Computation of Swing-up Signal for Inverted Pendulum Using Dynamic Optimization

Stepan Ozana, Martin Pies, and Radovan Hajovsky

VSB-Technical University of Ostrava,
Faculty of Electrical Engineering and Computer Science,
Department of Cybernetics and Biomedical Engineering,
17. listopadu 15/2172, 70833 Ostrava, Czech Republic
{stepan.ozana,martin.pies,radovan.hajovsky}@vsb.cz
http://www.fei.vsb.cz/en

Abstract. The paper deals with computation and implementation of swing-up pulse for inverted pendulum educational model. It presents derivation of the full nonlinear mathematical model, formulation of dynamic optimization task and its solution by use of DYNOPT toolbox that finds control signal which causes rising of the pendulum into upright position. The control scheme containing this control signal is implemented in PAC (programmable automation controller) WinPAC-8000 with the use of REX Control System. The computed swing-up pulse is used for simulation in Simulink environment. The real measurement of state variables is also presented.

Keywords: Control design, Educational products, MATLAB, Regulators.

1 Introduction

The paper deals with design of swing-up control signal for physical model of inverted pendulum. The regulation of the pendulum itself in the upright position is quite complex model to control, therefore it is highly recommended to use algorithms based on so called modern control theory. So far, linear quadratic control has been successfully tested under real conditions. This work is inspired by other articles, for example the one referred in [1]. The previous work of this papers authors has been extended by adding swing-up problem so that the regulation of the model starts with special input signal that lifts up the pendulum. As soon as the angular position is within predefined interval, the regulation is switched to the automatic mode and the model becomes controlled by the state LQR controller designed according algorithms [3]. This is the most common way how to tackle the swing-up problem, yet there are more ways and algorithms how to compute the appropriate control signal responsible for lifting up [2]. For the work described in this paper a numerical approach based on DYNOPT tool which is a special MATLAB third-party product designed to compute basic problems of dynamic optimization.

2   Mathematical Model

To be able to design and implement controller for inverted pendulum model, it is necessary to derive accurate mathematical model. This paper describes method of Lagrange equations of the second kind. Lagrange equations make up general form of Newton equations, because they make it possible to form movement equation even in the fields where Newton equations have no sense. The used method allows creating movement equations for a set of mass points (bodies) by introducing so called generalized coordinates. Other frequent methods use Newton movement rules themselves.

2.1   Lagrange Equations

Derivation of Lagrange equations of the second kind is based on principle of virtual work, by which the plant is in a balance if virtual work $\delta\omega$ caused by all forces in the system is zero.

$$\delta\omega = \sum_{i=1}^{n} Q_i \delta q_i = 0$$

Q_i generalized force acting in the direction
    of i-th coordinate
q_i i-th generalized coordinate

Forces acting in the systems can be divided into conservative and nonconservative. Conservative forces keep energy balance of the system, total sum of kinetic and potential energy is not affected by conservative forces. On the other hand, due to nonconservative forces the energy balance of the system change. These forces are for example dampening forces that depends on velocity.

The basic form of Lagrange equations is (1):

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} = Q_i^* \text{ for } i = 1, 2, \ldots, n \tag{1}$$

Lagrange function L, also referred to as kinetic potential, is defined as (2):

$$L = K - P \,[\text{J}] \tag{2}$$

K kinetic energy of the whole system
P potential energy of the whole system

Movement equations for introduced generalized coordinates thus can be set up by use of scalar quantities only (kinetic and potential energy).

General procedure for application of Lagrange equations of the second kind is as follows:

1. Definition of independent generalized coordinates $q_1, q_2, \ldots, q_n$.
2. Determination of kinetic energy K as a function of derivatives (velocities) of generalized coordinates $\dot{q}_1, \dot{q}_2, \ldots, \dot{q}_n$, and generalized coordinates $q_1, q_2, \ldots, q_n$.
3. Determination of potential energy P as a function of generalized coordinates $q_1, q_2, \ldots, q_n$. This function characterizes influence of all conservative forces.
4. Determination of Lagrange function.
5. Determination of generalized nonconservative forces $Q_1^*(t), Q_2^*(t), \ldots, Q_n^*(t)$.
6. Performing derivation of movement equations.

