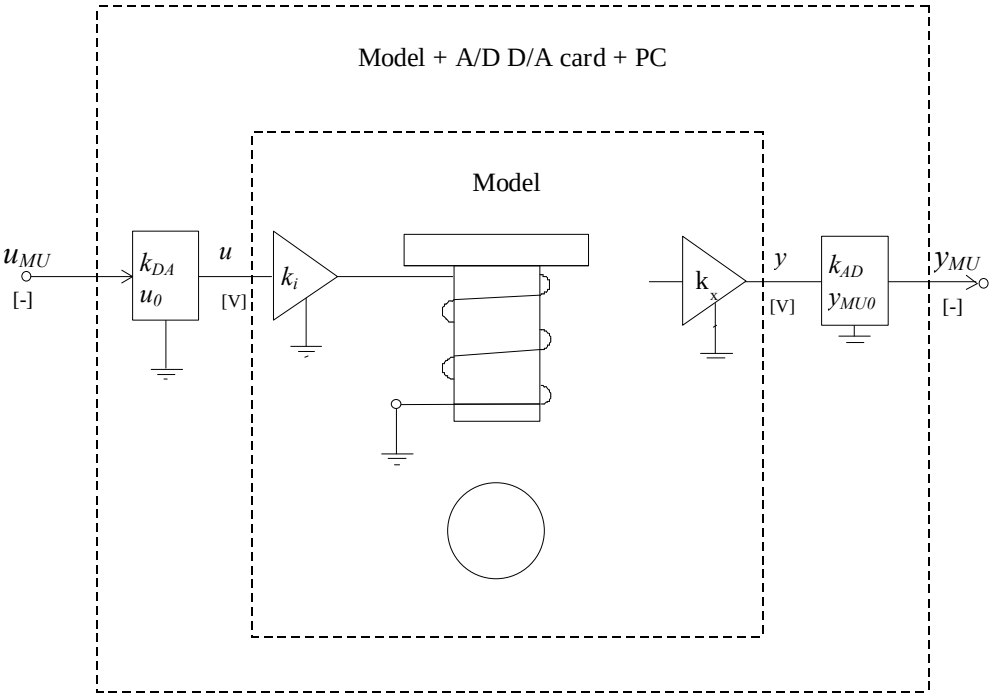

Figure 1: Interface to the CE152 Magnetic Levitation model.

A system configuration for the CE152 follows from the **Figure 1** where the system is connected to IBM PC compatible computer. The scheme shows, that the model interface can be considered at two different levels:

- physical level - input and output voltage to the coil power amplifier and from the ball position sensor.
- logical level - voltage converted by the data acquisition card and scaled to +-1 machine unit [MU].

Because experiments in this manual are implemented in Matlab environment, the later convention is used.

The core part of the model as shown in the **Figure 2** is a steel ball hanging in the magnetic field of the coil. The position of the ball is measured with a magnetic position sensor. The current for

the coil is amplified by an external amplifier and therefore is directly proportional to the input voltage. The model is connected to the PC via an universal data acquisition card, like the HUMUSOFT AD622 or MF624.

From the input-output view, the Magnetic Levitation Model can be approximated by a single input single output nonlinear dynamic system of order 2 or 3 depending on the modelling precision and with astatism of order 2.

The objective of this manual is to present typical problems and suggest experiments, giving some guidelines for the problem solution. The need for consulting control theory textbooks is minimal however the beginners should use them to learn principles not described in this manual. The text is divided into blocks. Each block contains the formulation of the problem followed by the description of the solution principles. Number of the experiments related to the problem is listed and typical results are given. It is important to note that the numerical results shown in the manual are relative and might be different from model to model due to differences in drives, propellers etc.

The manual is divided into chapters. The Chapter 2) is devoted to mathematical modelling. General model structure valid under some simplifications is derived in terms of nonlinear state space description and the corresponding block diagram. This chapter also contains experimental identification of model parameters. Experiments for measurement of parameter values are presented and typical results are given. Chapter 3) deals with linear control experiments - both conventional PIDs and state space LQ controllers. Chapter 4) contains some experiments in nonlinear control - feedforward controller design and compensation of model nonlinearity.

1.2. Windows Demonstration program

The Windows demonstration program is a fully menu-driven, user-friendly windows-based program to simplify your first experiments with the system. Using the Windows demonstration program is also the easiest way to test the model functionality in case of doubts. The program allows the user to enter the PID constants, define the sampling period, experiment duration and choose the controller setpoint.

1.3. Simulink Demonstration

The Simulink demonstration contains controller examples simplifying your first experiments with the system. Simulink models can be used for both real-time and non real-time simulations. Real-time simulation which can be used as real-time model controller requires Extended Real Time Toolbox (supplied with the model). Simulink models can be found in the `RT_TBX` subdirectory on installation floppy.

To repeat the experiments described in this manual, the Control Toolbox is needed (linear controller design). For your own experiments may be also useful to install System Identification Toolbox and Optimization Toolbox.

Simulink models allows the user to enter the PID constants, define the sampling period, experiment duration and choose the shape and parameters of the desired trajectory from the predefined set of functions available from signal generator block (constant, step, saw etc.) available from signal generator block. Model behaviour can be observed by scopes and stored for later analysis, model verification or controller optimisation by To File or To Workspace blocks (if Real Time Toolbox is used).

2. Modelling

The objective of this chapter is to derive the nonlinear model in the Simulink environment. This model will be used later for:

- experimental parameter estimation
- off-line PID controller tuning
- state space controller design.

Even if the system dynamics is relatively simple, an attempt to model it in details leads to a model which is complicated and difficult to use. Therefore the model was built in two steps - a basic model containing only features, significant for controller design and the advanced model giving detailed description of system behaviour.

We propose two ways how to get the model structure and parameters. The first one is a systematic modelling method based on variations approach, i.e. Lagrange's equation, the second method is based on the force balance. In the following text is used the later method.

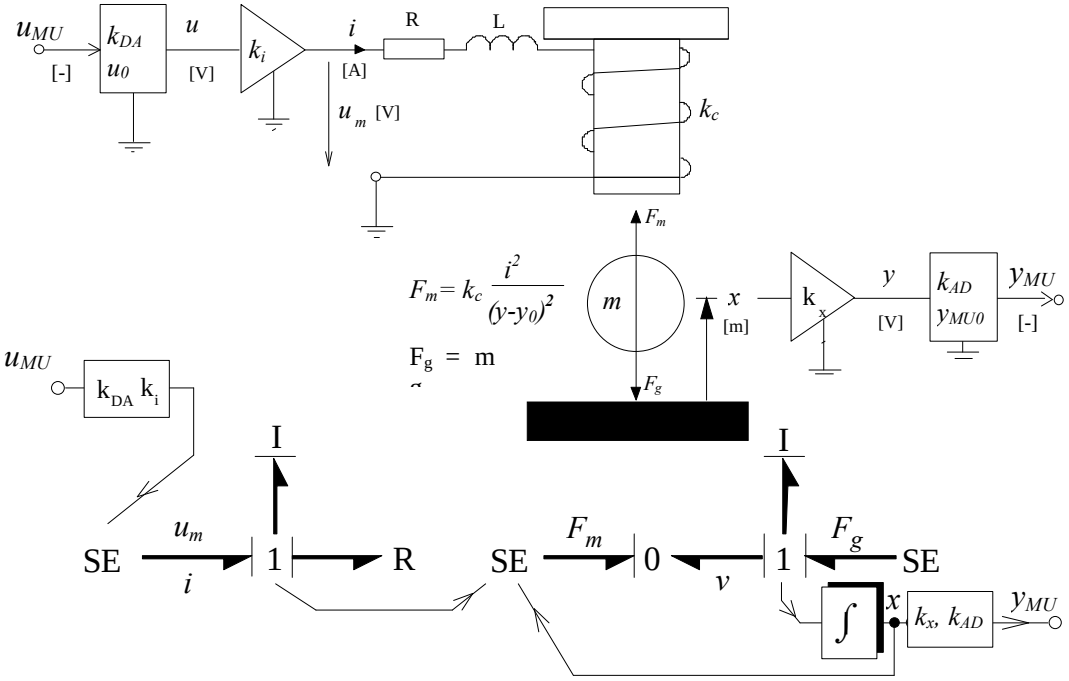

Figure 2: Principal scheme and bondgraph of the magnetic levitation model.

2.1. Model structure

The model shown in **Figure 2** consists from the following blocks:

- D/A converter
- the power amplifier
- the ball & coil subsystem,

C the position sensor
C A/D converter

The advanced version of the model contains in addition to these blocks the block of position limits and more detailed blocks of the ball and coil subsystem and power amplifier. In the simple model, only the electromagnetic motion force, gravity and static properties of the amplifiers are taken into account. Damping as well as position limits are neglected.

2.1.1. Ball & coil subsystem

The motion equation is based on the balance of all forces acting on the ball, i.e. gravity force F_g, electromagnetic force F_m and the acceleration force:

$$\begin{aligned} F_a &= F_m - F_g \\ F_M &= \frac{i^2 k_c}{(x - x_0)^2} \\ f_g &= m_k g \\ F_a &= m_k \ddot{x} \end{aligned} \qquad \textbf{(1)}$$

where

F_a = accelerating force [N]
i = coil current [A]
x = ball position [m]
k_c = coil constant
x_0 = coil offset [m]
m_k = ball mass [kg]
g = gravity constant [m.s^{-2}]

Some influences are neglected, e.g. the magnetic and mechanic ball damping and the influence of the limits.

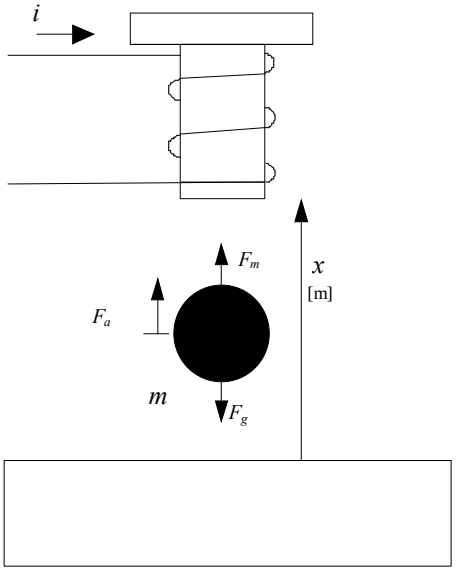

Figure 3: The ball and coil subsystem.

