

Digital Pendulum System

GETTING STARTED

33-005-1M5

MATLAB 5 Version



Feedback

Feedback Instruments Ltd, Park Road, Crowborough, E. Sussex, TN6 2QR, UK.

Telephone: +44 (0) 1892 653322, Fax: +44 (0) 1892 663719.

email: @fdbk.demon.co.uk World Wide Web Homepage: <http://www.fbk.com>

Manual: 33-005-1M5 Ed02 981130

Printed in England by FI Ltd, Crowborough

Feedback Part No. 1160-330051M5



**DIGITAL PENDULUM SYSTEM
GETTING STARTED**

PREFACE

Notes



DIGITAL PENDULUM SYSTEM GETTING STARTED

PREFACE

THE HEALTH AND SAFETY AT WORK ACT 1974

We are required under the Health and Safety at Work Act 1974, to make available to users of this equipment certain information regarding its safe use.

The equipment, when used in normal or prescribed applications within the parameters set for its mechanical and electrical performance, should not cause any danger or hazard to health or safety if normal engineering practices are observed and they are used in accordance with the instructions supplied.

If, in specific cases, circumstances exist in which a potential hazard may be brought about by careless or improper use, these will be pointed out and the necessary precautions emphasised.

While we provide the fullest possible user information relating to the proper use of this equipment, if there is any doubt whatsoever about any aspect, the user should contact the Product Safety Officer at Feedback Instruments Limited, Crowborough.

This equipment should not be used by inexperienced users unless they are under supervision.

We are required by European Directives to indicate on our equipment panels certain areas and warnings that require attention by the user. These have been indicated in the specified way by yellow labels with black printing, the meaning of any labels that may be fixed to the instrument are shown below:



CAUTION -
RISK OF
DANGER

Refer to accompanying documents



CAUTION -
RISK OF
ELECTRIC SHOCK



CAUTION -
ELECTROSTATIC
SENSITIVE DEVICE

PRODUCT IMPROVEMENTS

We maintain a policy of continuous product improvement by incorporating the latest developments and components into our equipment, even up to the time of dispatch.

All major changes are incorporated into up-dated editions of our manuals and this manual was believed to be correct at the time of printing. However, some product changes which do not affect the instructional capability of the equipment, may not be included until it is necessary to incorporate other significant changes.

COMPONENT REPLACEMENT

Where components are of a 'Safety Critical' nature, i.e. all components involved with the supply or carrying of voltages at supply potential or higher, these must be replaced with components of equal international safety approval in order to maintain full equipment safety.

In order to maintain compliance with international directives, all replacement components should be identical to those originally supplied.

Any component may be ordered direct from Feedback or its agents by quoting the following information:

- | | |
|------------------------|----------------------------|
| 1. Equipment type | 2. Component value |
| 3. Component reference | 4. Equipment serial number |

Components can often be replaced by alternatives available locally, however we cannot therefore guarantee continued performance either to published specification or compliance with international standards.



DIGITAL PENDULUM SYSTEM GETTING STARTED

PREFACE



DECLARATION CONCERNING ELECTROMAGNETIC COMPATIBILITY

Should this equipment be used outside the classroom, laboratory study area or similar such place for which it is designed and sold then Feedback Instruments Ltd hereby states that conformity with the protection requirements of the European Community Electromagnetic Compatibility Directive (89/336/EEC) may be invalidated and could lead to prosecution.

This equipment, when operated in accordance with the supplied documentation, does not cause electromagnetic disturbance outside its immediate electromagnetic environment.

COPYRIGHT NOTICE

© **Feedback Instruments Limited**

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without the prior permission of Feedback Instruments Limited.

ACKNOWLEDGEMENTS

Feedback Instruments Ltd acknowledge all trademarks.

IBM, IBM - PC are registered trademarks of International Business Machines.

MICROSOFT, WINDOWS 95, WINDOWS 3.1 are registered trademarks of Microsoft Corporation.

MATLAB is a registered trademark of Mathworks Inc.



TABLE OF CONTENTS

1. INTRODUCTION AND DESCRIPTION	1-1
2. MODEL AND PARAMETERS	2-1
3. CONTROL ALGORITHMS	3-1
3.1 Rule-based control	3-2
3.2 LQ control	3-2
3.3 PID control	3-3
4. COMPUTER CONTROL	4-1
5. THE DIGITAL PENDULUM SYSTEM	5-1
5.1 Hardware:	5-1
5.2 Software:	5-1
5.3 Pendulum, Cart, Encoder interface and DC motor driver	5-2
5.4 Operation of the system	5-4
6. STARTING, STOPPING AND TESTING	6-1
6.1 Starting procedure	6-1
6.2 Testing and troubleshooting	6-4
6.3 Stopping procedure	6-5
7. FAMILIARISATION USING DEMOS	7-1
7.1 Crane PID controller demo	7-1
7.2 Swinging-up and Upright Stabilisation Demo	7-2
7.3 Crane LQ controller demo	7-4
8. TUNING OF PARAMETERS	8-1
8.1 Swing, Upright and Down Stabilisation demo	8-1
8.2 Definitions and setting of parameters	8-8
8.3 Examples of tuning	8-15
9. REFERENCES	9-1



**DIGITAL PENDULUM SYSTEM
GETTING STARTED**

CONTENTS

Notes



1. INTRODUCTION AND DESCRIPTION

One of the simplest problems in robotics is that of controlling the position of a single link using a steering force applied at the end. Pole-balancing systems are impressive demonstration models of missile stabilisation problems. The crane used at shipping ports is another example of non-linear electromechanical systems having a complex dynamic behaviour and creating challenging control problems. Mathematically both are just a pendulum in a stable or unstable position.

The **pendulum-cart set-up** consists of a pole mounted on a cart in such a way that the pole can swing free only in vertical plane. The cart is driven by a DC motor. To swing and to balance the pole the cart is pushed back and forth on a rail of limited length.

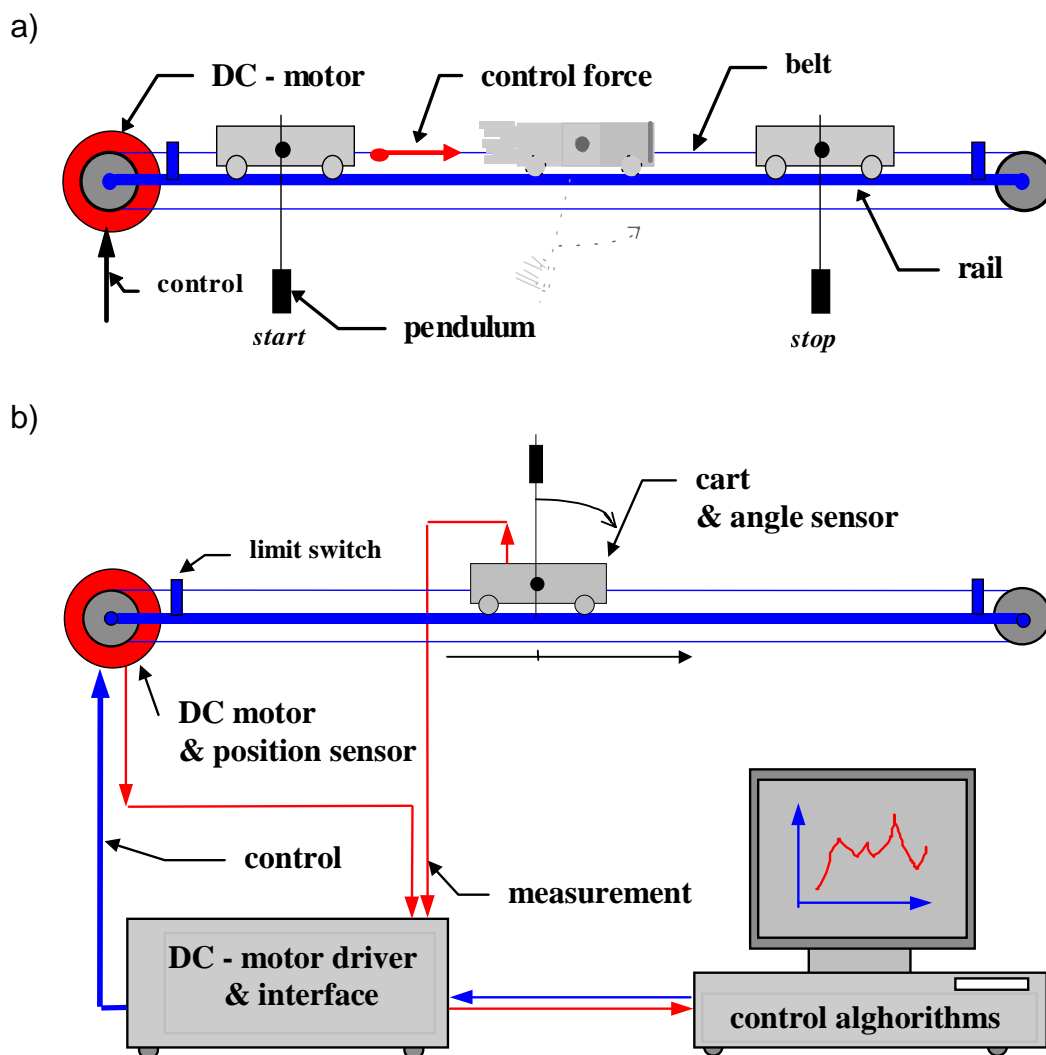


Fig. 1.1 a) The pendulum-cart set-up, b) pendulum control system.



The vertical stationary positions of the pendulum (upright and down) are equilibrium positions when no force is being applied. In the upright position a small deviation from it results in an unstable motion. Generally the pendulum control problem is to bring the pole to one of the equilibrium positions and preferably to do so as fast as possible, with few oscillations, and without letting the angle and velocity become too large. After the desired position is reached, we would like to keep the system in this state despite random perturbations. Manual control of the cart-pole system is possible only for simple tasks e.g. for moving the cart from one place on the rail to another. For more complicated tasks (such as stabilising the pole in an upright position) a feedback control system must be implemented (Fig. 1.1) .

The purpose of the inverted pendulum control algorithm is to apply a sequence of forces of constrained magnitude to the cart, such that the pole starts to swing with an increasing amplitude without the cart overriding the ends of the rail. Firstly the pole is swung up to the vicinity of its upright position and then, once this has been accomplished, the controller maintains the pole vertically and at the same time brings the cart back to the centre of the rail. Therefore two independent control algorithms are implemented for this purpose:

- a swinging algorithm, and
- a stabilising algorithm.

Only one of two control algorithms is active in each control zone. These zones are shown in

Fig. 1.2.

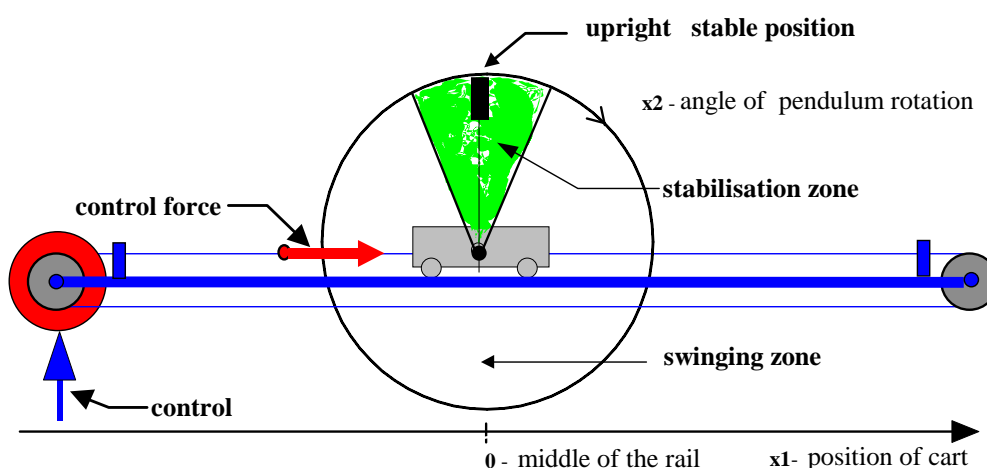


Fig. 1.2 Activity zones of two control algorithms



The swinging control algorithm is a heuristic one, based on energy rules. The algorithm steers the pole up thus increasing its total energy. There is a trade-off between two tasks: to *swing* the pendulum to the upright position and to *centre* the cart on the rail.

Due to the presence of disturbances and parameter uncertainties, a robust behaviour is more important than the optimal character of the control strategy. The switching moments are calculated according to a simple rule. The characteristic feature of control is its “bang-bang” character. Swinging up the pole may result in over-reaching the upper unstable equilibrium point.

To achieve a “soft” landing in the vicinity of the upright position (“stabilisation zone” in

Fig. 1.2), a routine called the “soft landing arbiter” checks whether the kinetic energy of the pole, minus the energy loss due to friction, is sufficient to raise the centre of gravity of the pole to its upright position. If the condition is satisfied then the control is set to zero and the “bang-bang” character of the control is finished. After the pole has entered the stabilisation zone the system can be treated as linear and the control is switched to the stabilising algorithm.

Due to the limited length of the rail a routine called “length control” is introduced, to reinforce centring of the cart and prevent over-running the edges of the rail. The rule is very simple. When the position given by the parameter “length” is reached, then the maximal force is applied to the cart steering it back away from this position.



**DIGITAL PENDULUM SYSTEM
GETTING STARTED**

CHAPTER 1

INTRODUCTION AND DESCRIPTION

Notes



2. MODEL AND PARAMETERS

The state of the system is the vector $x = \text{col}(x_1, x_2, x_3, x_4)$, where x_1 is the cart position (distance from the centre of the rail), x_2 is the angle between the upward vertical and the ray pointing at the centre of mass, measured counter-clockwise from the cart ($x_2 = 0$ for the upright position of the pendulum), x_3 is the cart velocity, and x_4 is the pendulum angular velocity.

The pendulum rotates in a vertical plane around an axis located on a cart. The cart can move along a horizontal rail, lying in the plane of rotation. A control force u , parallel to the rail, is applied to the cart. The mass of the cart is denoted by m_c and the mass of the pendulum, by m_p . l is the distance from the axis of rotation to the centre of mass of the pendulum-cart system. J is the moment of inertia of the pendulum-cart system with respect to the centre of mass.

T_c denotes the friction in the motion of the cart, and D_p is the moment of friction in the angular motion of the pendulum, proportional to the angular velocity: $D_p = f_p x_4$. The force of reaction of the rail V acts vertically on the cart. As the horizontal co-ordinate of the centre of mass is equal to $x_1 - l \sin x_2$ and the vertical to $l \cos x_2$, the motion equations are as follows

$$(m_c + m_p) (x_1 - l \sin x_2)'' = F - T_c,$$

$$(m_c + m_p) (l \cos x_2)'' = V - (m_c + m_p)g,$$

$$J x_2'' = (u - T_c) l \cos x_2 + V l \sin x_2 - D_p.$$

$(.)''$ denotes the second derivative with respect to time t and $(.)'$ denotes the first derivative with respect to time t . The first two equations describe the translation of the centre of mass, while the third describes the rotation of the whole system around the centre of mass. After the elimination of V and simple calculations we obtain the state equations (for $t \geq 0$)

$$x_1' = x_3,$$

$$x_3' = \frac{a(u - T_c - \mu x_4^2 \sin x_2) + l \cos x_2 (\mu g \sin x_2 - f_p x_4)}{J + \mu l \sin^2 x_2}$$

$$x_2' = x_4,$$

$$x_4' = \frac{l \cos x_2 (u - T_c - \mu x_4^2 \sin x_2) + \mu g \sin x_2 - f_p x_4}{J + \mu l \sin^2 x_2} \quad (2.1)$$



where

$$a = l^2 + \frac{J}{m_t + m_p}, \quad \mu = (m_c + m_p)l. \quad (2.2)$$

The admissible controls are bounded

$$|u(t)| \leq M. \quad (2.3)$$

The cart friction T_c in the model is a non-linear function of the cart velocity x_3 . As an approximation one can assume $T_c = f_c x_3$. The rail has a finite length and hence the cart position x_1 is bounded:

The typical parameters of the cart-pole set-up are given in Table 1.

