Magnetic

Levitation System 2EM

(MLS2EM)

User's Manual

Printed by InTeCo Ltd
phone./fax: (48) 12 430-49-61, e-mail: inteco@kki.krakow.pl

Table of contents

Introduction

The *Magnetic Levitation System with 2 Electromagnets* (*MLS2EM*) is a complete (after assembling and software installation) control laboratory system ready to experiments. The is an ideal tool for demonstration of magnetic levitation phenomena. This is a classic control problem used in many practical applications such as transportation - magnetic levitated trains, using both analogue and digital solutions to maintain a metallic ball in an electromagnetic field. *MLS2EM* consists of two electromagnets, the suspended hollow steel sphere, the sphere position sensors, computer interface board and drivers, a signal conditioning unit, connecting cables, real time control toolbox and a laboratory manual. This is a single degree of freedom system for teaching of control systems; signal analysis, real-time control applications such as MATLAB. MLS is a nonlinear, open-loop unstable and time varying dynamical system. The basic principle of MLS operation is to apply the voltage to an upper electromagnet to keep a ferromagnetic object levitated. The object position is determined through a sensor. Additionally the coil current is measured to explore identification and multi loop or nonlinear control strategies. To levitate the sphere a real-time controller is required. The equilibrium stage of two forces (the gravitational and electromagnetic) has to be maintained by this controller to keep the sphere in a desired distance from the magnet. When two electromagnets are used the lower one can be used for external excitation or as contraction unit. This feature extends the MLS application and is useful in robust controllers design. The position of the sphere may be adjusted using the set-point control and the stability may be varied using the gain control. Two different diameter spheres are provided. The band-width of lead compensation may be changed and the stability and response time investigated. User-defined controllers may be tested.

1.1 Laboratory set-up

A schematic diagram of the laboratory set-up is shown in Fig. 1.

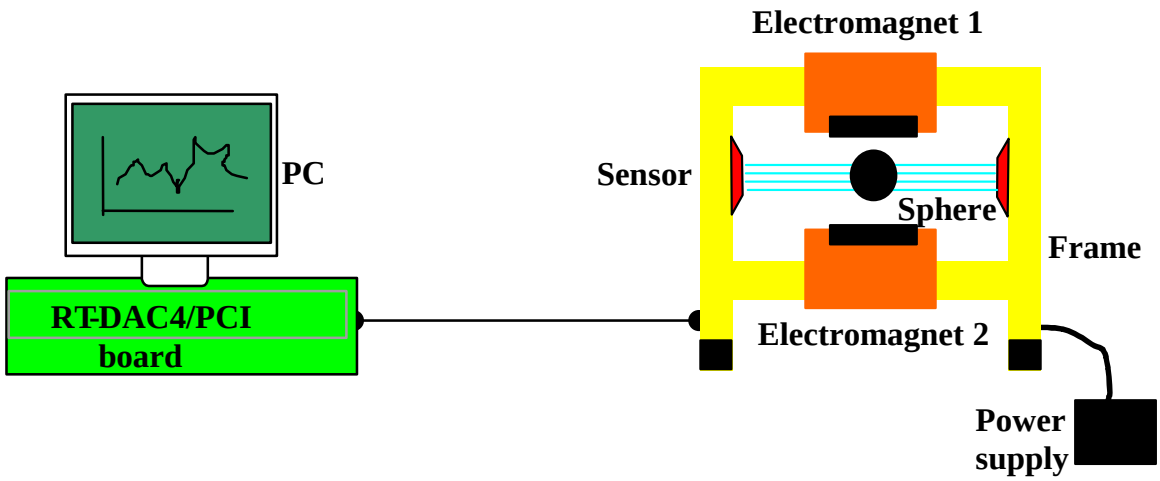


Fig. 1. MLS2EM laboratory set-up

One obtains the mechanical unit with power supply and interface to a PC and the dedicated RTDAC4/PCI I/O board configured in the Xilinx® technology. The software operates in real time under MS Windows® 2000/XP using MATLAB®/Simulink and RTW and RTWT toolboxes.
Control experiments are programmed and executed in real-time in the MATLAB/Simulink environment. Thus it is strongly recommended to a user to be familiar with the RTW and RTWT toolboxes. One has to know how to use the attached models and how to create his own models.
The control software for the MLS2EM is included in the *MLS2EM toolbox*. This toolbox uses the RTWT and RTW toolboxes from MATLAB.
MLS2EM Toolbox is a collection of M-functions, MDL-models and C-code DLL-files that extends the MATLAB environment in order to solve MLS modelling, design and control problems. The integrated software supports all phases of a control system development:

- on-line process identification,
- control system modelling, design and simulation,
- real-time implementation of control algorithms.

MLS2EM Toolbox is intended to provide a user with a variety of software tools enabling:

- on-line information flow between the process and the MATLAB environment,
- real-time control experiments using demo algorithms,
- development, simulation and application of user-defined control algorithms.

MLS2EM Toolbox is distributed on a CD-ROM. It contains software and the *MLS2EM User's Manual*. The *Installation Manual* is distributed in a printed form.

1.2 Hardware and software requirements.

Hardware
Hardware installation is described in the *Assembling* manual. It consists of:

- Two electromagnets
- Ferromagnetic objects
- Position sensor
- Current sensors
- Power interface
- RTDAC4/PCI measurement and control I/O board
- Pentium or AMD based personal computer.

Software
For development of the project and automatic building of the real-time program is required. The following software has to be properly installed on the PC:

- MS Windows 2000 or Windows XP. MATLAB/Simulink, Signal Processing Toolbox and Control Toolbox from MathWorks Inc. to develop the project.
- Real Time Workshop to generate the code.
- Real Time Windows Target toolbox.

- The *MagLev* toolbox which includes specialised drivers for MLS2EM, These drivers are responsible for communication between MATLAB and the RT-DAC4/PCI measuring and control board.
- MS Visual C++ to compile the generated code if MATLAB version 6.5 is used.

1.3 Features of MLS

- Aluminium construction
- Two ferromagnetic objects (spheres) with different weight
- Photo detector to sense the object position
- Coil current sensors
- A highly nonlinear system ideal for illustrating complex control algorithms
- None friction effects are present in the system
- The set-up is fully integrated with MATLAB®/Simulink® and operates in real-time in MS Windows® 2000/XP
- The software includes complete dynamic models.

1.4 Typical teaching applications

- System Identification
- SISO, MISO, BIBO controllers design
- Intelligent/Adaptive Control
- Frequency analysis
- Nonlinear control
- Hardware-in-the-Loop
- Real-Time control
- Closed Loop PID Control

1.5 Software installation

Insert the installation CD and proceed step by step following displayed commands.

2 ML Main Window

The user has a rapid access to all basic functions of the MLS System from the *MLS2EM* Control Window. In the Matlab Command Window type:

MLS2EM_Main

and then the *Magnetic Levitation Main* window opens (see Fig. 2).

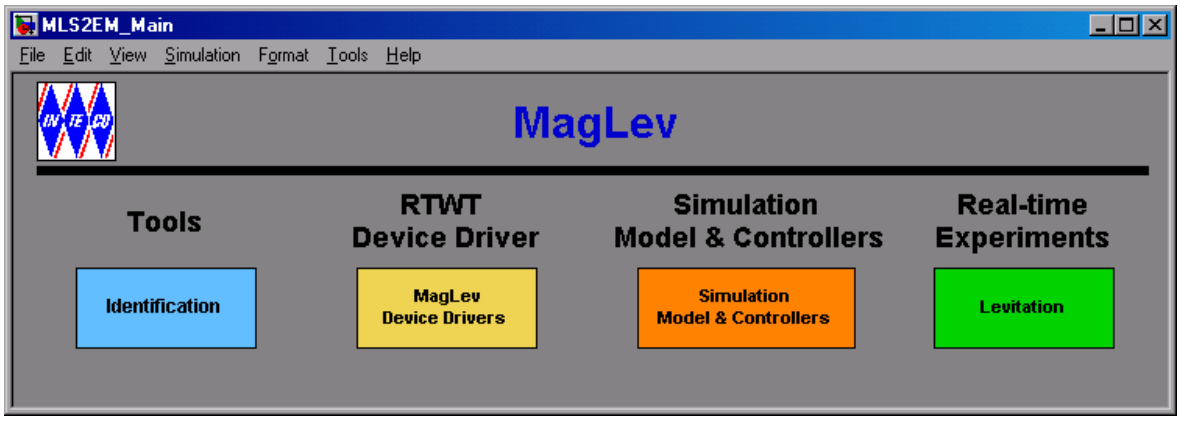


Fig. 2. The Magnetic Levitation Main window

In the MLS2EM_Main window one can find: testing tools, drivers, models and demo applications. You can see a number of pushbuttons ready to use.

The MLS2EM_Main window shown in Fig. 2 contains four groups of the menu items:

- Tools – identification
- RTWT Device Driver – MagLev device drivers
- Simulation model and controllers
- Real-time experiments – levitation

Section 2 is divided into four subsections. Under each button in the MLS2EM_Main window one can find the respective portion of software corresponding to the problem announced by the button name. These problems are described below in four consecutive subsections.

2.1 Identification

If we click the identification button the following window (see Fig. 3) opens. There are the default values of all parameters defined by the manufacturer. Nevertheless, a user is equipped with a number of identification tools. He can perform the identification procedures to verify and if necessary modify static and dynamic characteristics of MLS2EM.

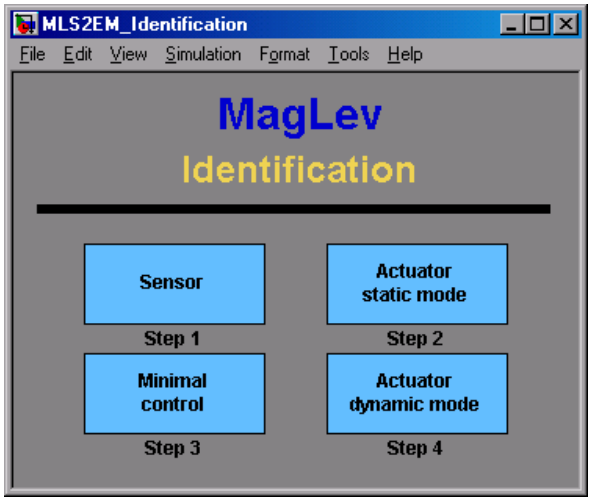


Fig. 3. The identification window

Four identification steps have been preprogrammed. They are described below.

2.1.1 Sensor

In this subsection the position sensor characteristics is identified.

If you click the Sensor button the following window opens (see Fig. 4)

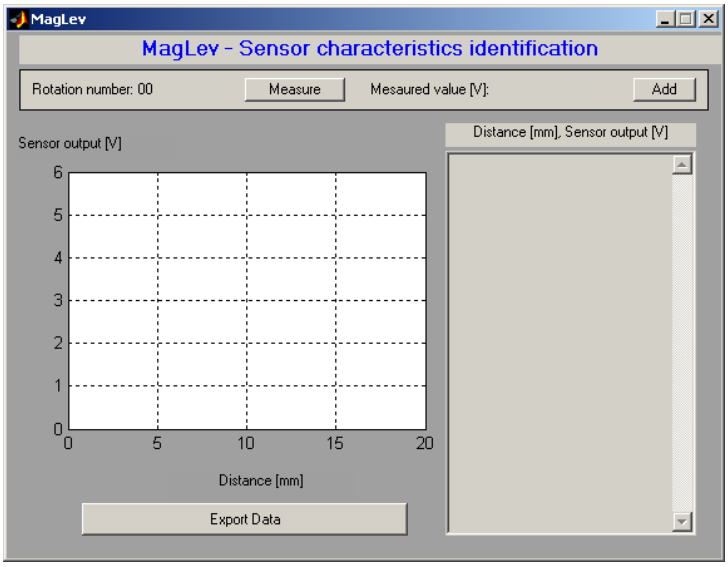


Fig. 4. Sensor signal in [V] vs. the sphere distance from the electromagnet in [mm]

The following procedure is required to identify the characteristics.

1. Screw in the screw bolt into the bottom electromagnet.
2. Screw in the black sphere and lock it by the nut. Notice, that **the sphere is fixed to the bottom electromagnet and frame recpectively!**
3. Turn round the screw so the sphere is in touch with the bottom of the top electromagnet.
4. Make sure that the power is on.
5. Start the measuring and registration procedure. It consists of the following steps:
6. Push the *Measure* button – the voltage from the position sensor is stored and displayed as *Measured value [V]*. One can correct this value by measuring it again.
    - Push the *Add* button – the measured value is added to the list. A rotation number value is automatically enlarged by one (see Fig. 5).

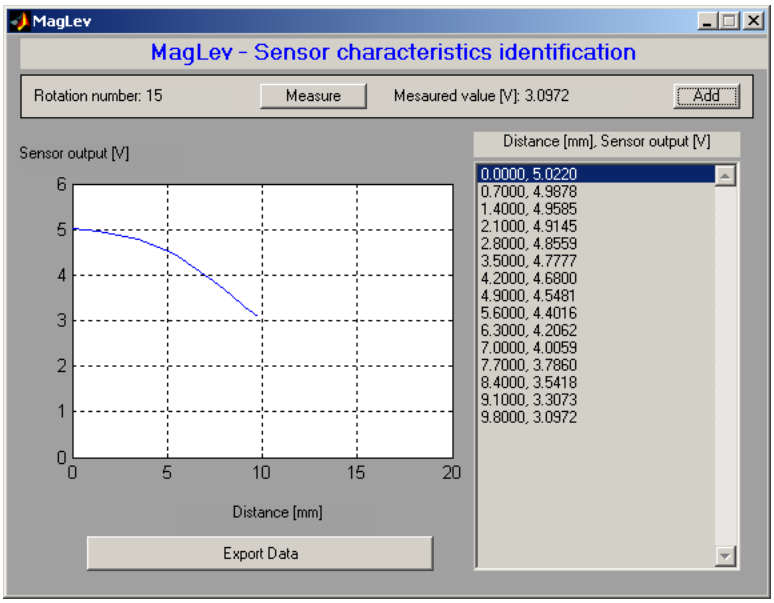


Fig. 5. Characteristics of the sphere position sensor

- Manually make one full rotation of the screw.
- Repeat three last steps so many times as none change in the voltage vs. position characteristics is observed.

7. Push the *Export Data* button – the data are written to the disc (see Fig. 6). Data are stored in the *MLS2EM_Sensor.mat* file as the *SensorData* structure with the following signals: *Distance_mm*, *Distance_m* and *Sensor_V*.

In the Simulink real-time models the above characteristics is used as a Look-Up-Table model. The block named *Position scaling* is located inside the device driver block of MLS2EM (see Fig. 7). Notice, that the characteristics shows meters vs. Volts. In Fig. 6

there were shown Volts vs. meters. It is obvious that we require the inverse characteristics because we need to define the output as the position in meters.

Notice that the characteristics can be different due to manufacturing process and light conditions.

