CONTROL PRACTICES USING SIMULINK, ARDUINO AND LOW-COST HARDWARE

FELIPE M. LOBO[1], CELSO J. MUNARO[1], LUCAS C. DE REZENDE[1].

1. *Post-Graduate Program in Electrical Engineering, Federal University of Espírito Santo - UFES, Av. Fernando Ferrari, 514 - 29075-910, Vitória, Espírito Santo, Brasil*
E-mails: felipemachadolobo@hotmail.com, cjmunaro@gmail.com, rezendelc94@gmail.com

Abstract— In this paper, Arduino, Matlab/Simulink and low-cost hardware are combined to propose control practices that explore very important concepts taught in control theory class. The activities are programed graphically on Simulink and the corresponding code is transferred and executed in the microcontroller attached to real plants. This system allows capturing data for modeling, using Matlab workspace for analysis, design and testing digital controllers under real time supervision on Matlab environment. Activities are proposed for modeling, analysis and control for both continuous and discrete time domain, and the concepts covered are highlighted. Some results obtained in the activities are presented, requiring reflections about how to apply in real world systems the concepts taught in class.

Keywords— Control education, embedded systems, Simulink, Arduino.

1 Introduction

On a 2008 survey carried out by the IEEE Control Systems Society, highlighted that universities 'overrate the quality of their graduate students in terms of satisfying industry needs'. A significant majority of industry respondents consider mathematical modeling of physical systems to be a valuable skill. Also, 72% think "hands-on experience" is the area that most needs to be strengthened to better prepare control engineers (COOK and SAMAD, 2009).

Laboratory activities are essential to the control engineering formation for motivation and better understanding of the concepts taught in class (RECK and SREENIVAS, 2015), (ALBAYRAK et al, 2015). The use of simple plants with data acquisition boards working as interfaces with environments for analysis, modeling and design is a suitable approach for this purpose (BARBER et al, 2013).

One of the most used microcontrollers in the market is Arduino, which is an open-source electronic platform based on easy-to-use hardware and software that permits innumerous control applications (RECK and SREENIVAS, 2015), (SORIANO et al, 2014), (SOBOTA et al, 2013) and (SANTOS et al, 2014).

An interesting feature of this platform is to be supported by Matlab, allowing its use for data acquisition and real time control, with supervision under Simulink (MATHWORKS, 2016). Any task can be programmed using the blocks from Simulink: the code for the task is generated, compiled and uploaded to the microcontroller. The data from the task can be visualized in real time and becomes available in Matlab workspace.

These activities are supported by a Simulink add-on called Real-Time Workshop (RTW). According to BARBER et al. (2013), RTW can be used for modeling and control a servomotor using the External Mode. However, there are concepts for analysis and control that were not explored, such as root locus and Bode plot. REGUERA (2015) describes many ways to connect Simulink with a microcontroller and a DC motor used as a plant. However, the education methodology describing how the tools can be used for teaching control was not covered.

Another way to explore concepts like modeling, analysis and control is using educational kits, like the ones sold by dSpace, Feedback and Quanser (TEIXEIRA and SALLES, 2009) and (GREPL, 2011). Despite those platforms capabilities, they are expensive and less versatile than microcontrollers.

The main purpose of this study is to fulfill those needs using low-cost hardware integrated with Matlab/Simulink, which is one the most used software for teaching engineering.

2 Framework Description

The use of a microcontroller results in a versatile and low-cost choice for the hardware. Three microcontrollers are supported by RTW: Raspberry Pi, BeagleBone and Arduino. Raspberry Pi does have no built-in analog-to-digital converter and BeagleBone is the most expensive. Arduino has a built in analog-to-digital (A/D) converter and is the cheapest option as well. Arduino Due board was selected for this project, with the features: 12 PWM outputs, 12 analog inputs, USB connection, 2 DAC (digital to analog converter), operates with 3.3V.

The system requirements for the computer to run RTW are: Intel or AMD x86 processor with 2 GB of RAM and 1GB of space for the software. Most universities and educational institutes already have license for the Matlab software. The version used in this study was the R2015a, and at least R2013a version is required.