Table 1. Parameters of the pendulum-cart set-up

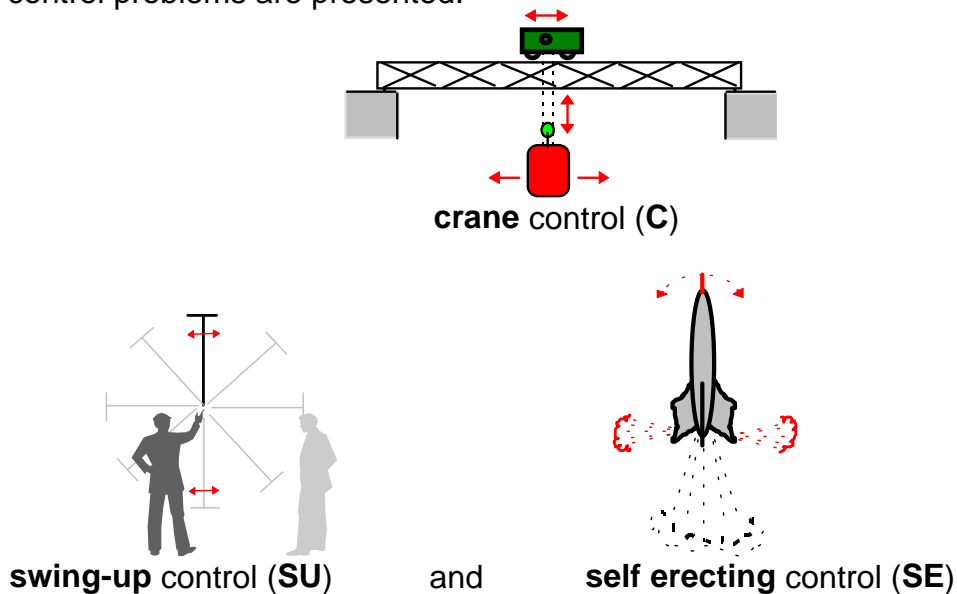
name of parameter	value of parameter
track limits	± 0.5 m
gravity g	9.81 m/s ²
Distance between mass centre and axis of rotation l	0.017 m
mass of cart m_c	1.12 kg
mass of pole m_p	0.11 kg
magnitude of control force M	17.0 N
moment of inertia of system J	0.0136 kgm ²
friction coefficient of pole rotation f_p	negligible
friction coefficient of cart f_c	0.05 Ns/ m

Notice, that the model of the pendulum-cart set-up is an example of a SIMO system: a single control input and multi outputs (states) and can be used to demonstrate the advantages of closed-loop control.



3. CONTROL ALGORITHMS

Three control problems are presented:



A human or a computer may operate as a controller. A human operator may be fairly successful in the C problem, but completely fails in the first two problems. The SU and SE problems require a fast operating controller and only a computer can manage such fast and complex tasks.

In this section a brief review of computer control algorithms is given but for a detailed description of the control algorithms refer to the Teaching Manual, 33-005-4.

The following control algorithms are introduced:

- **rule-based** (SU)
- **LQ** (C and SE)
- **PID** (C)
- **fuzzy*** (C)
- **time-optimal*** (SU and C)
- **adaptive*** (SU, C and SE)

* These controllers will be added in the extended version of toolboxes

The algorithmic formulas given below use the notation of Section 2.



3.1 Rule-based control

The simple swinging up formula has the form:

$$u = \text{sign}[x_4(|x_2| - \pi/2)]$$

It results in a normalised value of ± 1 of the control u . The control value is proportionally decreased by the parameter *Max*, taken from the range $[0, 1]$. To avoid rotation of the pole the total energy of the pole is calculated in each numerical step. If this energy is sufficient to steer the pole to its upright position then the control is set to zero. If the parameter *Friction* is greater than zero then a compensation factor for static friction is introduced:

$$u = u_{old} + \text{sign}(u_{old})\text{Friction}$$

Then the limit of the cart position is checked. If the cart reaches this limit then the maximum value of control (e.g. ± 1) is generated with the sign opposite to the sign of cart position.

3.2 LQ control

The linear-quadratic controller has the form:

$$u = - (K_1\varepsilon_1 + K_2\varepsilon_2 + K_3\varepsilon_3 + K_4\varepsilon_4)$$

where:

ε_1 = desired position of the cart - measured position of the cart,

ε_2 = desired angle of the pendulum - measured angle of the pendulum,

ε_3 = desired velocity of the cart - observed velocity of the cart,

ε_4 = desired angular velocity of the pendulum - observed angular velocity of the pendulum.

K_1, \dots, K_4 are positive constants.

The optimal feedback gain vector $K = [K_1, \dots, K_4]$ is calculated such that the feedback law $u = -K\varepsilon$; where $\varepsilon = [\varepsilon_1, \dots, \varepsilon_4]$ minimises the cost function:

$$J = \text{Integral} \{x'Qx + u'Ru\} dt - \text{where } Q \text{ and } R \text{ are the weighting matrices,}$$

subject to the state equation:

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



The last equation is the linearised model of the cart-pendulum system.

3.3 PID control

The PID controller has the form:

$$u = K_p \varepsilon_1(k) + K_i \sum_{k=0}^n \varepsilon_1(k) + K_d [\varepsilon_1(k) - \varepsilon_1(k-1)]$$

where:

ε_1 = desired position of the cart - measured position of the cart

K_p is gain coefficient

K_i is integration gain

K_d is derivative gain

The constants K_p , K_i and K_d can be chosen according to the Ziegler-Nichols rule.



**DIGITAL PENDULUM SYSTEM
GETTING STARTED**

**CHAPTER 3
CONTROL ALGORITHMS**

Notes



4. COMPUTER CONTROL

A block diagram of a computer-controlled process is given in Fig. 4.1

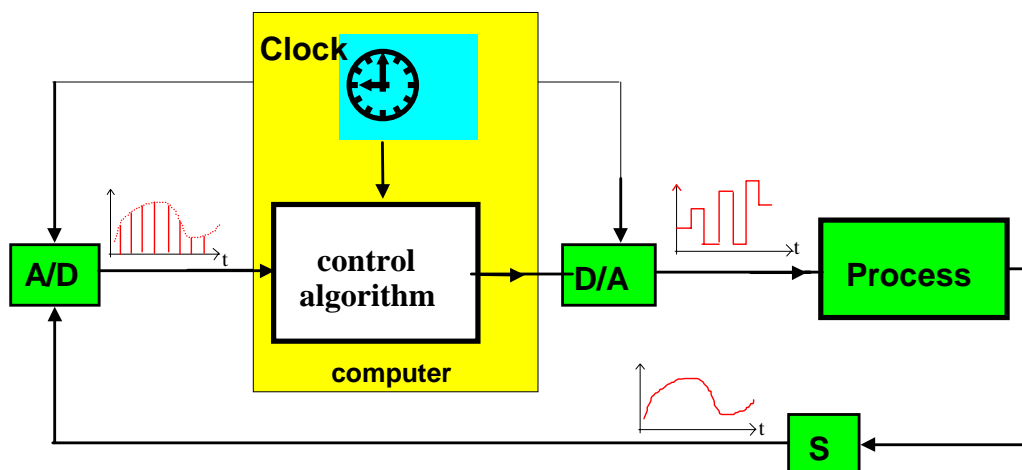


Fig. 4.1. Computer controlled process.

The system contains six blocks: the process, sensors (S), A/D and D/A converters, control algorithm, and a clock. The operation of the converters and the control algorithm is controlled by the software clock. The time between successive conversions of the signal to digital form is called the sampling period (T_0). The clock supplies a pulse every T_0 seconds, and the A/D converter sends a number to the computer every time an interrupt arrives. The control algorithm computes the value of the control variable and sends it as a number to the D/A converter. It is assumed that the D/A converter holds the signal constant over the sampling period; periodic sampling is normally used. It is possible to use different sampling periods for the A/D and D/A converters.

An application of the general digital control system schema for pendulum control is given in block diagram form in Fig. 4.2. Two process states are measured: the cart position x_1 and the pendulum angle x_2 . Process states are measured as continuous signals and converted to digital by optical encoders (sensors S1 S2). The reference input (desired value of the cart position x_1) can be generated in a digital form using a desired position generator. The software timer is used to supply interrupts for the system: The basic clock activates the periodic sampling of optical decoders outputs and synchronises the computation of controller outputs (u) and periodic digital-to-analog (D/A) conversion.

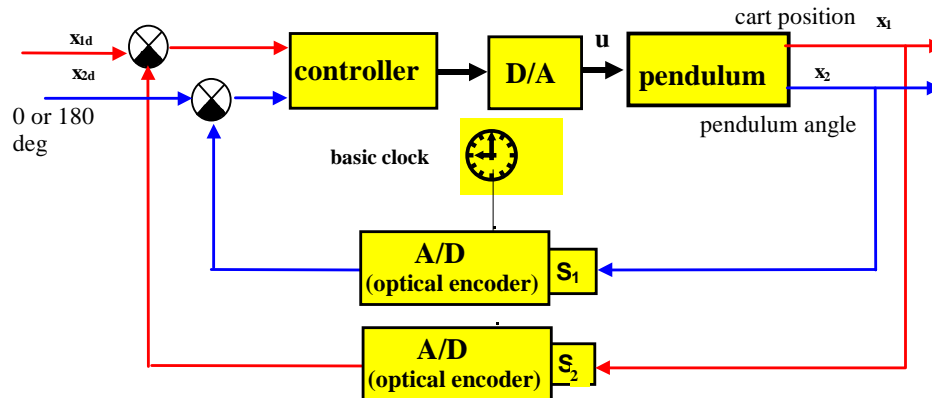


Fig. 4.2 Digital control of the pendulum-cart system (basic block diagram).

The pendulum-cart system is controlled in real-time. The term "real-time" is often used but seldom defined. One possible definition is [3]:

"Real-time is the operating mode of a computer system in which the programs for the processing of data arriving from the outside are permanently ready, so that their results will be available within predetermined periods of time; the arrival times of the data can be randomly distributed or be already determined depending on the different applications."

The real-time software for pendulum control is structured around particular internal signals (**events**) into a set of **tasks**. Each task implements the processing required by a corresponding event. A **task scheduler** recognises the events and activates or suspends the tasks. In the simplest case, when all tasks require processing at the same frequency, a sequential organisation of the tasks can be implemented (Fig. 4.3).

The time frame of each task is fixed. It is assumed that the longest task job takes no longer than the period of time generated by the software timer.

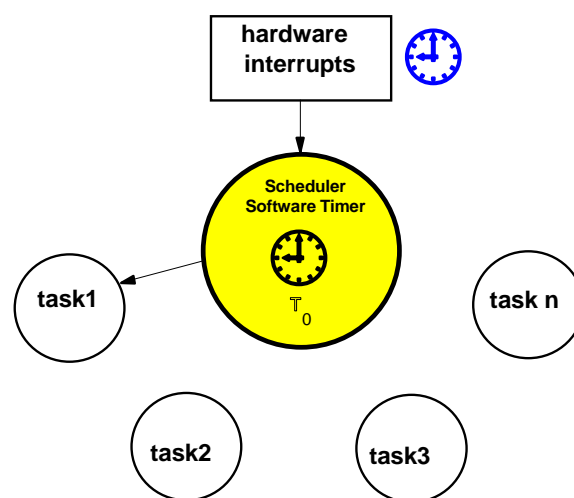


Fig. 4.3 Sequential real-time system



A list of real-time tasks creating an environment suitable for developing pendulum-cart digital controllers is given in Table 2.

Table 2. Real-time tasks for pendulum-cart control system.

data acquisition task	reads analogue and digital inputs from all input lines.
control task	calculates the control output
output task	transfers the controls to the output devices.
communication task	transmits the shared data between RTK and <i>MATLAB</i>
self-diagnostic task	checks the operation of the scheduler .

Real time tasks and the task scheduler are organised in the form of a real-time kernel (RTK) which supervises the set of real-time tasks. Several control algorithms can be embedded in the real-time kernel which is activated by a hardware clock. Selection of the control algorithms and tuning of their parameters is made by means of the communication interface from the *MATLAB* environment. Two levels of priority are determined: higher priority for real-time tasks and lower priority for MS-WINDOWS tasks (Fig. 4.4).

When RTK tasks are performed, the code allocation mechanism of MS-WINDOWS is blocked. Time slots available between the processing of higher priority tasks are used for tasks of lower priority. The communication interface is also used for RTK configuration (setting of sampling period, tuning of the excitation source etc.). A cyclic memory buffer is created in order to enable process data flow between the RTK and the *MATLAB* environment.

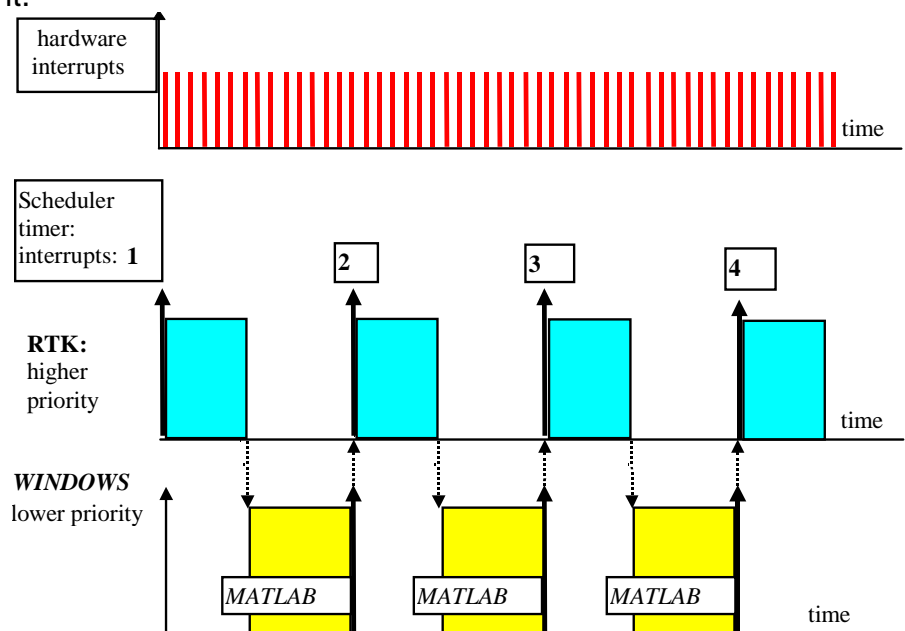


Fig. 4.4 Multitasking in the *MS-DOS/WINDOWS* environment.