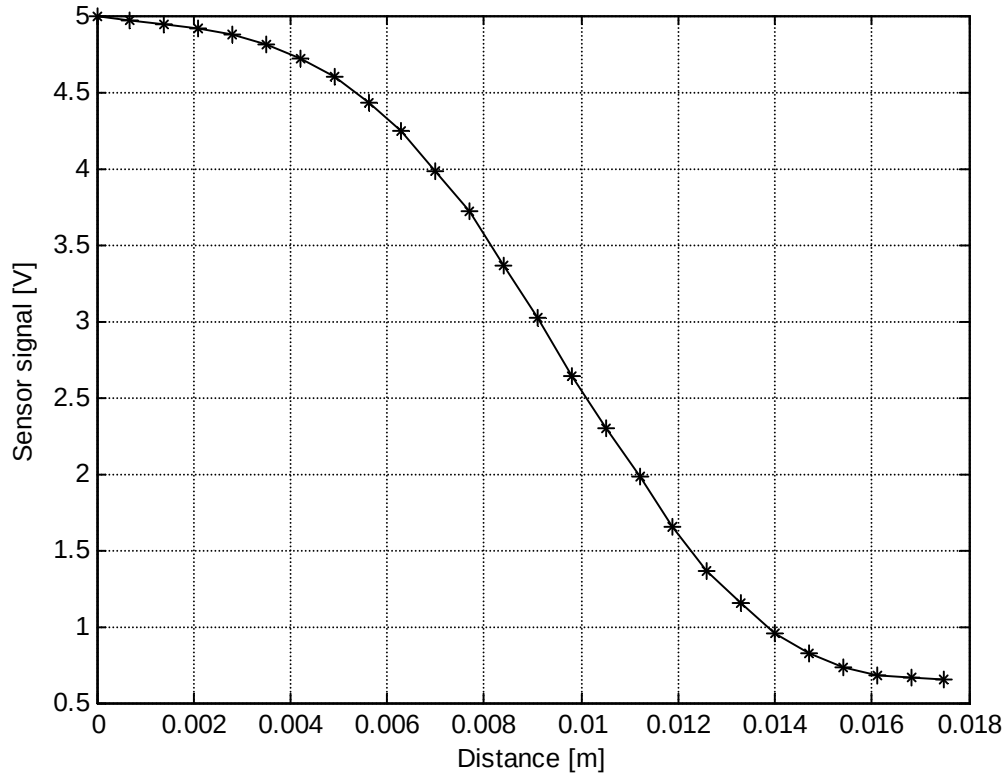


Fig. 6. The sensor characteristics after being measured and exported to the disc

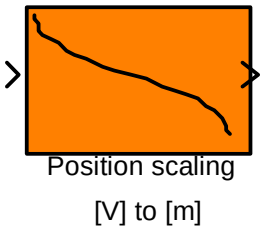


Fig. 7. The Simulink Look Up Table model representing the position sensor characteristics

If we click this block the window shown in Fig. 8 opens. Any time you like to modify the sensor characteristics you can introduce new data related to the voltage measured by the sensor. The voltage corresponds to the distance of the sphere set by a user while the identification procedure is performed. The sensor characteristics is loaded from the *MLS2EM_sensor.mat* file which has been created during the identification procedure. If the curve of the *Position scaling* block is not visible please load the file with data.

The sensor characteristics can be approximated by a polynomial of a given order. For example, we can use a fifth order polynomial.

$$P(x) = p_5 x^5 + \ldots + p_0$$

$p_5 = -25697073504.59$, $p_4 = 1245050011.25$, $p_3 = -18773635.92$, $p_2 = 79330.24$, $p_1 = -150.21$ and $p_0 = 5.015$.

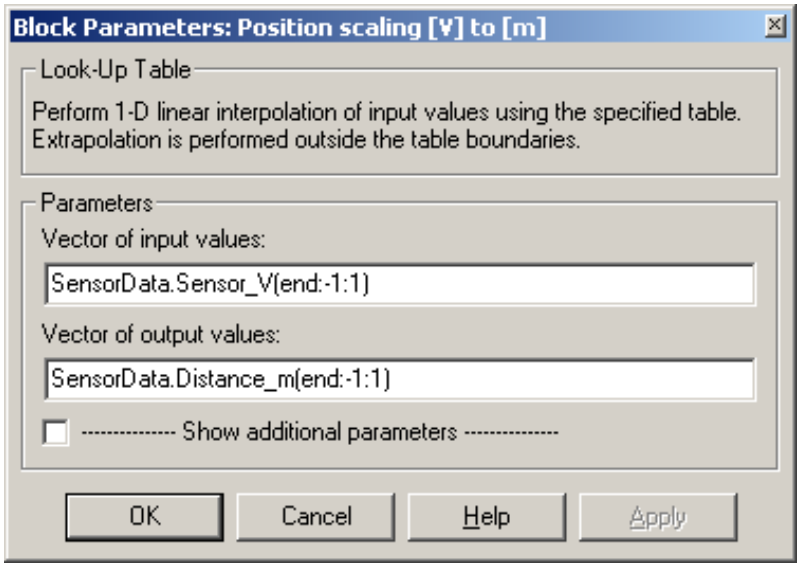


Fig. 8. Look-Up Table to be fulfilled with vectors of input and output values

The approximated polynomial (the red line) is shown in Fig. 9. The polynomial approximation will be not used in this manual due to the fact that the entire model is built in Simulink. Therefore we recommend to model the characteristics as a Look-Up Table block (see Fig. 7 and Fig. 8).

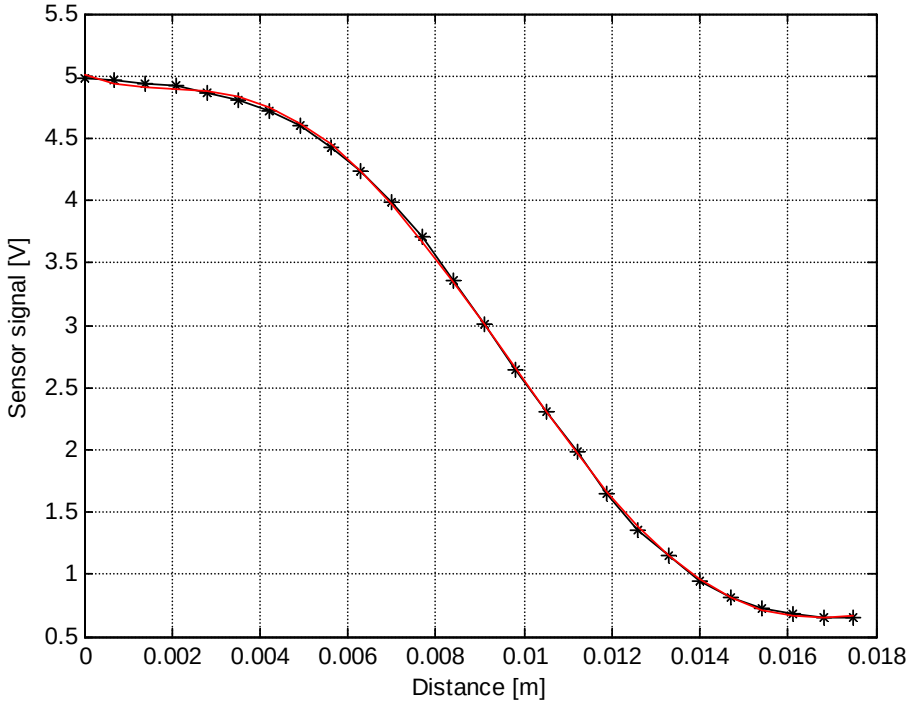


Fig. 9. The sensor characteristics approximated by the fifth order polynomial

2.1.2 Actuator static mode

In this subsection we examine static features of the actuator i.e. the electromagnet. Notice, that **the sphere is not present**!

Click the *Actuator static mode* button and the window shown in Fig. 10 opens.

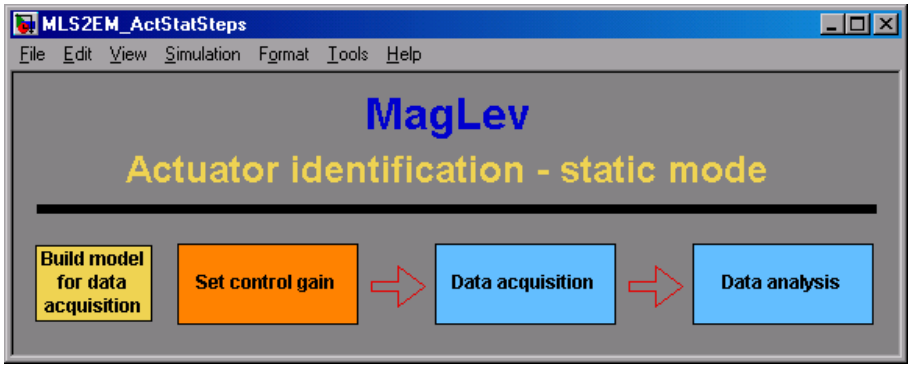


Fig. 10. Identification window of a static current/voltage characteristics

Now, we can perform button by button the operations depicted in Fig. 10. We begin from the *Build model for data acquisition* button. The window of the real-time task shown in Fig. 11 opens and the RTW build command is executed (the executable code is created).

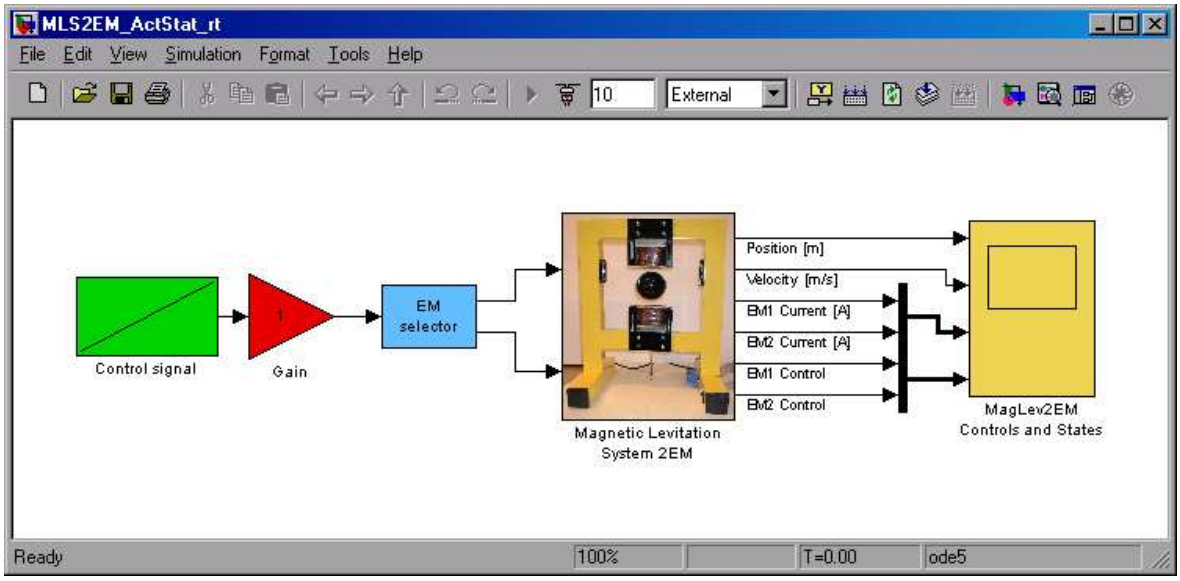


Fig. 11. Real-time model built to examine the current in the electromagnetic coil

Click the *Set control gain* button. It results in activation of the model window and the following message is displayed (see Fig. 12):

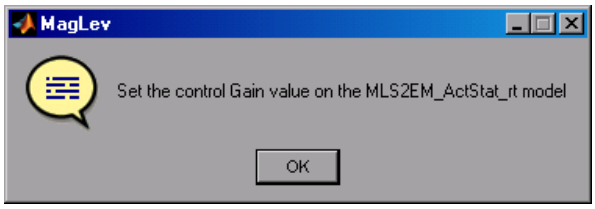


Fig. 12. Message – Set the "Control Gain"

In Fig. 11 one can notice the *Control signal* block. In fact the control signal increases linearly. We can modify the slope of this signal changing the *Control Gain* value.

Click the *Data acquisition* button. Within 10 seconds data are acquired and stored in the workspace.

Click the *Data analysis* button. The collected values of the coil current are displayed in Fig. 13.

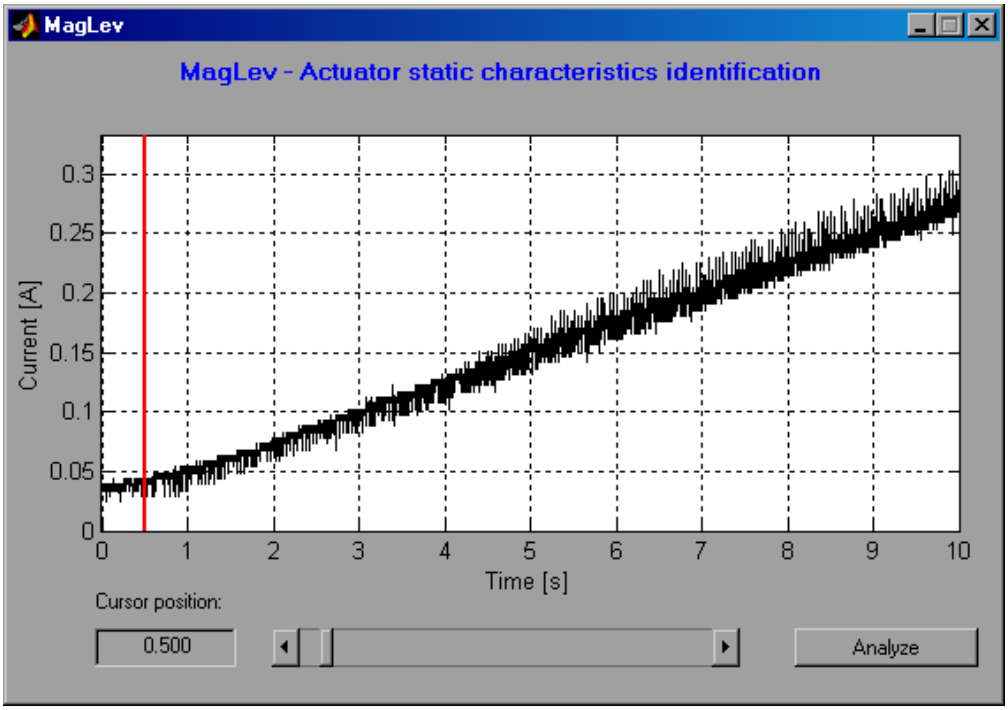


Fig. 13. Current in the electromagnetic coil

The characteristics is linear except a small interval at the beginning. We can locate the cursor at the point where a new line slope starts (see the red line in the picture). We can move the cursor in two ways: by writing down a value into the edition window or by drugging the slider. In this way the current characteristics is prepared to be analyzed in the next step. The line is divided into two intervals: the first – from the beginning of measurements to the cursor and the second – from the cursor to the end of measurements.

After setting the cursor position, consequently, click the *Analyze* button. The following message (see Fig. 14) appears. We obtain the dead zone values corresponding to the control and current. The constants *a* and *b* of the linear part are the parameters of the line equation: $i(u) = a\,u + b$.

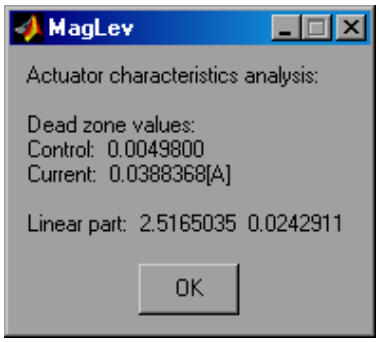


Fig. 14. Coefficients of the actuator characteristics

These parameters, namely: $u_{MIN} = 0.00498$, $x_{3MIN} = 0.03884$, $k_i = 2.5165$ and $c_i = 0.0243$ are going to be used in the simulation model in section 2.3.1 (see the differential equations parameters).

To obtain a family of static characteristics for linear controls with different slopes we repeat the following experiment. We apply a PWM voltage signal in the time interval from 0 to 10 s. The PWM duty cycles for the subsequent ten experiments are varying linearly in the ranges: [0, 0.1], [0, 0.2], ..., [0, 1.0] (see Fig. 16).

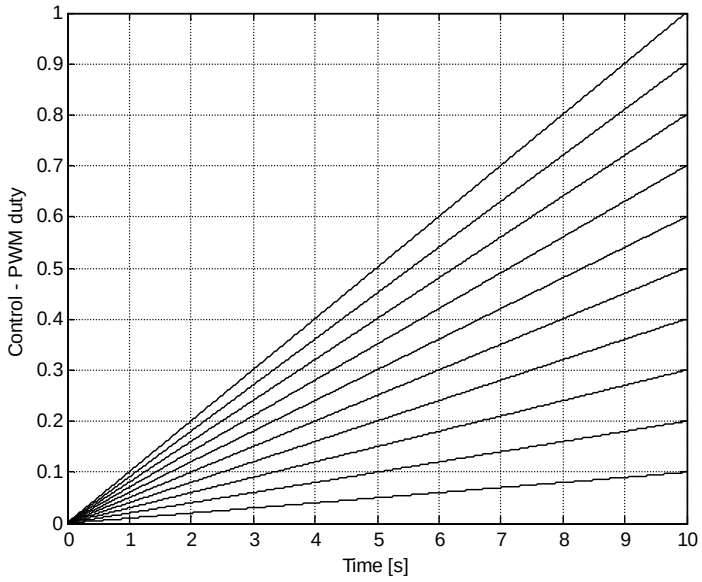


Fig. 15. Family of the input (PWM) characteristics

Consequently, we obtain diagrams of the currents corresponding to ten experiment (see Fig. 16). Each characteristics is approximated by a polynomial of the first order. Finally the entire current vs. PWM duty cycle relation is depicted (black points) in Fig. 17. The red line represents the linear approximation of measurements. We obtain the following numerical values of linear characteristics:

$$i(u) = k_i u + c\,; \quad a = 2.60798876298869, \quad b = -0.01077522109792.$$

The constant c is obtained for $u = 0$. The family of linear characteristics is used to obtain the coefficients k_i vs. control u.