2.1 RTW installation

The Library of Simulink Support Package for Arduino Hardware can be downloaded and installed by opening the Add-On window on Matlab. The user must create a MathWorks account to download the add-on.

After installation is complete, the Library should appear on the Simulink Library Browser as Simulink Support Package for Arduino Hardware.

2.2 Arduino Library

The blocks on the library installed are input or output blocks. In this study the blocks used are: Analog Input and PWM (output).

The Analog Input block converts the voltage of a specified analog input pin. The conversion uses 10 bits and the measured voltage must be between 0 and 3.3V. The user should provide two parameters: Pin Number and Sample Time.

The PWM block generates a PWM signal on the specified analog output pin. The duty cycle depends on the block input, which ranges from 0 to 255. The frequency of the pulses is approximately 490 Hz.

2.3 Simulink Configuration and External Mode

Simulink must be configured to work with the Arduino Due. In the options for simulation, External Mode is selected to operate in real-time with the external hardware.

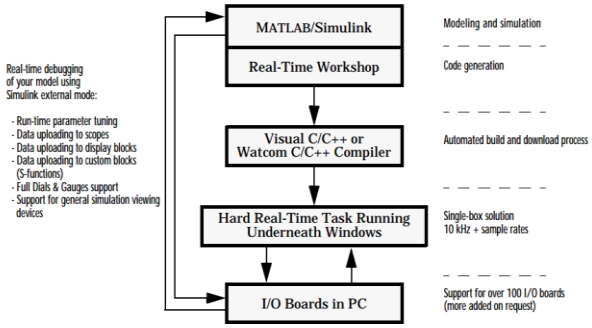

Figure 1. RTW operational cycle

When an execution is requested, RTW converts the block diagram design on Simulink into a C/C++ code, in order to compile and upload the code to the microcontroller. The code is executed in real-time with the sample time selected in the blocks, and the selected signals are sent to Simulink via USB connection. Thus, Simulink becomes a human machine interface to the microcontroller. Figure 1 shows RTW operational cycle.

During execution of the code, changes in parameters of the blocks are transmitted to the microcontroller, allowing changes in the tests. Examples of possible changes are values of a constant block, time delay, gains, parameters of PID blocks. However, changes in block diagram structure or in the sample time require a new compilation and download of the code.

The speed of the communication between the microcontroller and the Simulink is limited by USB port. For the framework used in this study, the minimum sample time for real time monitoring was 40ms. However, the sample time for tasks performed in the microcontroller is less than 1ms.

3 Modeling, analysis and control

The topics taught in control system involve modeling, analysis and design. This study proposes experiments related to such concepts that can be explored in the described framework. The use of a digital controller is twofold: it allows to analyze important aspects related to discretization and is very close to modern control systems, which are rarely analog.

The basic concepts are explored in a very simple first order system consisting of an RC circuit with time constant of 1s and time delay emulated in the microcontroller, very easy to model. Then, the challenge of modeling, analysis and control design of DC motor speed control is proposed.

In the sequence, experiments are proposed for important topic covered in control systems, highlighting the motivation, Simulink block diagram and required components, concepts and activities, and the results of some of the experiment.

3.1 Modeling

Obtaining the transfer function for the system under study is the first step, which can be done by the characterization of the system by its step response. This activity aims to: obtain a continuous and discrete model, analyze the effect of time delay, deciding about the choice of the sample time.

The block diagram shown in Figure 2 is used for this activity. Toggling the manual switch *SW1* allow the application of steps to the PWM block. The transport delay block allows the use of the desired time delay, which is implemented in the microcontroller. The delayed PWM signal is applied to the RC circuit and the voltage in the capacitor is measured by the analog input block. The two gain blocks are used to have inputs and outputs working in the same range of 0 to 3.3V. The scope block is configured to save the variables in the workspace with the same sample time of the microcontroller. Data type conversion blocks are included and set as double, as required by the mux block.