DIGITAL PENDULUM SYSTEM GETTING STARTED

CHAPTER 4

COMPUTER CONTROL

Fig. 4.5 illustrates, how the blocks of the control system from Fig. 4.2 are located in the hardware & software environment of the pendulum control system.

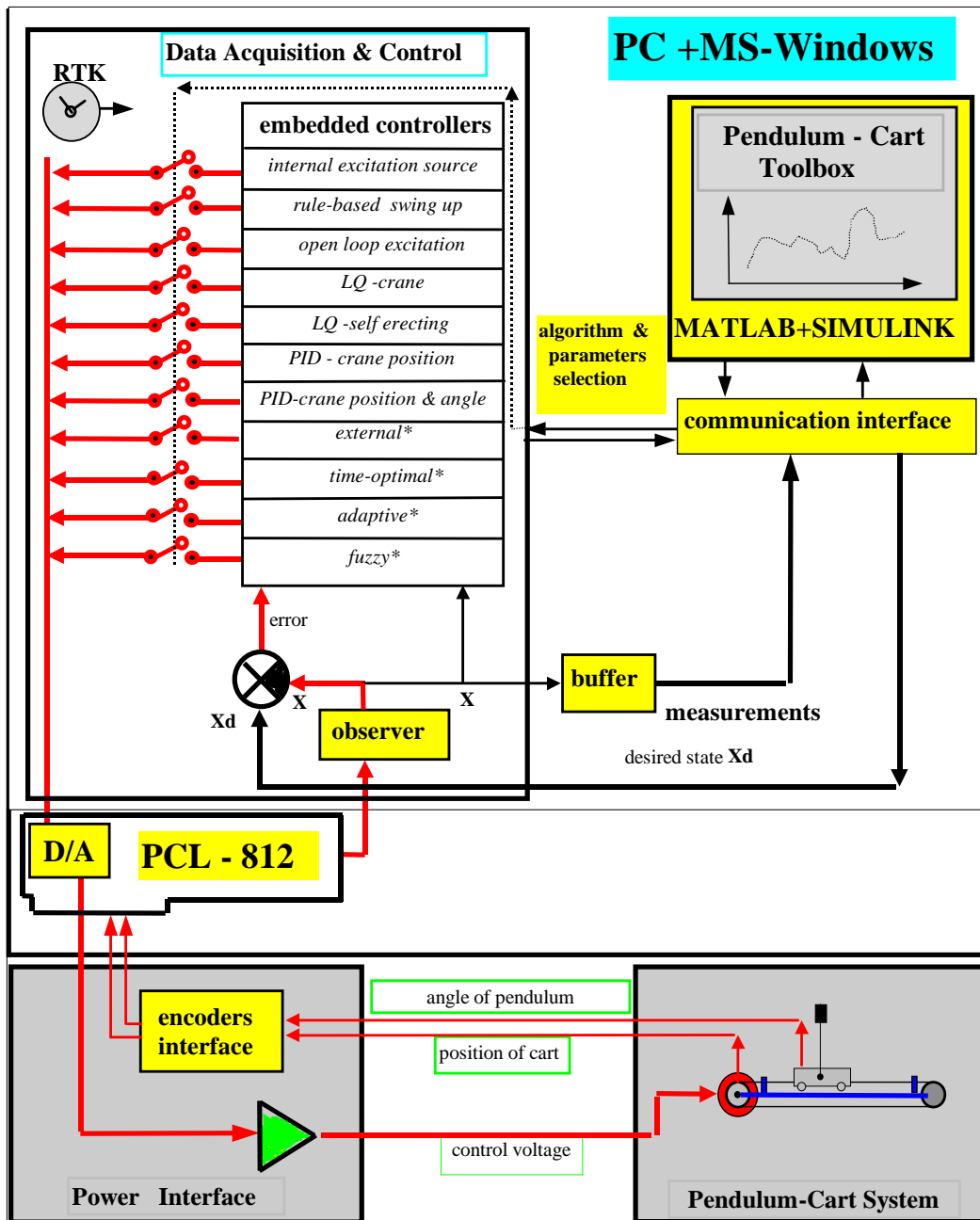


Fig. 4.5 Digital control of the pendulum (hardware and software configuration).

* will be added in new toolboxes.



5. THE DIGITAL PENDULUM SYSTEM

The pendulum-cart system consists of:

- PC computer equipped with I/O board,
- pendulum and cart,
- encoder interface and DC motor driver ,
- control software

The following basic PC configuration is required:

5.1 Hardware:

IBM-PC 486DX or higher compatible computer:, VGA graphics, 16 MB memory, mouse, PCL-812 data acquisition board.

PCL 812 PC-Card.

The system is equipped with a data acquisition card for controlling the model. The card is installed in the PC slot. Key features are:

- 12 bit A/D conversion. The maximum A/D sampling rate is 30kHz in DMA mode.
- 16 single-ended analogue input channels with 12 bits resolution.
- Switch selectable versatile analogue input ranges.
Bipolar: +/-1V, +/-2V, +/-5V, +/-10V.
- Two 12 bit monolithic multiplying D/A output channels with 12 bits resolution. An output range of from 0 to +5 can be created by using the on board -5V reference. External AC or DC references can also be used to generate other D/A output ranges.
- 16 TTL/DTL compatible digital input, and 16 digital output channels.

5.2 Software:

MS-WINDOWS 95 or NT, *MATLAB* version 5.x with *Simulink* ver.2.0, *Signal Processing Toolbox* and *Control Toolbox* from *MathWorks Inc.*

The control software for the pendulum-cart system consists of:

- **Real-time kernel (RTK)**
- **Pendulum Toolbox.**

The Real-time kernel supervises real-time tasks, creating a feedback control structure.

The Pendulum Toolbox is a collection of M-functions and C-code Mex-files that extends the MATLAB environment in order to solve pendulum modelling/design/control problems. A special memory buffer is created to enable process data flow between the real-time kernel and toolbox functions. See the Reference Manual, 33-005-2, for a description of the toolbox structure and functions.



5.3 Pendulum, Cart, Encoder interface and DC motor driver

The sensors attached to the pendulum-cart set-up, in combination with an appropriate converter, detect the angle of the pendulum and the position of the cart. The pendulum-cart set-up utilises two optical encoders as angle and position detectors (Fig.5.1). The first one is installed on the pendulum axis, the second on the DC motor axis. An optical encoder basically consists of a light source for emitting light, a light receiving device, and a rotary code disk with slits (Fig.5.2). The rotary disc is placed between the light source and the light receiving device. The optical encoder obtains the number of pulses which is proportional to the angle of rotation of the disk. The value of the rotation position of the disk is obtained by using a counter to accumulate the output pulses of the encoder.

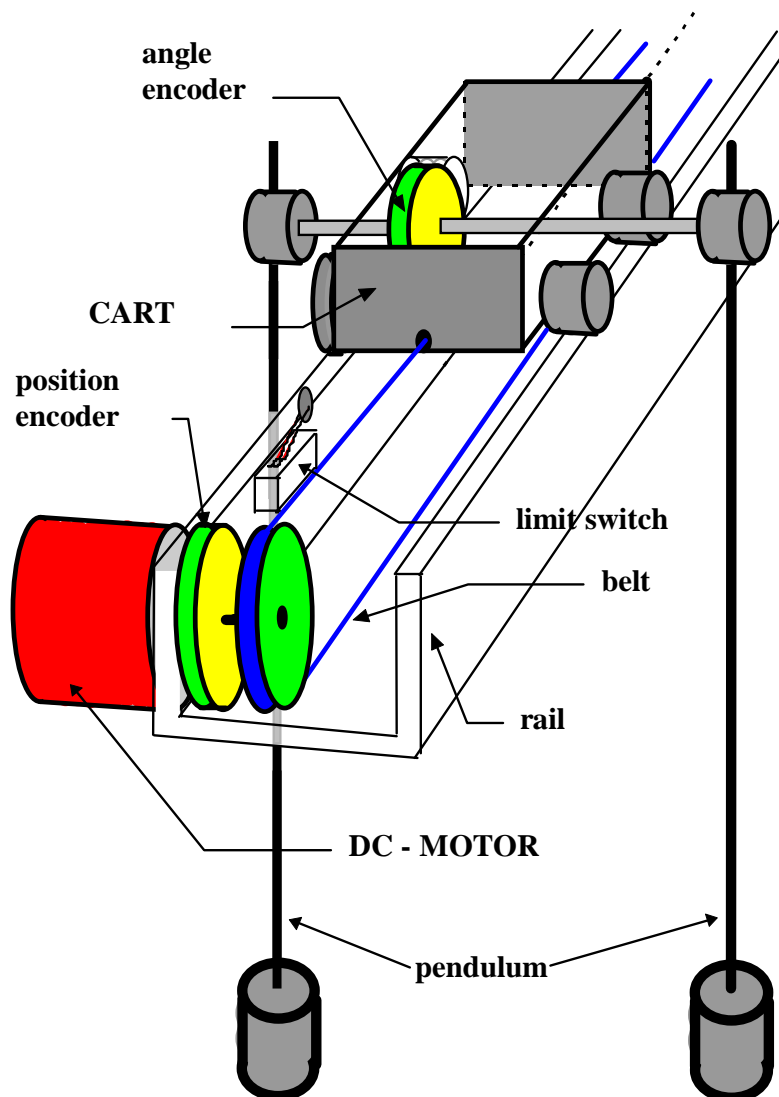


Fig. 5.1 Mechanical part and sensors of pendulum-cart set-up



DIGITAL PENDULUM SYSTEM GETTING STARTED

CHAPTER 5

THE DIGITAL PENDULUM SYSTEM

The pendulum-cart set-up uses incremental encoders. Fig. 5.2 illustrates the principle of determining the direction of rotation for an incremental optical encoder. The light beams emitted by two light sources (A and B) go through two rings of slits on the disc. The slits have a phase difference, so that the electric output of the receivers (A and B) are rectangular waves with a phase difference. The sign of the phase difference allows the direction of rotation to be determined.

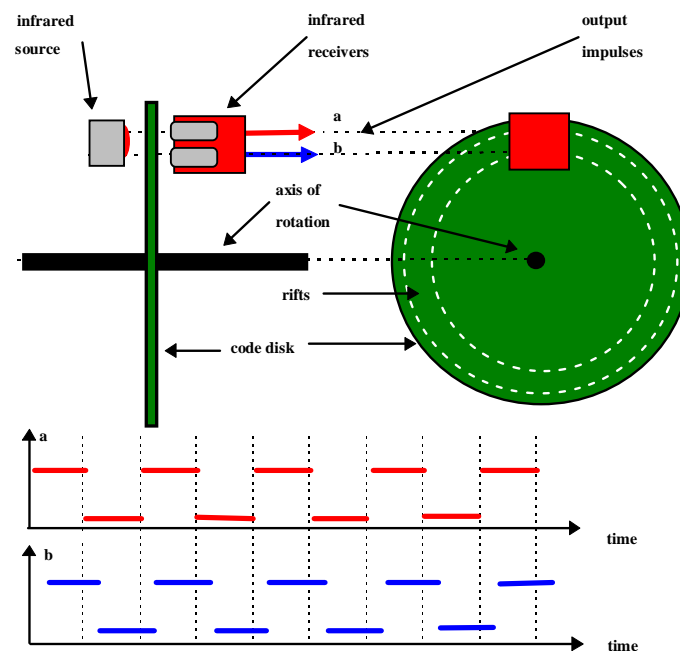


Fig. 5.2 Position Sensor

The following point should be noted when using an incremental encoder. After each experiment or power switching the cart should be moved to the centre of the rail before starting the next experiment, to allow the zero value of the position to be set.

The control signal flows from the computer through the D/A converter of the data acquisition board. The D/A output is wired to the power amplifier input which drives the DC motor. The power amplifier and encoder interface are located in the *Digital Pendulum Controller* box. This box is equipped with two switches: the main power switch and the switch for cutting off the DC motor power. At the rail ends there are two limit-switches which cut off the DC motor power when the cart overruns the limit points (Fig. 5.3).

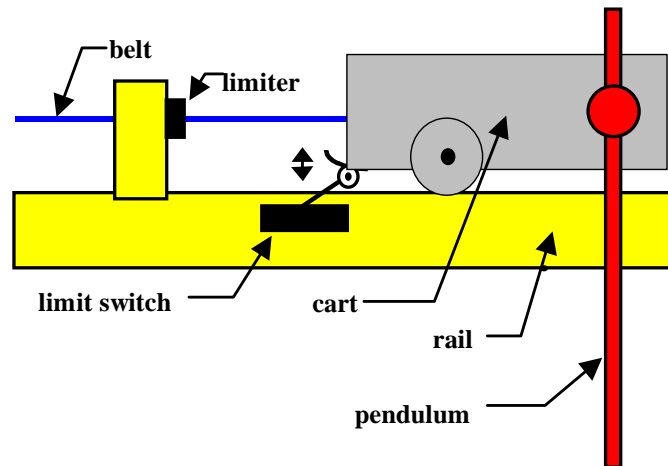


Fig. 5.3 Sensors of the limit points

5.4 Operation of the system

Before starting any experiments, perform the following steps:

- Make sure that the green *Start* button on the controller box is in its **off** position. (not lit)
- Switch-on the *power* switch on the controller box
- Bring the cart to the centre of the rail and stop oscillations of the pole.
- Reset the encoders(from Main Pendulum Menu in MATLAB)
- Press the green button motor start switch on the controller box
- Start the controlling software (e.g. the Simulink simulation) for an experiment



6. STARTING, STOPPING AND TESTING

6.1 Starting procedure

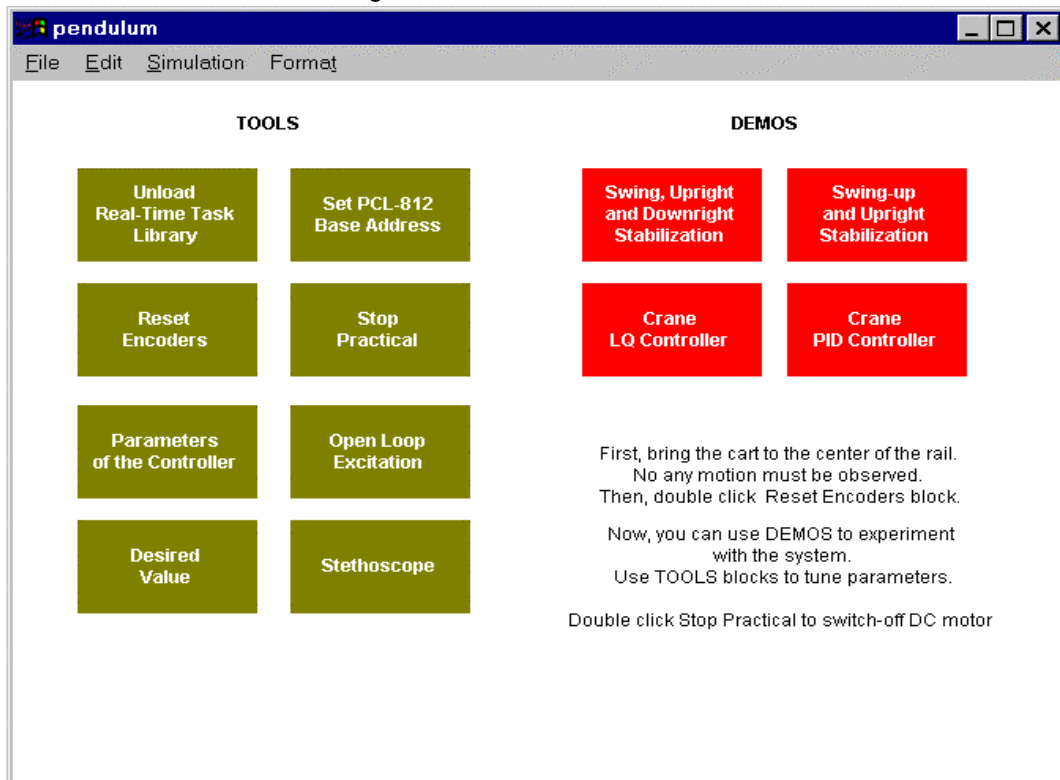
This manual is presented through a tutorial that closely follows the demonstrations in the M-file *pendulum*.