2.2   Physical Analysis of the Model

The scheme of physical model of inverted pendulum and its basic variables and parameters is given in Fig. 1. Its physical realization can be seen from Fig. 2.
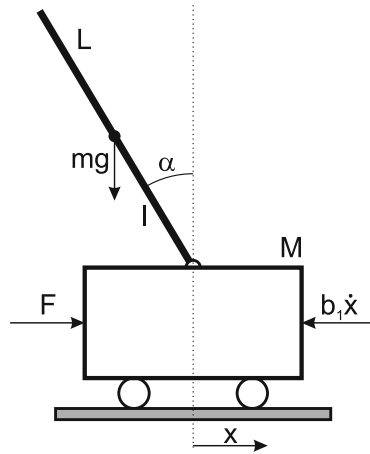

Fig. 1. Analysis of inverted pendulum model

m mass of the pendulum
g gravity
M mass of the cart
L length of the pendulum
l length between mass center and joint, $l = L/2$
F force (manipulated value)
x cart position
α pendulum angle
b_1 friction of the cart
b_2 friction of the pendulum
J inertia of the pendulum

Having introduced parameters of the system it is possible to move on towards determination of movement equations. Overall kinetic energy of the system is given by (3):
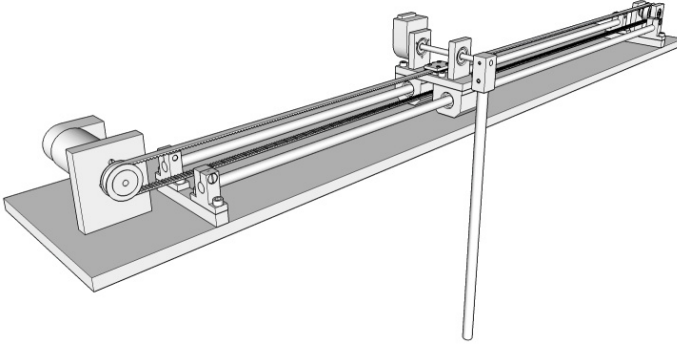
Fig. 2. Physical model of inverted pendulum

$$K = \frac{1}{2}M\dot{x}^2 + \frac{1}{2}m\dot{x}^2 + \frac{1}{2}J\dot{\alpha}^2 + ml\dot{x}\dot{\alpha}\cos\alpha[\text{J}] \tag{3}$$

Where J represents inertia of the pendulum given by (4)

$$J = \frac{1}{3}mL^2 \tag{4}$$

or, by use of Steiner formula (5),

$$J = \frac{1}{12}mL^2 + ml^2 \tag{5}$$

The equations for the first generalized coordinate (x) are given by (6)–(10):

$$\frac{\partial K}{\partial \dot{x}} = M\dot{x} + m\dot{x} + ml\dot{\alpha}\cos\alpha \tag{6}$$

$$\frac{\partial K}{\partial x} = 0 \tag{7}$$

$$Q_x = F - b_1\dot{x} \tag{8}$$

$$\frac{d}{dt}\left(\frac{\partial K}{\partial \dot{x}}\right) = (M+m)\ddot{x} + ml\ddot{\alpha}\cos\alpha - ml\dot{\alpha}^2\sin\alpha \tag{9}$$

$$\frac{d}{dt}\left(\frac{\partial K}{\partial \dot{x}}\right) - \frac{\partial K}{\partial x} = Q_x \tag{10}$$

The first output movement equation is then (11):

$$(M+m)\ddot{x} + ml\ddot{\alpha}\cos\alpha - ml\dot{\alpha}^2\sin\alpha = F - b_1\dot{x} \tag{11}$$

The equations for the second generalized coordinate (α) are given by (12)–(16):

$$\frac{\partial K}{\partial \dot{\alpha}} = J\dot{\alpha} + ml\dot{x}\cos\alpha \tag{12}$$

$$\frac{\partial K}{\partial \alpha} = -ml\dot{x}\dot{\alpha}\sin\alpha \tag{13}$$

$$Q_\alpha = mgl\sin\alpha - b_2\dot{\alpha} \tag{14}$$

$$\frac{d}{dt}\left(\frac{\partial K}{\partial \dot{\alpha}}\right) = J\ddot{\alpha} + ml\ddot{x}\cos\alpha - ml\dot{x}\dot{\alpha}\sin\alpha \tag{15}$$

$$\frac{d}{dt}\left(\frac{\partial K}{\partial \dot{\alpha}}\right) - \frac{\partial K}{\partial \alpha} = Q_\alpha \tag{16}$$

The second output movement equation is then (17):

$$J\ddot{\alpha} + ml\ddot{x}\cos\alpha = mgl\sin\alpha - b_2\dot{\alpha} \tag{17}$$

Equations (11) and (17) make up the full nonlinear model of the system.