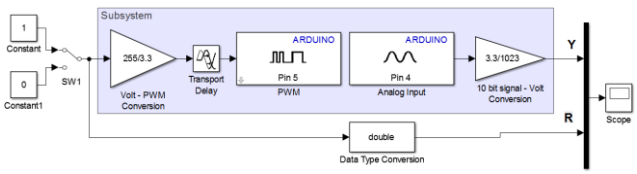

Figure 2. Block diagram for step response analysis

The maximum current that the PWM pin of the Arduino Due can deliver is 40mA. Thus, the resistor should have resistance higher than 100Ω. Since the minimum sample time using the external mode monitoring is 40ms, the capacitor is selected so that the time constant of the circuit is several times greater

the sample time. A capacitor of 100uF and a resistor of 10kohm provide a time constant of 1s.

Concepts and activities:

i) Continuous modeling: toggle the switch SW1, obtain the step responses, calculate the gain, time constant and time delay to obtain the transfer function $G(s)$. Simulate $G(s)$ and compare with the acquired data.

ii) Time delay approximation: use first and second order Padé approximation for the time delay and compare the simulated responses with the activity i).

iii) Selection of the sample time: using the step response as reference, discuss the minimum value for the sample time.

iv) Discrete modeling: analyze different techniques to discretize $G(s)$ obtaining $G(z)$. Compare the model response with the acquired data.

v) Zero order holder: identify in the diagram of Figure 2 where is the zero order holder.

vi) Quantization error: calculate the quantization error of the A/D converter and verify it in the sampled data. Check the minimum allowed variation of the PWM signal, given the PWM block has 8 bits.

3.2 Time Domain Analysis

Time domain analysis involves steady state and transient response when excited by standard signals. Using the RC with time constant of 1s and a transport delay of 0.2s in closed loop, one obtains an under damped system that can be used for transient analysis of the second order prototype. The damping of a second order prototype and the FOPTD (First Order Plus Time Delay) with the delay approximated by first order Padé is shown in Figure 3. The proportional gain is varied in both cases and damping is obtained via simulation. The small delay results in a transfer function with the Padé approximation with a zero far from the origin, reducing its effect on the response. Therefore, this system can be used to study a second order prototype. Using an inductor on the circuit could be another solution to work with a second order system. However, the inductance should be too high for the rise time to be around 1s as required. Figure 4 shows the block diagram used as reference for these activities.

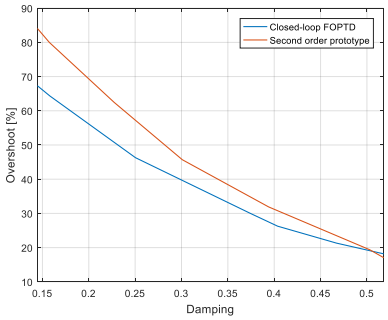

Figure 3. Damping versus overshoot for a second order prototype and a closed-loop FOPTD

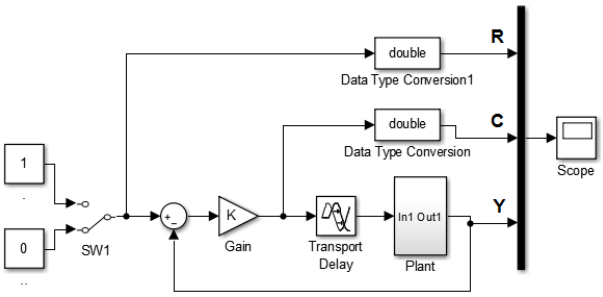

Figure 4. Diagram for time domain analysis

Concepts and activities:

i) Proportional gain and transient response: apply successive steps using switch SW1 and changing the value of the proportional gain K for each new step. When the value of K is changed, it is also changed in the program that is running on the Arduino. Stop the program after the tests and access the variables R, C and Y on the workspace. Analyze the relation of the gain with the overshoot and settling time.

ii) Pole location and transient response: estimate the value of damping and the damped natural frequency using the step response data of activity i) and plot the corresponding poles on the s plane.

Changing the proportional gain causes variations on the transient characteristics, like rise time, overshoot, settling time and peak time. Also, one can obtain the closed loop transfer function on different values of gain and associate with the location of poles on the complex plane.

iii) Proportional gain and steady state response: plot the gain versus the steady state error of the data obtained in the activity i). Using the open loop transfer function, calculate the steady state error and compare with the measured one. Be aware that system under analysis is type 0.