In the MS-WINDOWS environment start MATLAB by double clicking on the MATLAB icon. The MATLAB Command Window opens. Type:

pendulum

and then the window “pendulum” opens (see Fig. 6.1)

Fig. 6.1 The main control window



Go through the following steps:

- double click on the *Set PCL-812 Base Address* block. If your data acquisition board address agrees with the default value then accept it. Otherwise enter the address that you set on PCL-812 board during the installation,
- double click on the *Reset Encoders* block.



Now you are ready to start a practical.

Go through the following steps to avoid damaging your hardware:

- Choose one of the *DEMOS*. For example double click on the *Crane PID Controller* block. The window below (Fig. 6.2) will appear: Several simulated oscilloscopes ('scopes') are shown. Double clicking any of these displays opens window representing the scope. Scopes shown with more than one input will display more than one graphic trace.

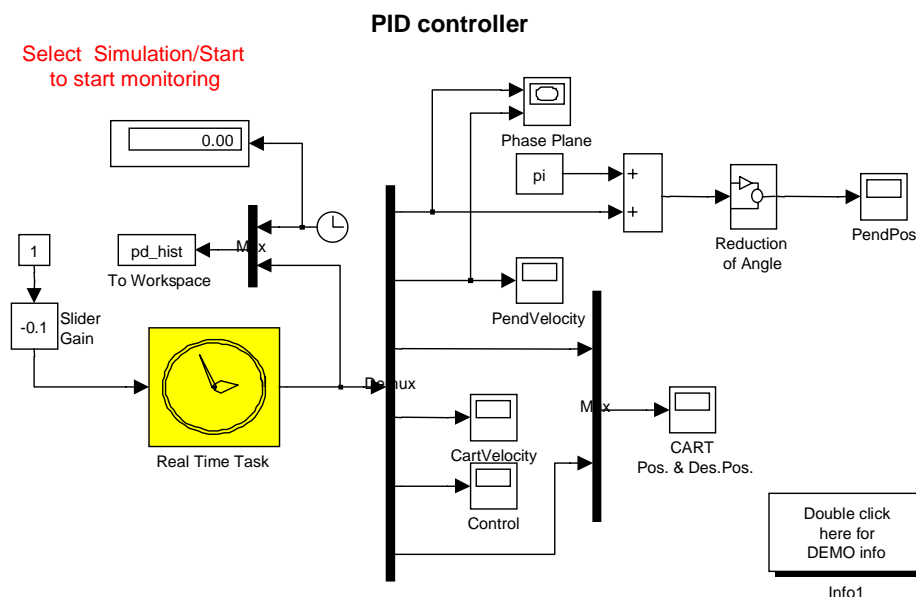


Fig. 6.2 Crane PID controller window

- First, check if measurements of angle and cart position are being transmitted. Click on the *Simulation* menu and select *Start* bar. Move the cart and rotate the pole by hand. This action should be visible in the scopes and graph windows of the screen (see Fig. 6.3).

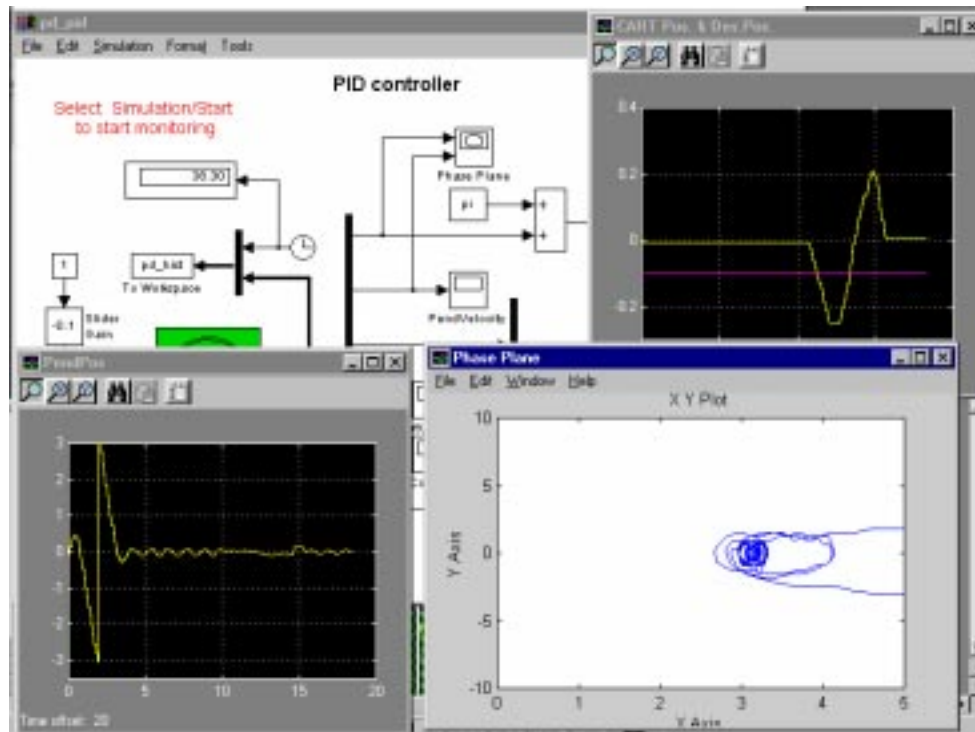


Fig. 6.3 Testing measurement signals. The DC motor is off.

Notice that the position of the cart changes (see the *CartPos* scope). The pendulum rotates about the $+\pi$ and $-\pi$ equilibrium points (look at the *Angle* and the *Phase Plane Angle/Velocity* graphic windows).

- Move the cart to the central position on the rail. Press down and hold one of the limit switches at the endpoints of the rail. The motor is disconnected.
- Switch-on the motor with the switch mounted on the front cover of the pendulum controller box.
- Release the limit switch being prepared for pressing it again, if you notice an unexpected behaviour of the system.

Usually the system works properly. The last precautions are not necessary if you are more familiar with the behaviour of the pendulum-cart system.

At the end of this first experiment check the crane mode. Double click on the *Slider Gain* block. The following window opens (Fig. 6.4)

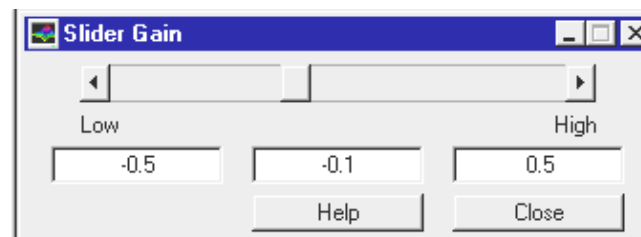


Fig. 6.4 Slider for reference position of the cart.

Dragging the slider results in the cart travelling from the initial to a new position.

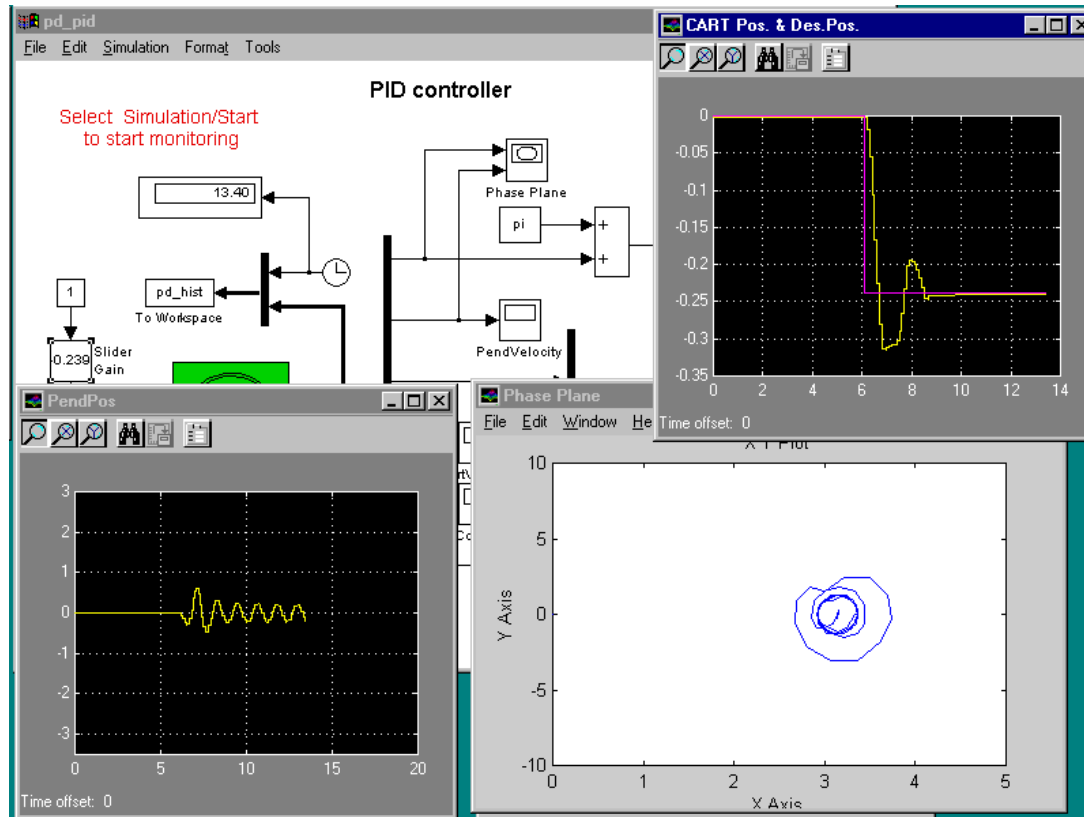


Fig. 6.5. Window of crane motion.

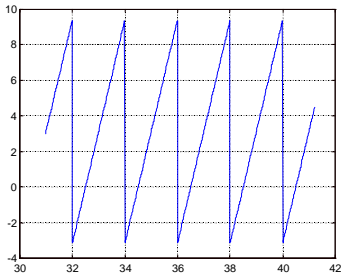
6.2 Testing and troubleshooting

We give a number of hints to avoid the most common faults that can occur during experiments with the system (see Table 6.1).

Table 6.1. Troubleshooting

Faulty action	Solution
Lack of signals or positive control feedback actions are observed on the screen	Refer to the Installation and Commissioning Manual 33-005-0M5
Constant value signals are observed or MS-WINDOWS system fails	Double click on <i>Set PCL-812 Base Address</i> block (<i>Main Control Window</i>). Check the value of address.



The cart is in the central position on the rail, no motion is observed but the corresponding displayed signals in the scope windows show that the cart position is not equal to zero, or the pendulum angle is not equal to $(2k+1) \pi$	Double click on <i>Reset Encoders</i> block
If a test is repeated and the signals are incorrect	Check if all parts of the system are fitted and screwed tightly. Pay attention to the encoders and the pole mountings.
An unexpected bumping of the cart into the rail bumpers has occurred.	Check correctness of measurements. If necessary reset encoders.
MS-WINDOWS system fails after starting one of the algorithms	<code>pd_call('SetSampleTime',new_value)</code> . Your computer is too slow to perform this action. Increase the <i>new_value</i> .
Moving the cart or pendulum results in the following time diagram, whatever the real motion pattern of the cart or pendulum was. 	Your base address is set to zero. Set it correctly.
After starting the simulation all signals have the expected correct shapes, but switching on the DC motor results in unpredictable behaviour of the system.	Check hardware

6.3 Stopping procedure

The practical may be stopped any time. Double click on the *Stop Practical* block in the *main control window*. If you wish to stop the visualisation process click once on the Stop bar in the Simulation menu. The real time library may be unloaded by double clicking the **Unload Real-Time library** block, but it is automatically unloaded when you close MATLAB.



**DIGITAL PENDULUM SYSTEM
GETTING STARTED**

CHAPTER 6

STARTING, STOPPING AND TESTING

Notes



7. FAMILIARISATION USING DEMOS

7.1 Crane PID controller demo

All the crane demos presented in this section deal with the same task: move the cart to an arbitrary chosen position (Fig.7.1).

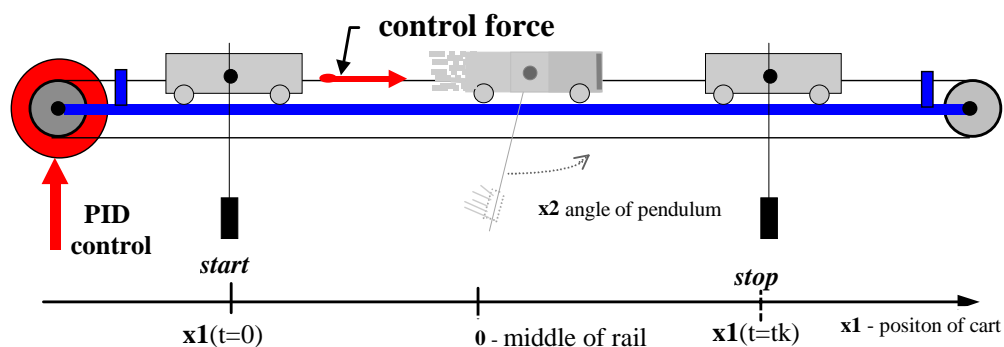


Fig. 7.1 Crane PID control problem

We start from the simple Crane demo to get some experience with the graphical Simulink software. The control objective is to move the cart along the rail from one point to another. A PID controller reducing the cart position error is used. The standard form of the screen is presented in Fig. 7.2. We can configure this screen by closing or opening scopes or by moving and changing their dimensions. The desired value of the cart position can be changed by dragging the slider. This is our simplest interaction with the stabilising PID controller. In Fig. 7.2 one can observe the results of such an experiment.

In the *DesPos* scope the desired value is defined by *Slider Gain*. The response of the system is visible in the other scopes and XY Plot. The control objective i.e. stabilisation in the new position is achieved after a few oscillations. The time response of the PID controller is also visible.

At the beginning we omitted a detailed description of all options offered by this demo. The parameters of the PID controller are set to default values, but may be tuned by opening the *Parameters of the Controller* block in the main menu of SETUP. It will be described later. Refer to the Teaching Manual - 33-005-4, where a more detailed description of the PID and other types of controllers, together with respective experiments, is given.

