Fourth LACCEI International Latin American and Caribbean Conference for Engineering and Technology (LACCEI'2006)
"Breaking Frontiers and Barriers in Engineering: Education, Research and Practice"
21-23 June 2006, Mayagüez, Puerto Rico

Active Learning of Control Theory Using Virtual Instrumentation

Igor Alvarado Sr., B.S.M.E.
Andean/Caribbean Sales Manager, National Instruments,Austin, Texas, USA, igor.alvarado@ni.com

Abstract

Engineering and science schools have always been challenged in terms of attracting and keeping new students, professors and researchers. In parallel, the industry is requiring new engineers and scientists with a practical, problem-solving focus capable of adapt themselves and respond to the various challenges of a highly competitive, global economy. In many countries, the number of engineering graduates each year is staggering. And even worse, most universities face a common problem: freshman engineering student attrition rates. On the other side, too many students bail out of engineering or science degree programs. As a result, universities are looking for new ways to make engineering education more engaging, more enjoyable, and more fun. The use of flexible, customizable software and hardware tools in an integrated teaching-learning environment is part of the answer to this and other challenges. It's been demonstrated that for control education, experiments have an immeasurable but profound impact, and control experiments provide a valuable link between theory and practice. This paper presents a possible answer to these and other challenges faced by the engineering and science academic sector.

Keywords

Virtual Instrumentation, Active Learning, Experimental Validation, Graphical Programming, Control Theory

1. Introduction

In order to make engineering, and more specifically control engineering more attractive and engaging to students, new tools and methodologies are required. Also, control engineering education must be more responsive and adapted to the new requirements and challenges that continuously changing technologies and a highly competitive global economy represent. Active learning is one of those methodologies and virtual instrumentation (VI) is one of the key enabling technologies. Active learning refers to a methodology that allows students to learn by experimental validation (hands-on learning). On the other side, the concept behind the virtual instrumentation paradigm is its ability is to create more powerful, flexible, and cost-effective instrumentation and control systems using standard, readily available technologies such as the Personal Computer (PC), the Personal Digital Assistant (PDA), micro-controllers (μC) or embedded computers, just to mention a few. As VI is built around standard, widely-accepted computing devices using software as the engine and interface, almost any end-user can built its own, personalized, virtual instrument or controller. These virtual instruments and controllers are key elements in the experimental validation of control theory. Two types of experimental validation have been suggested (Alleyne et al, 2003): control validation experiments and control technology experiments. Both types of experimental validation can take advantage of VI as an enabling technology or tool for its practical implementation.

Validation experiments are essential when applying the active learning model to control design teaching, since they provide an effective link between theory and practice. While it's true that a control experiment may be performed without a specific hardware implementation in mind, it's also useful to make a distinction between what has defined as technology-driven control experiments and system-driven control experiments (Bernstein et al, 2003). Research has shown that control experiments aimed at the constitutive technologies are technology driven. On the other side, system-driven control experiments have being defined as those in which "the objective is to understand the tradeoffs among hardware constraints, plant properties, and achievable performance from a systems point of view" (Bernstein et al, 2003). For a practical implementation of these experiments, flexible, off-the-shelf yet powerful hardware is needed. Usually, the PC represents the best platform for implementing these control experiments, especially at universities and technical schools because PCs are readily available. But while the hardware advances in the personal computer have driven significant performance improvements and cost reductions compared to traditional instruments and controllers, it is the software that empowers hundreds of thousands of engineers and scientists to take advantage of these benefits. Instead of being limited to the use of conventional text-based programming languages such as C or C++, or text-based modeling and simulation languages or packages, other unique software tools are needed. These new tools should provide a stimulating educational environment based on both, theoretical but highly interactive modeling and simulation tools, together with practical hands-on experiments that go all the way to the real-time implementation of the system being designed.