By increasing the proportional gain, it is possible to observe that reducing the steady state error reducing implies in an increase in overshoot.

3.3 Control Design

PID tuning for FOPTD systems can be tested on this environment. The proposed activities explore some of the methods as well as their performance comparison.

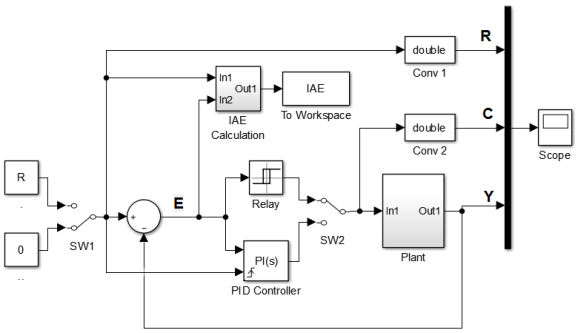

Figure 5. Closed loop system with a PID controller

The block diagram in Figure 5 shows a closed loop system that allows using a PID controller or a relay, depending on the switch SW2. The IAE (Integral Absolute Error) calculation block receives the error signal (E) and calculates the IAE each time the switch SW1 is toggled. A reset on rising edge on the IAE calculation and the PID controller allows many tests without stopping and recompiling the program.

Concepts and activities:

i) Tuning of PID controllers: using the given model, obtain the PID parameters using different methods such as Ziegler-Nichols, Cohen Coon and CHR. The test of different tuning methods can be done one after another without stopping the program, just by changing the parameters on the PID block and applying a new step. The variables will be available in the workspace after the program is stopped. Compare overshoot, rise time, settling time and IAE for each method.

ii) Time delay effect on different tuning methods: compare the different tuning methods changing the time delay. Each method has the best performance for different relation (time delay)/(time constant), that can be checked in this activity.

iii) Lambda tuning method: this method allows defining the time constant of the closed loop system. Test different time constants, from sluggish to agressive, and compare their performance using the IAE value.

iv) Relay-based PID tuning: this method offers an alternative to how to choose the parameters of the controller. The concept of the Nyquist criterion can be explored, since it supports the method.

Using the diagram showed in Figure 5, the value of *ref* and h (parameters in the relay block) must be chosen in order to generate the oscillations with a form similar to a triangular waveform and symmetrical in relation to the *ref* value (see Figure 6). Using the amplitude and the period of oscillation, the critic gain K_u and the controller parameters can be calculated. Using the parameters obtained, the performance can be compared with the previous methods.

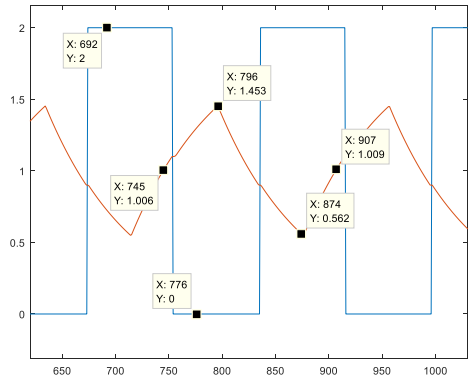

Figure 6. Input and Output of the system using the relay method

The Nyquist diagram of the FOPTD system (Figure 7) shows that for the closed loop system to become marginally stable, the proportional gain has to be $K_u = \frac{1}{0.262}$ and the period of oscillation $T_u = \frac{2\pi}{0.921}$. Thus, the limit cycle of Figure 6 can be analyzed using Nyquist diagram of Figure 7.

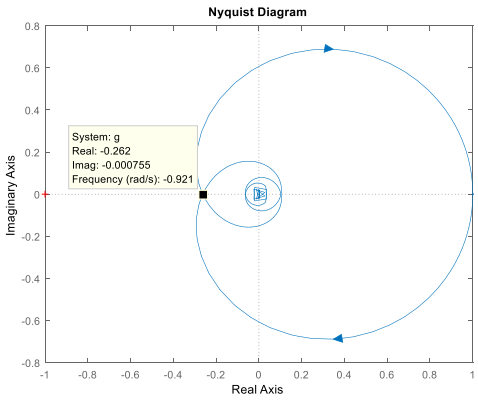

Figure 7. Nyquist diagram for the closed loop system with time delay

3.4 PID realization

Overall, control engineering students spend more time studying continuous data systems than working with digital systems designs. However, most of the PID controllers nowadays are implemented digitally.