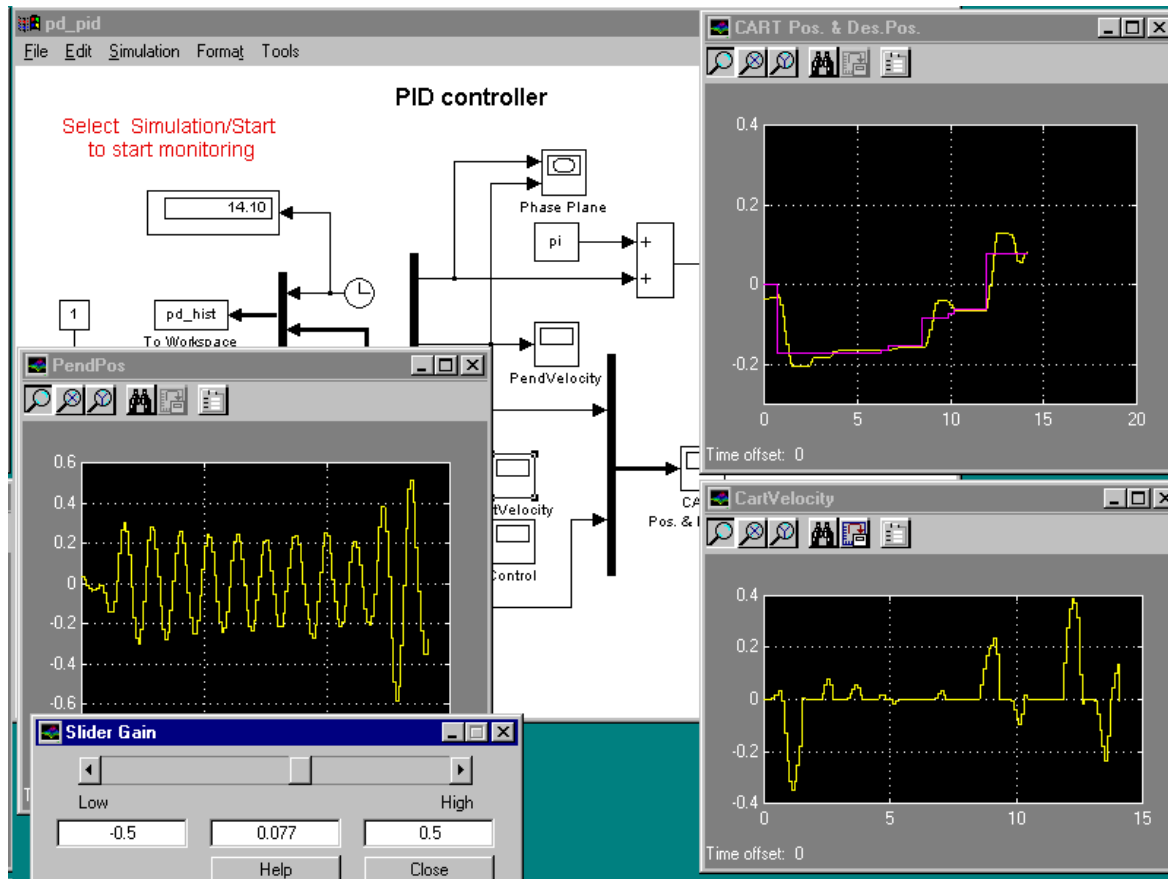


Fig. 7.2 First experiment - Crane PID controller

7.2 Swinging-up and Upright Stabilisation Demo

The second experiment is concerned with upright stabilisation. First, the pendulum is steered to its upright unstable position and then is kept erected by the Linear-Quadratic (LQ) controller.

Domains of the stabilising and swinging algorithms are shown in Fig.1.2.

Click on the *Swing-up* and *Upright Stabilisation* block in the main menu. The following window appears (Fig 7.3).

Note that the *Constant* block and *Slider Gain* block are now substituted by the external *Desired Position Generator*. The *To workspace* block is added to collect state and control data in the workspace of MATLAB (it is easy to plot them afterwards).



DIGITAL PENDULUM SYSTEM GETTING STARTED

CHAPTER 7

FAMILIARISATION USING DEMOS

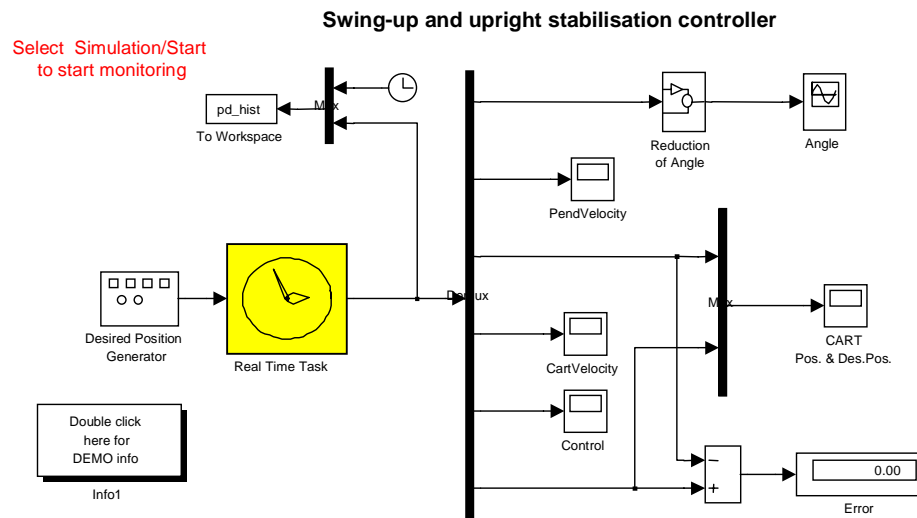


Fig. 7.3 Upright stabilisation window

The operation of the linear-quadratic controller is presented in Fig.7.4. You can observe certain differences between the configuration of this window and the previous one. However scopes and graphical tools can be added and removed from the SIMULINK library.

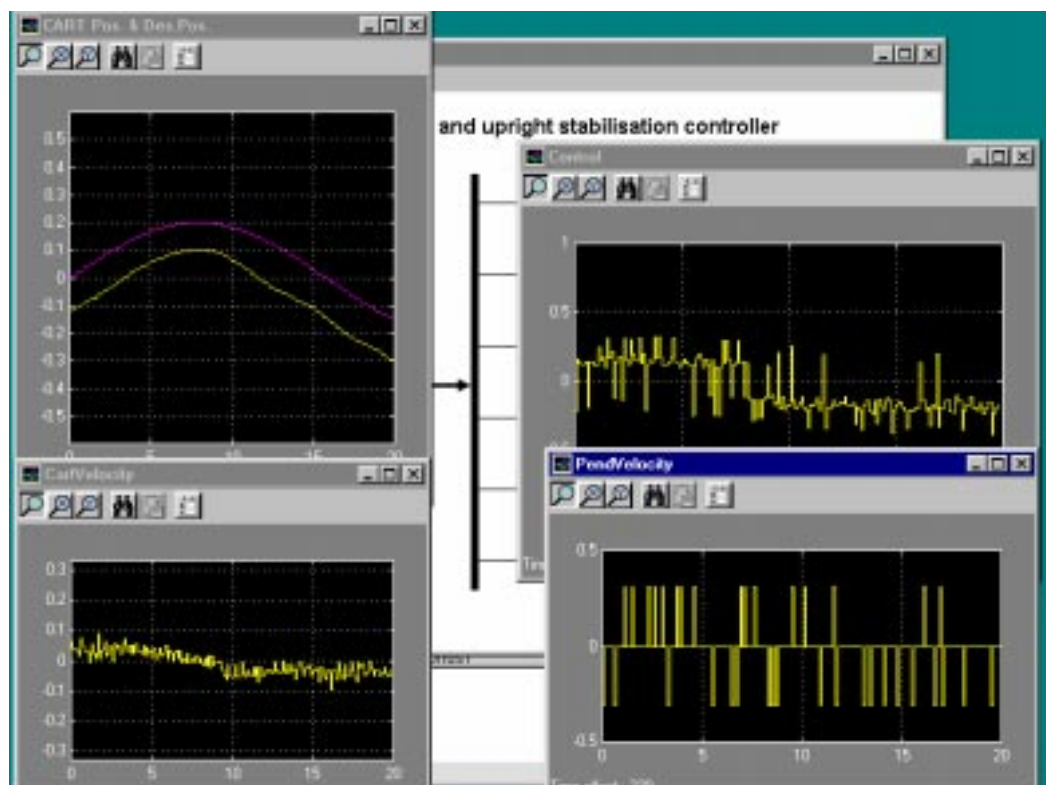


Fig. 7.4 Upright Linear-Quadratic control



You can start the experiment at the down position of the pendulum or you can rotate it by hand to the upright unstable position. The *Desired Position Generator* is set as a low frequency sinusoidal generator. It is active when the external generator usage flag is set to **on** (Fig. 7.5). Double click on the Real Time Task block then the following mask opens. Check if the external generator usage flag is set to **on**.

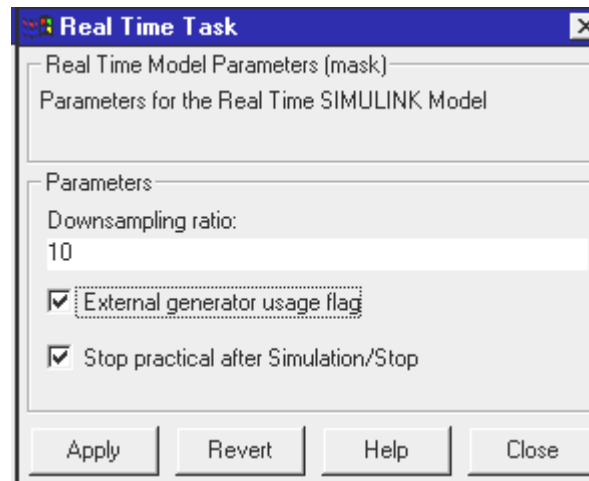


Fig 7.5 Real Model Parameters mask

Notice also that the *Stop practical after Simulation/Stop* flag is on. It means that practical will be stopped when the simulation is stopped.

The cart position is following the desired position. Simultaneously the cart is balancing the pole in its upright unstable position. In the left hand corner scope we observe the controller error as the difference between two signals. It is worthwhile to compare it with the *Control* scope and the graph of angle in the upright position.

7.3 Crane LQ controller demo

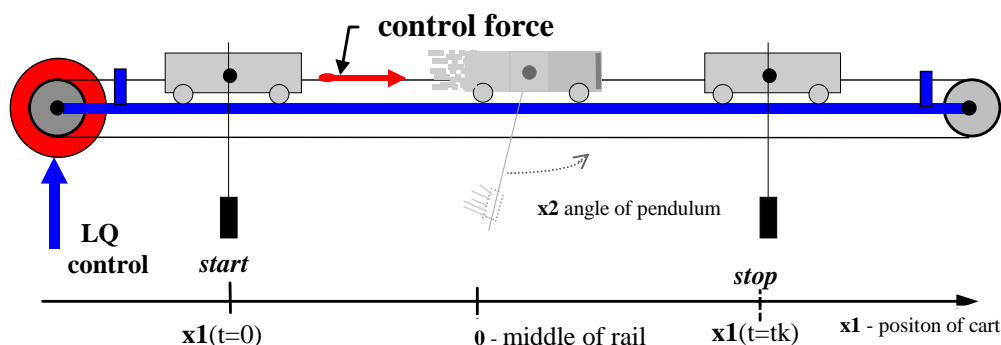


Fig. 7.6 Crane LQ control problem.



In the third experiment we are concerned with the crane problem again (as described in Section 7.1). This time we use an LQ controller (Fig. 7.6) instead of the PID controller. After clicking on *Crane LQ Controller* block the following window opens (Fig. 7.7). The *Desired Position Generator* is set to generate a saw tooth signal. The cart is tracking this signal with a LQ strategy of control.

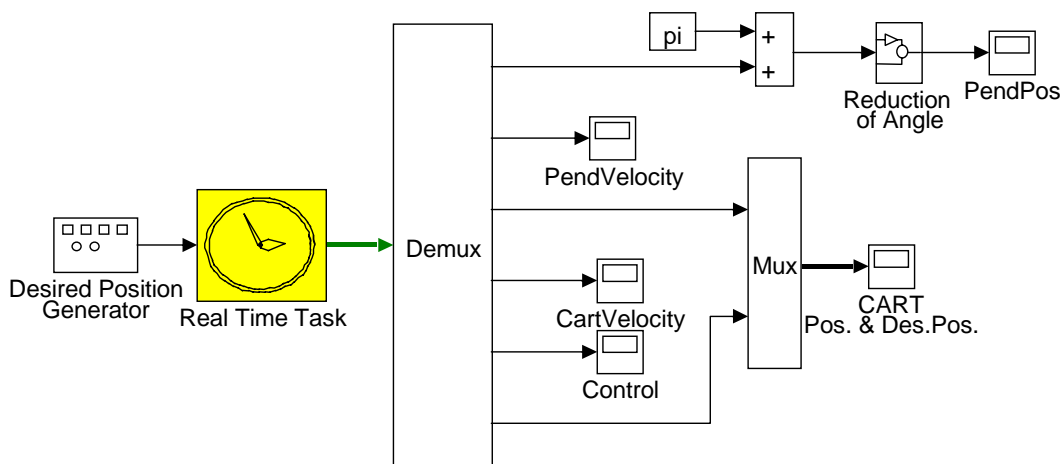


Fig. 7.7 Down stabilisation window

The records of this experiment are given in Fig. 7.8. In the *Control* scope we observe saturation of the controller. In these short time intervals when an abrupt change of desired position occurs the controller does not follow the LQ algorithm exactly.

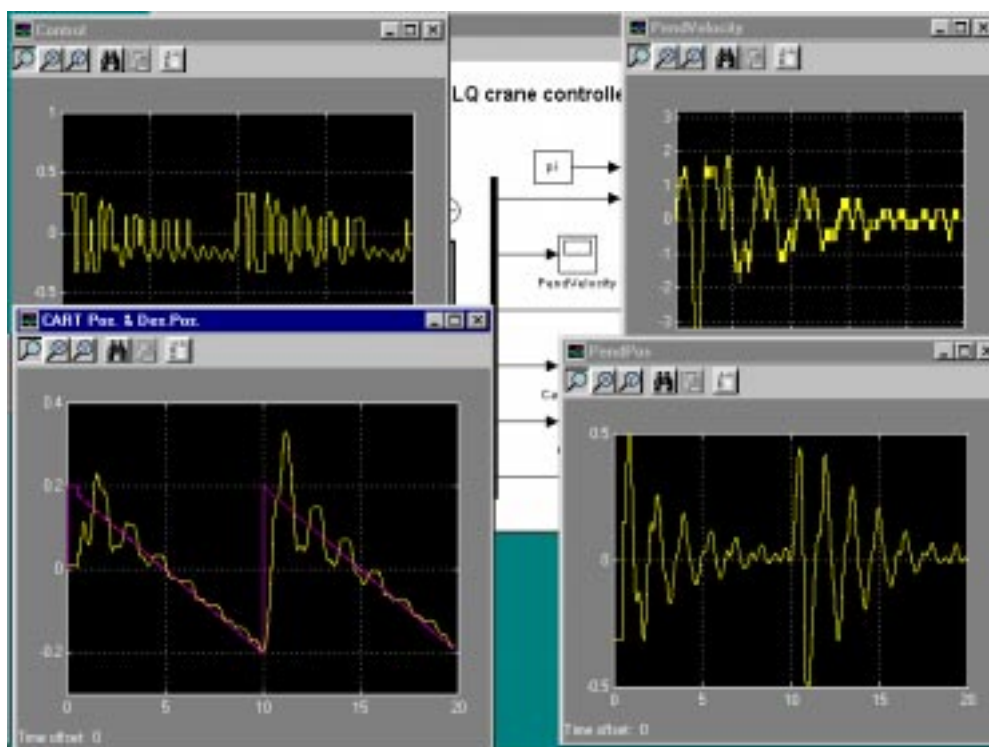


Fig. 7.8 Linear-Quadratic crane control.