2. Objectives

The objectives of this research are divided into two areas: one related to the effectiveness of the active-learning model (methodology-oriented) and the other related to the tools used for making the learn-by-experimentation experience possible (technology-oriented). Thus, one objective of this research is to demonstrate the effectiveness of the active learning model while the other objective is to demonstrate the applicability of VI in an integrated teaching-learning laboratory for control engineering. Regarding the first objective, the motivation for change in engineering education seems to be driven by the need to replace a strictly theoretical, engineering education that does not engage engineering students or does not meet the needs of the marketplace, by a new hands-on, active-learning approach that is accompanied by a strong theoretical background (Schwartz et al, 2000). This motivation for change is not new, since more than eight years ago the National Science Board indicated that only 4.5% of 24 year-olds in the United States pursues degrees in the natural sciences, mathematics and engineering fields (Science and Engineering Indicators, 1998). Science and Engineering indicators in other countries show similar or even worse results. The main cause for this seems to be a function of the learning environment (methodology) and the teaching tools (technology) typical in the natural sciences, mathematics and engineering fields which do not match the learning preferences of most of the student population. Students have preconceptions and research shows that the traditional "read, observe, listen to the lecture and memorize" method of teaching does not promote conceptual understanding and fails to challenge the preconceptions of students (Andre, 1997). Regarding the second objective, the technology and traditional tools used for teaching and learning seems to be not appropriate, are ineffective and in some cases, simply outdated. Virtual instrumentation and a graphical programming language are to be considered in this work, in order to evaluate its effectiveness and applicability in this control engineering experiments and hands-on control laboratories.

3. Scope of Work

Considering the wide range of topics, methodologies and techniques to choose from, a scope of work was defined. The scope of this work is limited to Control theory and more specifically, on linear-time invariant (LTI) systems of first and second order, with and without time-delay. Also, this work is focused on the representation of continuous systems with transfer functions, leaving other mathematical representations as optional (State-Space, Zero-Pole-Gain, etc.). Additionally, this work focuses on

classical control problems for single-input and single-output (SISO) open-loop and feedback (closed) loops, more specifically on Proportional-Integral-Derivative (PID) controllers. Finally, this work covers both software-only simulation of control systems and software-hardware, real-world control of plants (processes). Emphasis is put on the practical aspects of control design, not on the theory behind it, in order to maintain a experimental, hands-on approach for both, software-only simulations and hardware-based real-time implementation of control systems.

4. Methodology and Tools

In line with the objectives and the scope of this work, a simple methodology was defined. The methodology is based on the model-based design process, in which a student first understand the process or plant to be controlled, then defines the mathematical model that best defines the plant, designs the proper controller for the plant and simulates the controller-plant system in software. Then, the student develops a real-world prototype of the system which is later deployed in the best platform for the application such as a custom embedded real-time controller, a PC with a real-time operating system and DAQ boards, or a traditional industrial controller. This interactive, Design-Prototype-Deploy (DPD) methodology is implemented with state-of-the-art software and hardware tools that are easy to learn and use by entry-level students as well as senior or graduate students that need to develop very demanding, complex applications in a short time and with a relatively small budget. The DPD methodology has been tested with EE and ME students at leading universities in Latin America (Latam) in control design classes and laboratories. Most of these students are new to the tools used in this work (graphical programming language and virtual instrumentation) and must complete the DPD cycle in a period of six months or less (one semester or less). An introductory three to six hours hands-on seminar is conducted for groups of ten to twelve undergraduate or graduate students that are new to the tools to be used (graphical programming language and virtual instrumentation). Then, the tools are incorporated into their regular laboratory assignments and experiments such as DC motor control, inverted pendulum control, ball-and-beam control, magnetic or air levitation control, and many others (in many cases, the student is free to choose the experiment of their preference).It's known that one of the current challenges in engineering education is how to make it more engaging and more fun (methodology). Every engineering program is attempting to address the issue of student retention. The use of virtual instrumentation and graphical programming languages (technology) in classrooms and laboratories for freshman students can make engineering more enjoyable during the crucial first year. For example, virtual instrument-based robots can be used as a robotic or mechatronics teaching tool. Examples of this are nationwide student competitions in robotics such as RoboCup (robot-based soccer games). Using a graphical programming language together with different programmable "bricks" and wireless transmitters, students can download "instructions" to their robotic vehicle or device. Students are challenged to develop a virtual instrument-based application that makes their "robot" to grab a ping-pong ball and drop it through the hole in the center of a table. The vehicle includes motors, wheels, and a light sensor, enabling the vehicle to "follow" for example, dark lines on a table or on the floor. In the freshmen year, the software used (graphical programming language) only includes a very simplified set of features. As the student advances in his/her career, more sets of functions are incorporated into the programming languages, empowering the student for developing more complex, advanced applications. For example, when learning control theory, a student can develop very simple simulations of first and second-order systems. Figure 1 shows a simple example (First-Order System) created with a graphical programming language. The so-called block diagram represents the actual source code of the application. The "front-panel" represents the human-machine interface (HMI) for the application. In this example, the student can interactively introduce changes in the gain and time constant of the first-order system, and immediately see the changes in the transfer function (TF) of the system in the form of new coefficients for the numerator and denominator of the TF.