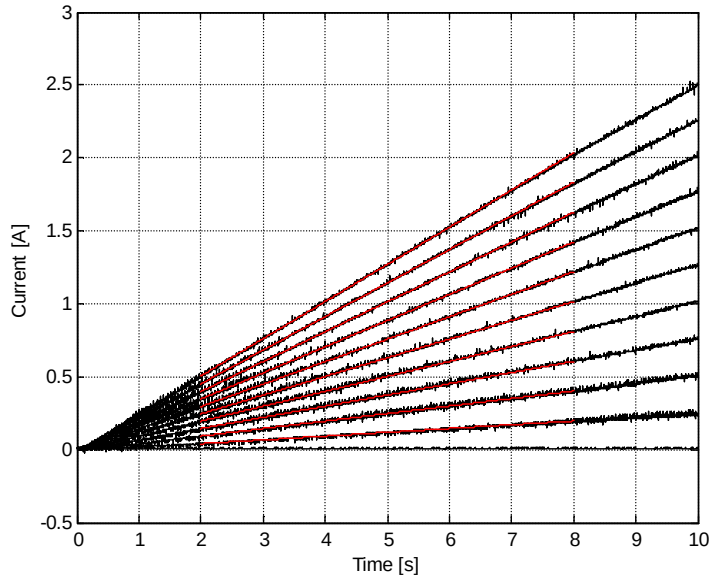


Fig. 16. Family of the output (current) characteristics

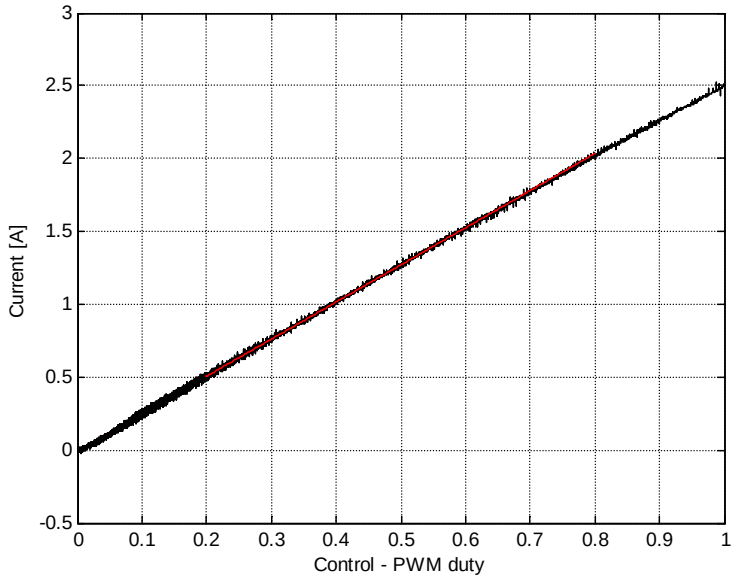


Fig. 17. Current vs. PWM duty cycle

Using the *EM Selector* block select the electromagnet to be controlled and repeat the whole procedure for both electromagnets.

2.1.3 Minimal control

In this subsection we examine the minimal control to cause a forced motion of the sphere from the bottom electromagnet toward the upper electromagnet against the gravity force. Notice, that in this experiment **the sphere is not levitating**! It is kept nearby the electromagnet in the operating range.

Click the *Minimal control* button and the window shown in Fig. 18 opens.

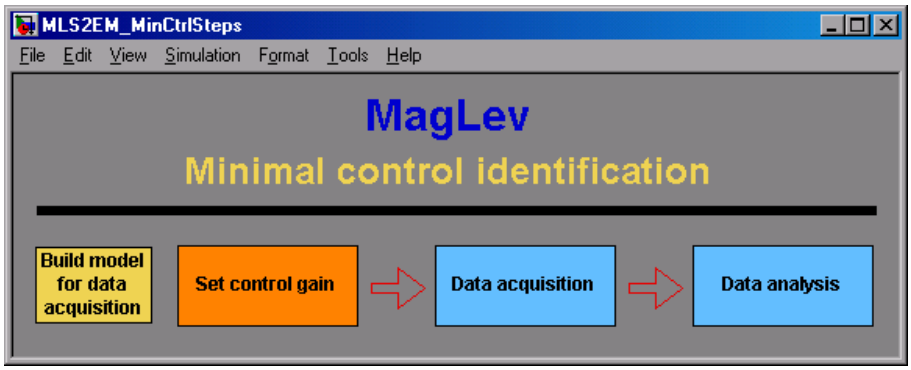


Fig. 18. Window to identify the minimal control vs. distance (between the sphere and electromagnet)

Now, we proceed button by button the operations depicted in Fig. 18 similarly to the procedure described in the previous subsection. We begin from the *Build model for data acquisition* button. The window of the real-time task shown in Fig. 19 opens.

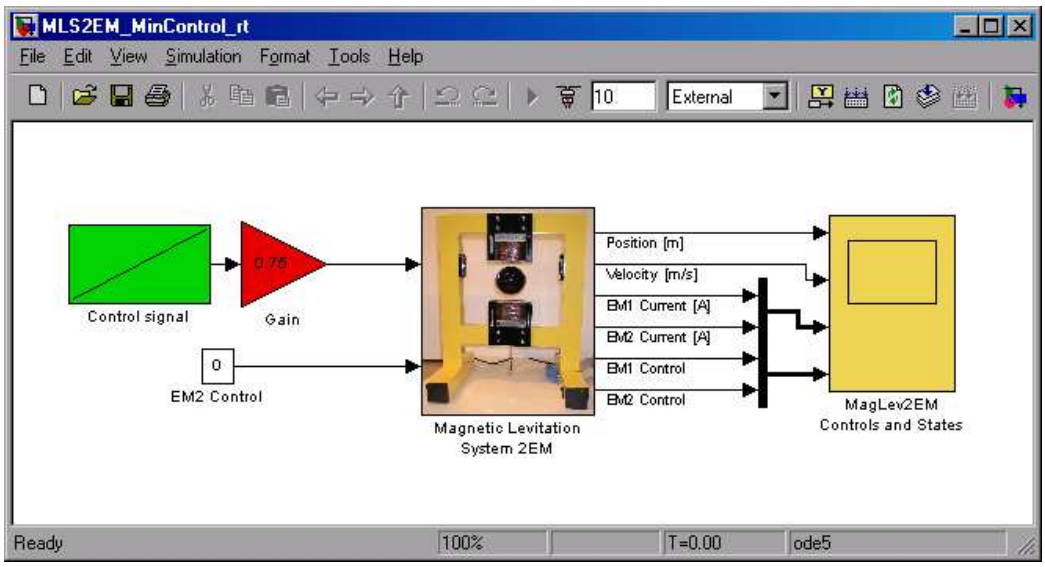


Fig. 19. Real-time model built to examine the minimal electromagnetic force

Click the *Set control gain* button. It results in activation of the model window and the following message is displayed (see Fig. 20).

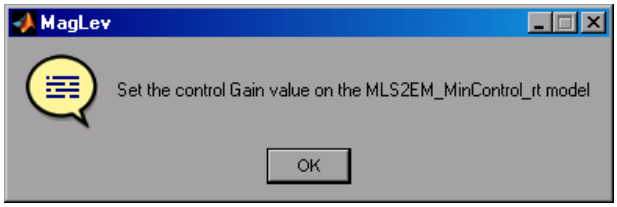


Fig. 20. Message – set the "Control Gain"

It means that we can set a duty cycle of the control PWM signal. The sphere is located on the support and the experiment starts. Click the *Data acquisition* button. A forced motion of the ball toward the electromagnet begins.

Click the *Data analysis* button. The collected values of the ball position are displayed in Fig. 21.

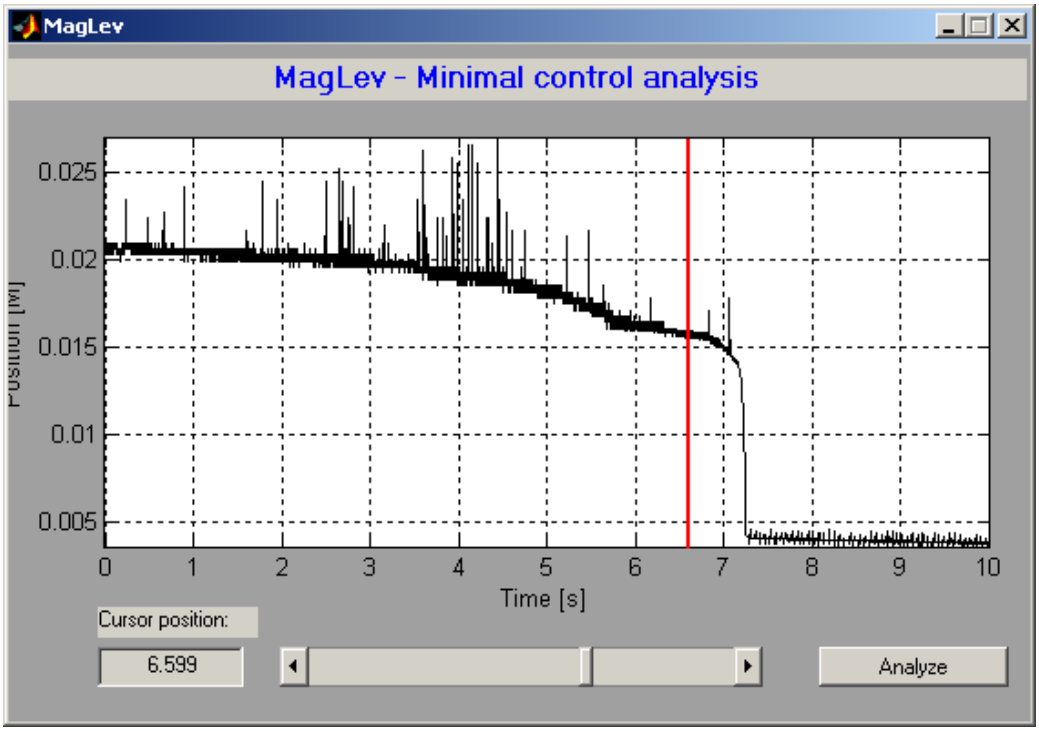


Fig. 21. The sphere motion

The sphere motion is visible. We can locate the cursor at the point slightly before a position jump occurs (takes place) (see the red line in the picture). We can move the cursor in two ways: by writing down a value into the edition window or by drugging the slider. In this way the acquired data are prepared to be analyzed in the next step.

After setting the cursor position, consequently, click the *Analyze* button. The following message (see Fig. 22) appears. This information means that the sphere located 15.82 mm from the electromagnet begins to move toward it when the PWM control over-crosses the 0.49485 duty cycle value.

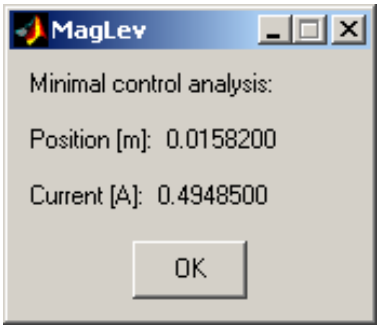


Fig. 22. Message of the experiment results

2.1.4 Actuator dynamic mode

In this subsection we examine dynamic features of the actuator i.e. the electromagnet. It means that the moving sphere generates an electromotive force (EMF). EMF diminishes the current in the electromagnet coil. Click the *Actuator static mode* button and the window shown in Fig. 23 opens.

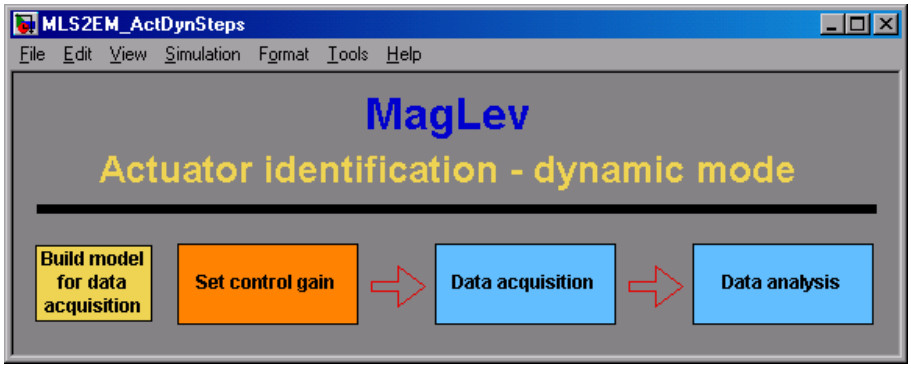


Fig. 23. Identification window of a dynamic current/voltage characteristics

A user should perform three experiments: without the sphere (*Without ball*), with the sphere located on the bottom electromagnet (*Ball on EM*) and with the sphere fixed to the rigid screw (*Ball fixed*).

We begin from the *Build model for data acquisition* button. The window of the real-time task shown in Fig. 24 opens. We have to set the control gain. If we are going to modify the control magnitude then we set the default gain to 1 and the subsequent duty cycles to: 0.25, 0.5, 0.75 and 1. Click the *Data acquisition* button and save data under a given file name.

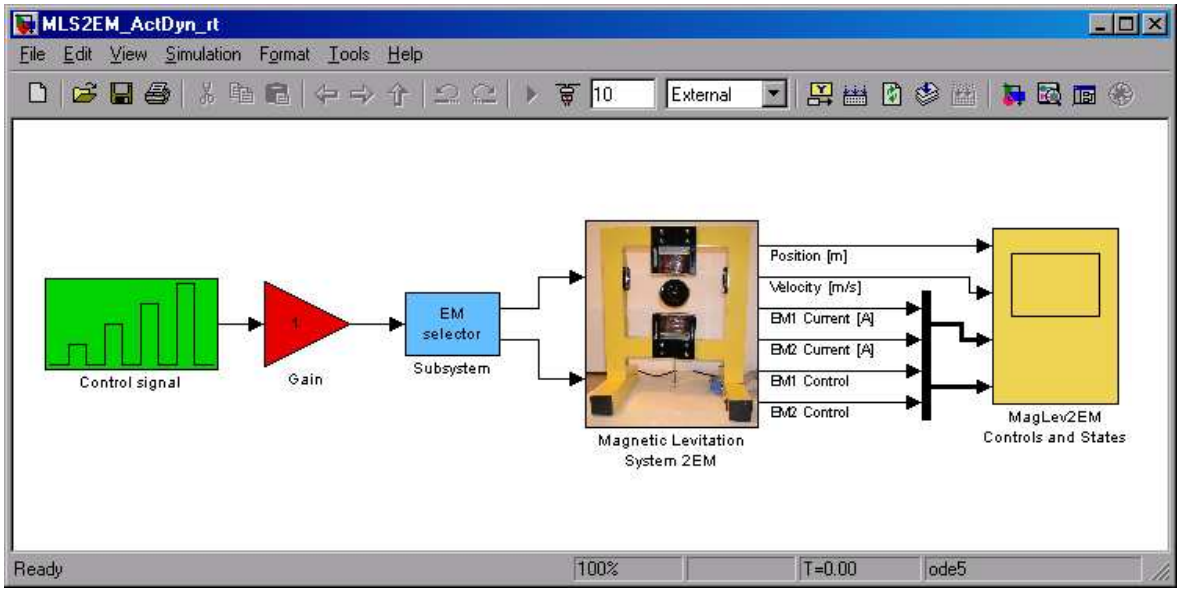


Fig. 24. Real-time model built to examine EMF influence on the coil current

Click the *Data analysis* button. It calls the *mls2em_find_curr_dyn.m* file. The following window opens (see Fig. 25). The parameters optimization procedure starts. The optimization routine is based on the *mls2em_current_m.mdl* model.

When *mls2em_find_curr_dyn.m* runs the optimization function *fminsearch* is executed. *Fminsearch* uses the *mls2em_opt_current.m* file.

The k_i and f_i parameters are iteratively changed during the optimization procedure. The current curve is fitted four times. This is due to the control signal form.