2.3   State Nonlinear Model

Firstly, we obtain the formulas (18) and (19) that represent evaluation of second derivatives $\ddot{x}$ and $\ddot{\alpha}$. This is easily carried out by use of equations (11) and (17):

$$\ddot{x} = \frac{F - b_1\dot{x} + ml\dot{\alpha}^2\sin\alpha - ml\ddot{\alpha}\cos\alpha}{M + m} \tag{18}$$

$$\ddot{\alpha} = \frac{mgl\sin\alpha - b_2\dot{\alpha} - ml\ddot{x}\cos\alpha}{J} \tag{19}$$

Introducing the state variables, input variables and output variables, we get

$$\begin{aligned}
x_1 &= x \\
x_2 &= \dot{x}_1 = \dot{x} \\
x_3 &= \alpha \\
x_4 &= \dot{x}_3 = \dot{\alpha} \\
u &= F \\
y_1 &= x_1 \\
y_2 &= x_3
\end{aligned} \tag{20}$$

With the following physical meanings

x_1 position of the cart   [m]
x_2 velocity of the cart $\left[\mathrm{m \cdot s^{-1}}\right]$
x_3 angular position of the pendulum   [rad]
x_4 angular velocity of the pendulum $\left[\mathrm{rad \cdot s^{-1}}\right]$
u force (manipulated value)   [N]
y_1 position of the cart   [m]
y_2 velocity of the cart $\left[\mathrm{m \cdot s^{-1}}\right]$

by substitution into (18) and (19) we get (21) and (22)

$$\dot{x}_2 = \frac{u - b_1 x_2 + mlx_4^2 \sin x_3 - ml\dot{x}_4 \cos x_3}{M + m} \tag{21}$$

$$\dot{x}_4 = \frac{mgl \sin x_3 - b_2 x_4 - ml\dot{x}_2 \cos x_3}{J} \tag{22}$$

Having substituted $\dot{x}_4$ described in (21) by (22) we get (23)

$$\dot{x}_2 = \frac{u - b_1 x_2 + mlx_4^2 \sin x_3}{M + m} \\ - \frac{ml \cos x_3 \left(mgl \sin x_3 - b_2 x_4 - ml\dot{x}_2 \cos x_3\right)}{J \left(M + m\right)} \tag{23}$$

Similarly, having substituted $\dot{x}_2$ described in (22) by (21) we get (24)

$$\dot{x}_4 = \frac{mgl \sin x_3 - b_2 x_4}{J} - \frac{ml \cos x_3 \left(u - b_1 x_2\right)}{J \left(M + m\right)} \\ - \frac{ml \cos x_3 \left(mlx_4^2 \sin x_3 - ml\dot{x}_4 \cos x_3\right)}{J \left(M + m\right)} \tag{24}$$

Making $\dot{x}_2$ and $\dot{x}_4$ single from (23), (24), and adding $\dot{x}_1$, $\dot{x}_3$ (that stayed the same during adjustments), we get the full final state description of the model, described by (25)–(28).

$$\dot{x}_1 = x_2 \tag{25}$$

$$\dot{x}_2 = \frac{gl^2 m^2 \cos x_3 \sin x_3 - Jlmx_4^2 \sin x_3 + Jb_1 x_2 - Ju - b_2 lmx_4 \cos x_3}{l^2 m^2 \cos^2 x_3 - J \left(M + m\right)} \tag{26}$$

$$\dot{x}_3 = x_4 \tag{27}$$

$$\dot{x}_4 = \frac{\left(M + m\right)\left(b_2 x_4 - glm \sin x_3\right) + lm \cos x_3 \left(lmx_4^2 \sin x_3 + u - b_1 x_2\right)}{l^2 m^2 \cos^2 x_3 - J \left(M + m\right)} \tag{28}$$

3 Dynamic Optimization

3.1 Introduction

The mathematical theory of dynamic programming used for solution of dynamic optimization problems dates to the early contributions of Bellman [4] and Bertsekas [5]. Dynamic programming was systematized by Richard E. Bellman. He began the systematic study of dynamic programming in 1955. The word "programming," both here and in linear programming, refers to the use of a tabular solution method and not to writing computer code.