2.1.2. The power amplifier

The power amplifier is designed as a source of constant current with the feedback current stabilisation **Figure 4**.

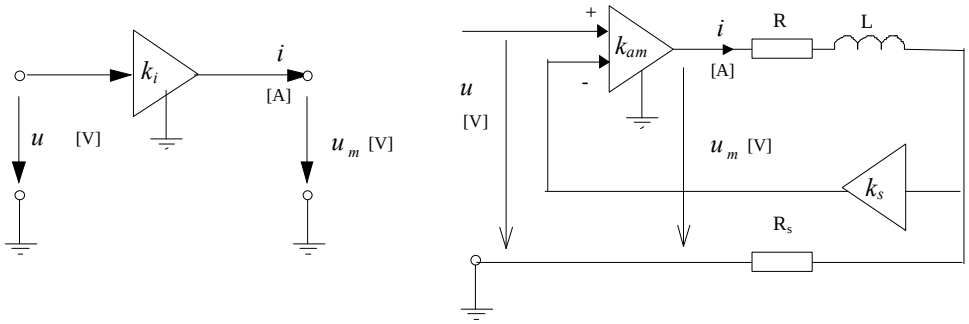

Figure 4: The power amplifier and its internal structure.

$$u_m = iR + \frac{di}{dt}L$$
$$u_m = K_{am}\left(u - R_s i\right) \qquad \textbf{(2)}$$

Because neither amplifier gain K_{am} nor the coil resistance R and inductance L are directly measurable, the amplifier and coil subsystem has to be modelled with the transfer function of 1st order with aggregated coefficients K_i - gain and T_a - time constant.

$$\frac{I(s)}{U(s)} = \frac{K_{am}}{R + K_{am}K_sR_s} \frac{1}{\frac{L}{R + K_{am}K_sR_s}s + 1} = k_i \frac{1}{T_a s + 1} \quad (3)$$

where

k_i = coil & amplifier gain [A/V]
T_a = coil & amplifier time constant [s]
K_{am} = amplifier gain [-]
K_s = current sensor gain [-]
R_s = resistance of the feedback resistor [Ohm]
R - coil resistance [Ohm]
L - coil inductance [H]

The amplifier is designed so that its time constant is negligible with respect to system dynamics and the current limits are not reached under standard operating conditions. Then the subsystem can be modelled just as a constant gain:

$$i = k_i u \quad (4)$$

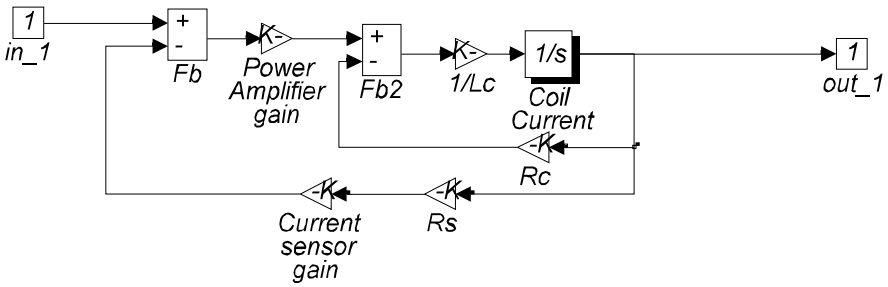

Figure 5: Simulation diagram of power amplifier.

2.1.3. D/A converter

This block includes the influence of the D/A acquisition card plus the software used for the data acquisition:

$$u = k_{DA}u_{MU} + u_0 \quad (5)$$

where

u = model input voltage [V]
$_{ump}$ = D/A converter input [MU]

$_{kea}$ = D/A converter gain [V/MU]
$_{up}$ = D/A converter offset [V].

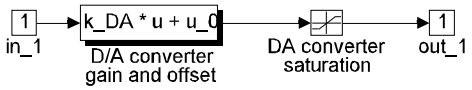

Figure 6: Simulation diagram of the D/A converter.

2.1.4. Position Sensor

An inductive position sensor is used to measure the ball position. The sensor can be approximated with a linear function:

$$y = k_x x + y_0 \quad \textbf{(6)}$$

where

$_k$ = position sensor gain [V/m]
y_0 = position sensor offset [m]
x = ball position [m]
y = model output voltage [V]

2.1.5. A/D converter

This block includes the influence of the A/D acquisition card plus the software used for the data acquisition. If the influence of limits is neglected, this relationship is described by a linear function :

$$y_{MU} = k_{AD} y + y_{MU0} \quad \textbf{(7)}$$

where

y = model output voltage [V]
y_{MU} = A/D converter output [MU]
k_{AD} = A/D converter gain [MU/V]
y_{MU0} = A/D converter offset [MU]

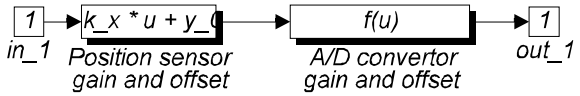

Figure 7: Simulation diagram of position sensor and A/D converter.

2.1.6. Complete system dynamics

Complete block diagram is to be assembled from the blocks listed above and is shown in **Figure 8**. The simplified motion equation of the system is:

$$m_k \ddot{x} = \frac{i^2 \, k_c}{(x - x_0)^2} - m_k g \qquad \textbf{(8)}$$

After substitution of **(4)** and **(6)** into **(8)** (for inputs in physical units):

$$m_k \frac{\ddot{x}}{k_x} = \frac{(k_i u)^2 \, k_c}{\left(\frac{y - y_0}{k_x} - x_0 \right)^2} - m_k g \qquad \textbf{(9)}$$

where

m_k = ball mass [kg]
y = ball position [m]
u = model input voltage [V]
k_c = power amplifier gain [A/V]
y_0 = power amplifier offset [V]
g = gravity acceleration constant [m.s-2]

Because k_c is not directly measurable, the term $k_c \, k_i^2$ is substituted by a current/force constant k_f.

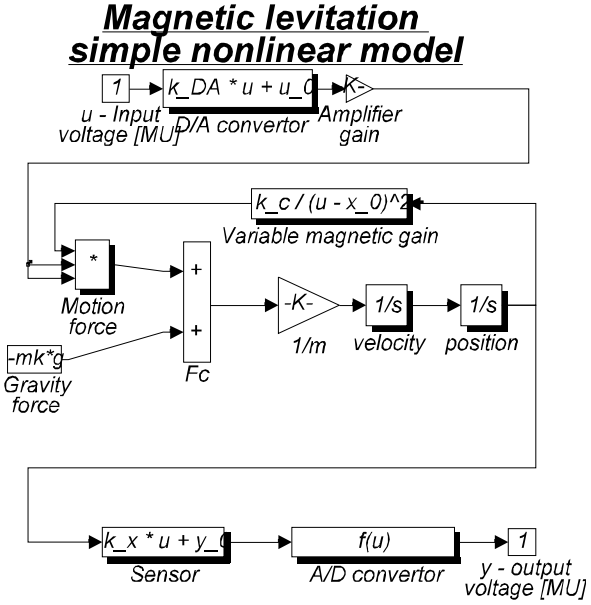

Figure 8: Simple model of the magnetic system.

2.2. Detailed model

The detailed model is based on the simple model, but in addition to this the influence of input amplifier dynamics, limits of the ball movements and ball damping is taken into account.
The **power amplifier dynamics** can be approximate by a 1st order linear system:

$$\frac{I(s)}{U(s)} = k_i \frac{1}{T_a s + 1} \tag{10}$$

To model the **damping**, the term k_{fv} is introduced into the equation **(8)**:

$$m_k \ddot{x} + k_{fv} \dot{x} = \frac{i^2 k_c}{(x - x_0)^2} - m_k g \tag{11}$$

where

k_{fv} = damping constant [N/m.s].

To model the **limits**, model constants have to vary according the ball position

$$\begin{aligned} m_k \ddot{x} + (k_{Fv} + k_{Fl}) \dot{x} + k_D x &= \frac{i^2 k_c}{(x - x_0)^2} - m_k g \quad for\ x<0 \\ m_k \ddot{x} + k_{fv} \dot{x} &= \frac{i^2 k_c}{(x - x_0)^2} - m_k g \quad for\ 0<x<L \\ m_k \ddot{x} + (k_{Fv} + k_{Fl}) \dot{x} + k_D (x - L) &= \frac{i^2 k_c}{(x - x_0)^2} - m_k g \quad for\ L<x \end{aligned} \tag{12}$$

where

k_{Fl} - limit constant - elasticity
k_{Dl} - limit constant - damping

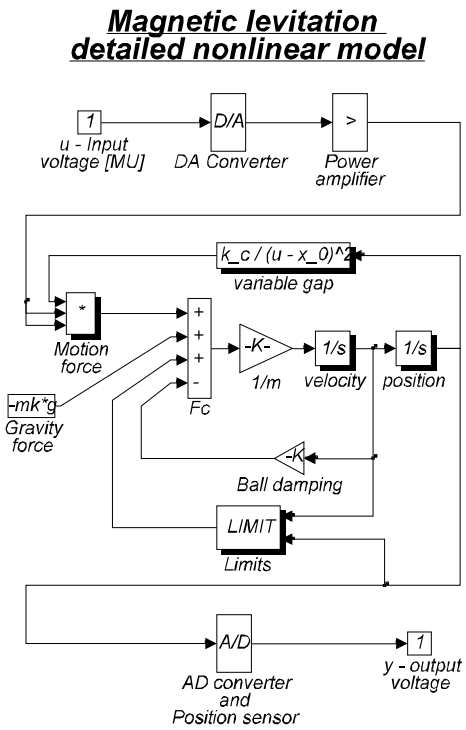

Figure 9: Detailed nonlinear model.

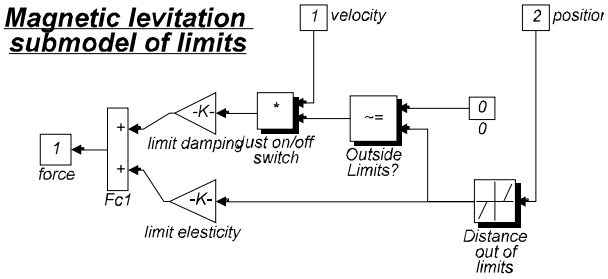

Figure 10: Advanced nonlinear model - submodel of the limits.