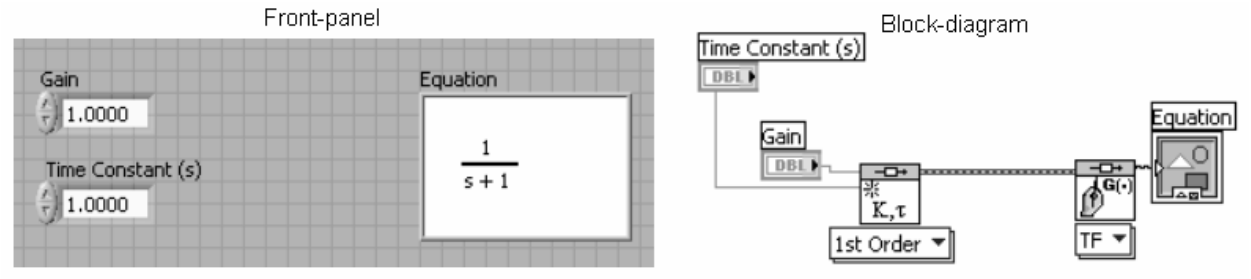

Figure 1: Front-panel with controls and indicators as related to the block diagram

5. Learning Control Theory with Virtual Instrumentation

Following the DPD methodology, the student will model the system, simulate it, verify it with a prototype, test it and deploy it with the same set of tools. The same procedure applies to the active teaching-learning process in for example, a control lab. As an example, a generic SISO, linear, time-invariant (LTI) closed-loop feedback control system is represented in a block diagram (Figure 2).

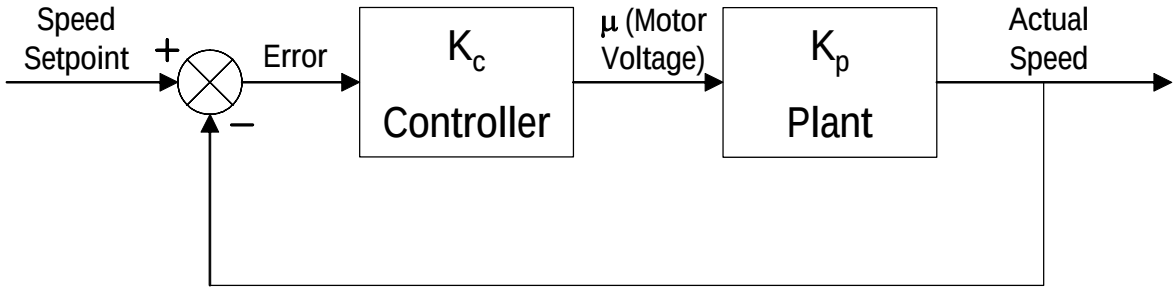

Figure 2: Controller and Plant System

In the following example, an electric motor (DC) controller is to be modeled, simulated, prototyped and tested with virtual instrumentation and graphical programming language.

5.1 Plant model:

The first step is to develop a model of the electric motor. This model will show the student how its parameters act and affect the behavior of the system. The model developed by the student contains some of the equations used to get the transfer function of the electric motor. As an option, a System Identification toolkit and a DAQ board can be used to "characterize" the motor and obtain the coefficients of its equations. For the purpose of simplicity in this work, three parameters that affect the motor transfer function are considered: K (constant), R (resistance), and J (inertia), as shown in Figure 3. It's assumed that the inductance (L) and the friction (b) are very small and negligible. So, the electrical and mechanical diagrams of the plant (DC Motor) are shown in Figure 3.