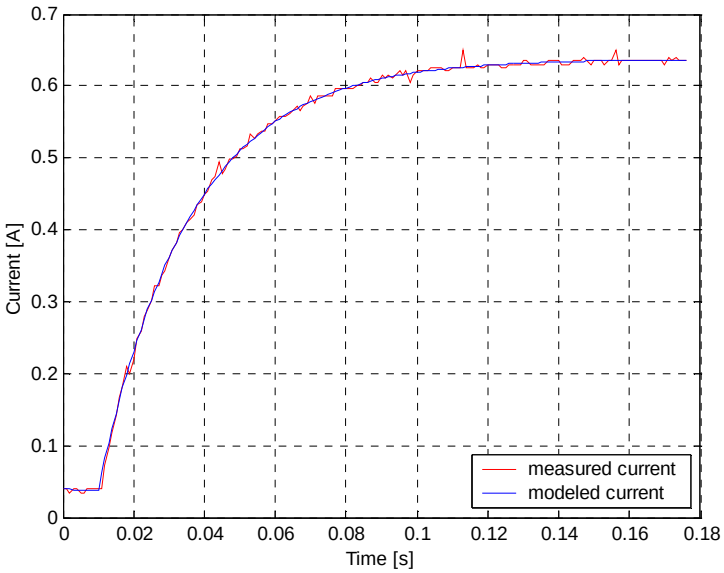


Fig. 25. Current curve – the fitting result of the optimization procedure

Finally the information about the mean values is displayed (see Fig. 26). The advanced user can use the functions code to perform a detailed analysis.

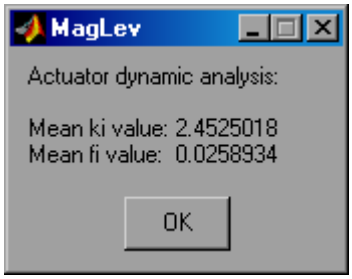


Fig. 26. Optimization results

Using the *EM Selector* block select the electromagnet to be controlled and repeat the whole procedure for both electromagnets.

2.2 MagLev device drivers

The driver is a software go-between for the real-time MATLAB environment and the RT-DAC4/PCI acquisition board. The control and measurements are driven. Click the *RTWT Device Drivers* button in the *Magnetic Levitation Main* window. The following window opens (see Fig. 27).

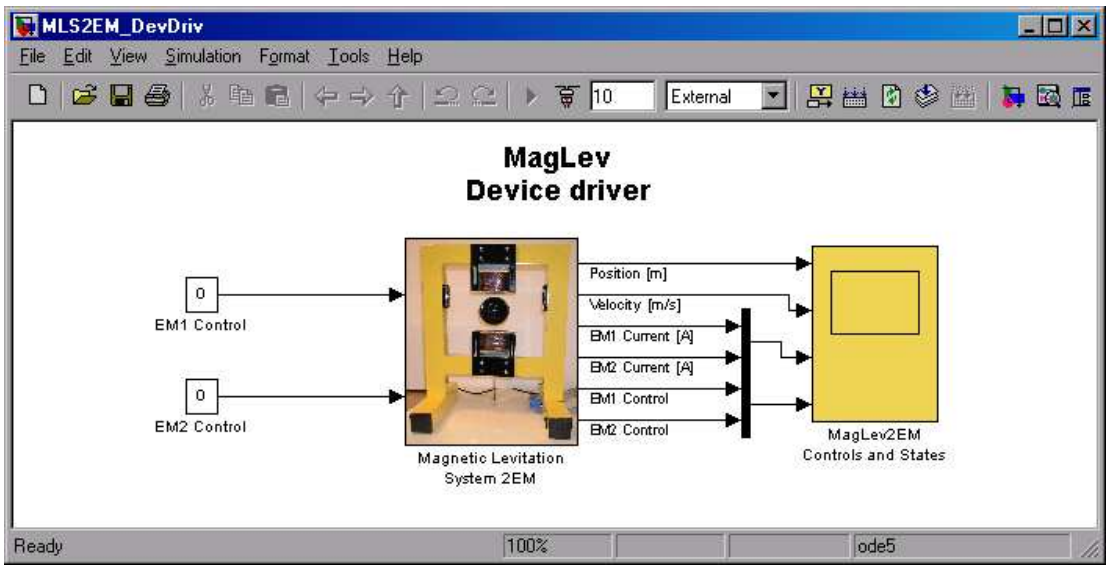


Fig. 27. RTWT MagLev device driver window

Notice that the scope block writes data to the *MLS2EMExpData* variable defined as a structure with time. The structure consists of the following signals: Position [m], Velocity [m/s], EM1 Current [A] and EM2 Current [A], EM1 Control [PWM duty 0÷1] and EM2 Control [PWM duty 0÷1]. The interior of the *Magnetic Levitation System 2EM* block (it means the interior of the driver block) is shown in Fig. 28.

In fact there are two drivers: MLS2EM_AnalogInputs and MLS2EM_PWM. There are also two characteristics: the ball position [m] vs. the position sensor voltage [V] and the coil current vs. the current sensor voltage [V]. The second one should be individually identified for the appropriate electromagnet. The driver uses functions, which communicates directly with logic stored at the RT-DAC4/PCI board. When one wants to build his own application he can copy this driver to a new model.

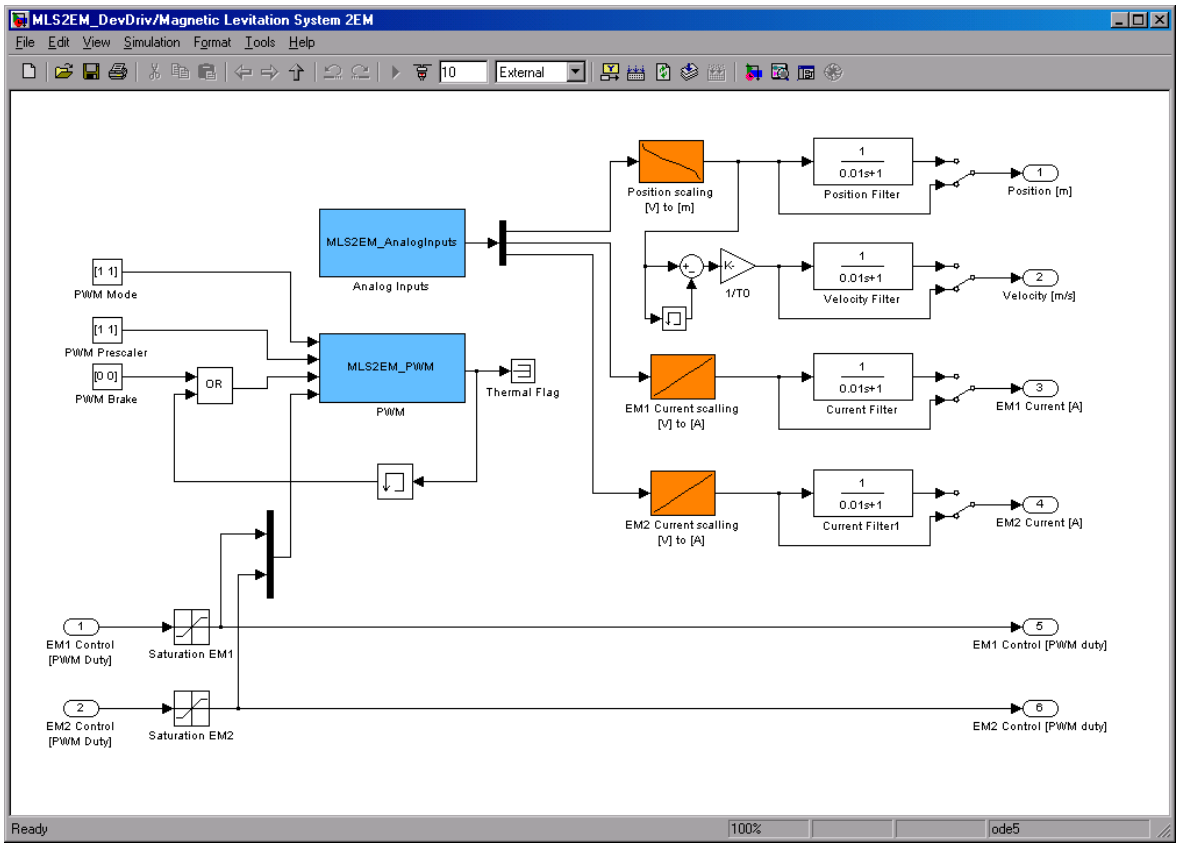


Fig. 28. Interior of the driver block

Do not introduce any changes inside the original driver. They can be introduced only inside its copy!!! Make a copy of the installation CD.

The Simulink Look-Up-Table model named *Position scaling* (see Fig. 7) representing the position sensor characteristics has been already described. Now let us present the second Simulink Look-Up-Table model named *Current scaling* (see Fig. 29).

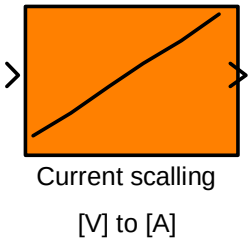


Fig. 29. The Simulink Look-Up-Table model representing the current sensor characteristics

To build the above characteristics it is necessary to measure the current of the electromagnet coil. The algorithm in the computer is the source of the desired value of the control in the form of the voltage PWM signal. This PWM is the input voltage signal transferred to the LMD18200 chip of the power interface. Due to a high frequency of the

PWM signal the measured current values correspond to the average current value in the coil. This characteristics has been built by the manufacturer. It is not recommended to repeat measurements by a user because to do so one must unsolder the input wires of the electromagnet. On the basis of the data given in the table below one can generate his own characteristics. For a fixed PWM frequency and a variable duty cycle the coil amperage is measured. The measured data are given below in the table.

PWM duty cycle	amperage [A]	voltage [V]
0	0	0.374811
0.1	0.25	0.262899
0.2	0.51	0.510896
0.3	0.77	0.752465
0.4	1.02	0.993620
0.5	1.28	1.229133
0.6	1.52	1.459294
0.7	1.74	1.651424
0.8	1.99	1.875539
0.9	2.21	2.076814
1	2.43	2.269865

The current [A] vs. voltage [V] characteristics is shown in Fig. 30.

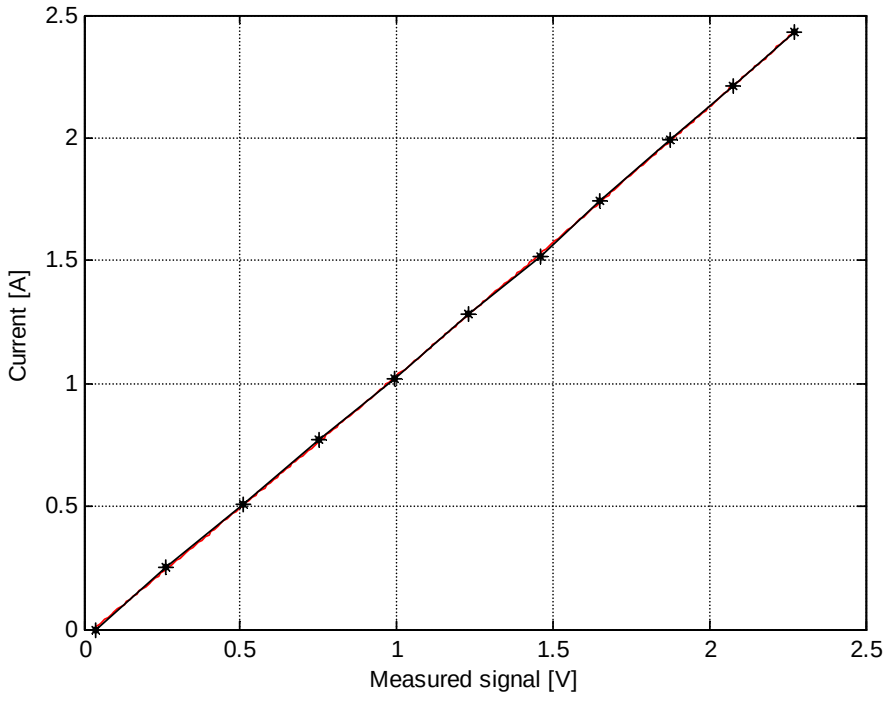


Fig. 30. Current vs. voltage characteristics approximated by the red curve

The characteristic can be approximated by a polynomial of the second order:

$$I(U) = a_2 U^2 + a_1 U + a_0$$

where:

I – current,

U – voltage from the A/D converter

a_0, a_1, a_2 - identified parameters of the polynomial

$a_2 =$ **0.0168**

$a_1 =$ **1.0451**

$a_0 =$ **–0.0317**

2.3 Simulation Model & Controllers

Click the *Simulation Model & Controllers* button in the *Magnetic Levitation Main* window. The following window opens (see Fig. 31).

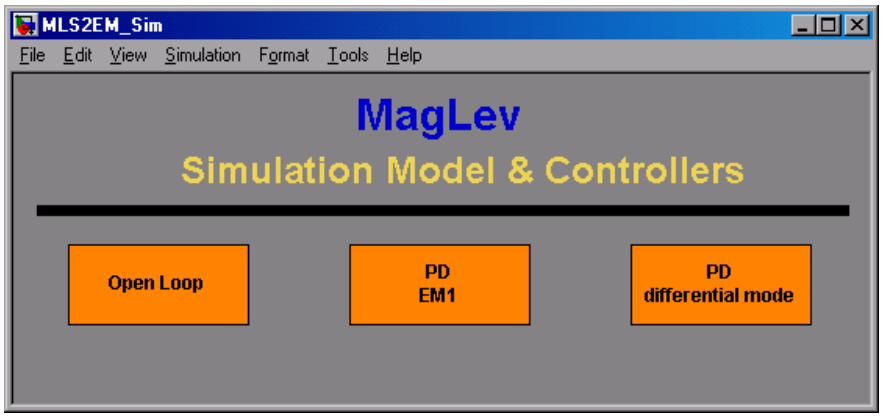


Fig. 31. Simulation Model & Controllers window

2.3.1 Open Loop

- **Simulink model**

Next, you can click the first *Open Loop* button. The following window opens (see Fig. 32). Notice that the scope block writes data to the *MLS2EMSimData* variable defined as a structure with time. The structure consists of the following signals: Position [m], Velocity [m/s], Currents [A], Controls [PWM duty 0÷1].

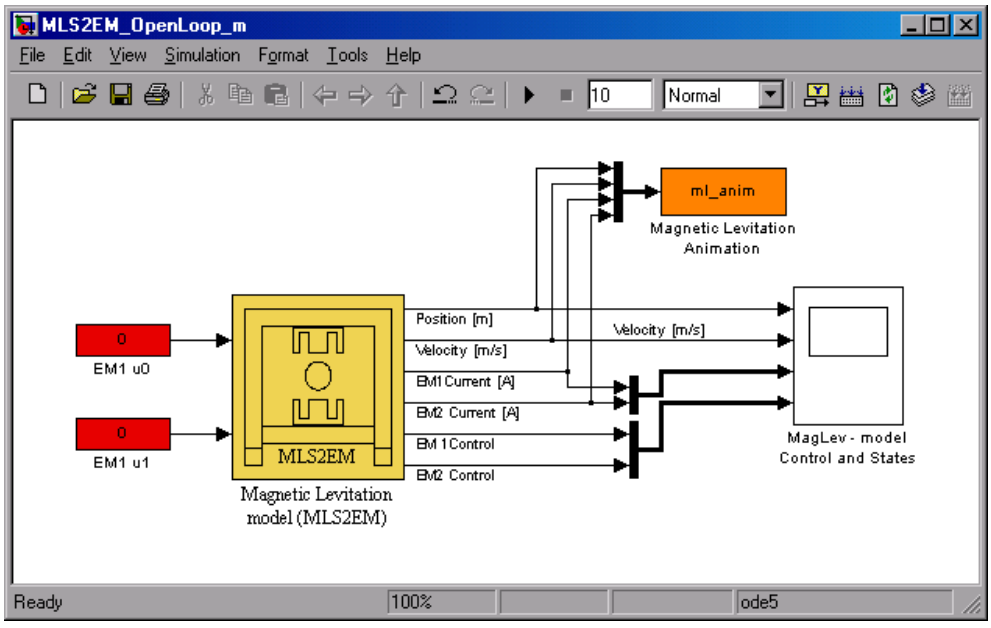


Fig. 32. Open-loop simulation

If you click the *Magnetic Levitation model* block the following mask opens (see Fig. 33).