3. Identification

There are two identification approaches for model parameter estimation:

- identification of the parameters of some black box model from operational input-output data.
- identification of the parameters of the model described above item by item by direct measurement or by dedicated experiments.

The first method is general, elegant but its use is limited to the cases when the system model is expected to be linear. Because the nonlinearity of the model is of the type x^2, the model can be linearised around the setpoint with sufficient accuracy. The need to use operational data for the identification will cause some problems because the system is unstable in the open loop.
The second method is time consuming but gives good understanding of system behaviour taking into account physical structure of system functioning. The disadvantage is that not all parameters are accessible for direct measurement, therefore some of the parameters have to be estimated from the input-output data.

3.1. Parameters of A/D and D/A converters

The standard version of the CE152 Magnetic Levitation model is equipped with the HUMUSOFT MF614 data acquisition board interfaced via the Extended Real Time Toolbox for Matlab And Simulink. The A/D converter maps the input range -5V/+5V to the range -1 MU/+1 MU, the D/A converter maps the range -1 MU/+1 MU to the range -10V/+10V. This leads to the following set of converter constants:

k_{DA} = 10 [V/MU]
u_0 = 0 [V]
k_{AD} = 0.2 [MU/V]
y_{MU0} = 0 [MU]

3.1.1. The power amplifier

Typical parameters of each component of the coil and power amplifier are:

K_{am} = 100 [-]
K_s = 13.33 [-]
R_s = 0.25 [Ohm]
R_c =3.5 [Ohm]
L_c =0.03 [H]
k_i = 0.33 [A/V]
T_a = 10^{-5} [s]

3.2. The position sensor

The calibration of the position sensor is straightforward. Because the position sensor is supposed to be linear, it is sufficient to measure the sensor output voltage at both travel limits and to solve the equation **(11)** Typical model dimensions are:

D_k = 12.7 . 10^{-3} [m]
L_0 = 19 . 10^{-3} [m]
$L = L_0 - D_k$; = 6.3 . 10^{-3} [m]

Typical data obtained from a calibration experiment are shown in **Table 1**:

i	y_{MU} [MU]	y_i [V]	x_i [m]
1	0.0054	0.054	0
4	0.5078	5.078	L

Table 1: Typical data for position sensor calibration

When calculating sensor parameters, we have to remember, that the travelling distance of the ball L has to be calculated as a difference between physical distance of the coil and the sensor L_0 and the ball diameter D_k . Now, we can calculate the constants of the position sensor:

$y_0 = y_1$ =0.0540 [m]
$_k = (y_4 - y_1) / (x_4 - x_1)$ =797.4603 [V/m]

3.3. Ball & coil subsystem

The properties of the ball & coil subsystem can be well approximated with the equation **(11)**. Because the magnetic field is nonhomogenous, the equation is just an approximation but holds well everywhere except in the immediate proximity of the position sensor (**Figure 11**), i.e. except upper 10% of the ball positioning range.

The static calibration process is based on the static balance between the constant gravity force and the electromagnetic force proportional to the square of the coil current. Because the system is unstable, the state of equilibrium is achieved either by an auxiliary controller, or slowly increasing the coil current. In the moment, when the ball starts to move towards the coil, the motion force is supposed to be just equal to the gravity.

There are two possibilities how to obtain the coefficients of the equation **(11)**:

- to measure ball position and coil current in two distinct points. It is recommended to avoid the top and bottom 10% of the ball travelling distance to avoid nonlinearities and to choose the distance of calibration points at least 50% of the distance not to loose the precision.
- to measure an arbitrary number of points inside the 10%-90% of the ball position range and to use an interpolation technique to fit the coefficients of the equation **(11)** to the measured data.

3.3.1. Ball mass

To estimate dynamic properties of the ball & coil subsystem, only the ball mass has to be measured. This can be done either by weighting the ball or calculated approximately from known ball dimensions and steel density. The value of ball mass will be needed later to calculate coil constants as well as a parameter of the motion equation **(1)**:

$D_k = 12.7\ 10^{-3}$ [m]
$\rho = 7800$ [kg.m^{-3}]
$V_k = 4/3\ (\ (D_k/2)^3 = 1.07\ .\ 10^{-6}$ [m^3]
$m_k = \rho\ V_k = 8.37\ .\ 10\text{-}3$ [kg]

3.3.2. Coil constant - two point calibration

If just two pairs of calibration data are available as shown in **Table 2**, coefficients of the equation **(11)** can be estimated by the formula **(13)**.

i	$_{ump}$ [MU]	u_i [V]	y_{MU}[MU]	x_i [m]
2	0.3	3	0.05	0.56 . 10-3
3	0.1	1	0.50	6.2 . 10-3

Table 2: Typical data for two point calibration

$$x_0 = \frac{\frac{x_3}{u_3} - \frac{x_2}{u_2}}{\frac{1}{u_3} - \frac{1}{u_2}} \qquad k_f = k_i^2 k_c = m_k g \frac{(x_2 - x_0)^2}{u_2^2} \tag{13}$$

where

u_2 , u_3 - equilibrium model input voltages at calibration points [V]
x_2 , x_3 - equilibrium ball positions [m]
k_f - aggregated coil/amplifier constant $k_i^2 k_c$.

Typical results:

$k_f = 0.653 . 10^{-6}$
$x_0 = 9.02 . 10^{-3}$

Note: in the identification process, the aggregated gain k_f is calculated. It has to be separated into the power amplifier gain k_i (which can be calculated from the parameters of amplifiers components) and the coil gain k_c .

3.3.3. Coil constant - LQ optimal calibration

If more then 2 pairs of input data are available, the interpolation curve can not match all of them exactly - some interpolation technique is needed to fit the parameters in an optimal way. The most natural criterion of optimality is the sum of squared errors of estimated equilibrium force:

$$J = \sum_i \left(\frac{u_i^2 k_f}{(x - x_0)^2} - m_k g \right)^2 \tag{14}$$

Finding the minimum of the criterion is a standard problem of nonlinear least squares and can be

solved either by some dedicated routines - e.g. `leastsq(...)` from Matlab Optimisation Toolbox or by a universal function minimisation procedure. In our example was chosen the simplex procedure `fmins(...)` because is a part of standard Matlab. The search terminated in less than 50 iterations. Use of some more advanced technique from the Optimisation Toolbox speeds up the calculations if the Optimisation toolbox is available. The easiest way, how to obtain sufficient volume of recorded data is to use closed loop control with slowly changing setpoint. Sample data are shown in **Figure 12**.
Comparison between results of the 4-point and least squares optimisation are shown in **Figure 11**.

Typical results:

$k_f = 0.606 \,.\, 10^{-6}$
$x_0 = 8.26 \,.\, 10^{-3}$

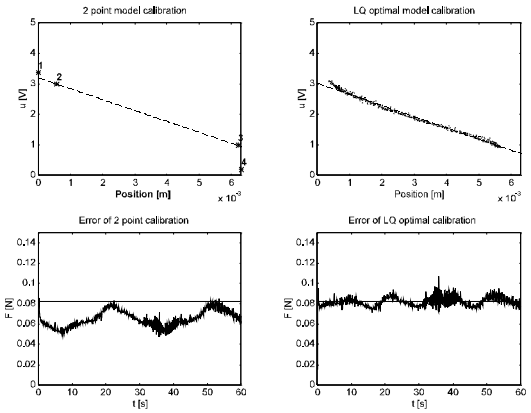

Figure 11 Calibration of static model parameters.

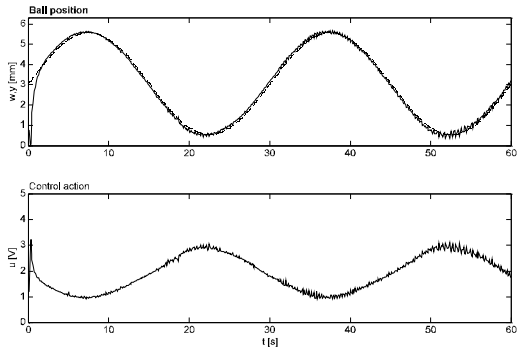

Figure 12 Data for calibration of static model parameters obtained by closed loop control with very slow change of the desired ball position.

3.4. Model verification by open loop simulation

If model constants are identified on the part by part basis, the behaviour of the complete model has to be verified from the input-output view. The standard method of the verification is the open loop simulation - recorded input data are connected to the model input and the model output is compared with the recorded output data. Minor changes in model parameters can be done to improve the match between real and simulated response.
This method can not be used with the Magnetic Levitation Model because of the unstable pole. **Figure 13** shows that in this case the ball position moves from one limit to another.

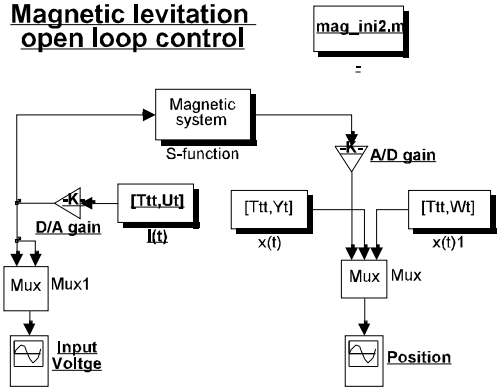

Figure 13: Open loop simulation - block diagram.

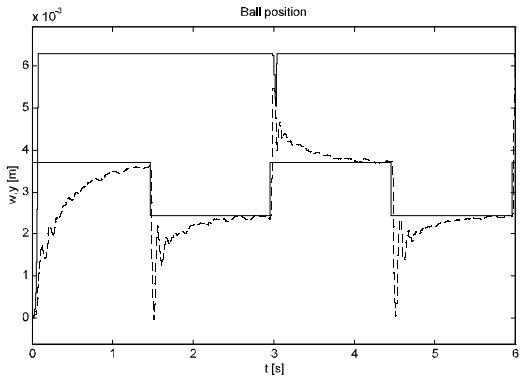

Figure 14: Open loop simulation. Desired, measured (dashed) and simulated (solid) position.