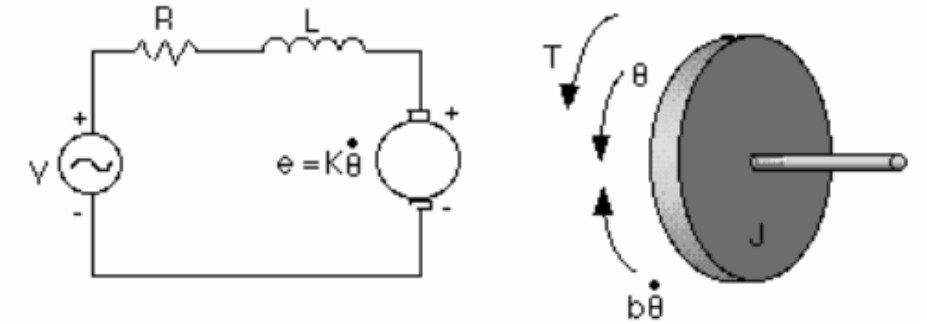

Figure 3: DC Motor Electrical and Mechanical Models

Equations for both models (electrical and mechanical) are developed. The circuit diagram of the electric motor consists of a voltage input (V), resistor (R), inductor (L), and back EMF voltage (e) connected in series. In the mechanical diagram, torque (T), position (θ), friction (b), and inertia. The torque of the motor is proportional to the current consumed by the electric motor. After simplifying and substituting values in the equations for each model, and by taking the Laplace Transform, the resulting transfer function of this plant (motor) is obtained:

$$\frac{\text{Angular Speed}}{\text{Input Voltage}} = \frac{\omega(s)}{V(s)} = \frac{K}{JRs + K^2}$$

With this model of the plant (motor), our control system can now be represented as shown in Figure 4.

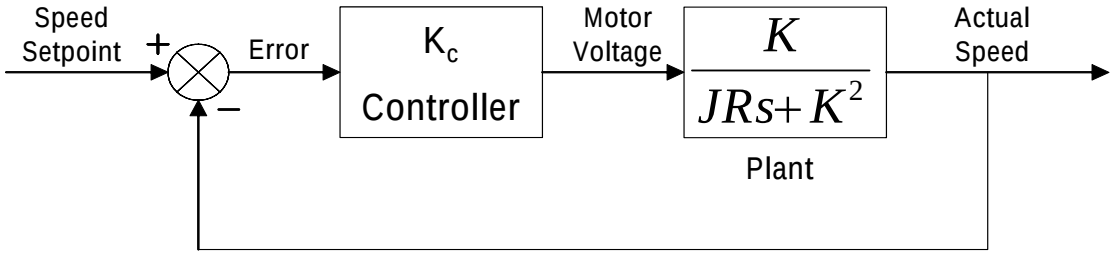

Figure 4: Updated Controller and Plant System

Now that an accurate model of the plant (electric motor) is available, the student proceeds to interactively change the values of K, J and R and evaluate the results in real-time.

5.2 Plant Analysis and Simulation:

Once the plant is mathematically modeled, it can then be dynamically simulated and analyzed. For doing this, the student switches to the Analysis or Simulation module of the application. At this stage of the design process, the student analyzes the Root-Locus Plot (other options or techniques are available), step response, bode magnitude, and bode frequency of the plant model (Figure 5). In fact, the student can adjust the model parameters and see how the behavior of the system changes. The root locus method helps evaluate the effect on the roots of the equation by modifying the parameters of the controller. In the example shown bellow, a gain Kc has been put into a feedback loop with the model for the electric motor. With this system, there is only one pole (a first order system):

$$s = \frac{-K^2}{JR}$$

The Root Locus Plot is similar to the Pole-Zero Map except that it shows the trajectory of the poles and zeroes as the parameters of the controller are modified. This can also be used to evaluate the stability and performance of the controlled plant on a later stage.