As the analytical solutions are generally very difficult, chosen software tools are used widely. These software packages are often third-party products bound for standard simulation software tools on the market. As typical examples of such tools, TOMLAB and DYNOPT could be effectively applied for solution of problems of dynamic programming. We can classify the dynamic programming tasks concerning the type of final time (free/fixed) and final point (free/fixed), thus we can distinguish 4 combinations: problem with free time and free end point, problem with free time and fixed end point, problem with fixed time and free end point, problem with fixed time and fixed end point.

3.2 DYNOPT

DYNOPT is a set of MATLAB functions for determination of optimal control trajectory by given description of the process, the cost to be minimized, subject to equality and inequality constraints, using orthogonal collocation on finite elements method.

The actual optimal control problem is solved by complete parameterization both the control and the state profile vector. That is, the original continuous control and state profiles are approximated by a sequence of linear combinations of some basis functions. It is assumed that the basis functions are known and optimized are the coefficients of their linear combinations. In addition, each segment of the control sequence is defined on a time interval whose length itself may also be subject to optimization. Finally, a set of time independent parameters may influence the process model and can also be optimized.

4 Solution of the Task in DYNOPT

4.1 Adjusting of the Model for DYNOPT

It is the problem with free time and fixed end point, because we dont know the time when the pendulum reaches the vertical position. The objective function is defined as (29):

$$J = \int_0^t dt \tag{29}$$

and it has to be minimized, finding the unknown final time t_f. For that reason, the current model of the system will be added by one more state variable $x_5 = t$. Overall system is then described by basic equations (25)–(28), plus (30).

$$\dot{x}_5 = 1 \tag{30}$$

Then the objective function becomes as described by (31).

$$J = x_5\left(t_f\right) = \int_0^{t_f} \dot{x}_5 dt \tag{31}$$

This is required by Dynopt toolbox as the assignment has to be set up in so called Mayer form.

4.2   Solution on DYNOPT

The solution of the problem in DYNOPT lies in setup of needed scripts `confun.m`, `graph.m`, `objfun.m`, `process.m` according DYNOPT guide [6]. The core of the computation is defining the system itself as $F = f\left(x, u, p, t\right)$ represented by equations (25)–(28) plus (30), then the derivatives $\frac{\partial F}{\partial x}$ and $\frac{\partial F}{\partial u}$. The numerical computation consists of iterations which leads to the final control signal as shown in Fig. 3.
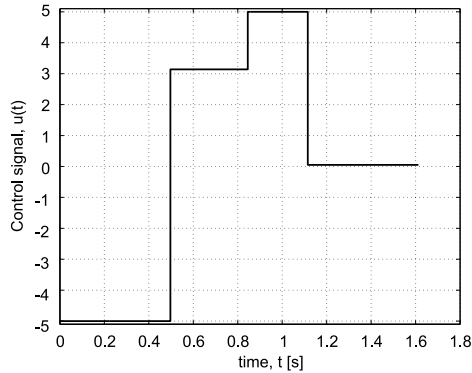

Fig. 3. Result of DYNOPT numerical computation finding the swing-up control signal

Following pictures Fig. 4 to Fig. 7 show the variables x_1, x_2, x_3, x_4 computed by DYNOPT compared to simulation in Simulink.
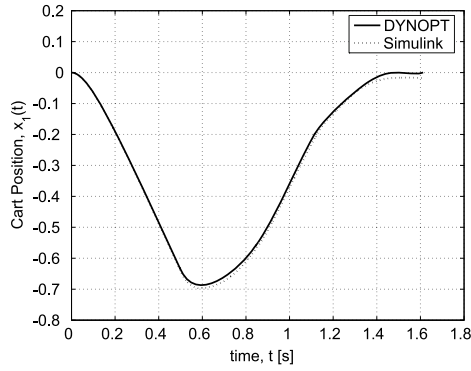
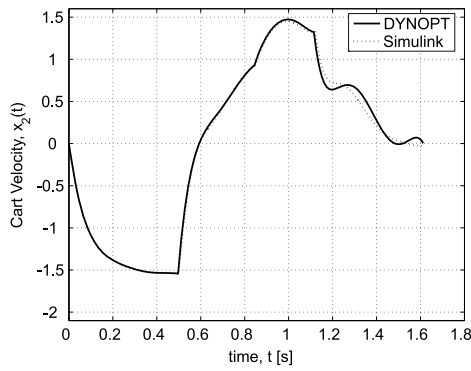**Fig. 4.** Comparison of Cart Position in Simulink and DYNOPT