**DIGITAL PENDULUM SYSTEM
GETTING STARTED**

CHAPTER 7

FAMILIARISATION USING DEMOS

Notes



8. TUNING OF PARAMETERS

8.1 Swing, Upright and Down Stabilisation demo

This is the most complex demo. Double click on the *Swing, Upright and Down Stabilisation* block in the main *pendulum* menu and the following window will open (Fig 8.1).

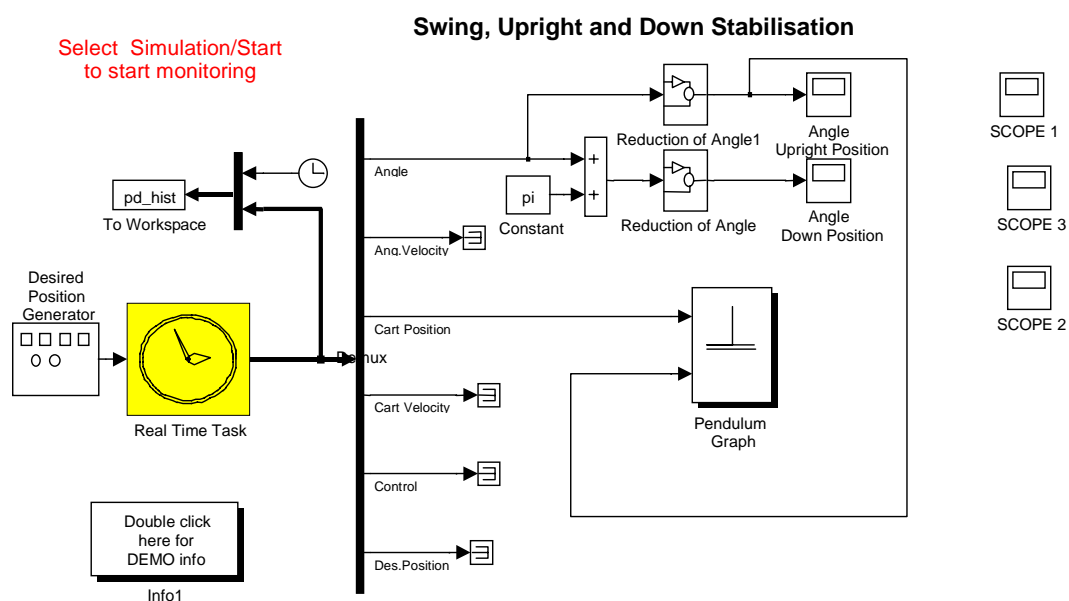


Fig. 8.1 Swing, Upright and Down Stabilisation window

Now we describe the main blocks of this window. Double clicking on the *Reduction of Angle* or *Reduction of Angle1* blocks show the interior of these blocks (Fig. 8.2).

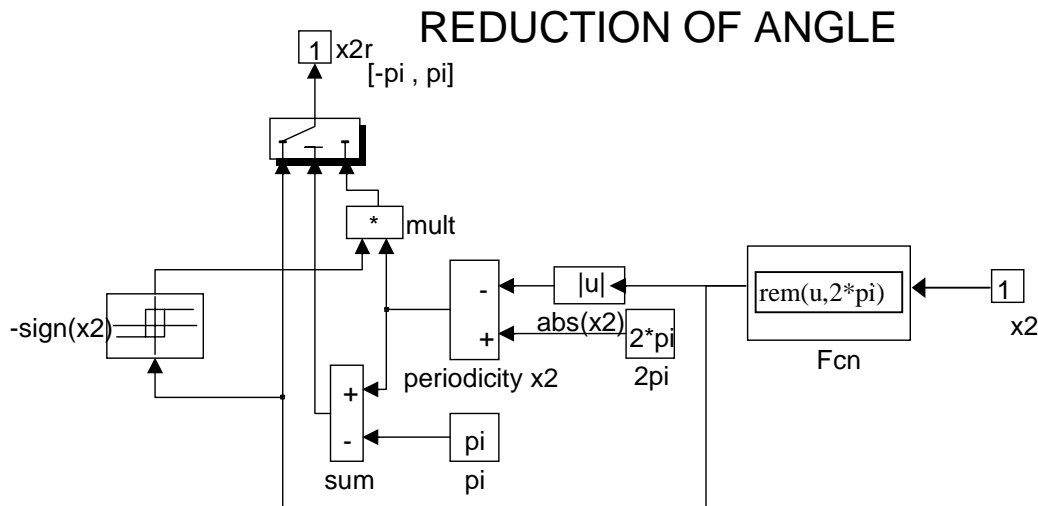


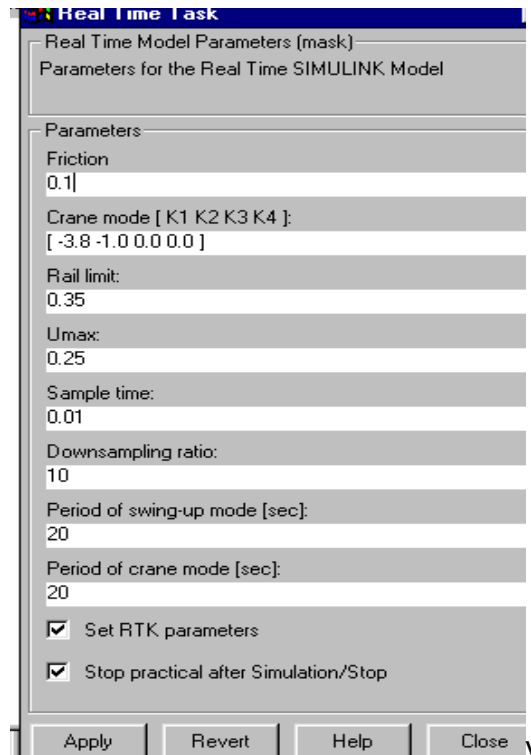
Fig. 8.2 Reduction of angle block

In this block an arbitrary angle measured in radians is reduced to the interval $[-\pi, +\pi]$. It is easy to see from Fig. 8.2 how this task is performed. However, we list some comments here to clarify its operation.

The MATLAB function *REM* (remainder after division) has the following definition: the remainder $REM(x,y)$ is $x - n \cdot y$ where $n = \text{fix}(x./y)$ is the integer part of the quotient $x./y$.

An arbitrary angle is reduced to the interval $(-2\pi, +2\pi)$ at the output of FCN block REM. If the transformed value of angle is inside the interval $(-\pi, +\pi)$ then it is not transformed further. If the angle value is outside $(-\pi, +\pi)$ interval then the following blocks become active: *periodicity x2*, *-sign(x2)* and *mult*. These blocks map the interval $[-2\pi, -\pi]$ into $[0, +\pi]$ and $[\pi, +2\pi]$ into $[-\pi, 0]$ respectively.

Double-clicking on *Real Time Task* block opens the dialog box (Fig. 8.3). This block is masked.

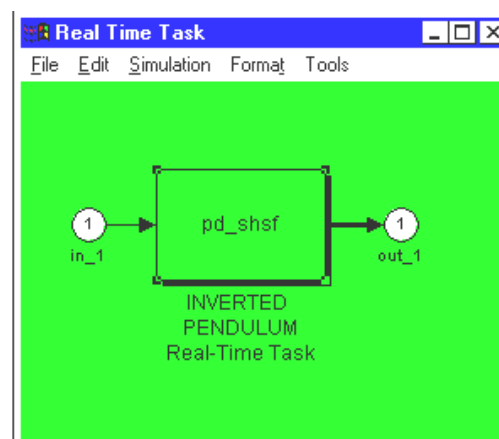


*Double click the
Real time task
block in Fig 8.1*

Fig. 8.3 Mask of *Real Time Task* block

If it is unmasked the following window appears (Fig. 8.4). (to unmask it go to the Edit menu and select “**Look Under Mask**”).

The detailed description of the S-function *sf_show* which is embedded in the *Real Time Task* block is given in the Reference Manual – 33-005-2 (S-function – Example 1).



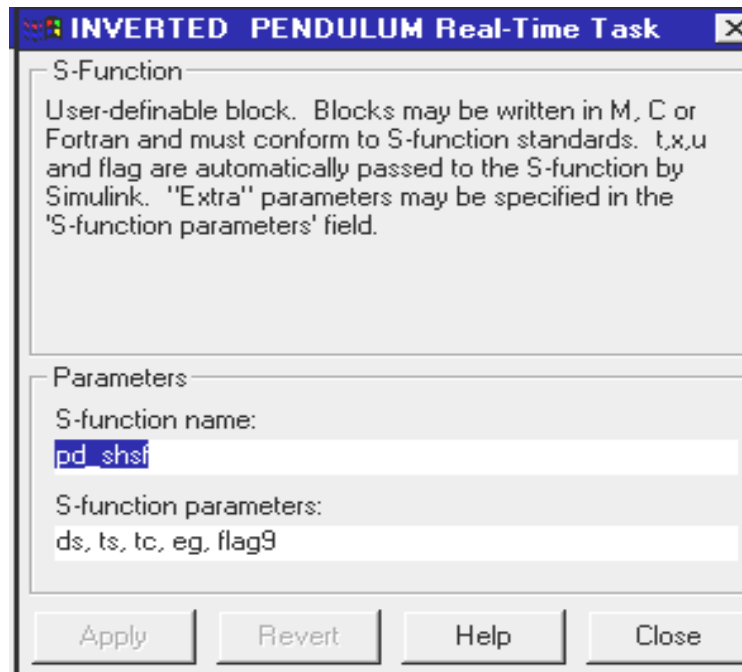
*Edit menu
“Look Under Mask”*

Fig. 8.4 Unmasked *Real Time Task* block

To convert the S-function *sf_show* into a block, we put its name into the S-function dialogue box in the Subsystem function name field (Fig. 8.5). (To get this dialogue box double click the block in Fig 8.4.)

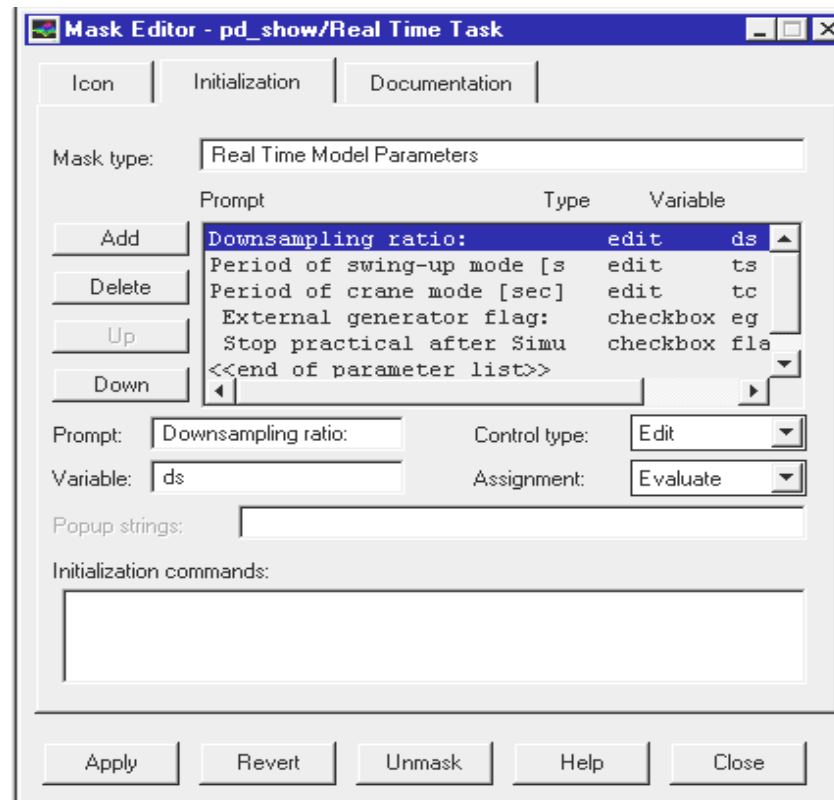


The Function parameters field allows additional parameters to pass to the S-function at each iteration (following the standard t, x, u and flag variables). In our case the parameters: ds , ts , tc , eg are passed. The definitions of these parameters are given in Fig. 8.6, obtained by selecting “**Edit Mask**” from the Edit menu.



*Double click the
block in Fig 8.4*

Fig. 8.5 *sf-show* embedded in function block



Edit menu
"Edit Mask"

Fig. 8.6 Mask block definitions

It should be remembered that it is a demo session and all programs, masked blocks of S-functions, ect. are set up for the particular demonstration.

If it is desired to go further and create a new demo involving additional parameters, different goals of control, etc., then the S-function *sf_show* should be modified and a new mask for it created. Any block can be masked, including subsystem and S-function blocks. The mask definition is given in Fig. 8.6. Mask modification is recommended only for advanced users.

If we wish to introduce a new parameter, for example named *lr*, into the mask presented in Fig. 8.6 then we write *length of rail* at the end in the *Dialog string* window separated by a vertical bar. Each character '|' results in writing the string following '|' in the next dialog window of the mask. We write *lr=@5;* in *Initialization command* window. This means that in fifth created window we will introduce a numeric value of parameter *lr* after masking. Now this new parameter can be used by *sf_show* s-function like any of the previous defined parameters: *ds*, *ts*, *tc* and *eg*.

As far as our demo is concerned we only focus on choosing numerical values of defined parameters. At our disposal there are four parameters: *Downsampling ratio -ds*, *Period of swing-up mode -ts*, *Period of crane mode -tc*. The numerical value 25 of the parameter *ds*



means that only 1 sample of 25 samples is transferred from the Real Time Task to the output of the S-function. This parameter is introduced to provide a trade off between the slow graphical visualisation process and the fast data collection process. The parameters t_s and t_c define the period in seconds of the upright and down stabilisation modes.

The *Pendulum Graph* block representing animation of the pole movement is described in the Reference Manual – 33-005-2 (S-function - Example 2).