Simulink automatically generates the PID code when using it on a block diagram and upload to the microcontroller. The realization of the PID can be made using elements from Simulink environment, comparing different methods.

Concepts and activities:

i) PID controller realization: realize a PI controller using the different methods to describe the respective difference equations by

a block diagram on the Simulink. Run the program with the realizations and compare them. Compare the realization with the one made with the PID block of the Simulink.

Figure 8 shows a PID realization using approximations of the derivative and integral parts.

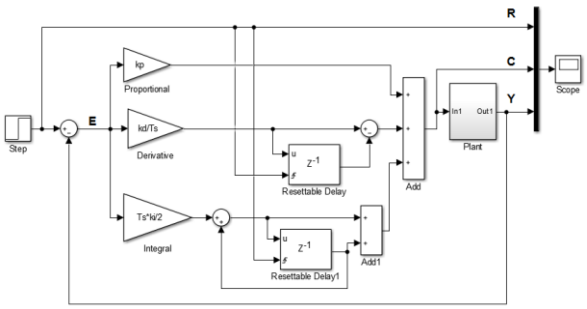

Figure 8. PID realization block diagram

ii) Derivative action: implement a first order filter related to derivative part, analyzing its effect on the response. Change the derivative gain to differentiate the output rather than the error. Compare the effect on the step response. Compare with the effect of the PID block filter.

3.5 Application to a DC motor

After reviewing the basic concepts, the next step is to apply such concepts to control the speed of a DC motor. This plant is very usual in textbooks for modeling and control examples, and allows the introduction of sensor and actuators. The speed measurement is obtained by converting pulses from an optical encoder fixed on the motor shaft and an LM393 speed sensor to detect the pulses that are converted to voltage using a frequency to voltage circuit based on LM331 (TEXAS INSTRUMENTS, 2015). The components are chosen to reduce ripple and to make the voltage proportional to the pulses for all range. The motor driver uses a NPN bipolar junction transistor to apply the 12V to a small DC motor whose current consumption is 100mA for 4000 rpm nominal speed. The driver circuit also has a Normally Closed (NC) Push Button that includes a resistor in series with the motor, representing a disturbance in the process. Figure 9 shows the motor driver circuit. For the following experiments, the input is the PWM pulses and the output is the voltage proportional to motor speed.

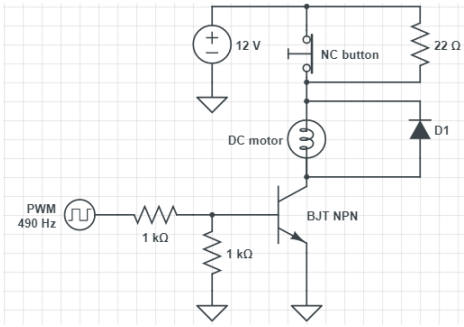

Figure 9. DC motor driver circuit

Concepts and activities:

i) Regions of operation: since real systems always are nonlinear to some extent, apply a unit PWM ramp input from 0 to 255.

The result is shown in Figure 10 which clearly shows that the input-output behavior can be considered linear in two regions, around 1500 and 3000 rpm. Dead zone can be also quantified with this experiment.

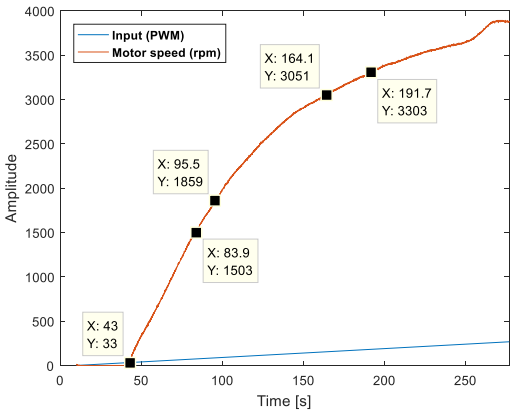

Figure 10. Unit ramp response of the system

ii) Motor modeling: apply steps of different amplitudes and collect the responses. An operating region must be selected, according to Figure 11. First and second order models including time delay if necessary (COELHO, 2004) can be fitted to this data.