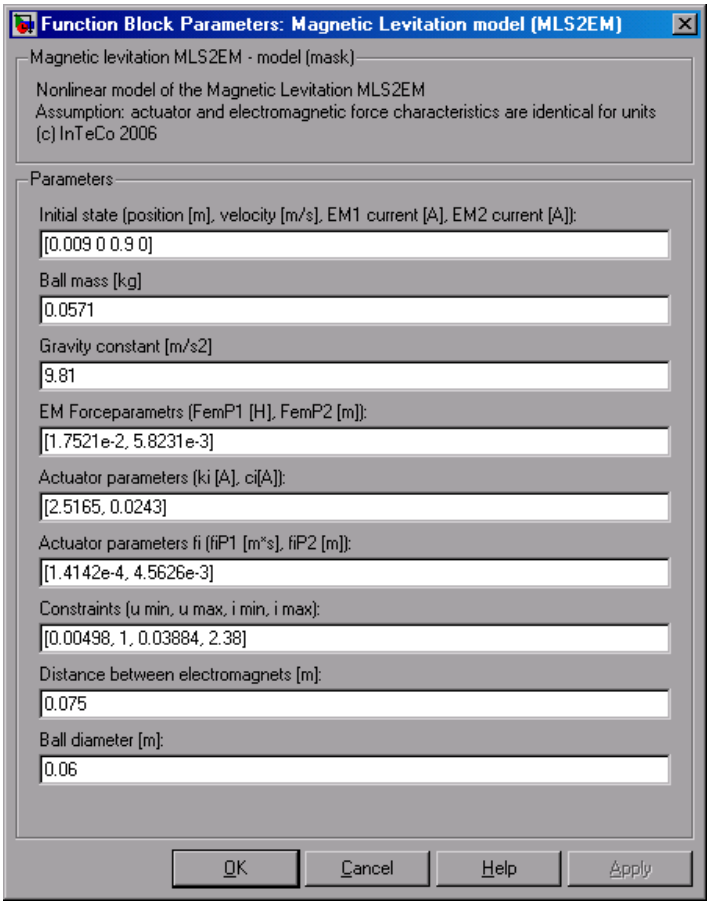


Fig. 33. Mask of the Magnetic Levitation model MLS2EM

In Fig. 32 enter into the *File* option and choose *Look under mask*. The interior of the *Magnetic Levitation model (MLS2EM)* block shown in Fig. 34 opens.

Please note that we assume that the actuator and electromagnetic force characteristics are the same for both units.

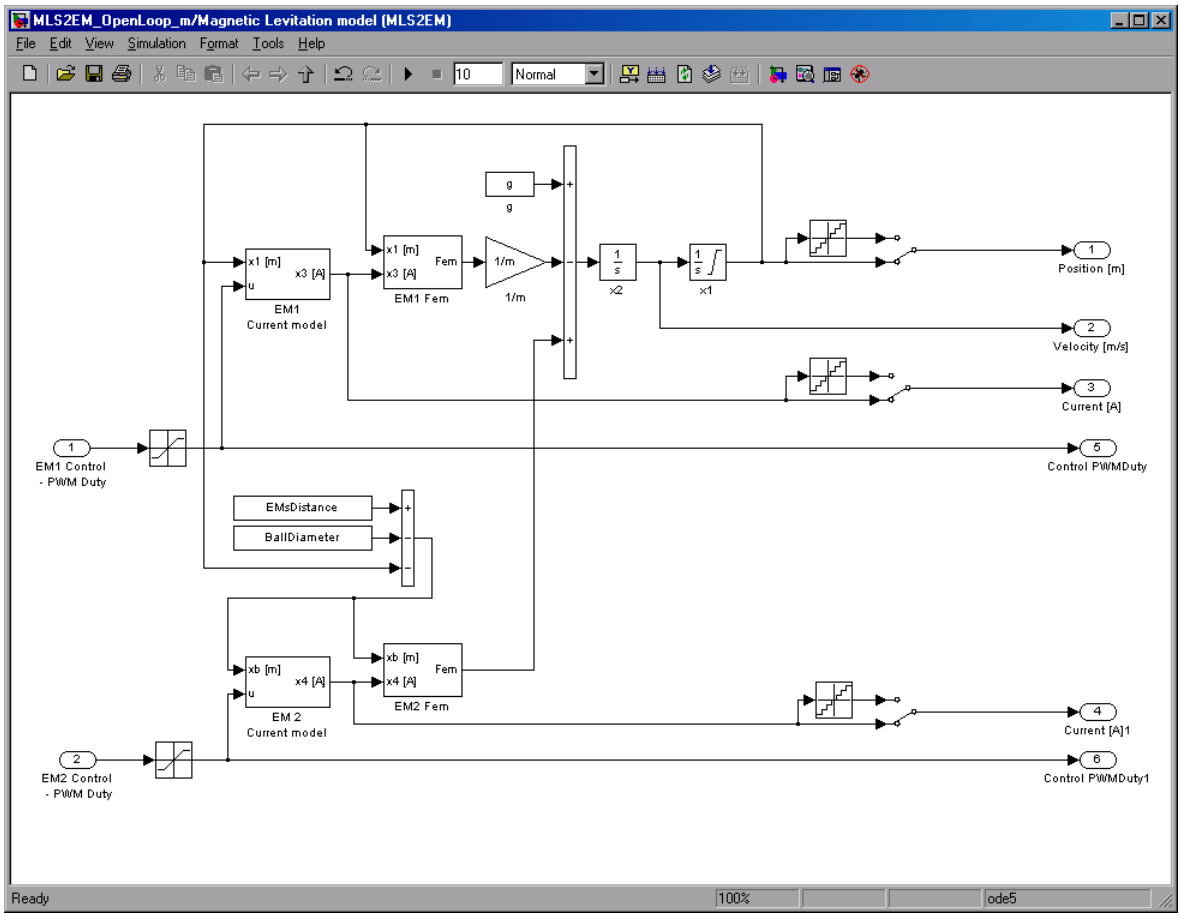


Fig. 34. Interior of the MLS2EM model

Notice two integrator blocks in Fig. 34. In fact we deal with third order dynamical system. The third integrator related to the coil current is visible in Fig. 35.

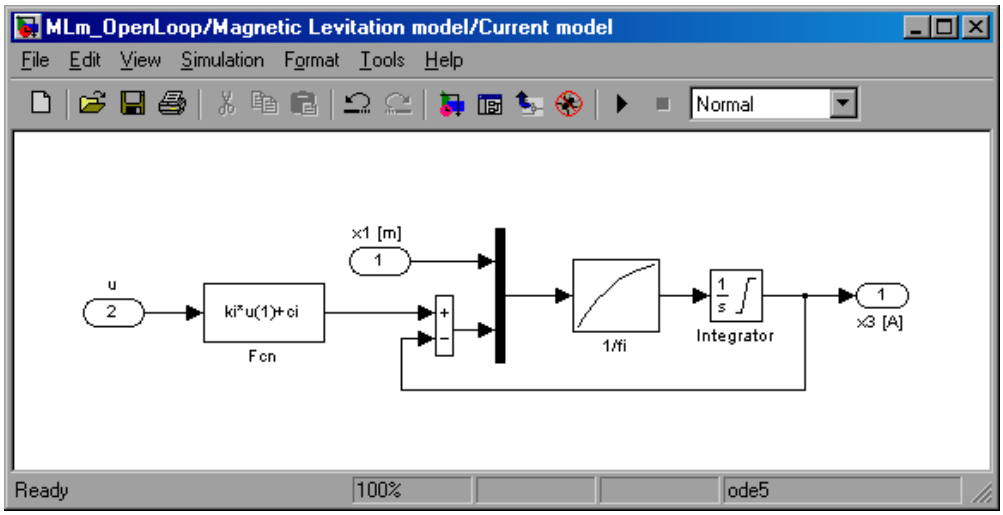


Fig. 35. Interior of the *Current model* block

The Simulink model is also equipped with the animation block. When a simulation starts the following window opens (see Fig. 36). The animation screen is updated in every sample

time. All state variables: the ball position and velocity, and also the coil currents are animated.

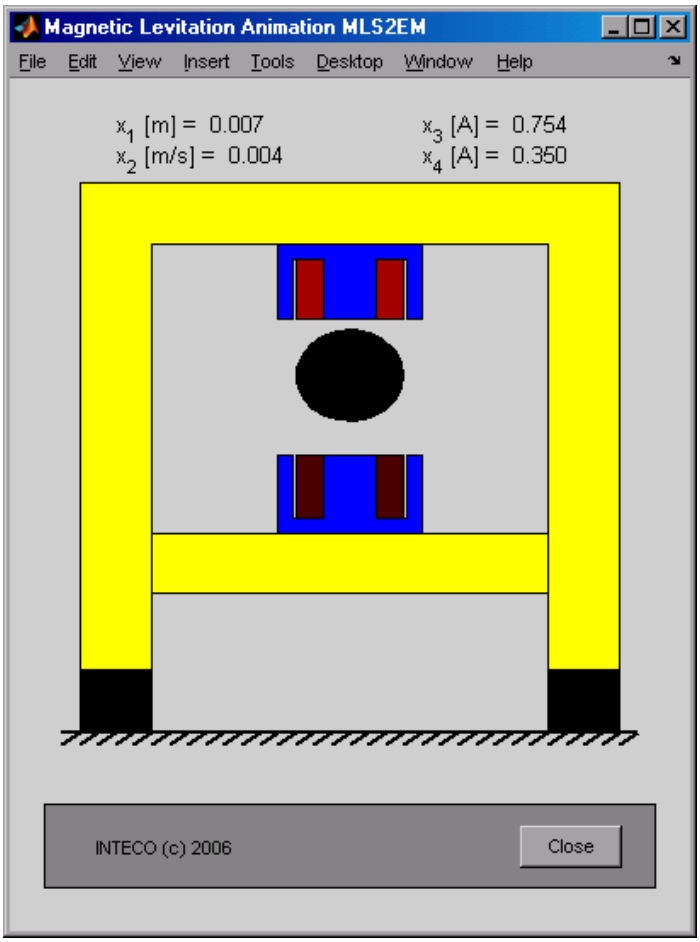


Fig. 36. MLS2EM animation

- **Mathematical model**

The schematic diagram of the MLS2EM system is shown in Fig. 37. Two electromagnetic forces and gravity force act on the ferromagnetic sphere located between electromagnets. The lower electromagnet can be used for external force excitation or as additional force to the gravity force.

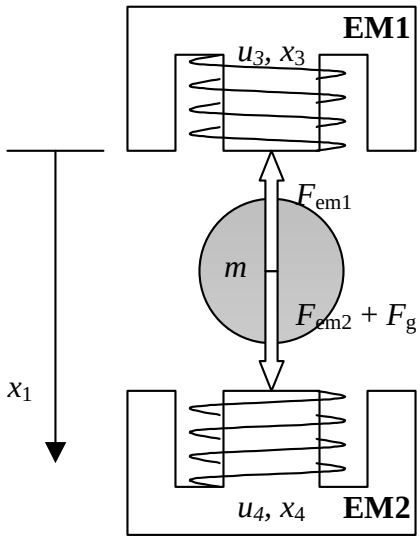


Fig. 37. MLS2EM diagram

The Simulink model is consistent with the following nonlinear mathematical model

$$\dot{x}_1 = x_2$$
$$\dot{x}_2 = -\frac{F_{em1}}{m} + g + \frac{F_{em2}}{m}$$
$$\dot{x}_3 = \frac{1}{f_i(x_1)}\left(k_i u_1 + c_i - x_3\right)$$
$$\dot{x}_4 = \frac{1}{f_i(x_d - x_1)}\left(k_i u_2 + c_i - x_4\right)$$

where:

$$F_{em1} = x_3^2 \frac{F_{emP1}}{F_{emP2}} \exp(-\frac{x_1}{F_{emP2}}) \ , \ F_{em2} = x_4^2 \frac{F_{emP1}}{F_{emP2}} \exp(-\frac{x_d - x_1}{F_{emP2}})$$

$$f_i(x_1) = \frac{f_{iP1}}{f_{iP2}} \exp(-\frac{x_1}{f_{iP2}}) \text{ for both actuators}$$

where:

$$x_1 \in [0,\ 0.016], \quad x_2 \in \Re, \quad x_3 \in [i_{MIN},\ 2.38],\ x_4 \in [i_{MIN},\ 2.38]$$

$u_1 \in [u_{MIN},\ 1],\ u_2 \in [u_{MIN},\ 1]$

The parameters of the above equations are given in the table below

Parameters	**Values**	**Units**
m	0.0571 (big ball)	[kg]
g	9.81	[m/s^2]
F_{em1}, F_{em2}	functions of x_1 and x_3	[N]
F_{emP1}	$1.7521 \cdot 10^{-2}$	[H]
F_{emP2}	$5.8231 \cdot 10^{-3}$	[m]
$f_i(x_1)$	function of x_1	[1/s]
f_{iP1}	$1.4142 \cdot 10^{-4}$	[m·s]
f_{iP2}	$4.5626 \cdot 10^{-3}$	[m]
c_i	0.0243	[A]
k_i	2.5165	[A]
x_d	distance between electromagnets minus ball diameter (this parameter is modified by the user)	[m]
i_{MIN}	0.03884	[A]
u_{MIN}	0.00498	

The electromagnetic force vs. position diagram is shown in Fig. 38 and the electromagnetic force vs. coil current diagram is shown respectively in Fig. 39.

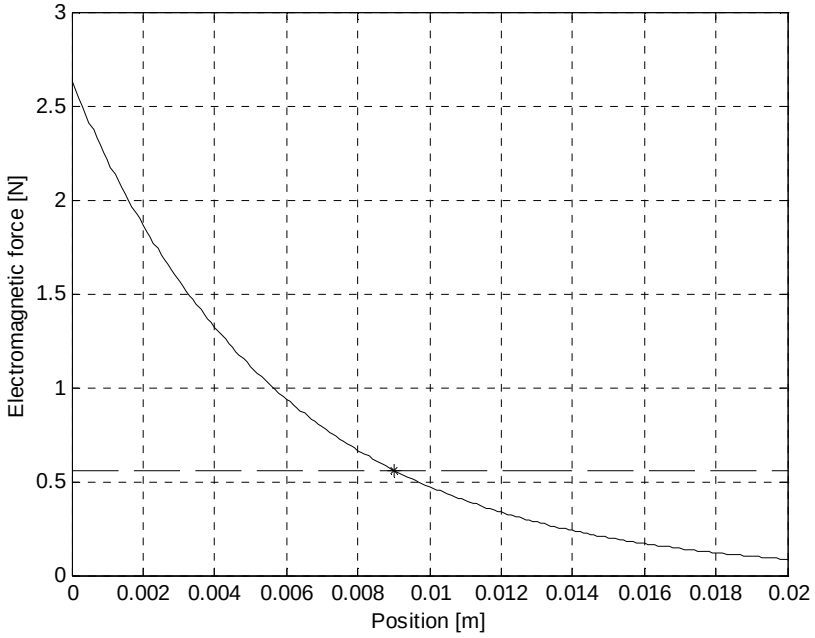


Fig. 38. Electromagnetic force vs. position. The gravity force of the big ball (dashed horizontal line) is crossing the curve at the 0.009 m distance from the electromagnet

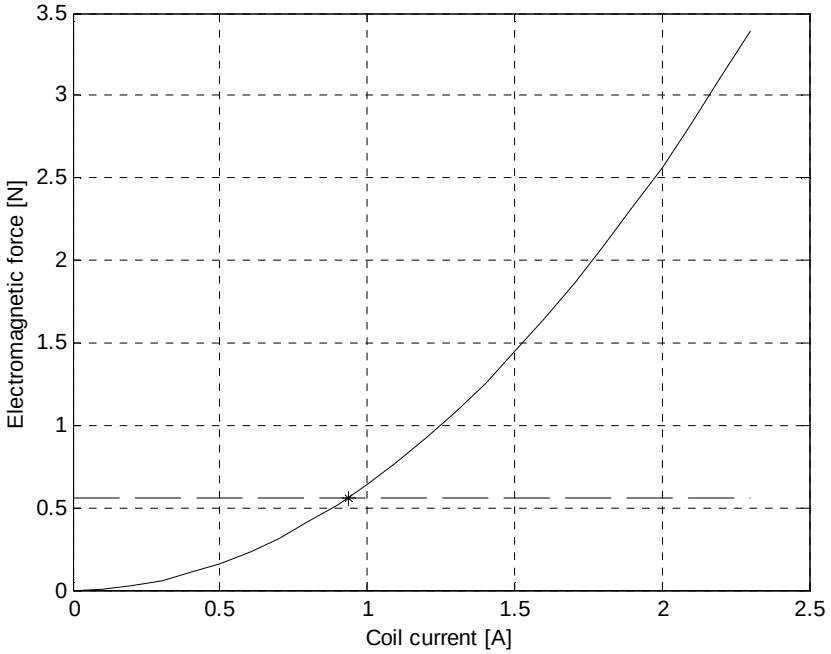


Fig. 39. Electromagnetic force vs. coil current. The gravity force of the big ball (dashed horizontal line) is crossing the curve at the 0.9345 A coil current

The electromagnetic force depends on two variables: the ball distance from the electromagnet and the current in the electromagnetic coil. This is clearly presented in Fig. 38 and Fig. 39. We can show these dependencies in three dimensional space (see Fig. 40). The ball is stabilized at $[x_1, x_2, x_3, x_4] = \mathrm{col}(9 \cdot 10^{-3},\quad 0,\quad 9.345 \cdot 10^{-1},\ 0)$. It means that the

ball velocity remains equal to zero. The ball is levitating kept at the 9 mm distance from the bottom of the upper electromagnet. The 0.9345 A current flowing through the magnetic coil is the appropriate value to balance the gravity force of the ball.