Now we are ready to start the Swing, Upright and Down Stabilisation demo. If not already open, double click the *Swing, Upright and Down Stabilisation* block in the *pendulum* screen to open the following window (Fig. 8.7)

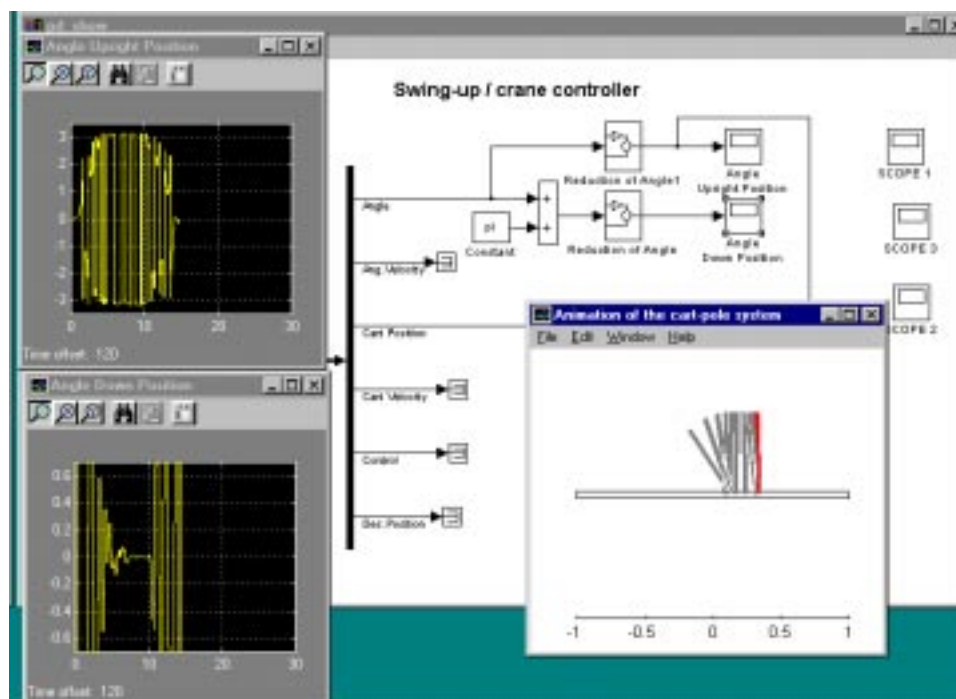


Fig. 8.7 Swing, Upright and Down Stabilisation window. Experiment 1

The window in the upper right hand corner of Fig. 8.7 shows an animated image of the pendulum. The pole is “dancing” on the rail. The time diagrams at the bottom of the screen correspond to the two different control objectives: upright, down and again upright stabilisation. In this experiment Desired Position for the cart is set from the Desired Position Generator block. Double clicking on the this block opens the following window (Fig. 8.8)

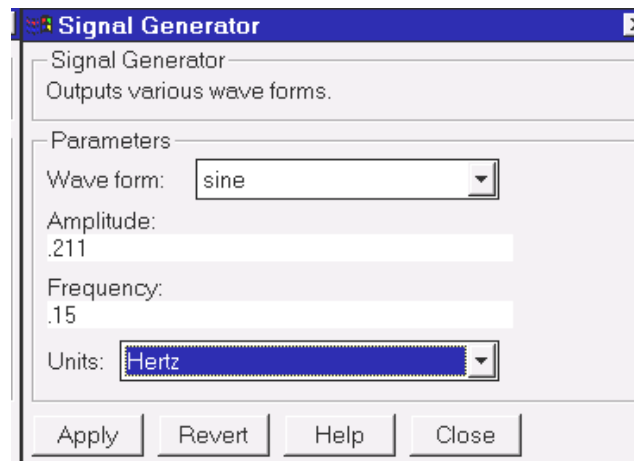


Fig. 8.8 Desired position generator setting

A sinusoidal wave of frequency 0.211Hz and peak value 0.15 is selected. The results of the experiment are given in Fig. 8.9.

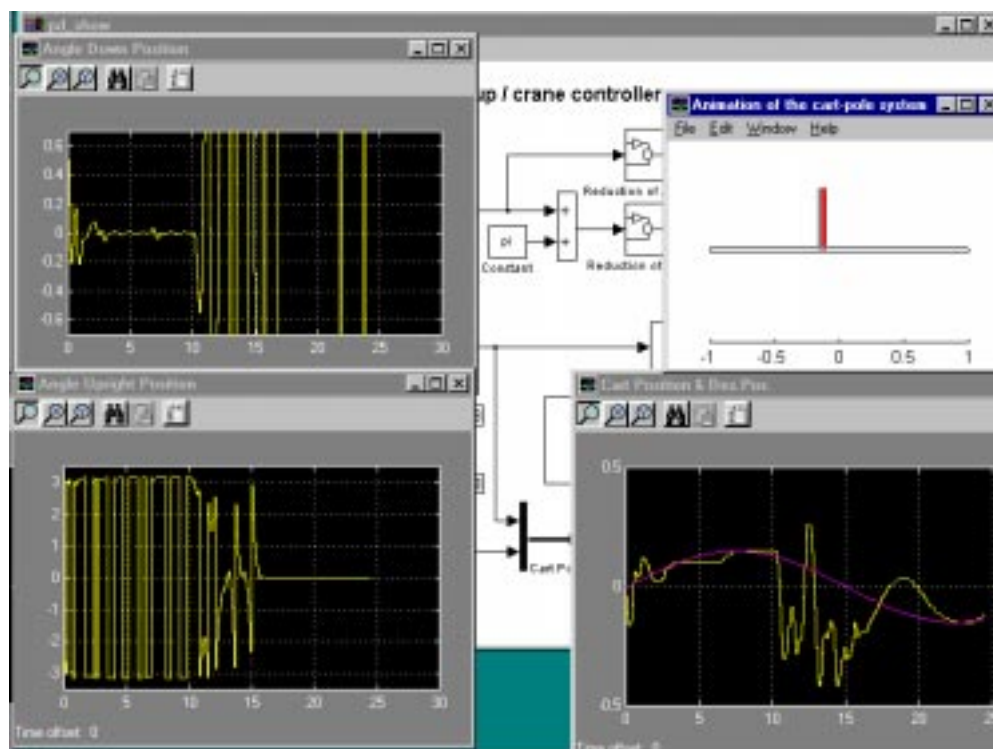


Fig. 8.9 Swing, Upright and Down Stabilisation window. Experiment 2

The cart is tracking the signal from the generator and simultaneously the control objective is achieved. In the *Cart's Position* scope it can be noticed that the sinusoidal shape is disturbed. This is the result of the trade off between the two control actions: balancing the pole in the upright unstable position and tracking the desired sinusoidally changing value of cart position.



The following experiments show how to tune parameters in this demo.

8.2 Definitions and setting of parameters

There are twenty parameters stored in the RTK in the vector form. To read this vector write the following in the MATLAB Command Window:

```
p=pd_call('getp');
```

where: p is a vector of 20 elements. Each parameter occupies a fixed position in the vector p .

To change the value of a parameter in this vector, write the command:

```
p(i)=new value of parameter
```

where $i=1 \dots 20$, for example $p(2)=1; p(3)=0.5$; the parameters $p(2)$ and $p(3)$ are changed.

and then

```
a=pd_call('setp',p);
```

 the vector p is sent to RTK

where p is a new value of this vector.

Most of the vector parameters are linked with the pendulum model and the control algorithms. Every parameter is called by its name. The list of these parameters is given below.

1. *Friction* - static friction (Coulomb friction). The value of this coefficient is equal to the minimum control value required to move the cart.

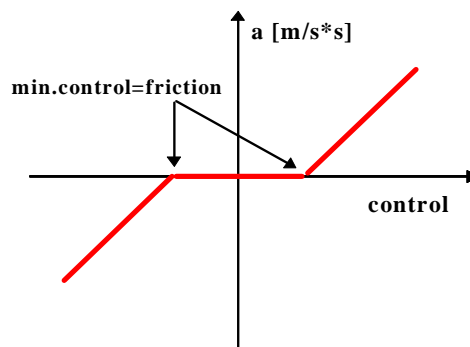


Fig. 8.10. Acceleration vs. control

To change this parameter call the following functions :



```
p=pd_call('getp');
p(1)= new value of the Friction % the first parameter of vector p is changed.
a=pd_call('setp',p);
```

2. K_1, K_2, K_3, K_4 - parameters of the LQ controller. Before setting these parameters the LQ controller must be selected.

Control calculated by the LQ algorithm is:

$$u = K \cdot X \quad (1)$$

where :

$X = [x_1 \ x_2 \ x_3 \ x_4]^T$ - is the vector of states: cart position, pendulum angle, cart velocity and pendulum angular velocity.

$K = [K_1 \ K_2 \ K_3 \ K_4]$ - weights of the control law (1) :

To introduce new values of parameters call the sequence of functions:

```
p=pd_call('SetAlgNo',0); % all algorithms are switched off
p=pd_call('getp');

p(2)= new value of K1;
p(3)= new value of K2;
p(4)= new value of K3;
p(5)= new value of K4;

a=pd_call('setp',p);
p=pd_call('SetAlgNo',2); % LQ control algorithm is chosen
```

3. *Rail Length* is the working zone of the cart. The range of this parameter is -0.6m to 0.6m. If the cart position exceeds the *Rail Length* value the limit control steers it back towards the rail centre. Note that suitable working length of rail is 0.3 m.

The following commands are executed to change the *Rail Length* value :

```
p=pd_call('getp');
p(6)=new value of rail length;
a=pd_call('setp',p);
```

4. *Max.* is the maximum control value. The range of this parameter in normalised form is from 0 to 1. If the control calculated by an algorithm exceeds the maximum value then the control value is set to the maximum control value. The default *Max.* value is set to 0.25.



The parameter *Max.* may be changed by the following sequence of commands:

```
p=pd_call('getp');
p(7)= 0.3;           % new value of Max.
a=pd_call('setp',p);
```

5. Let us denote: $W = \frac{I}{m}$, where:

I is inertia moment of the pendulum
 m is mass of the pendulum

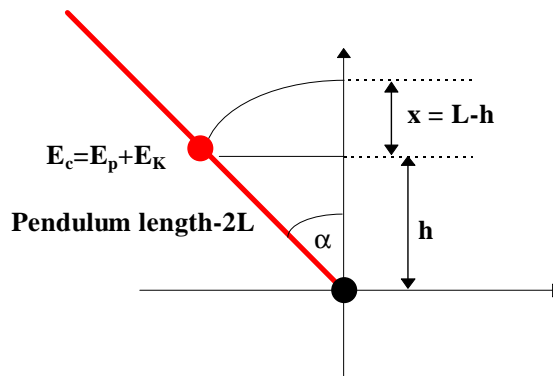


Fig. 8.11. Variables used in calculations of pendulum energy

$$x = L - h = L - L \cos \alpha = L(1 - \cos \alpha)$$

$$E_p = -mgL(1 - \cos \alpha) = mgL(\cos \alpha - 1) \text{ - is potential energy of pendulum,}$$

g - is gravity constant

$$E_k = \frac{I\omega^2}{2} \text{ - is kinetic energy of pendulum}$$

$$I = \frac{1}{3}m(2L)^2 = \frac{4}{3}mL^2$$

$$E_T = E_p + E_k \text{ - is total energy}$$

$$E_c = \frac{I\omega^2}{2} + mgL(\cos \alpha - 1) = 0 \text{ } \wedge (m)$$

$$\frac{I}{m} \frac{\omega^2}{2} = gL(1 - \cos \alpha)$$

$$W = \frac{m}{I} = \frac{\omega^2}{2gL(1 - \cos \alpha)} \text{ - is constant coefficient}$$

To obtain and plot data required for calculation of the parameter W execute the following commands (Fig. 8.12, Fig. 8.13):

```
h=pd_call('GetHistory');
```



```
plot(h(1,:)-h(1,1), h(2,:)); grid;    % angle vs. time
```

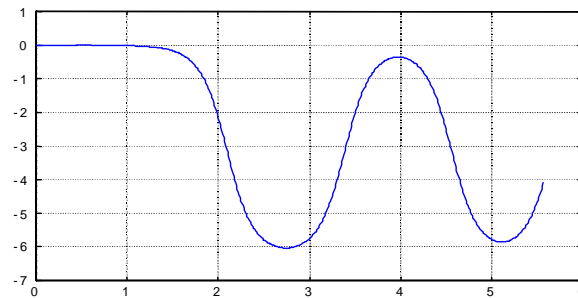


Fig. 8.12. Angle vs. time

```
plot(h(1,:)-h(1,1), h(3,:)); grid;    % angular velocity vs. time
```

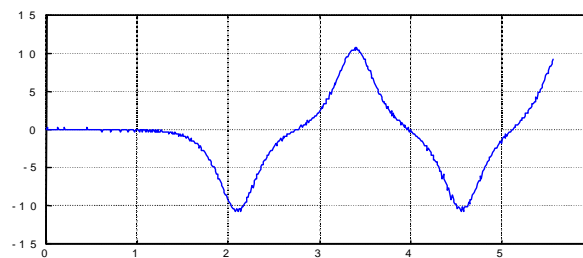


Fig. 8.13. Angular velocity vs. time

The following statements calculate and plot K vs. time (Fig. 8.14)

```
alpha=h(2,:);  
  
omega=h(3,:);  
for i=1:length(alpha)  
    W(i)=omega(i)^2/(2*g*L*(1-cos(alpha(i))));  
end  
plot(h(1,:)-h(1,1),W); grid
```

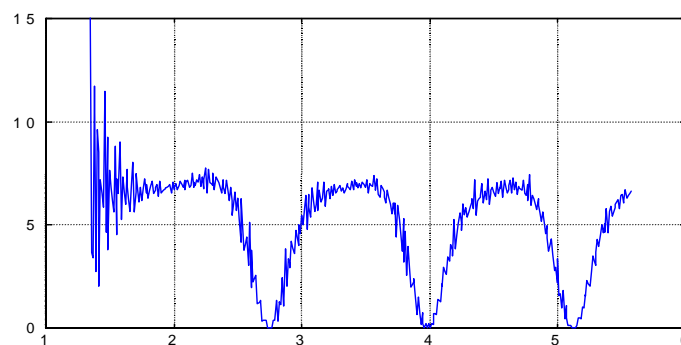


Fig. 8.14. The parameter W vs. time



It can be noticed that the W coefficient varies with time. The calculation errors are visible on the plot. The minimal calculation error of W is obtained for the maximal angular velocity. The time diagram of W is plotted in Fig. 8.15.

```
plot(h(1,150:250)-h(1,1), W(150:250)); grid
```

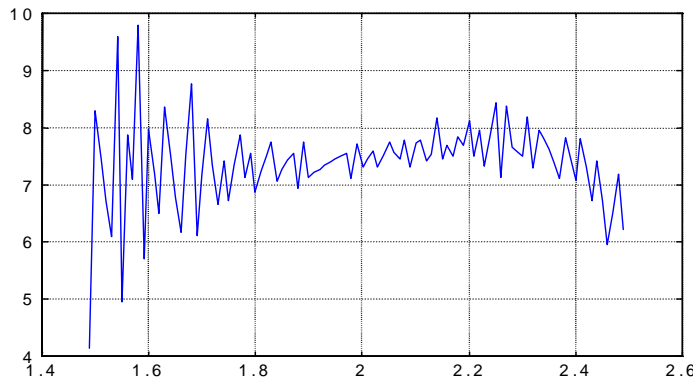


Fig. 8.15. The parameter W vs. time (subset of Fig.8.14.).

Fig. 8.15 shows a fragment of a same plot in an enlarged scale. In this time interval W is calculated with a minimum error.