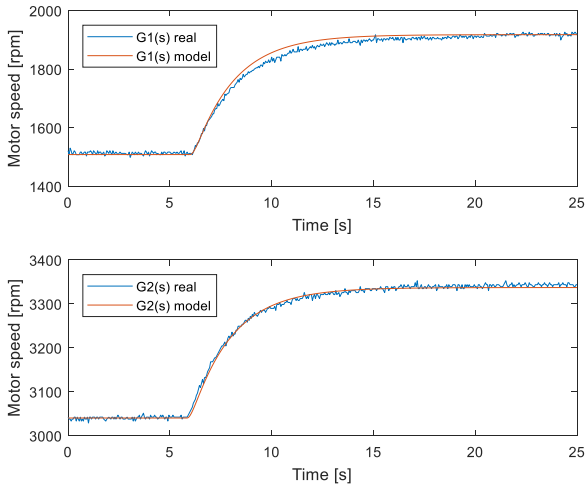

Figure 11. Unit step response for two different operating regions compared to their corresponding models

A comparison of the step response for 1500 rpm region of operation, $G_1(s)$, with the model obtained and a step response for 3000 rpm, $G_2(s)$, and its respective model is shown Figure 11.

iii) Design specifications: using the responses obtained in the activity ii), choose the speed range of operation, rise time, settling time, overshoot and maximum steady state error for step and ramp input.

Based on open loop transient and steady state characteristics, one can define the specifications of the closed loop system to improve these characteristics.

iv) Direct synthesis controller tuning: use the methods described in (CHEN, 2002) to choose the reference model that has the desired performance and verify the possibility to meet them.

If a first order model is selected in activity ii), the simulation of the designed controller will hardly be similar to the test in the DC motor, that tends to show overshoot. This observation highlights the importance of closed loop model validation. In this case, the choice of the second order model from activity ii) should be considered for the controller design. Most of the control systems textbooks do not cover this subject. Equation 1 shows the transfer function for the 1500 rpm operating point.

$$G_1(s) = \frac{31.5}{(2s+1)(0.2s+1)} \qquad (1)$$

With the mathematical model obtained, direct synthesis can be used. The desired τ_C of the closed loop system with the controller must be compatible with the design specifications defined in activity iii) and compared with the real system response. The specifications chosen can be impossible to be met given the system dynamics, quantization and sample time limitations, saturation, so it has to be changed if necessary.

v) Root locus analysis and design: the specifications of activity iii) can be used to define the area on the s plane to contain the closed loop poles. The activity involves changing the gain and zero locus of the PI controller to meet the project specifications. The design can explore better solutions to those using direct synthesis.

vi) Bode plot analysis: the choice of phase margin is not straightforward to meet time domain specifications. The design can be done increasing the phase margin until overshoot is satisfied. On the other hand, it becomes clear that the presence of time delays and higher order models are easily handled in this approach. The activity should result in phase margin and bandwidth that meet the design specifications of activity iii).

vii) Disturbance rejection: an additional specification in control design is disturbance rejection. A simple disturbance can be created switching a resistor in series with the motor. The designed controller can be evaluated in such conditions. An index to measure performance should be discussed. The conditions for mitigation of steady state error can be checked considering the model used for design. The tradeoff between disturbance rejection and tracking can be explored in the design of the controller. The effect of proportional gain on disturbance rejection is shown in Figure 12.