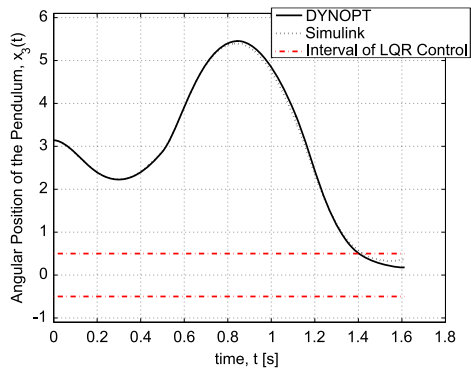**Fig. 5.** Comparison of Cart Velocity in Simulink and DYNOPT

Fig. 6. Comparison of Angular Position in Simulink and DYNOPT
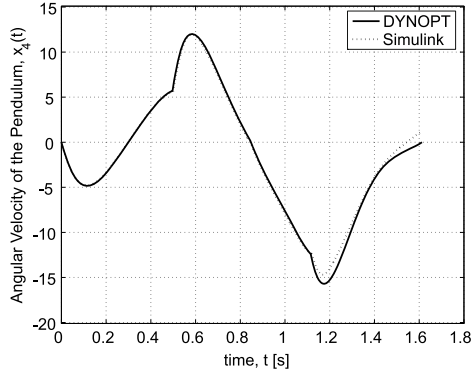
Fig. 7. Comparison of Angular Velocity in Simulink and DYNOPT

5   Implementation

5.1   Block Scheme of Solution

The scheme of control circuit given in Fig. 8 is a combination of control scheme and electronic components, altogether representing the idea how to control the inverted pendulum model. It uses analogue and digital input and output modules (AI, AO, DO) of the programmable automation controller. It also shows electronic elements SG3524N and LM18200T that represents hardware current (torque) controller, according functional diagram, see reference [7], page 2. Connection diagram with the bridge LM18200T can be seen from reference [8], page 11. The middle part of the scheme represents system observer designed based on LQG technique, using Kalmann filter. This is used to generate approximations of two state variables which are not measured (velocities of the cart and pendulum $\dot{x}$, $\dot{\alpha}$). These approximated state variables together with two other measured variables (cart and pendulum position x, α) are then used as input to state (LQR) controller represented by matrix K.

The switch referred to as "T" represents switching to the automatic mode which is triggered once the angular position of the pendulum is close to the vertical position 0 rad, this is predefined as an interval of angles between -0.5 to $+0.5$ rad, as indicated by red dashed lines in Fig. 6.

5.2   REX Control System + WinPAC

REX control system is an advanced tool for design and implementation of complex control systems for automatic control. Basically it consists of two parts: the development tools and the runtime system.The control algorithms are composed from individual function blocks, which are available in the extensive function block library called RexLib. This library covers all common areas of automation and robotics. Moreover, several unique advanced function blocks are contained [11].

The algorithms are composed of individual function blocks, which are available in extensive function block libraries. These libraries cover not only all common fields of automation and regulation but offer also a variety of elements for high-level control algorithms. Runtime version of the REX control system is available for industrial PLC/PAC WinPAC and ViewPAC or their predecessor WinCon of the ICPDAS company.
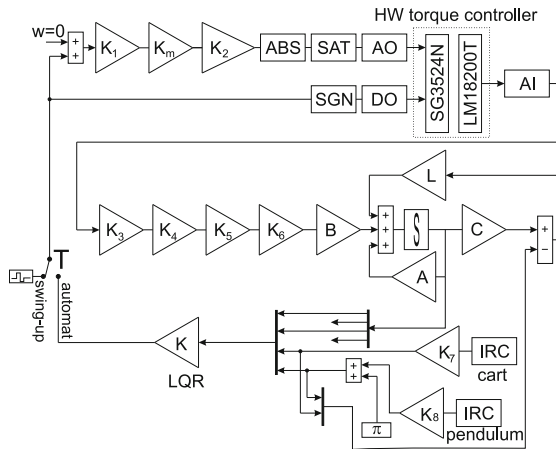