The parameter W may be changed by the following sequence of commands:

```
p=pd_call('getp');
```

```
p(8)= new value of the W;
```

```
a=pd_call('setp',p);
```

6. R is the half length of the pole.

The parameter R may be changed by the following sequence of commands:

```
p=pd_call('getp');
```

```
p(9)= new value of the R;
```

```
a=pd_call('setp',p);
```

7. *Stab. zone* is the stabilisation zone in the upright position of the pendulum. LQ controller is active in this zone (Fig. 8.16)

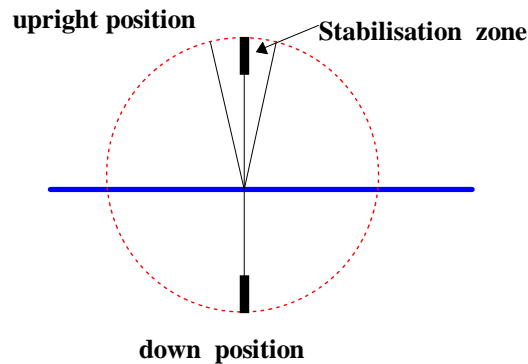


Fig. 8.16. Activity zones of two control algorithms.

The parameter *Stab. zone* may be changed by the following sequence of commands:

```
p=pd_call('SetAlgNo',0);    % all algorithms are switched off
p=pd_call('getp');
p(10)= new value of the Stab. zone;
a=pd_call('setp',p);
p=pd_call('SetAlgNo',2);    % LQ control algorithm is chosen
```

8. K_p , K_i , K_d are the parameters of the PID controller in the upright and down position of the pendulum. Before setting these parameters the PID controller must be selected.

The PID control is described by the following formula:

$$u = K_p \varepsilon(k) + K_i \sum_{k=0}^n \varepsilon(k) + K_d [\varepsilon(k) - \varepsilon(k-1)]$$

where:

ε = desired position of the cart - measured position of the cart
 K_p is gain coefficient
 K_i is integration constant
 K_d is derivation constant

To introduce new values of parameters call the sequence of functions:

```
p=pd_call('SetAlgNo',0);    % all algorithms are switched off
p=pd_call('getp');
p(2)= new value of the  $K_p$ ;
p(3)= new value of the  $K_i$ ;
p(4)= new value of the  $K_d$ ;
a=pd_call('setp',p);
p=pd_call('SetAlgNo',5);    % PID control algorithm is selected
```



9. *Sample Time* defines the sample period of the system. The minimum value of *Sample Time* is determined by the clock of the computer used for control. It can not be less than 0.001 s.

The parameter *Sample Period* may be changed by the following command:

```
pd_call('SetSampleTime',0.01);
```

To read the current value of this parameter enter the following :

```
pd_call('GetSampleTime');
```



8.3 Examples of tuning

This section explains how to use GUI tools for tuning of parameters described in the previous section.

1. Tuning of *Friction*.

Double click on the *Swing-up and Upright Stabilisation* block in the main *pendulum* menu and the corresponding demo window will open. Double click once again in the *pendulum* menu on *Parameters of Controller* block. Return to main experiment –*Swing-up and Upright Satbilisation* and select Start from the Simulation menu on the top bar. The real time experiment will start.

The following window with controller parameters is activate Fig. 8.17.

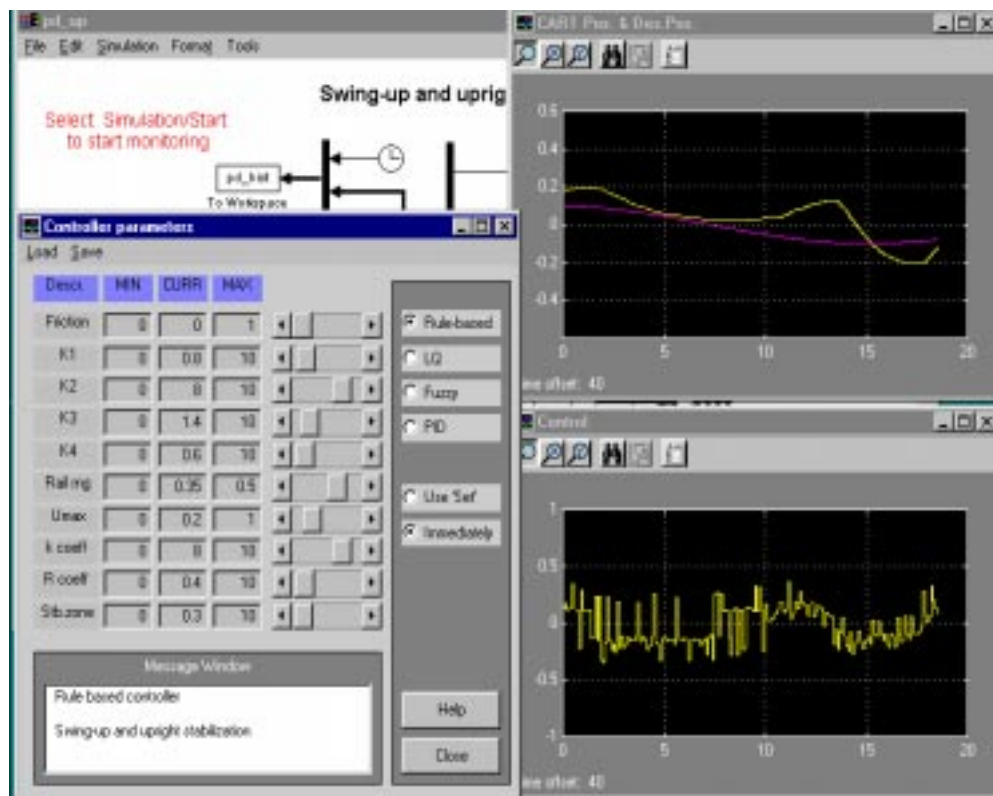


Fig. 8.17. Tuning of *Friction*

The pole is stabilised in its upright position. The cart is following the sinusoidal desired position signal. All parameters are set to their default values. We can read them as *cur* (current value) in *Pendulum Parameters of Controller* window. For example we can tune the first parameter *Friction*. A new value of this parameter may be assigned by dragging the slider or typing the value in the box called "CUR" and send it to the Real Time Kernel by selecting SET button. The result of this action is shown in Fig. 8.17.



If the friction value is too high then the system starts to tremble as the amplitude of control increases.

2. Tuning of *Max.* (maximal swinging force).

Start to swing the pendulum with a low value of the parameter *Max.* The stabilisation zone is achieved. Next, press the limit switch, the motor is off, and the pendulum falls down. Drag the *Max.* slider to a new value, for example 0.1, then release the limit switch. The pendulum starts to swing again. This experiment is demonstrated in Fig. 8.18 and 8.19.

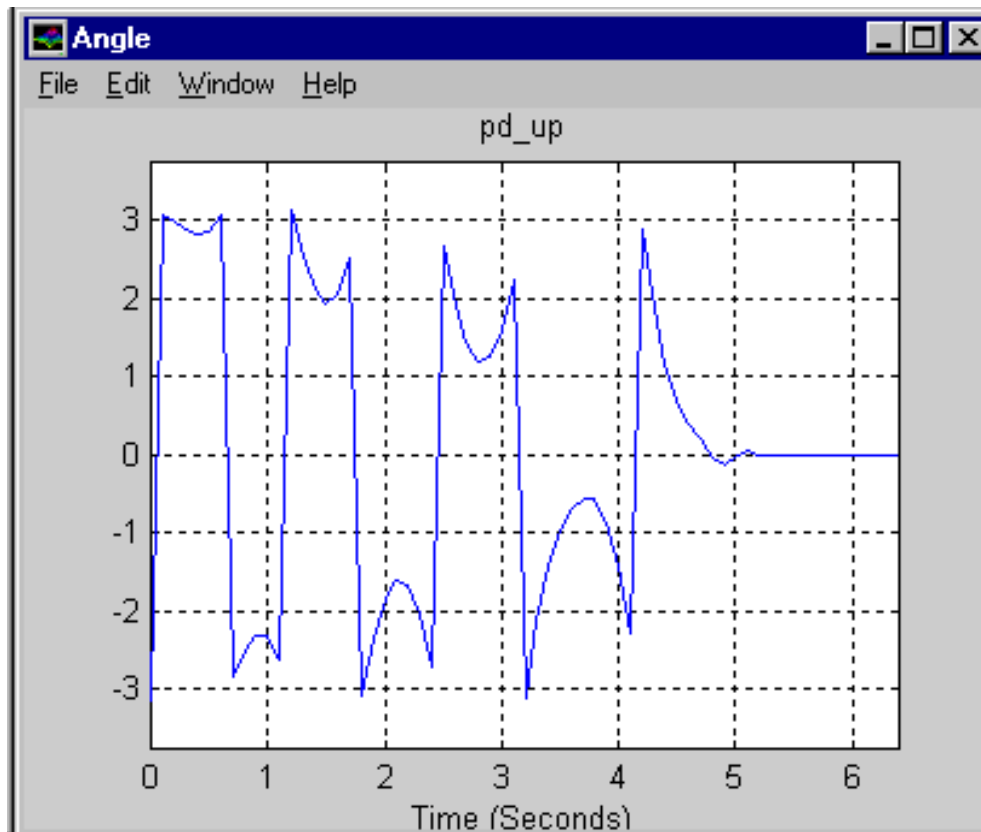


Fig. 8.18. Maximal swinging force tuning. Default value of U_{max}

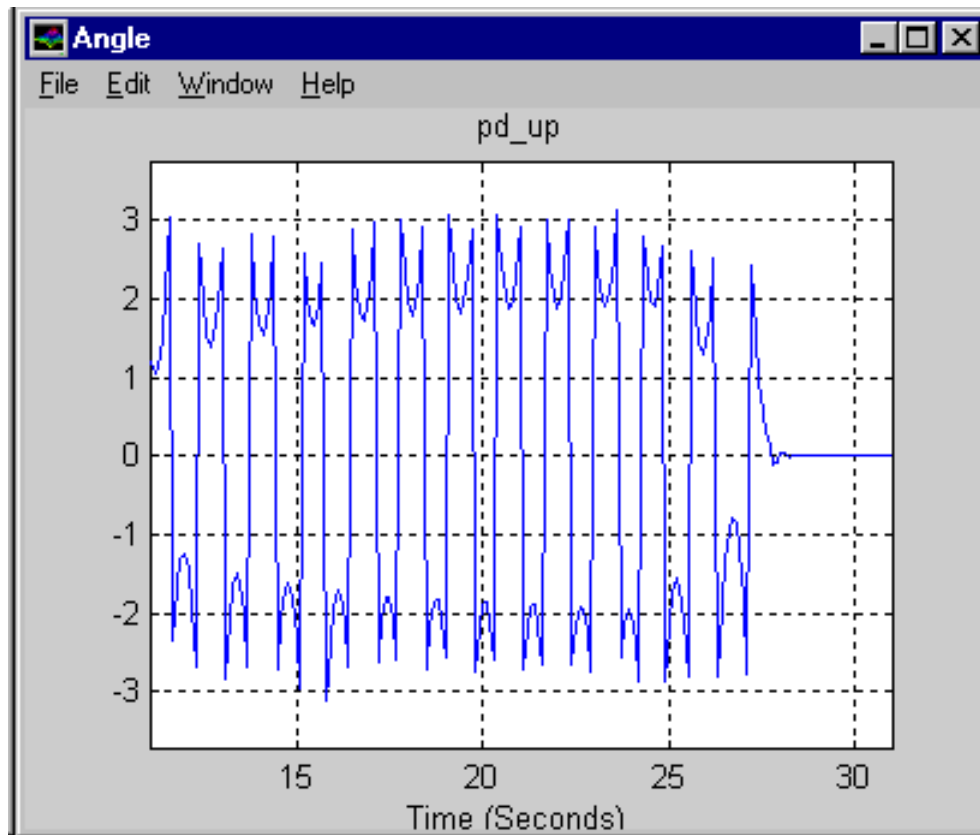


Fig. 8.19. Maximal swinging force tuning. A higher value of U_{max}



3. Tuning of *Stab. zone*.

Start from a small stabilisation zone value. Repeat the actions as in the previous example enlarging the stabilisation zone value. The results are presented in Fig. 8.20. It can be seen that the control objective is reached faster with a larger stabilisation zone.

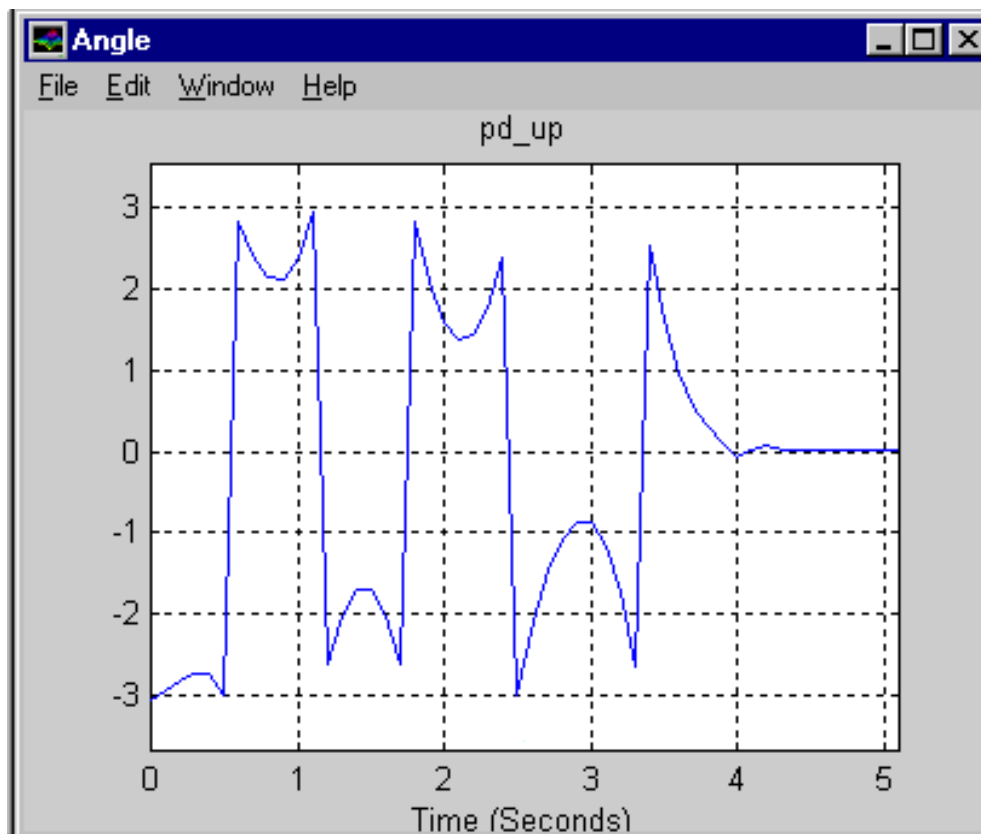


Fig. 8.20. Stabilisation zone tuning.



9. REFERENCES

- [1].Aström K.J. and Wittenmark B., *Computer Controlled Systems*, Prentice-Hall, 1989.
- [2].Franklin G.E., Powell J.D and Workman M.L., *Digital Control of Dynamic Systems*, Addison-Wesley (second edition 1990) .
- [3]. Halang W.A., *Real-time systems*, World Scientific, London, 1992.



**DIGITAL PENDULUM SYSTEM
GETTING STARTED**

**CHAPTER 9
REFERENCES**

Notes