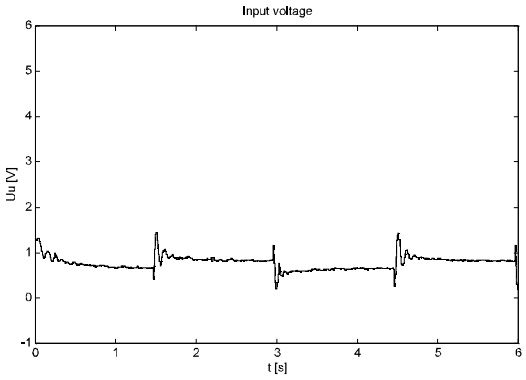

Figure 15: Open loop simulation - measured and simulated controller output.

3.5. Model verification by closed loop simulation

In these cases where the open loop simulation can not be used, the whole subsystem system/controller has to be simulated. In this case is the desired trajectory is used as an input signal and both the ball position and the controller output are used as output signals and compared with actual values. Results of the closed loop simulation have to be interpreted with some precautions because the close match between simulated and recorded ball position does not imply close match between system and model parameters. If model parameters have to be accepted, the controller output must match the recorded value as well. An example of closed loop simulation is in the **Figure 17**.

3.5.1. Parameters of the detailed model

Additional parameters which appear in the advanced model can not be measured directly or by a dedicated experiment. The only possibility is the trial-and-error method using real experimental data. Typical values of the ball damping constant K_{fv} is:

$$K_{Fv} = 0.02 \text{ [N s m}^{-1}\text{]}$$

Parameters of the position limits - elasticity and damping - were set with respect to the numerical stability of the simulation even if their parameters are apparently unrealistic. Simulation limits are significantly softer then the real limits to avoid problems with the stiffness of the model. This intentional simplification has only minor effect on modelling accuracy because the ball under standard condition only rarely hits the limits:

B_l = 34 [N s m^{-1}]
K_l = 330000 [N m^{-1}]

3.6. Model linearisation

There are two alternative approaches how to obtain constants of a linear state-space model A, B, C ,D around the setpoint x_0 from the nonlinear model

- by numerical linearisation
- by analytical linearisation of the equations (**(3)-(10)**).

The standard form of state space model is:

$$\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u} \\ \boldsymbol{y} &= \boldsymbol{C}\boldsymbol{x} + \boldsymbol{D}\boldsymbol{u} \end{aligned} \qquad \textbf{(15)}$$

where:

u - system input [$_{ump}$]
x - system state [$i\ v\ x$]
y - system output [y_{MU}]
A, B, C, D - parameters of linear model.

The components of system state **x** were chosen coil current, ball position and ball velocity. To make the state-space description more general, system state is measured in metric units.

System input **u** and output **y** were scaled to machine units as measured by the Matlab RealTime toolbox to simplify the controller design.

3.6.1. Analytical linearisation

Exact value of the coefficients **A, B, C** and **D** can be obtained by analytical linearisation of the nonlinear state space model. The model has the general form

$$\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{\Phi}(\boldsymbol{x},\boldsymbol{u}) \\ \boldsymbol{y} &= \boldsymbol{\Psi}(\boldsymbol{x},\boldsymbol{u}) \end{aligned} \qquad \textbf{(16)}$$

This procedure gives exact values of the coefficients of the state-space model **(9)**:

$$A = \begin{bmatrix} -\frac{1}{T_a} & 0 & 0 \\ \frac{2\, i_{00}\, k_c}{(x_{00}-x_0)^2 m_k} & -\frac{K_{Fv}}{m_k} & \frac{-2\, i_{00}^2\, k_c}{(x_{00}-x_0)^3 m_k} \\ 0 & 1 & 0 \end{bmatrix} \quad B = \begin{bmatrix} \frac{k_{DA}}{T_a} \\ 0 \\ 0 \end{bmatrix}$$

$$C = \begin{bmatrix} 0 & 0 & k_x k_{AD} \end{bmatrix} \quad D = \begin{bmatrix} 0 \end{bmatrix} \qquad \textbf{(17)}$$

where

i_{00} = nominal coil current
v_{00} = nominal ball velocity
y_{00} = nominal ball position

3.6.2. Numerical linearisation

The MATLAB procedure `linmod` supplies numerical estimates of the matrices ABCD. The difference between analytical and numerical estimate is under 1%.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% numerical model linearisation
%
[ An, Bn, Cn, Dn ] = linmod('mag_mod', x_init, i00);
```

The procedure is based on the approximation formula

$$A_{i,j} = \frac{\Phi_i(x+dx) - \Phi(x)}{dx_j} \qquad \textbf{(18)}$$

$$dx_j = \begin{cases} \varepsilon & for\ k=j \\ 0 & else \end{cases}$$

where ρ is the step of numerical differentiation, usually $\rho = 10^{-5}$. Similar formulas are applied for matrices B, C and D.

4. Linear control

This chapter describes experiments with PID and Linear Quadratic controller design. Experiments are done both in simulation and using real hardware. Because of the unstable pole, some of standard methods for controller parameter design can not be used (Ziegler-Nichols) or give poor results (frequency domain based sampling period design).

4.1. Continuous PID controller

Standard PID controller can be described by a 2-nd order transfer function:

$$G_r(s) = K_p + \frac{K_i}{s} + K_d s = \\ = K \frac{(T_{z1}s+1)(T_{z2}s+1)}{s} \tag{19}$$

where

G_r= controller transfer function
K_p= controller proportional constant
K_i=controller integral constant
K_d=controller derivative constant
K= controller gain
T_{z1}, T_{z2}= time constants corresponding with controller zeros.

In our experiments, the modified version of the controller is used:

$$Y_r(s) = \left(K_p + \frac{K_i}{s} \right) E(s) + K_d s\ Y(s) \tag{20}$$

The modification of the standard version is that the derivative term processes only the output of the system and not the error signal. This option makes the control action more smooth while preserving the stability of the controller.

4.2. Discrete PID controller

We have two options how to convert PID constants into a z-polynomial suitable for the implementation in the PC environment:

C direct discretisation using approximate formulas for numerical differentiation - conversion from PID constants into a difference equation (And then to a polynomial in z if needed).

- indirect method.

The indirect method includes two steps:
C conversion of PID constants into the polynomial in *s* operator
C conversion from a polynomial in s operator into a polynomial in z operator.

The direct method uses the rectangular approximation for the integral term K_i:

$$I(t) = K_i \int_0^t e(\tau)\, d\tau$$
$$I(kT_s + T_s) = I(kT_s) + K_i T_s e(kT_s) \quad \textbf{(21)}$$

and the backward difference approximation for the derivative term:

$$D(t) = K_d \frac{dy(t)}{dt}$$
$$D(kT_s) = \frac{y(kT_s) - y(kT_s - T_s)}{T_s} \quad \textbf{(22)}$$

The P-term is identical with the continuous form:

$$P(t) = K_p e(t) \quad \textbf{(23)}$$

The resulting controller uses the formula:

$$u(kT_s) = P(kT_s) + I(kT_s) + D(kT_s) \quad \textbf{(24)}$$

The first step of the indirect method was already described in the equation **(20)**, the *s* to *z* conversion can be done by any of standard methods - see the Matlab `c2dm(...)` method. The popular zero order hold method uses the substitution:

$$s = \frac{z-1}{zT_s} \quad \textbf{(25)}$$

where

T_s = sampling period.

This approach gives the result already obtained by the direct method:

$$G_r(s) = K_p + K_i \frac{zT_s}{(z-1)} + K_d \frac{(z-1)}{zT_s} =$$
$$= \frac{K_p z(z-1) + K_i T_s z^2 + K_d / T_s (z-1)^2}{z(z-1)} \quad \textbf{(26)}$$

4.3. Rough estimation of PID constants

Because of the unstable pole, none of the standard methods for the design of the PID constants can be used. Therefore the trial-and-error method using the closed loop simulation model was used (**Figure 17**). The first guess on PID constants is:

$K_p = 6$
$K_i = 4$
$K_d = 0.02$

4.4. Root locus PID tuning

Root locus method of PID design allows to find the optimal controller gain supposing that the location of controller zeros and poles, i.e. the ratio of the constants K_p/K_i and K_p/K_d was already obtained by some other method.

Time constant corresponding to the position of controller zeros T_{z1} and T_{z1} are chosen using the frequency response of the system.

Choice of the proper gain is a trade-off between minimum overshot, reasonable control action, and fast transient response. The choice of the gain is supposed by the MATLAB function `rlocus` which calculates the root locus trajectory and `rlocfind` which allows to select the proper controller gain K by pointing on the desired root location (**Figure 16**). In our particular case the design objective was to move the unstable pole as far to the left as possible.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% root locus PID design
%

rx = rlocus(num, den, omega);

plot(real(rx),imag(rx))

[k, p]= rlocfind( num, den );
```

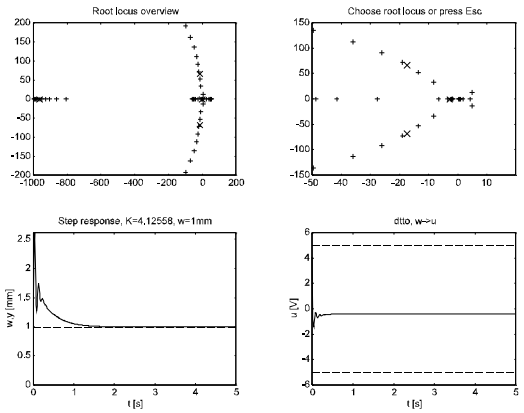

Figure 16: Root locus PID controller tuning.

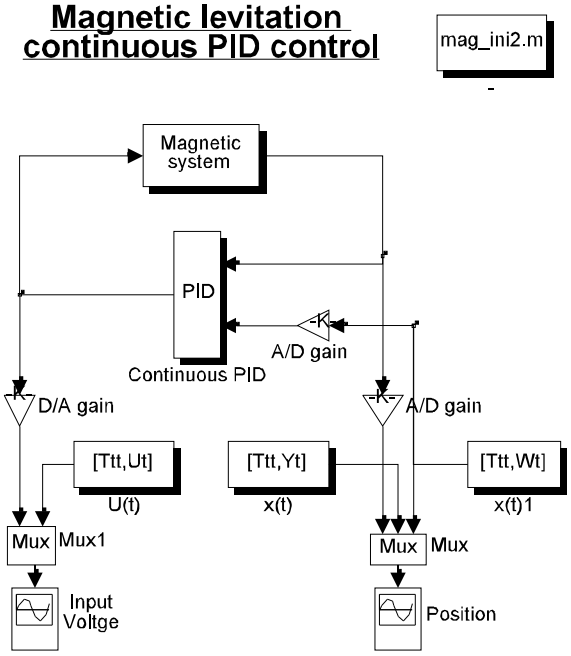

Figure 17: Simulation of closed loop PID control - simulation diagram.