Fig. 8. Block control scheme of the software solution

The block scheme in Fig. 8 has been implemented on programmable automation controller PAC WinPAC-8000, see [9]. The creation of control algorithm for this PAC is performed at two steps.

Firstly, it is creation of executive task that defines target platform and the main tick (time period) of the process. The executive can handle up to 5 tasks, particularly one fast QuickTask, and 4 slower tasks referred to as Level0 – Level3. Each of slower tasks has predefined its own tick based on the main tick and factor (priority). Currently the inverted pendulum model is connected to a QTask with 4 ms sampling period. Processor scheduling is controlled by REX core executive running on a target platform (WinCE), [10].

The second step of implementation of control algorithm is creation of control scheme containing blocks for reading/writing from/to IO modules of the automation controller. The control scheme is similar and compatible with Simulink environment. This scheme together with executive scheme is stored as *.mdl* file so as it can be open and even edited in Simulink provided REXLib library is installed on the computer, [11]. The advantage of compatibility between REX and Simulink is the possibility of tunning and verification of the algorithms without loading the program to the real hardware.

Besides described hardware setup (WinCE, WinPAC-8000, REX) the proposed approach using REX Control System allows implementation on many

other modern and common platforms, such as Embedded PC/Single-Board PC + Linux/Xenomai + B&R I/O modules or usage of Raspberry PI or Arduino boards.

Swing-up pulse was implemented in REX Control System according Fig. 3 with use of DELM and CNR blocks, see Fig. 9
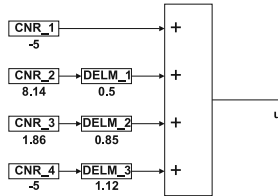

Fig. 9. Block control scheme of the software solution

After implementation of swing-up pulse into real control circuit, the functionality was tested under real conditions. Measured cart position and angular position are shown in Fig. 10.
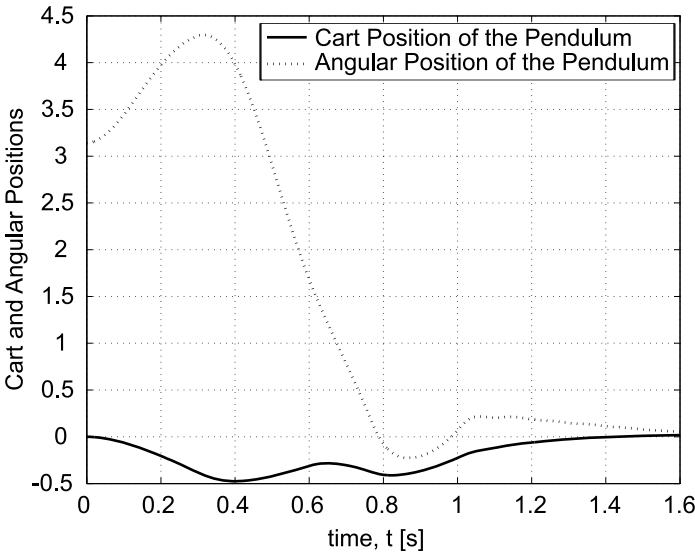

Fig. 10. Cart and Angular Positions Measured on Real Inverted Pendulum

6   Conclusions

The paper demonstrates use of DYNOPT Toolbox to design and implement swing-up control signal for physical model of inverted pendulum. The swing-up impulse for inverted pendulum designed and calculated by DYNOPT toolbox

was implemented by a sequence of four pulses according Fig. 3 and Fig. 9, with overall length (final time) $T_f = 1.685$ s.