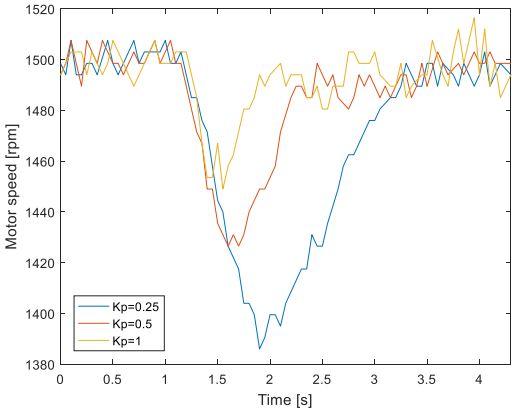

Figure 12. Disturbance rejection for different controllers

4 Conclusions and future studies

A framework consisting of Arduino, Matlab/Simulink and low-cost hardware was proposed to perform laboratory experiments to explore control concepts taught in class. Many commercial control kits require another system for data acquisition. The methodology here proposed allows data

acquisition for modeling, supervision, analysis, design and implementation of controllers. The controllers are implemented digitally, since this is the reality of current control systems.

The main concepts about modeling, analysis and design are covered by the proposed experiments. Changes made graphically in Simulink diagrams are easily transferred and performed on Arduino, with monitoring via Simulink scope. Such framework stimulates the students to create solutions to real control problems.

Based on the assumption that Matlab is available, all required items to perform the proposed activities are low-cost products on the shelf.

5 References

Albayrak, A.; Albayrak, M.; Bayir, R. (2015). Design of Matlab/Simulink based development board for fuzzy logic education. 2015 IEEE International Conference on Fuzzy Systems.

Barber, R.; Horra, M; Crespo, J. (2013). Control Practices Using Simulink with Arduino As Low Cost Hardware. 10th IFAC Symposium Advances in Control Education, volume 46, issue 17. Sheffeld, UK.

Chen, D.; Seborg, D. E. (2002). PI/PID Controller Design Based on Direct Synthesis and Disturbance Rejection. Industrial & engineering chemistry research. Santa Barbara, USA.

Coelho, A. A. R.; Coelho, L. dos S. (2004). Identificação de Sistemas Dinâmicos Lineares. Florianópolis, Brazil.

Cook, J. A.; Samad, T. (2009). Controls Curriculum Survey. A CSS Outreach Task Force Report, IEEE Control Systems Society.

Grepl, R. (2011). Real-Time Control Prototyping in MATLAB/Simulink: Review of tools for research and education in mechatronics. 2011 IEEE International Conference on Mechatronics.

Mathworks (2016). Simulink Real Time user's guide. Available on: http://users.isr.ist.utl.pt/~alex/micd0506/rtw.pdf. Access in 10 dec 2016.

Reck, R. M.; Sreenivas, R. S. (2015). Developing a new affordable DC motor laboratory kit for an existing undergraduate controls course. 2015 American Control Conference (ACC). IEEE. Urbana, USA.

Reguera, P.; García, D.; Domínguez, M.; Prada, M. A.; Alonso, S. (2015). A Low-Cost Open Source Hardware in Control Education. Case Study: Arduino-Feedback MS-150. IFAC – Papers Online. León, Espanha.

Santos, C. M. M.; Costa, B. L. G.; Silva, R. A.; Scalassara, P. R. (2014). Desenvolvimento de um Módulo de Controle de Nível Utilizando o Kit Arduino. XX Congresso Brasileiro de Automática. Belo Horizonte, Brazil.

Teixeira, H. T.; Salles, J. L. F. (2009). Desenvolvimento de uma interface com o usuário no matlab para controle e monitoramento de processos para o laboratório de ensino de controle da UFES. COBENGE 2009.

Texas Instruments. Datasheet: LMx31x Precision Voltage-to-Frequency Converters. Electronic Publication, 2015.

The Mathworks Inc. Simulink Real Time user's guide. Available on: http://users.isr.ist.utl.pt/~alex/micd0506/rtw.pdf. Access in 10 dec 2016.

Sobota, J.; Pi˘sl, R.; Balda, P.; Schlegel, M. (2013). Raspberry Pi and Arduino boards in control education Faculty of Applied Sciences. 10th IFAC Symposium Advances in Control Education, volume 46, issue 17. Sheffeld, UK.

Soriano, A.; Marin, L.; Vallés, M.; Valera, A., Albertos, P. (2014). Low Cost Platform for Automatic Control Education Based on Open Hardware. IFAC Proceedings of the 19th World Congress, volume 47, issue 3. Cape Town, South Africa.