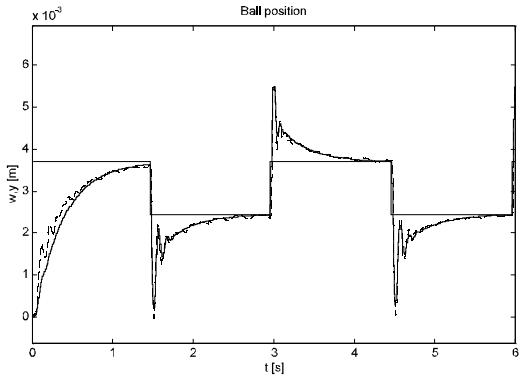

Figure 18: Simulation of closed loop PID control - Desired, measured (dashed) and simulated (solid) position.

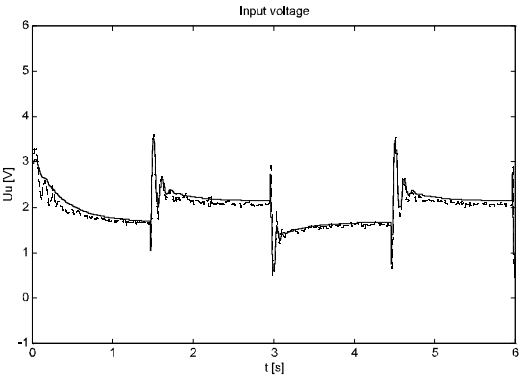

Figure 19: Simulation of closed loop PID control - measured (dashed) and simulated (solid) controller output.

4.5. Controller discretisation

Methods mentioned above allow to estimate the controller constants K_p, K_i and K_d. The next step is to set the sampling frequency T_s and to discretise the controller. As in the case of estimation of PID constants, standard methods used for the choice of the sampling period give too long sampling periods resulting in the unstable discrete controller.

Standard methods for the choice of the sampling period use either the closed loop frequency response and the 20 dB increase of position (**Figure 20**) error or 20 dB decay of closed loop position transfer function (**Figure 21**). The bandwidth at -20 dB corresponds with the critical frequency 100 Hz.

It can be verified by the nonlinear simulation (**Figure 22**) , that the controller with this sampling frequency is unstable. To reach reasonable degree of stability, the sampling frequency has to be increased at least 10 times, i.e. to 1 kHz.

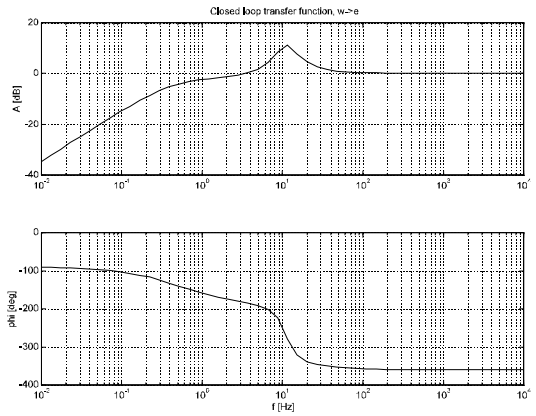

Figure 20: Closed loop transfer function with well tuned PID, desired position to position error.

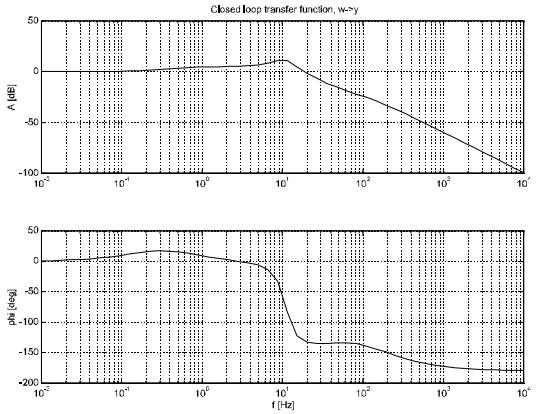

Figure 21: Closed loop transfer function with well tuned PID, desired position to actual position.

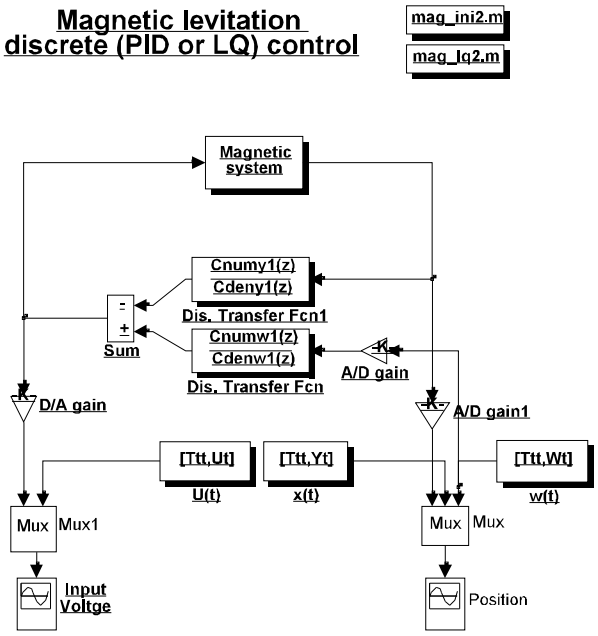

Figure 22: Simulation of discrete PID control - simulation diagram.

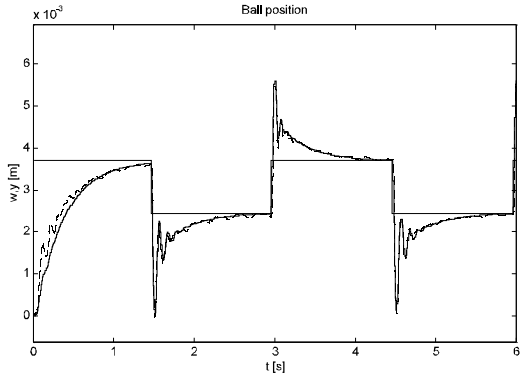

Figure 23: Simulation of discrete PID control - desired, measured (dashed) and simulated (solid) ball position.

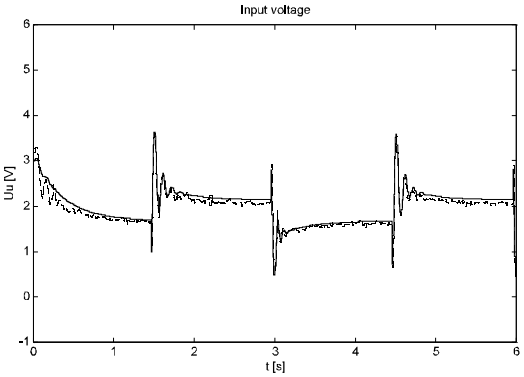

Figure 24: Simulation of discrete PID control - measured (dashed) and simulated (solid) controller output.

4.5.1. Implementation using RealTime toolbox

There are two possibilities how to implement a discrete controller in RealTime toolbox:

- using the MATLAB language and to communicate with the physical model using the commands `rtrd` (RealTime Read) and `rtwr` (RealTime Write). This solution works well with sampling periods 20 ms and more and allows e.g. implementation of nonlinear controllers.
- run the controller as a background process using the `rtlink` command. In this case the controller is limited to a linear polynomial transfer function in z operator, but the sampling period can be shorter then 1ms .

4.6. Experiments with PIDs

4.6.1. Empirical parameter tuning

Figure 25-**Figure 27** shows how the controller performance changes with varying controller parameters.

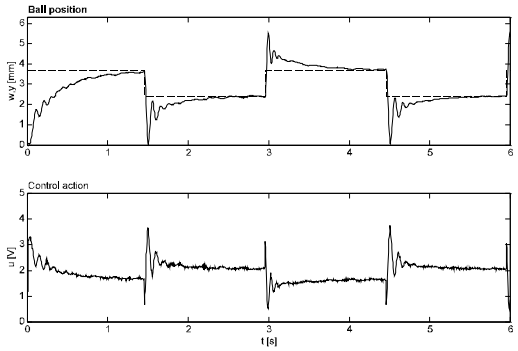

Figure 25: Closed loop control - experimental data - near optimal PID parameters - $K_p = 6$, $K_i = 4$, $K_d = 0.02$.

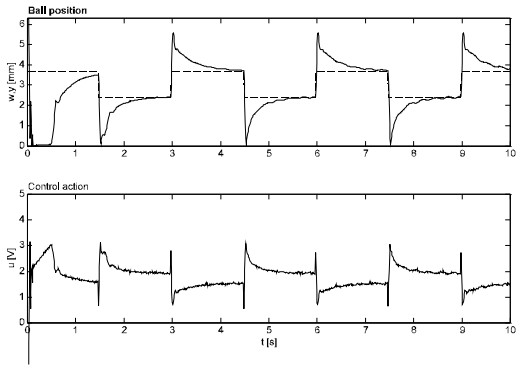

Figure 26: Closed loop control - experimental data - test of influence of PID parameters - K_p = 3, K_i = 4, K_d = 0.02.

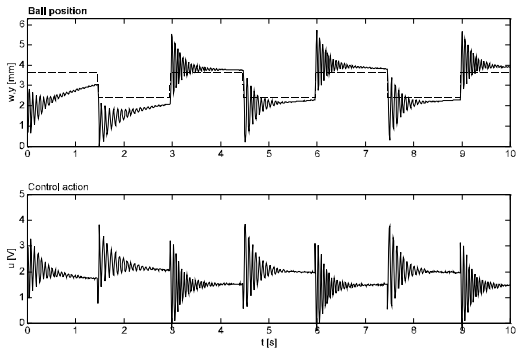

Figure 27: Closed loop control - experimental data - test of influence of PID parameters K_p = 5, K_i = 3, K_d = 0.017.