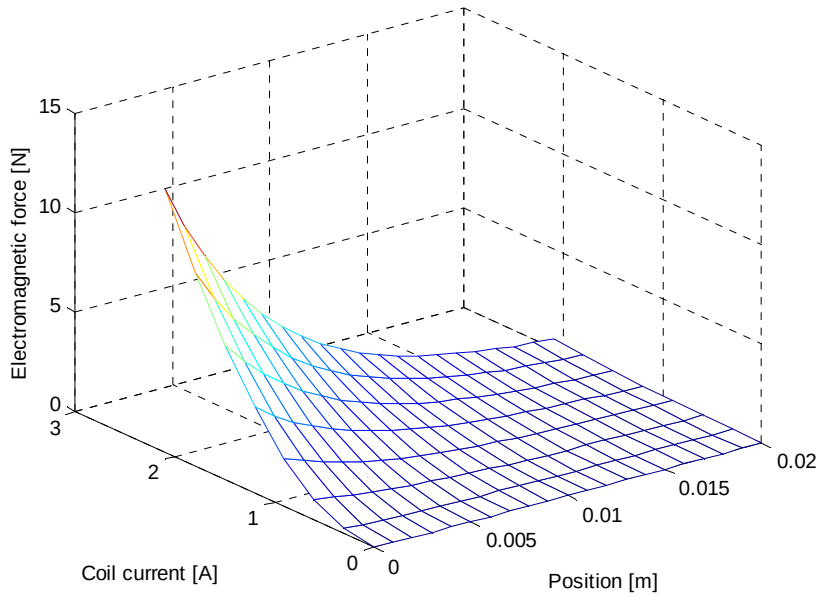


Fig. 40. Electromagnetic force vs. coil current and distance from the electromagnet.

In Fig. 41 the $f_i(x_1)$ diagram is shown.

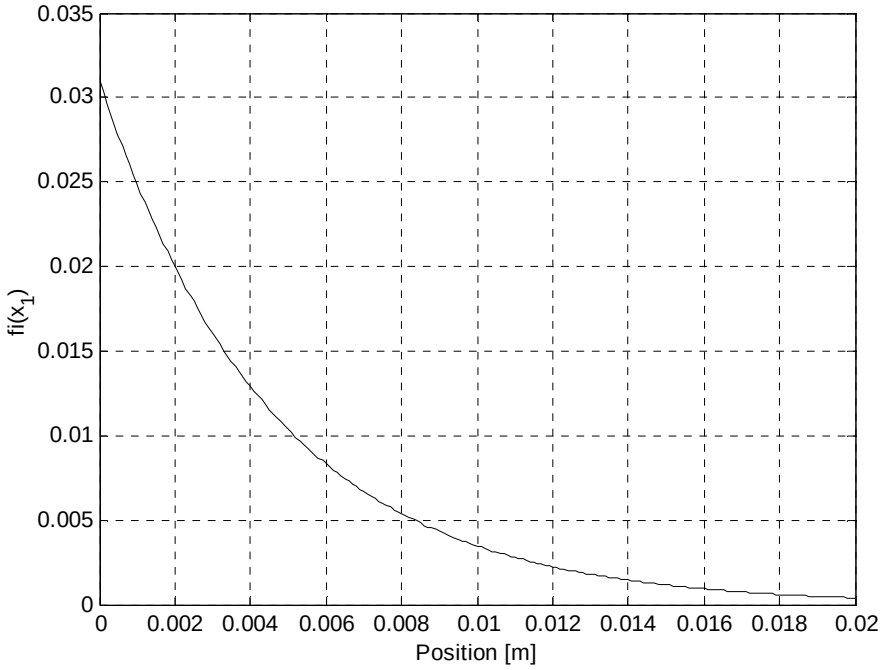


Fig. 41. Function $f_i(x_1)$

- **Linear continuous model**

MLS2EM is a highly nonlinear model. It can be approximated in an equilibrium point by a linear model. The linear model can be described by four linear differential equations of the first order in the form:

$$\dot{x} = Ax + Bu$$

$$y = Cx$$

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ a_{2,1} & 0 & a_{2,3} & a_{2,4} \\ a_{3,1} & 0 & a_{3,3} & 0 \\ a_{4,1} & 0 & 0 & a_{4,4} \end{bmatrix}, \qquad B = \begin{bmatrix} 0 \\ 0 \\ b_3 \\ b_4 \end{bmatrix}$$

The elements of the A matrix are expressed by the nonlinear model parameters in the following way:

$$a_{2,1} = \frac{x_{30}^2}{m}\frac{F_{emP1}}{F_{emP2}^2} e^{-\frac{x_{10}}{F_{emP2}}} + \frac{x_{40}^2}{m}\frac{F_{emP1}}{F_{emP2}^2} e^{-\frac{x_d - x_{10}}{F_{emP2}}},$$

$$a_{2,3} = -\frac{2x_{30}}{m}\frac{F_{emP1}}{F_{emP2}} e^{-\frac{x_{10}}{F_{emP2}}},$$

$$a_{2,4} = \frac{2x_{40}}{m}\frac{F_{emP1}}{F_{emP2}} e^{-\frac{x_d - x_{10}}{F_{emP2}}}$$

$$a_{3,1} = -(k_i u + c_i - x_{30})(x_{10} / f_{iP2}) f_i^{-1}(x_{10}),$$

$$a_{4,1} = -(k_i u + c_i - x_{40})(x_{10} / f_{iP2}) f_i^{-1}(x_d - x_{10})$$

$$a_{3,3} = -f_i^{-1}(x_{10}),$$

$$a_{3,4} = -f_i^{-1}(x_d - x_{10})$$

$$b_3 = k_i f_i^{-1}(x_{10}),$$

$$b_4 = k_i f_i^{-1}(x_d - x_{10})$$

The C vector elements correspond to an applied controller. For example, The PD controller shown in the next subsection requires C in the form:

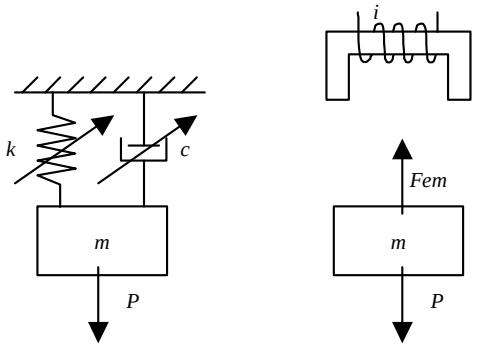
$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}.$$

- **Active suspension**

*

One of electromagnets of MLS2EM can be analyzed as the single-degree-of-freedom mass-spring-damper system with controllable stiffness and damping. Using the non-contact actuator both parameters are controlled by the formulated control strategy.

Let's assume that the nonlinear model of MLS2EM is simplified to the upper electromagnet only and is linearized at the selected steady-state point, that the coil current is fixed and actuator dynamics can be neglected. In this case the obtained model of the magnetic levitation system is controlled directly by the coil current and is simplified to the linear second order system. Let's assume that the second order open loop system is described by the following formula where α and β correspond to $a_{2,1}$ and $a_{2,3}$ respectively under above assumptions.



$$\dot{x} = \begin{bmatrix} 0 & 1 \\ \alpha & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ \beta \end{bmatrix} u$$

$$u(t) = -K_P x_1(t) - K_D x_2(t)$$

The closed loop system has two poles which can be located on the imaginary axis representing the marginally stable system but generally they are located in the left half plane to obtain desired performance. Choosing appropriate values of poles we can obtain the required dynamic behavior of the closed loop system.

The obtained second order closed loop system is characterized by programmable stiffness and damping settings.

$$\ddot{x} = (\alpha - \beta K_P)x - \beta K_D \dot{x}$$

The controller parameters K_P and K_D can be designed to satisfy requirements of the closed loop performance determined by the damp free natural frequency ω_n and the damping ratio ζ.

$$K_P = (\omega_n^2 + \alpha)\beta^{-1},\ K_D = 2\zeta\omega_n\beta^{-1}$$

Choosing the appropriate value of ω_n we can control the speed of the system response to the external disturbance. Highest natural angular frequency gives faster system response. Setting the appropriate value of damping ratio we can control the damping mode. In most cases the critically damped mode can be used because it is the most stable mode characterized by the asymptotic stability and time constant ω_n^{-1}. Note that α and β presented in the linearized system strongly depend on the selected steady-state point. Using the digital form of the PD controller the derivative component can be calculated as

difference quotient. The sampling frequency affects proportionally the derivative gain of the controller.

2.3.2 PD

If you click the PD EM1 button the following windows open (see Fig. 42).

The interior of the *Magnetic Levitation model MLS2EM* block is shown in Fig. 34.

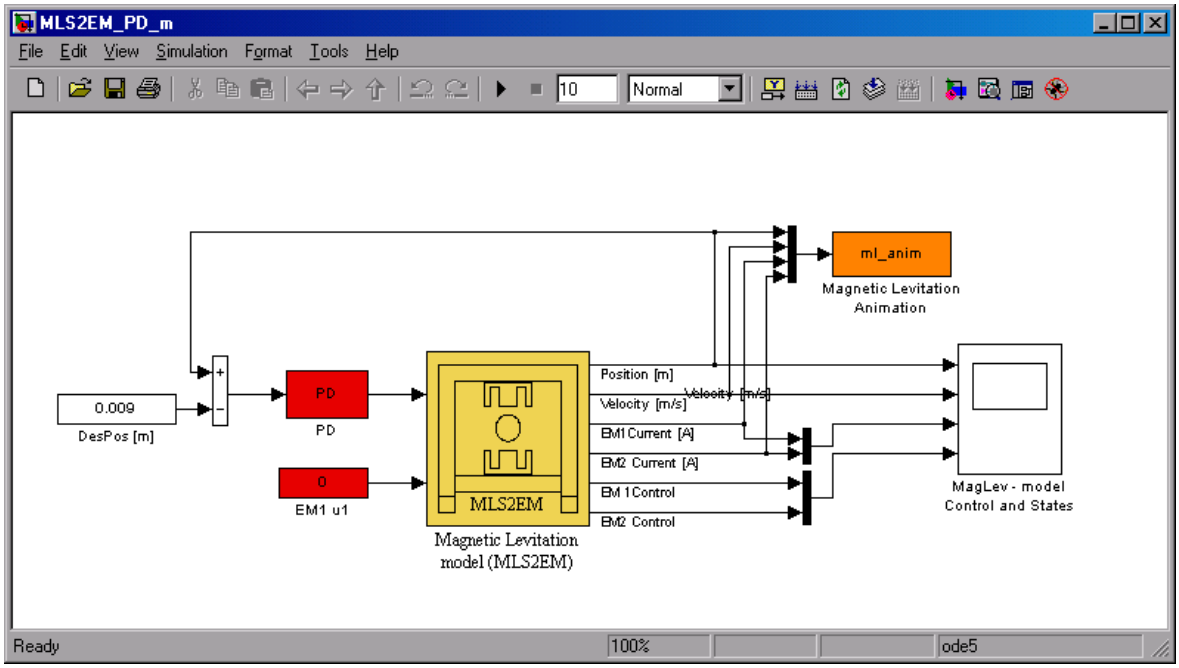
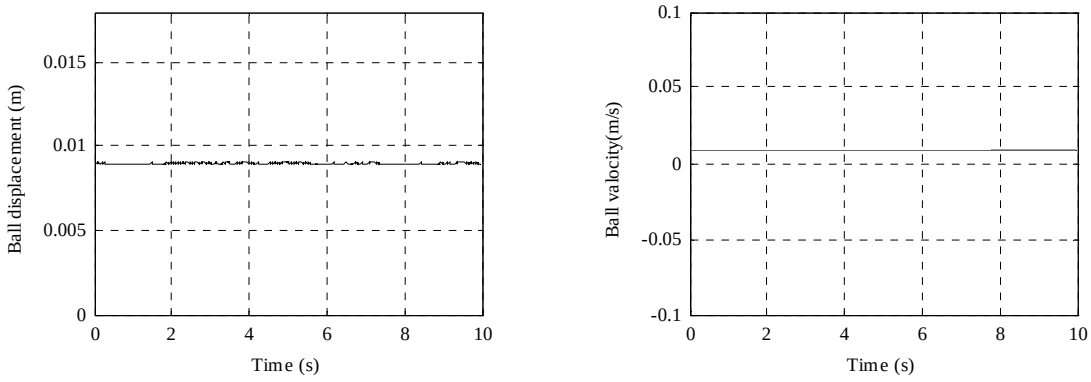


Fig. 42. PD simulation

The parameters given bellow are used in the PD controller.

K_P	K_D	u0
55	5	0.3611

The simulated stabilization results are shown below.



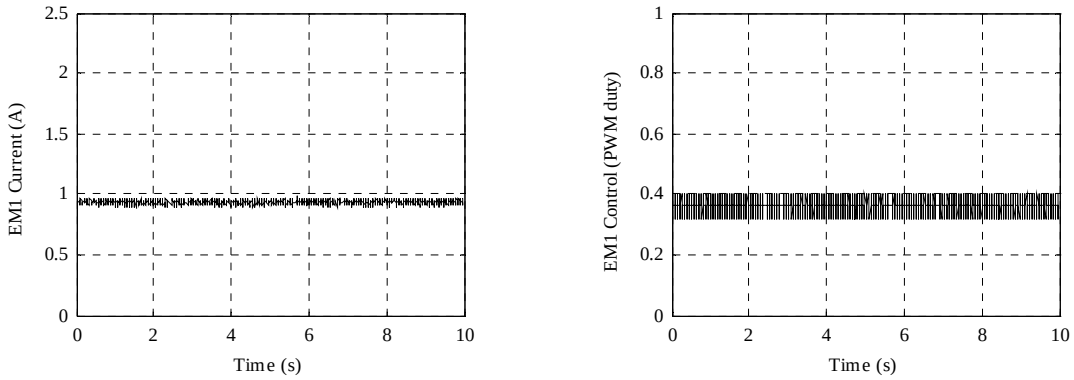


Fig. 43. PID simulation – the desired position is a constant.

2.3.3 PD Differential mode

If you click the PD differential mode button the following windows opens (see Fig. 44).