As it can be seen from Fig. 4 to Fig. 7, all of the state variables in T_f must be close to zero value because all positions and velocities are zero in upright position. After the system is agitated by swing-up pulse, the rod is getting close to upright position and automatic LQR control is applied immediately. Based on long-term experience, the swing-up problem is sensitive to constructional features of the system. Not all the swing-up attempts are successful when the cart moves through the railways, as even slight friction changes may cause a failure. Further work will require adding a signal of cart absolute position that can represent a flag indicating successful or unsuccessful swing-up. Based on this absolute position, the swing-up impulse may be adjusted in case of need. For example, it can handle some situations and conditions under which the swing-up is impossible at all. Typically, the cart needs approximately 2/3 of the length to be able to erect so if the cart is in the middle at the beginning than the swing-up is impossible due to insufficient space for swaying the rod.
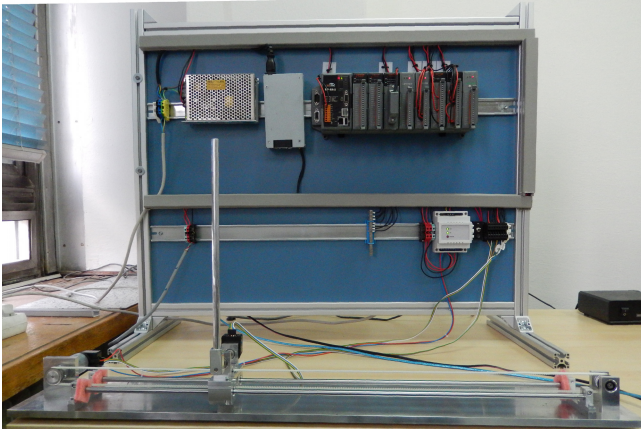

Fig. 11. Laboratory model of inverted pendulum at the Department of cybernetics and biomedical engineering

The model is currently used for educational purposes at the Department of cybernetics and biomedical engineering for analysis and synthesis of the systems, representing a nonlinear very complex control system but attractive at the same time. Both swing-up and consequent LQR control have been successfully implemented and tested. The results implemented on a real system are documented in the form of YouTube video accessible through reference [12].

References

1. Przemyslaw, H.: Stabilization of the cart-pendulum system using normalized quasi-velocities. In: Proceedings of the 17th Mediterranean Conference on Control & Automation, pp. 827–830 (2009)
2. Kocian, J., Koziorek, J., Ozana, S.: An approach to identification procedures for PID control with PLC implementation. In: 2012 IEEE 17th International Conference on Emerging Technologies and Factory Automation, ETFA 2012 (2012)
3. Tewari, A.: Modern Control Design With MATLAB and SIMULINK. Wiley, Chichester (2002)
4. Bellman, R.: Dynamic Programming. Princeton University Press, Princeton (1998)
5. Bertsekas, D.: Dynamic Programming and Stochastic Control. Academic Press, New York (1976)
6. Cizniar, M., Fikar, M., Latifi, M.A.: Matlab DYNamic OPTimisation code – DYNOPT, Institute of Information Engineering, Automation, and Mathematics, Department of Information Engineering and Process Control, Bratislava, Slovak Republic, Tech. Rep (2013), `http://www.kirp.chtf.stuba.sk/moodle/mod/resource/view.php?id=5464` (accessed on May 25, 2014)
7. Texas instruments, Regulating pulse width modulators SG3524 datasheet (2009), `http://www.ti.com/lit/ds/symlink/sg2524.pdf` (accessed on May 25, 2014)
8. National Semiconductor, 3A 55V H-bridge LMD18200 datasheet (2012), `http://www.ti.com/lit/ds/symlink/lmd18200.pdf` (accessed on May 20, 2014)
9. ICP DAS Co. Ltd., Winpac-8441/8841 WinCE Based Programmable Automation Controller (2013), `http://www.icpdas-usa.com/documentation/Quickstarts/wp-8441_wp-8841.pdf` (accessed on May 20, 2014)
10. Balda, P., Schlegel, M., Stetina, M.: Advanced Control Algorithms + Simulink Compatibility + Real-time OS = REX. In: 16th Triennial World Congress of International Federation of Automatic Control, Prague, vol. 16, pp. 121–126 (2005)
11. REX Controls, ZCU Plzen, Czech Republic (2009), `http://www.rexcontrols.com` (accessed on May 22, 2014)
12. Ozana, S.: Inverted pendulum with swing-up (2013), `http://www.youtube.com/watch?v=rOt1FiJNVjA` (accessed on May 20, 2014)