4.6.2. Influence of setpoint

To discuss the influence of setpoint on the performance of a PID controller with constant parameters, **Figure 28** to **Figure 31** show the closed loop behaviour. The controller was optimised for the setpoint 3 mm - **Figure 29**. Figures **Figure 28** and **Figure 30** show modest degradation of the parameters with ball position varying by 0.5 mm, the **Figure 31** shows significant degradation of the closed loop behaviour for ball position close to the top limit.
Controller parameters remain unchanged for all experiments- K_p = 6, K_i = 4, K_d = 0.02.

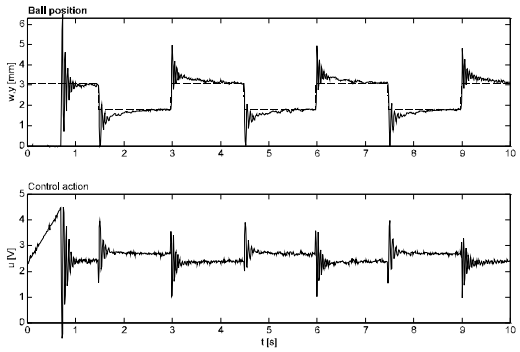

Figure 28: Closed loop control - experimental data - test of influence of mean ball height - ball height slightly below optimum.

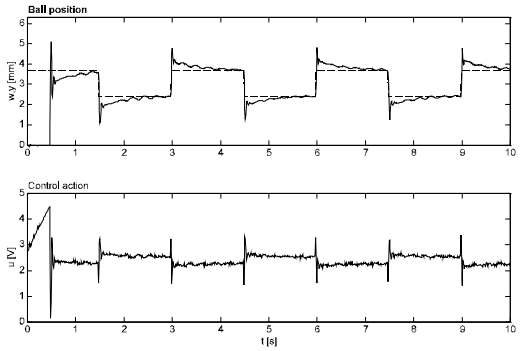

Figure 29: Closed loop control - experimental data - test of influence of mean ball height - optimum controller setting.

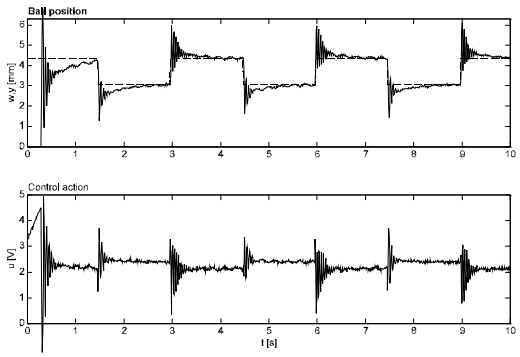

Figure 30: Closed loop control - experimental data - test of influence of mean ball height - ball height slightly above optimum.

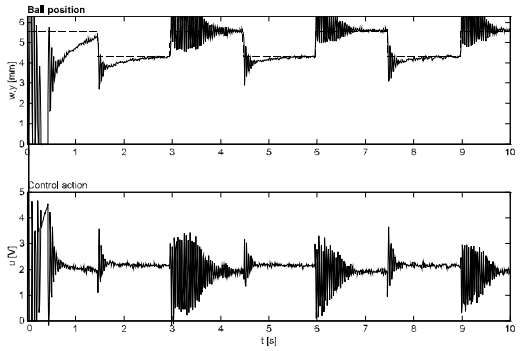

Figure 31: Closed loop control - experimental data - test of influence of mean ball height - ball position close to top limit.

4.6.3. Closed loop simulation

Values of some of the model parameters can not be estimated by mathematical and physical modelling because their nature is too complex (ball damping) or because they are used only as an approximation of nonlinear complex elements (amplifier time constant). Rough estimates of their values can be found by comparing the model response with measured data for different values of these parameters.

K_{fv}	0.02	ball damping	[N/m.s]
T_a	1e-3	amplifier time constant	[s]

Table 4: **Typical values of additional model parameters**

4.7. Linear quadratic controller

The main idea of the Linear Quadratic (LQ) design method is to find a control law u which represents a tradeoff between the sum of squares of error and the sum of squares of controller output. To measure this criterion, the performance index J is introduced as a weighted sum of squares for each nonzero initial state $\boldsymbol{x}$:

$$J = \int_{t=0}^{t=\infty} \boldsymbol{x}^T \boldsymbol{Q} \boldsymbol{x} + \boldsymbol{u}^T \boldsymbol{R} \boldsymbol{u} \qquad (27)$$

where matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$ are square, positively semidefinite weight matrices with dimensions which are compatible with the state vector $\boldsymbol{x}$ and input vector u.

If the system is a linear system described by a state space model

$$\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{Ax} + \boldsymbol{Bu} \\ \boldsymbol{y} &= \boldsymbol{Cx} + \boldsymbol{Du} \end{aligned} \tag{28}$$

where:

u - system input
x - system state
y - system output
A, B, C, D - parameters of linear model.

the control law can be found in the simple form of linear combination of system states:

$$\boldsymbol{u} = \boldsymbol{K_x x} \tag{29}$$

where

$\mathbf{K_x}$ is the feedback gain vector.

Matrices **Q** and **R** are square, positively semidefinite weight matrices with dimensions which are compatible with vectors **x** and **u**.

The optimal value of the feedback gain vector **K** is obtained from the steady state solution **S** of the Riccati equation:

$$\mathbf{0} = \boldsymbol{S_\infty A} + \boldsymbol{A}^T\boldsymbol{S_\infty} - \boldsymbol{S_\infty B}\,\boldsymbol{R}^{-1}\boldsymbol{B}_T\boldsymbol{S_\infty} + \boldsymbol{Q} \tag{30}$$

by the substitution:

$$\begin{aligned} \boldsymbol{u_t} &= -\left(\boldsymbol{R}^{-1}\boldsymbol{B}^T\boldsymbol{S_\infty}\right)\boldsymbol{x_t} = \\ &= -\boldsymbol{K_x x_t} \end{aligned} \tag{31}$$

In Matlab, LQ controller design is supported by the function `lqr` :

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% continuous LQ controller design
%
[ Kx ]  =  LQR( Ax, Bx, Q, R );
```

Note that if the position, velocity and the integral of the position error are chosen as system states (which is the case of the magnetic levitation apparatus), the PID controller can be considered as a special case of the state space controller with the control law:

$$\boldsymbol{K_x} = \begin{bmatrix} 0 & K_d & K_p & K_i \end{bmatrix} \tag{32}$$

Known constants of a PID controller K_p, K_i and K_d can be used not only as constants of the state feedback vector, but also directly as elements of the weight matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$. As a first guess, a diagonal matrix with squares of PID-like constants on its main diagonal can be used.
Because scaling of both matrices by the same factor does not influence the result of the controller design, the matrix $\boldsymbol{R}$ is scaled to 1.

$$\boldsymbol{R} = \begin{bmatrix} 1 \end{bmatrix}$$

$$\boldsymbol{Q} = \begin{bmatrix} 0 & & & \\ & K_d^2 & & \\ & & K_p^2 & \\ & & & K_i^2 \end{bmatrix} \tag{33}$$

4.7.1. State enhancement

It can be verified by a simple experiment, that the controller based solely on system states stabilises well the linear model, but is not stable with the nonlinear system. The reason is that the standard state space controller is missing the integral term in the control action. In fact it is a state-space alternative to the PD controller.

To overcome this difficulty, the state vector is enhanced by the integral of the output state. In practice, the integral of the output error is used instead of the integral of output and therefore the space controller based on the this enhanced state vector acts as an PID controller.
Here are the coefficients of the enhanced state space model:

$$\boldsymbol{A} = \begin{bmatrix} -\frac{1}{T_a} & 0 & 0 & 0 \\ \frac{2\, i_{00}\, k_c}{(x_{00}-x_0)^2 m_k} & -\frac{K_{Fv}}{m_k} & \frac{-2\, i_{00}^2\, k_c}{(x_{00}-x_0)^3 m_k} & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad \boldsymbol{B} = \begin{bmatrix} \frac{k_{DA}}{T_a} \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{34}$$

$$\boldsymbol{C} = \begin{bmatrix} 0 & 0 & k_x k_{AD} & 0 \end{bmatrix} \quad \boldsymbol{D} = \begin{bmatrix} 0 \end{bmatrix}$$

4.7.2. Observer design

Because only the ball position is directly measurable, other states of the system must be obtained from a dynamic system called an estimator or state observer. In our case we reconstruct the ball velocity and the coil current.
The observer is implemented as an state-space linear model whose parameters are identical to system parameters. The observer gain vector $\boldsymbol{L}$ is designed to minimise the difference between the actual and estimated model output.

$$\begin{aligned} \dot{\boldsymbol{x}}_m &= \boldsymbol{A}_m\boldsymbol{x}_m + \boldsymbol{B}_m\boldsymbol{u}_m + \boldsymbol{L}_x(\boldsymbol{y}-\boldsymbol{y}_m) \\ \boldsymbol{y}_m &= \boldsymbol{C}_m\boldsymbol{x}_m + \boldsymbol{D}_m\boldsymbol{u}_m \end{aligned} \qquad \textbf{(35)}$$

where:

u, x, y - system input ,state, output
$\mathbf{u_m}$, $\mathbf{x_m}$, $\mathbf{y_m}$ - observer model input ,state, output
$\mathbf{A_m}$, $\mathbf{B_m}$, $\mathbf{C_m}$, $\mathbf{D_m}$ - parameters of observer model.
$\mathbf{L_x}$ **-** observer feedback gain

The observer gain $\boldsymbol{L}$ can be obtained by any technique used for state-space controller design, e.g. by pole placement or Linear Quadratic design. In Matlab environment,
the LQ observer design procedure `lqe` is used:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% continuous LQ observer design
%
[ Lx ] = lqe( An, G, Cn, Q, R );
```

Tuning knobs of the estimator design - matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$ have identical meaning as matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$ in controller design. Good initial choice for the matrix $\boldsymbol{Q}$ is a diagonal matrix with one nonzero element on its main diagonal - the square of the inverse filter time constant $1/T_f^2$. Reasonable value of filter time constant T_f for ball position is cca 1ms.

$$\boldsymbol{R} = 1$$

$$\boldsymbol{Q} = \begin{bmatrix} 0 & & & \\ & 0 & & \\ & & 1/T_f^2 & \\ & & & 0 \end{bmatrix} \qquad \textbf{(36)}$$

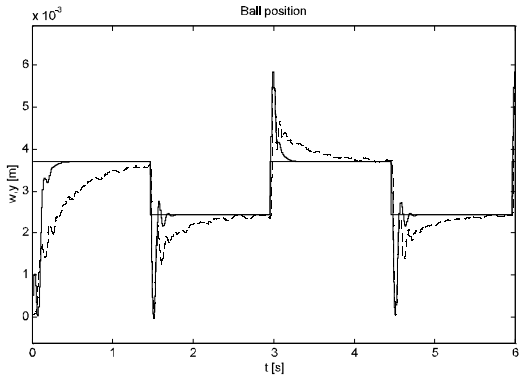

Figure 32: **Simulation of continuous state space control (solid line) compared with real PID controller (dashed line) - actual and desired position.**

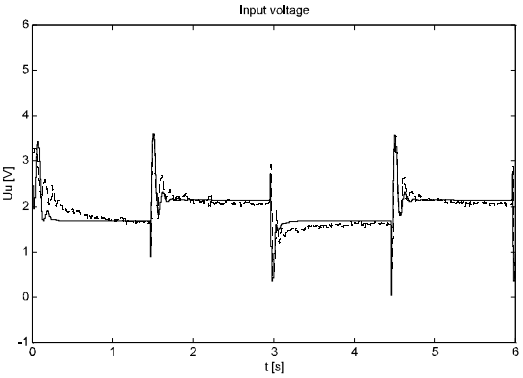

Figure 33: Simulation of continuous state space control (solid line) compared with real PID controller (dashed line) - controller output.