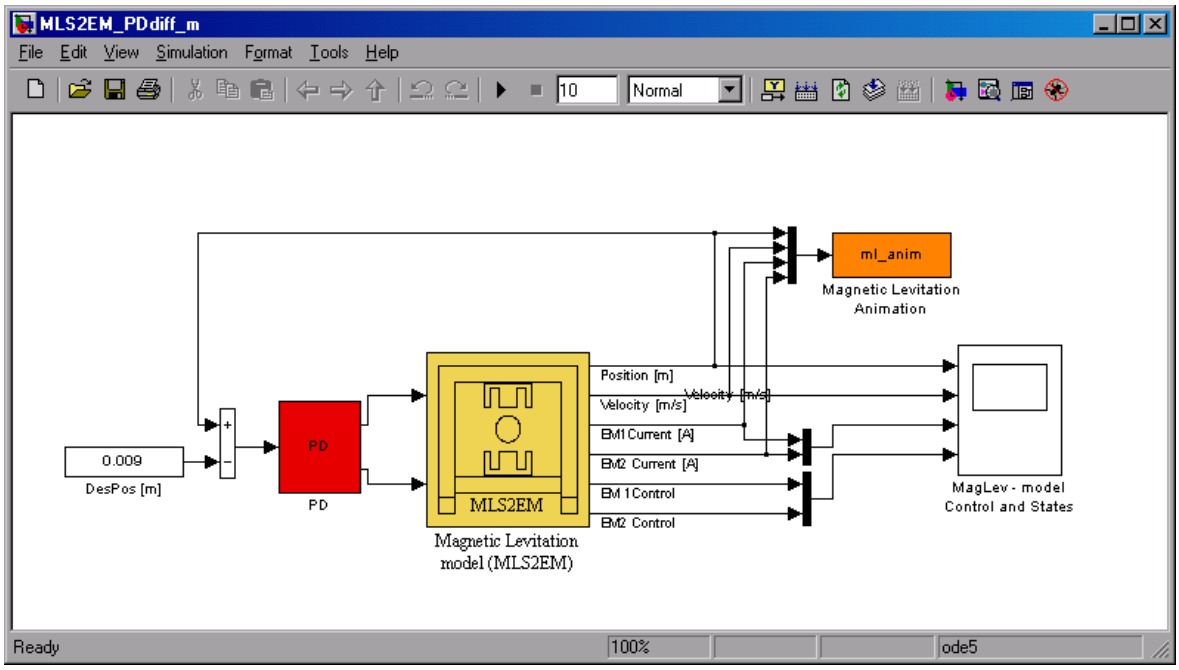


Fig. 44. PD differential mode simulation

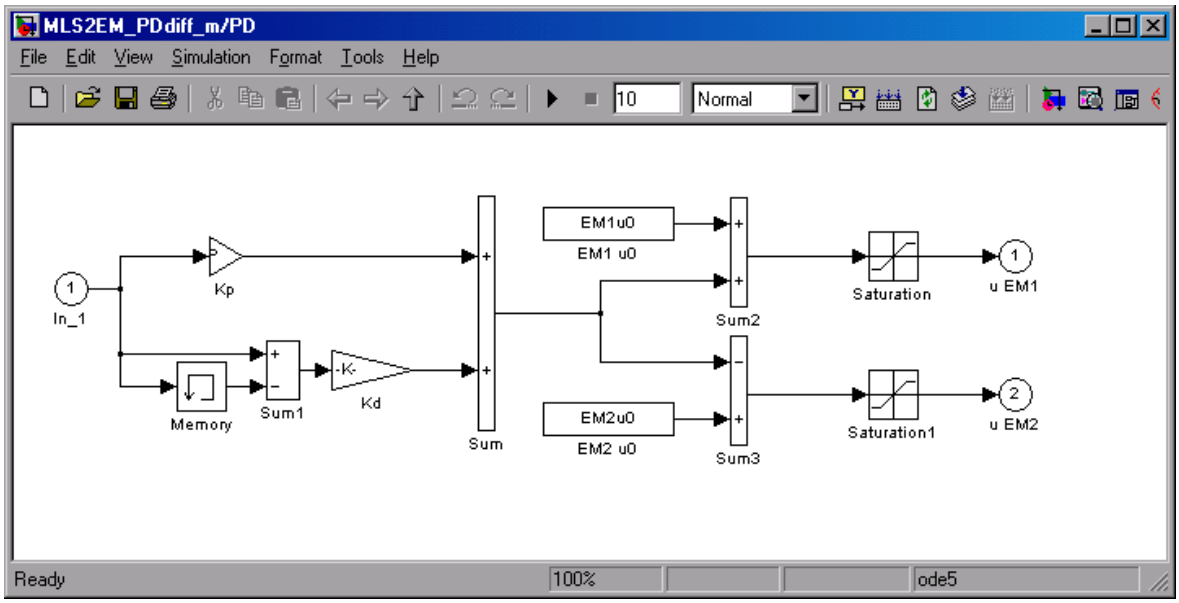
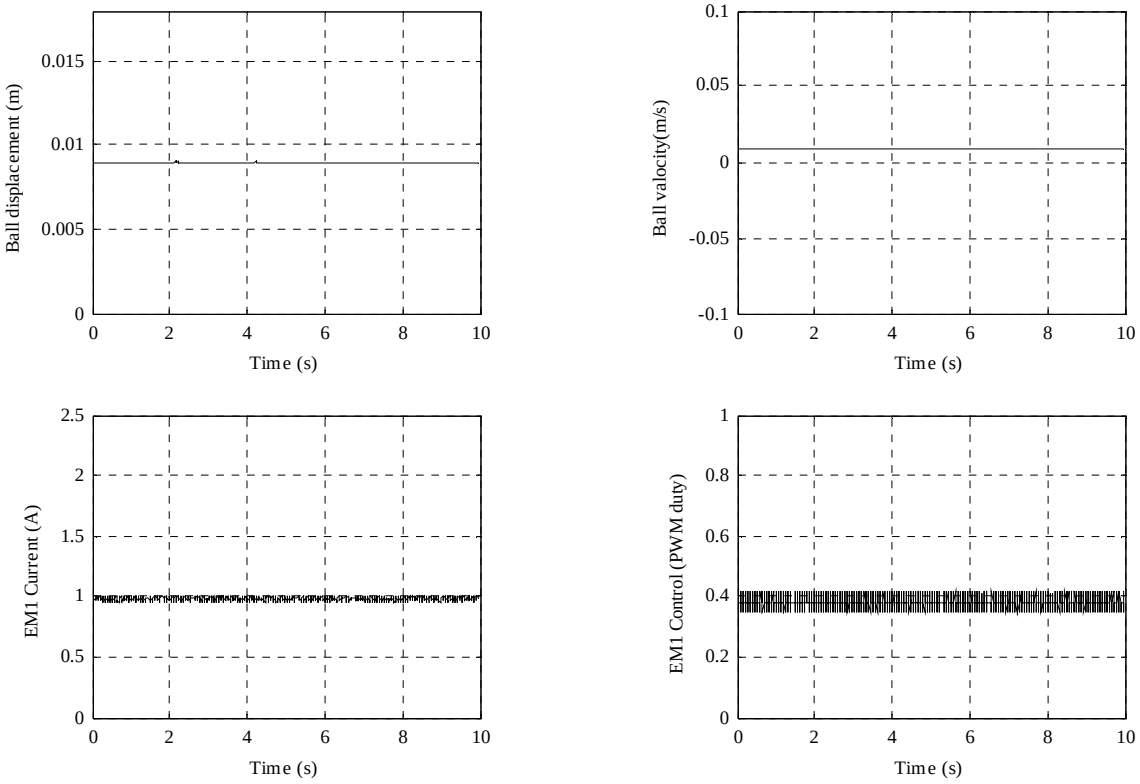


Fig. 45. Interior of the PD differential controller

The interior of the PD controller working in the differential mode is shown in Fig. 45.



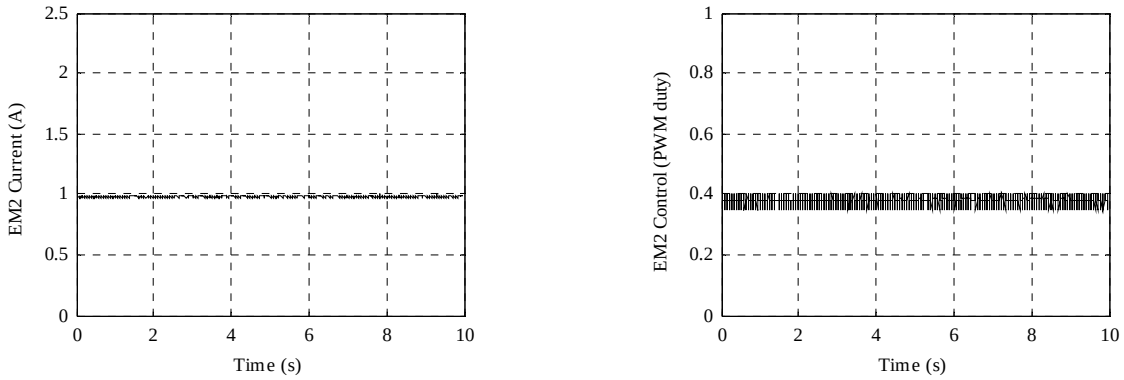


Fig. 46. LQ simulation – the desired position is a constant

2.4 Levitation

All simulation experiments can be repeated as real-time experiments. In this way one can verify accuracy of modelling. If we double click the *levitation* button in the *MLS2EM Main* window the following window opens (see Fig. 47).

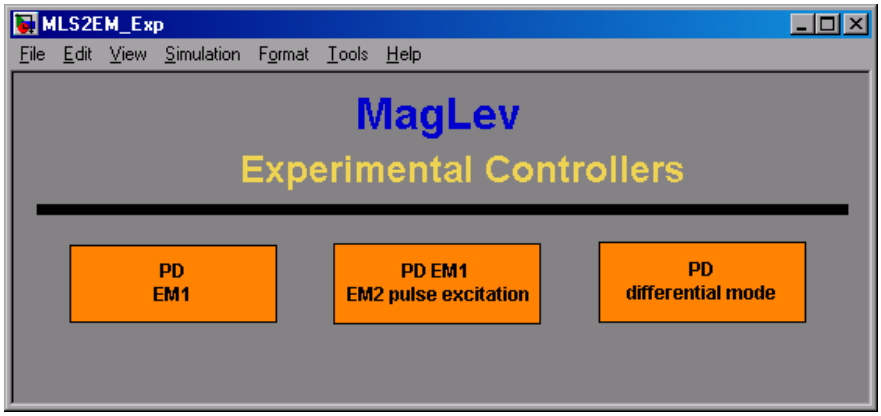


Fig. 47. Experimental controllers

Now, we can choose the controller we are interested in. We start from the PD control.

2.4.1 PD applied to EM1

Double click the PID button. The real-time PID controller opens (see Fig. 48). The results of the real-time experiment are shown in: Fig. 49.

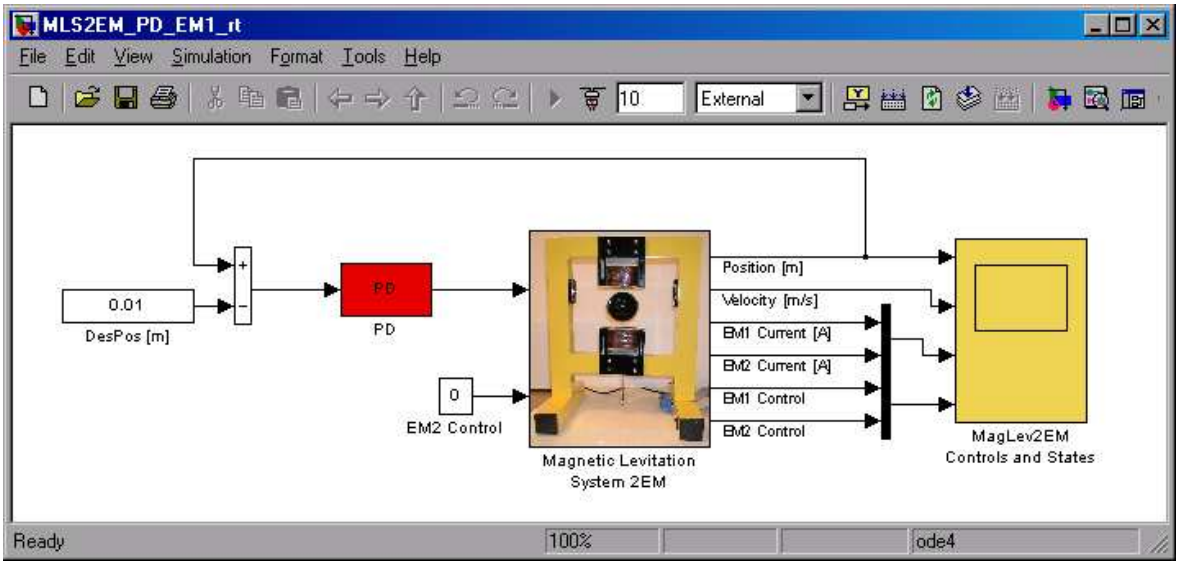


Fig. 48. PID real-time experiment.

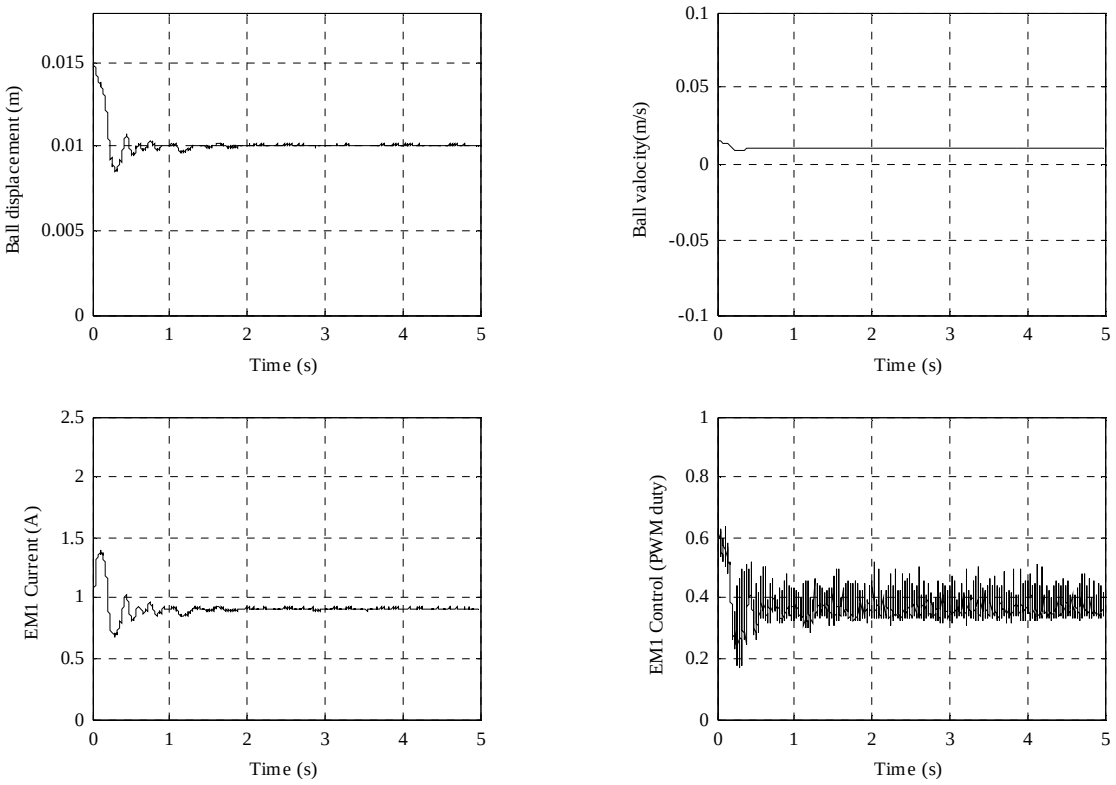


Fig. 49. PD real-time experiment. The desired position as a constant.

2.4.2 PD EM1 + EM2 pulse excitation

Double click the LQ button. The real-time LQ controller opens (see Fig. 50). The results of the real-time experiment are shown in: Fig. 51.

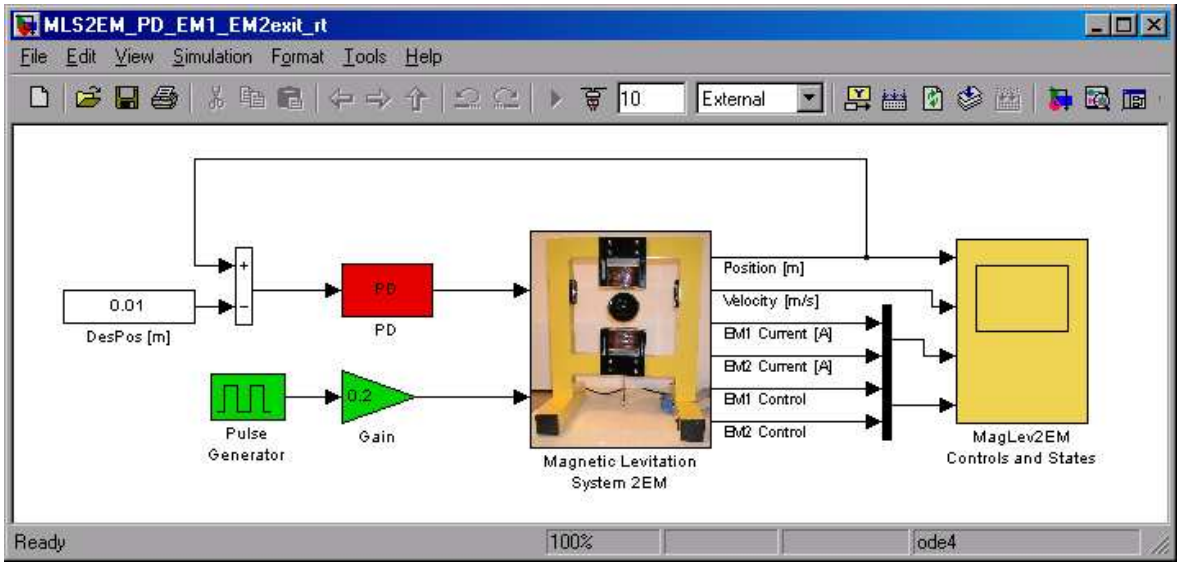


Fig. 50. PD control and pulse excitation real-time experiment.