4.7.3. Implementation

Model parameters, observer gain and controller gain are merged together to build state space description of the controller $\boldsymbol{A_r}$, $\boldsymbol{B_r}$, $\boldsymbol{C_r}$, $\boldsymbol{D_r}$: In the regulator case where the controller has to bring the system from any nonzero initial state to zero (**Figure 34**), the state space controller description has the following form:

$$
\begin{aligned}
\dot{x}_r &= A_r x_r + B_r y \\
y &= C_r x_r + D_r y \\
A_r &= A - BK_x - L_X C \\
B_r &= L_x \\
C_r &= K_x \\
D_r &= 0
\end{aligned}
\qquad \textbf{(37)}
$$

where

A,B,C,D - are constants of the linear model

K_u,L_u - are controller and observer gain vectors designed by Matlab functions `lqr` and `lqe`.

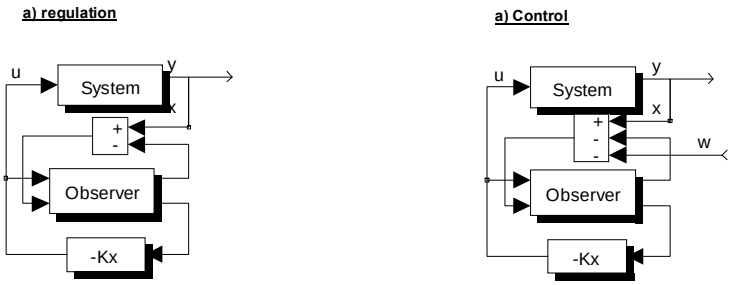

Figure 34: Regulator and tracking version of the state-space controller.

In the case of a controller for tracking (**Figure 34** right) , the ball position y is replaced with the tracking error (w-y).

Continuous state space description of the controller is converted into the discrete one with sampling period T_s (***I*** stands for an identity matrix of dimensions compatible with ***M***). Zero order hold is used as the conversion method (Matlab function `c2d(...)`).

$$
\begin{aligned}
x_{k+1} &= M_r x_k + N_r y_k \\
u_k &= C_r x_k + D_r y_k \\
M_r &= \exp(A_r T_s) \\
N_r &= \left(M_r - I\right) A_r^{-1} B_r
\end{aligned}
\qquad \textbf{(38)}
$$

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% convert controller to discrete
%
[ Ard, Brd ]= c2d( Ar, Br, Ts );
```

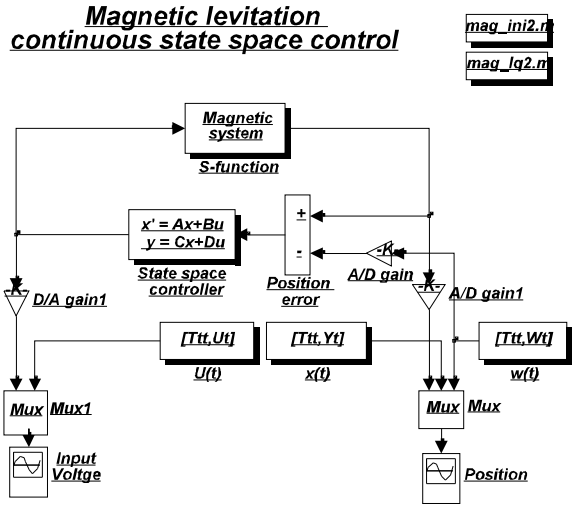

Figure 35: Continuous state-space control - a simulation diagram.

Discrete state space controller is converted into discrete polynomial controller (Matlab function `ss2tf(...)` or `c2dm(...)`).

$$
\begin{aligned}
z\boldsymbol{x}(z) &= \boldsymbol{M}_r\boldsymbol{x}(z) + \boldsymbol{N}_r\boldsymbol{y}(z) \\
\boldsymbol{u}_k &= \boldsymbol{C}_r\boldsymbol{x}(z) + \boldsymbol{D}_r\boldsymbol{y}(z) \\
G_r(z) &= \frac{u(z)}{y(z)} = \frac{b(z)}{a(z)} = \\
&= \boldsymbol{C}_r\left(z\boldsymbol{I} - \boldsymbol{M}_r\right)^{-1}\boldsymbol{N}_r + \boldsymbol{D}_r
\end{aligned}
\qquad \textbf{(39)}
$$

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% state space to polynomial transfer function in z
%
[ numr, denr ] = ss2tf(Ard, Brd, Cr, Dr, 1);
```

Discrete polynomial controller in *z* operator is implemented in Real Time Toolbox for Matlab as a polynomial controller running on the background.

4.7.4. Experiments

Although state-space controller should be stable for any setting of the design parameters, the controller with pre-set parameters is not stable. This is the result of the nonlinearity. To overcome this difficulty, the integral gain was increased cca. 20 times.
Although simulation results (**Figure 32**) indicate that LQ optimal controller should outperform PID, experimental results did not confirm this opinion.

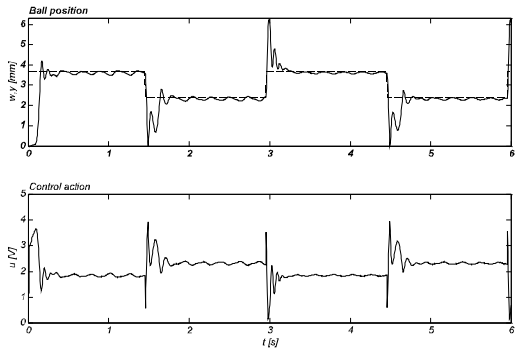

Figure 36: Closed loop control - experimental data - tuning LQ controller $W_p = 4$, $W_i = 120$, $W_d = 0.02$.

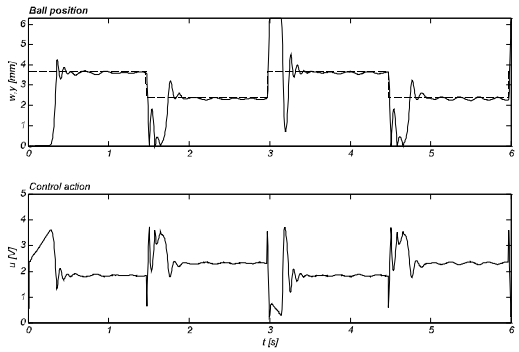

Figure 37: Closed loop control - experimental data - tuning LQ controller $W_p = 4$, $W_i = 200$, $W_d = 0.02$.

5. Nonlinear control

There are three simple experiments in nonlinear control design: nonlinear feedforward design and two experiments with the linearisation of the nonlinear relationship force/position/current. Surprisingly, removing these nonlinearities does not bring the expected effect because - in some limits - the quadratic relationship between coil current and the magnetic force compensates the inverse quadratic relationship position/force. Consequently, the static relationship current/force is almost linear (**Figure 11** top right). Therefore the linear control law is close to be optimal.

5.1. Feedforward compensation

The objective of the feedforward compensation is to eliminate the steady state control action. Therefore the controller response is faster, because the controller answers directly to the change of the desired position before the position error is built up. Therefore, the static relationship current/force has to be used. Solving the motion equation **(1)** under the assumption, that the motion force F=0 and dy/dt = 0, the feedforvard compensation voltage is given by the formula:

$$u_{ff} = -\sqrt{\frac{m_k g}{k_x}}\left(x - x_0\right) \qquad \textbf{(40)}$$

where:

u_{ff} - feedforward compensation voltage [V]

x - ball position [m]

Surprisingly, even if the system is nonlinear, the expression for the feedforward voltage is linear because the quadratic relation position - force compensates the quadratic relationship current/force.

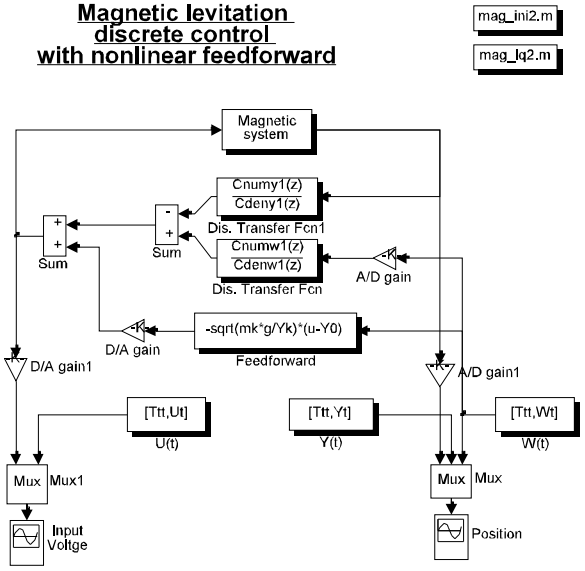

Figure 38: Simulation of feedforward control - simulation diagram.