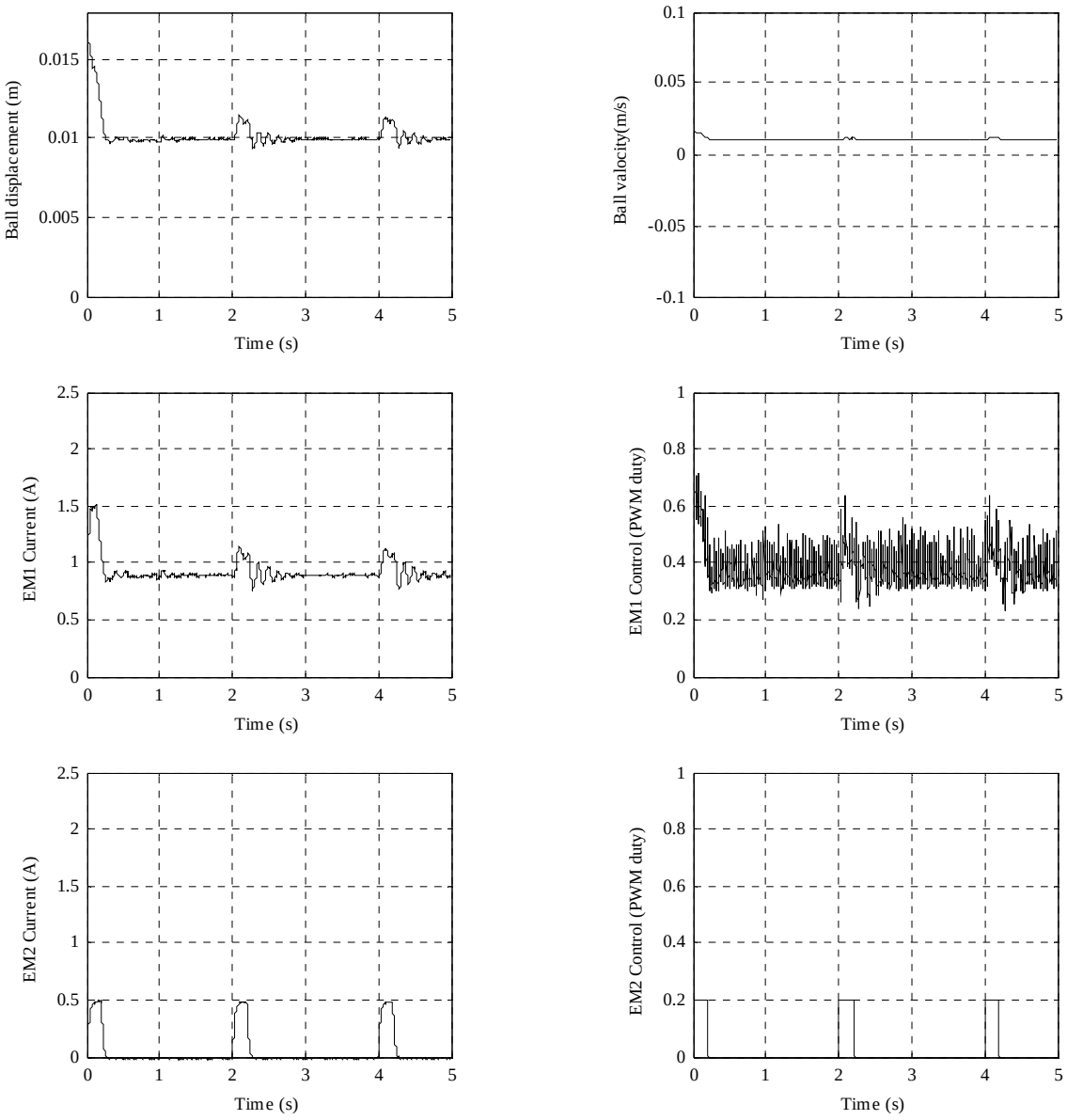


Fig. 51. PD stabilization and external pulse type excitation real-time experiment. The desired position as a constant.

2.4.3 PD differential control mode

Double click the LQ tracking button. The real-time LQ tracking controller opens (see Fig. 52). The results of the real-time experiment are shown in Fig. 53.

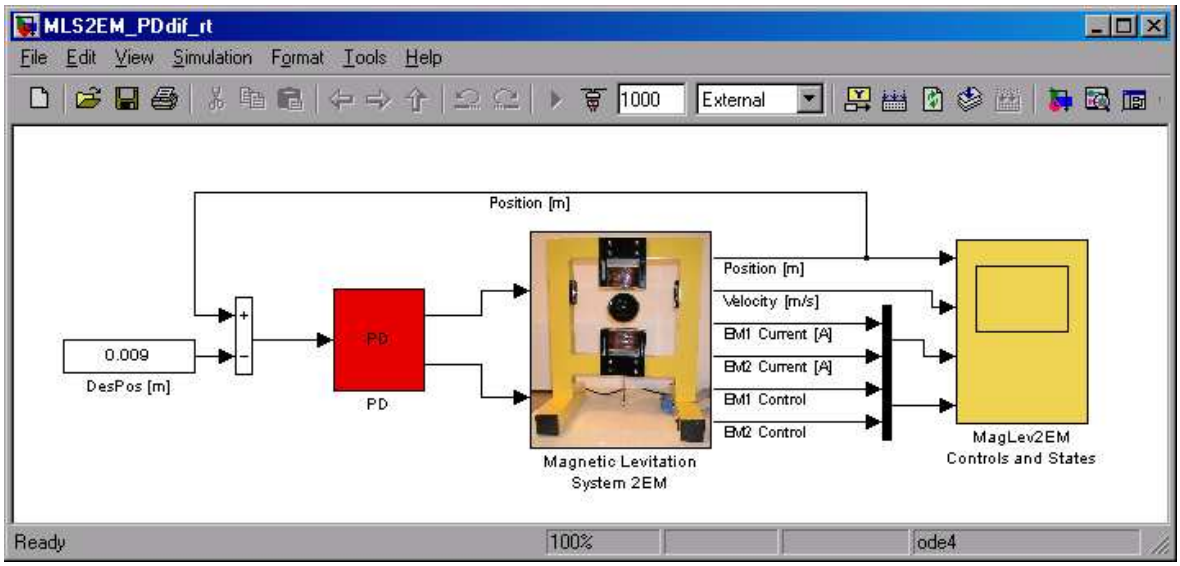


Fig. 52. PD differential control real-time experiment.

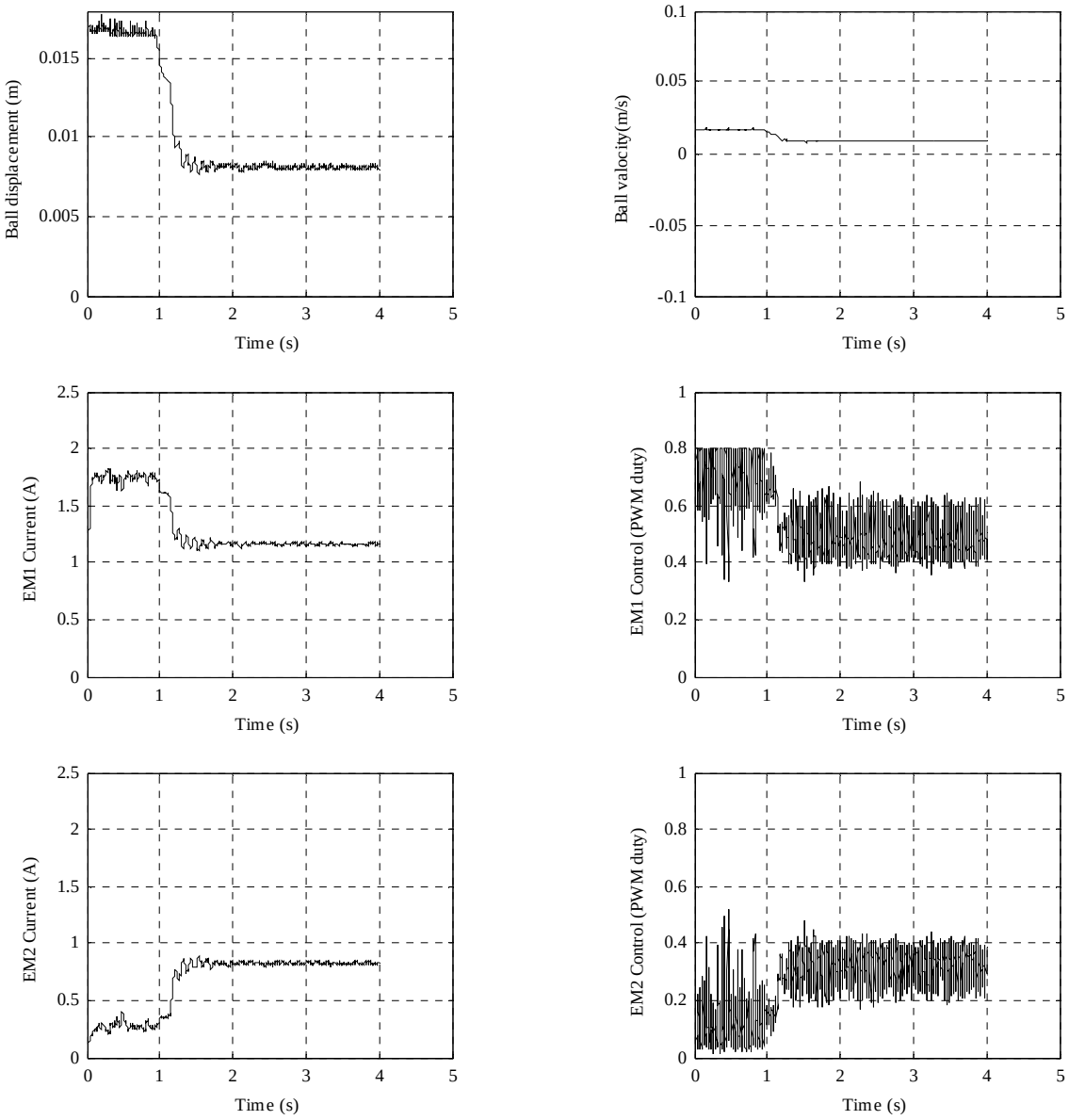


Fig. 53. PD operating in differential mode real-time experiment. The desired position as a constant.

3 Description of the Magnetic Levitation class properties

The *MagLev2EM* is a MATLAB class, which gives the access to all the features of the RT-DAC4/PCI board supported with the logic for the MLS2EM model. The RT-DAC4/PCI board is an interface between the control software executed by a PC computer and the power-interface electronic of the modular servo model. The logic on the board contains the following blocks:

- PWM generation block – generates the Pulse-Width Modulation output signal for the appropriate channel. Simultaneously the direction signal and the brake signal are generated to control the power interface modules. The PWM prescaler determines the frequency of the PWM wave;
- power interface thermal status –the thermal status can be used to disable the operation of the overheated actuator unit;
- interface to the on-board analog-to-digital converter. The A/D converter is applied to measure the position of the ball (light sensor) and to measure the coil current of the actuator.

All the parameters and measured variables from the RT-DAC4/PCI board are accessible by appropriate properties of the *MagLev2EM* class.
In the MATLAB environment the object of the *MagLev2EM* class is created by the command:
object_name = MagLev2EM;

for example *mls2em = maglev2em;*

The *get* method is called to read a value of the property of the object:
property_value = get(object_name, 'property_name');
The *set* method is called to set new value of the given property:
set(object_name, 'property_name', new_property_value);
The *display* method is applied to display the property values when the *object_name* is entered in the MATLAB command window.

This section describes all the properties of the *MagLev2EM* class. The description consists of the following fields:

Purpose	Provides short description of the property
Synopsis	Shows the format of the method calls
Description	Describes what the property does and the restrictions subjected to the property
Arguments	Describes arguments of the set method
See	Refers to other related properties
Examples	Provides examples how the property can be used

3.1 BaseAddress

Purpose: Read the base address of the RT-DAC4/PCI board.

Synopsis: *BaseAddress = get(mls2em, 'BaseAddress');*

Description: The base address of RT-DAC4/PCI board is determined by computer. Each C*ML2EM* object has to know the base address of the board. When a *CML2EM* object is created the base address is detected automatically. The detection procedure detects the base address of the first RT-DAC4/PCI board plugged into the PCI slots.

Example: Create the MagLev2EM object:
mls2em = MagLev2EM;
Display their properties by typing the command:
mls2em

```
Type:             InTeCo ML2EM object
BaseAddress:      54272 / D400 Hex
Bitstream ver.:   x901
Input voltage:    [ 0.8451  0.0244 0.0243][V]
PWM:              [ 0  0]
PWM Prescaler:    [ 0  0]
Thermal status:   [ 0  0]
Time:             0.00 [sec]
```

Read the base address:
BA = get(mls2em, 'BaseAddress');

3.2 BitstreamVersion

Purpose: Read the version of the logic stored in the RT-DAC4/PCI board.

Synopsis: *Version = get(mls2em, 'BitstreamVersion');*

Description: The property determines the version of the logic design of the RT-DAC4/PCI board. The magnetic levitation models may vary and the detection of the logic design version makes it possible to check if the logic design is compatible with the physical model.

3.3 PWM

Purpose: Set the duty cycle of the PWM wave.

Synopsis: *PWM = get(mls2em, 'PWM');*
set(mls2em, 'PWM', NewPWM);

Description: The property determines the duty cycle and direction of the PWM wave. The PWM wave is used to control the electromagnet so in fact this property is responsible for the electromagnet control signal. The *NewPWM* variable is a two element vector in the range from 0 to 1. The value of +1 means the maximum control, 0.0 means zero control.

See: *PWMPrescaler*

Example: set(mls2em, 'PWM', [0.5 0.2]);

3.4 PWMPrescaler

Purpose: Determine the frequency of the PWM wave.

Synopsis: *Prescaler = get(mls2em, 'PWMPrescaler');*
set(mls2em, 'PWMPrescaler', NewPrescaler);

Description: The prescaler value can vary from 0 to 16. The 0 value generates the maximal PWM frequency. The value 16 generates the minimal frequency. The frequency of the generated PWM wave is given by the formula:

$$PWM_{\text{frequency}} = 40MHz / 4095 * (Prescaler+1)$$

See: *PWM*

3.5 Stop

Purpose: Sets the control signal to zero.

Synopsis: *set(ml, 'Stop');*

Description: This property can be called only by the set method. It sets the zero control of the electromagnet and is equivalent to the *set(ml, 'PWM', 0)* call.

See: *PWM*

3.6 Voltage

Purpose: Read two voltage values.

Synopsis: *Volt = get(mls2em, 'Voltage');*

Description: Returns the voltage of three analog inputs. Usually the analog inputs are applied to measure the ball position and both coil currents.

3.7 ThermStatus

Purpose: Read thermal status flag of the power amplifier.

Synopsis: *ThermSt = get(mls2em, 'ThermStatus');*

Description: Returns the thermal flags of the power amplifier. When the temperature of a power amplifier is too high the flag is set to 1.

3.8 Time

Purpose: Return time information.

Synopsis: *T = get(mls2em, 'Time');*

Description: The *MagLev2EM* object contains the time counter. When a *MagLev2EM* object is created the time counter is set to zero. Each reference to the *Time* property updates its value. The value is equal to the number of milliseconds which elapsed since the object was created.

3.9 Quick reference table

Property name	Operation*	Description
BaseAddress	R	Read the base address of the RT-DAC4/PCI board
BitstreamVersion	R	Read the version of the logic design for the RT-DAC4/PCI board
PWM	R+S	Read/set the parameters of the PWM waves
PWMPrescaler	R+S	Read/set the frequency of the PWM waves
Stop	S	Set the control signals to zero
Voltage	R	Read the input voltages
ThermStatus	R	Read the thermal flags of the power amplifiers
Time	R	Read time information

- R – read-only property, S – allowed only set operation, R+S –property may be read and set

bottom