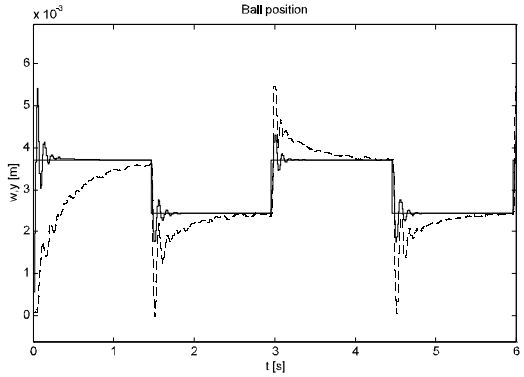

Figure 39: Simulation of feedforward control (solid line) compared with real PID controller (dashed line) - desired and actual ball position.

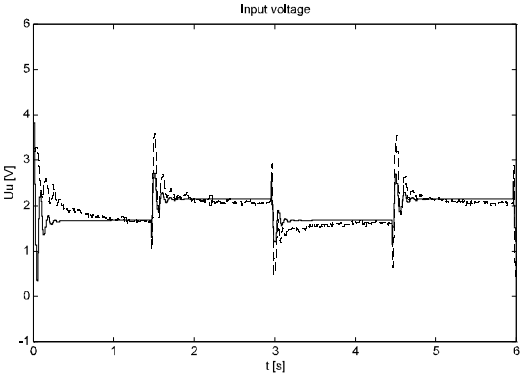

Figure 40:Simulation of feedforward control (solid line) compared with real PID controller (dashed line) - controller output.

5.2. Compensation of nonlinear relationship current/force

This nonlinearity causes that the controller gain increases with higher values of the coil current. To compensate this nonlinearity, the nonlinear function $u=f(u_u)$ is introduced in series with the controller (**Figure 41**).

The relation current/force is quadratic and can be linearised by square root. The compensation function was designed with two objectives:

- keep the parameters of the open loop at the at the nominal input voltage u_{00} unchanged
- keep the relation input voltage/motion force constant for each input voltage $0 < u < u_{max}$ and constant ball position x.

To achieve this objective, the compensation function has to meet three conditions:

- the relationship between controller input voltage and magnetic force $F(f(u))$ is linear with respect to the input voltage u
- the motion force for the nominal input voltage u_{00} is equal both in the compensated and in the uncompensated case, i.e. $F(f(u_{00}))$ is equal to $F(u_{00})$.
- the gain for the nominal input voltage u_{00} is equal both in the compensated and in the uncompensated case, i.e. $\mathrm{d}F(f(u_{00}))/\mathrm{d}u$ is equal $\mathrm{d}F(u_{00})/\mathrm{d}u$.

$$F = \frac{f(u)^2 \, k_c}{(x - x_0)^2}$$
$$f(u) = \sqrt{2u_{00}u - u_{00}^2} \qquad \textbf{(41)}$$

where

$f(u)$ - nonlinearity compensation function
u_{00} - nominal input voltage for linearisation
x - ball position
u - input voltage

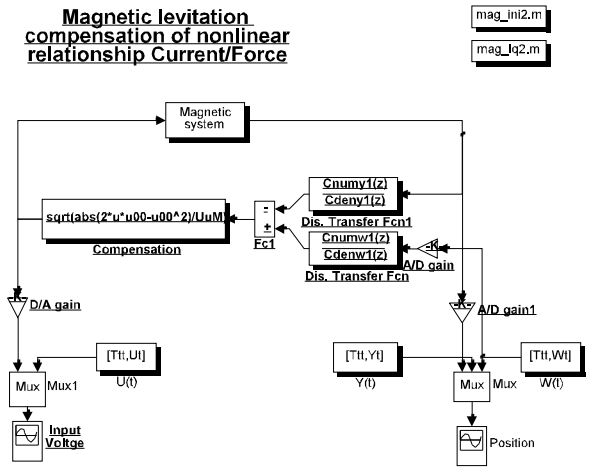

Figure 41: Simulated compensation of nonlinearity - I - current/force relationship - simulation diagram.

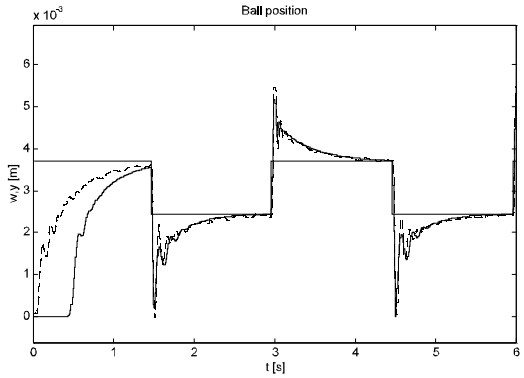

Figure 42: Simulated compensation of nonlinearity - I - current/force relationship - (solid line) compared with real PID controller (dashed line) - desired and actual position.

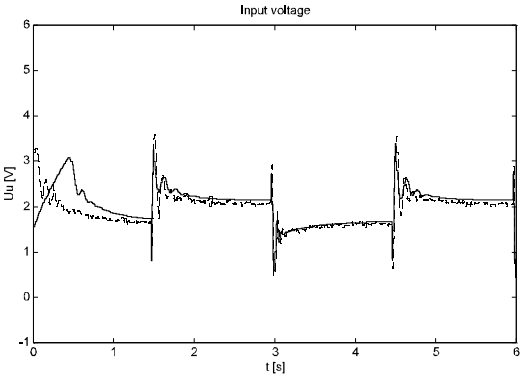

Figure 43: Simulated compensation of nonlinearity - I - current/force relationship - (solid line) compared with real PID controller (dashed line) - controller output.

5.3. Compensation of nonlinear relationship position/force

The inverse quadratic relationship position/force causes that the open loop gain decreases with growing distance between tha ball and the coil. Therefore the position-dependent gain *f(y)* is introduced. The compensation function was designed with two objectives:

- keep the parameters of the open loop at the at the nominal ball position x_{00} unchanged
- keep the relation input voltage/motion force constant for each position $0 < x < L$ and constant input force.

To achieve this objective, the compensation functionhas to meet three conditions:

- The relation between motion force *F(u,x)* and input voltage *u* is linear.
- The motion force for the nominal position x_{00} is equal to the motion force without compensation, i.e. $F(f(u,x),x_{00}))$ is equal to $F(u, x_{00})$.
- The gain for the nominal position x_{00} is equal to the gain obtained without compensation, i.e. $dF(f(u,x), x_{00})/dx$ is equal to $dF(u, x_{00})/dx$.

$$F = \frac{(u\ f(x))^2\ k_x}{(x_{00} - x_0)}$$
$$f(u,x) = u\ \frac{x - x_0}{x_{00} - x_0} \qquad \textbf{(42)}$$

where

F(u,x) - magnetic force at ball position *x* and model input voltage *u*
f(u,x) - nonlinearity compensation function
x_{00} - nominal ball position

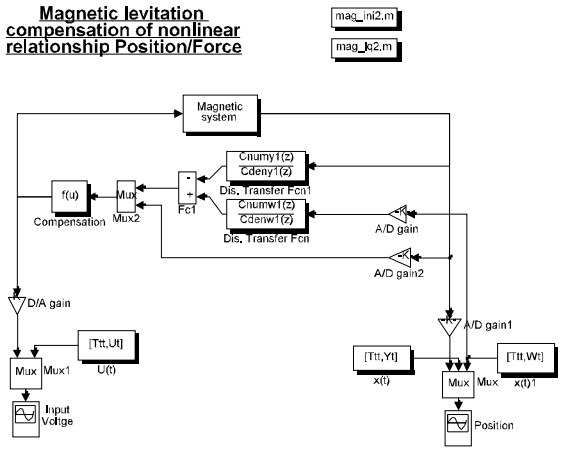

Figure 44: Simulated compensation of nonlinearity II - position/force relationship - simulation diagram.

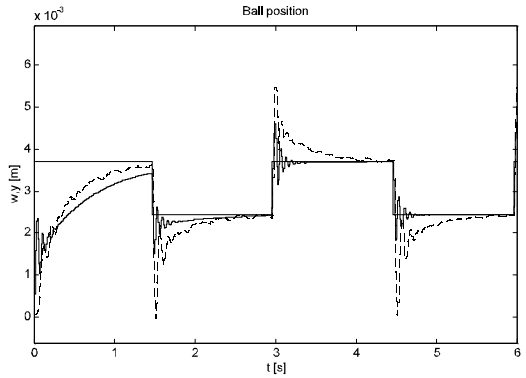

Figure 45: Simulated compensation of nonlinearity II - position/force relationship - (solid line) compared with real PID controller (dashed line) - desired and actual position.

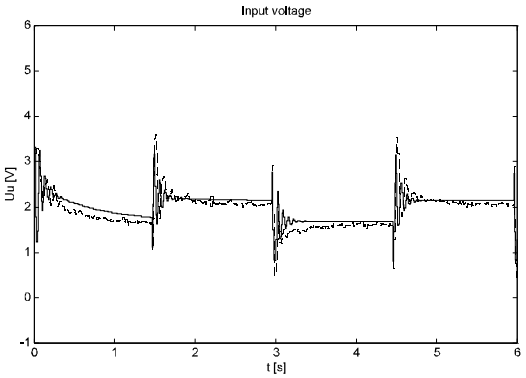

Figure 46: Simulated compensation of nonlinearity II - position/force relationship - (solid line) compared with real PID controller (dashed line) - controller output.

6. Bibliography

Astrom,K.J., B.Wittenmark: Computer-Controlled Systems: Theory and Design. Second Edition. Prentice-Hall Internation Editions, Englewood Cliffs, N.J. 1990

Brogan,W.L.: Modern Control Theory. Third Edition. Prentice-Hall, Inc., Englewood Cliffs, N.J. 1991

Kuo,B.C.: Automatic Control Systems - Sixth Edition. Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1991

Phillips,C.L., R.D.Harbor: Feedback Control Systems. Prentice-Hall, Inc., Englewood Cliffs, N.J. 1991

Nijmeijer,H.: Nonlinear Dynamical Control Systems. Springer-Verlag, 1990.

Isermann,R. : Digital Control Systems. Vol.1: Fundamentals,Deterministic Control. 2nd ed. Springer-Verlag,1989.

Isermann,R. : Digital Control Systems. Vol.2: Stochastic Control, Multivariable Control, Adaptive Control, Appl. 2nd rev.ed., Springer-Verlag, 1991.