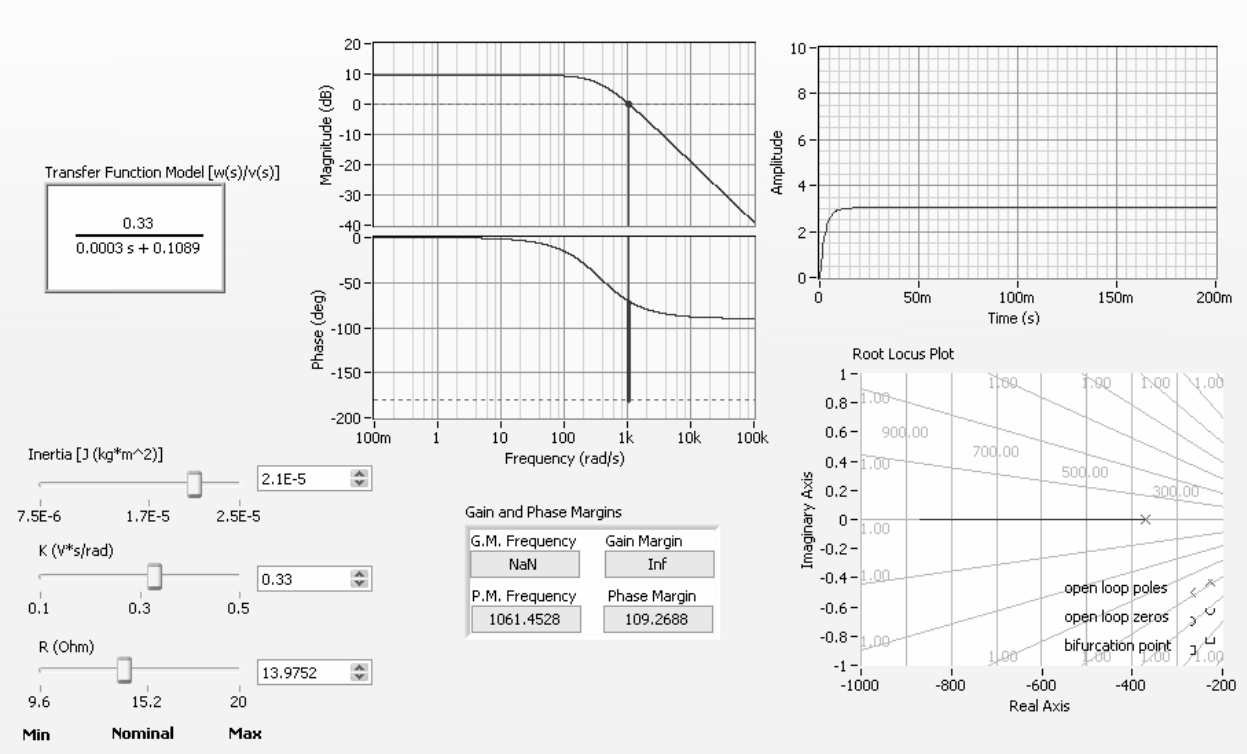

Figure 5: DC Motor model analysis

5.3 Control Design:

Once the DC motor is modeled and simulated, the student starts to take the steps for developing the corresponding controller (Figure 6). Different controller options can be applied. In this example only two controller options are evaluated. The first one is simply a proportional (P) controller. In this case, a proportional gain is placed in a feedback loop with the plant model. The second one is a proportional-integral (PI) controller. As an alternative, a proportional-integral-derivative PID controller could also be evaluated. In the first case, the value of Kc can be adjusted. As a result, the rise time of the electric motor rises and falls based on this Kc value. From the results, the student realizes that while the rise time improves as you get closer to 1, the response of the system actually never reaches 1, meaning that there is steady state error in the system when using the proportional controller in a feedback loop with the electric motor. Because there is steady state error, we will need to include an integrator into the system, becoming a Proportional-Integrator (PI) controller. The controller equation now includes an integrator with a time constant of Ti as follows:

$$Kc\left(\frac{s+1/T_i}{s}\right)$$

Once the controller equation is defined, the student can adjust the Kc value and see how it affects the response rate. The step response first shows a little of overshoot and then settle out at 1 (Figure 7). As the Kc value is further adjusted, it can be noted that the poles and zeroes on the root locus graph also

move along their trajectory. When the poles go into the imaginary plane, the response rate shows the overshoot which is indicative of a second order system. The Ti value can also be adjusted causing a change in the root locus graph. The Kc and Ti values can then be changed interactively until the best response of the system is obtained. With the controller defined, our modeled system is now represented as shown in Figure 6.

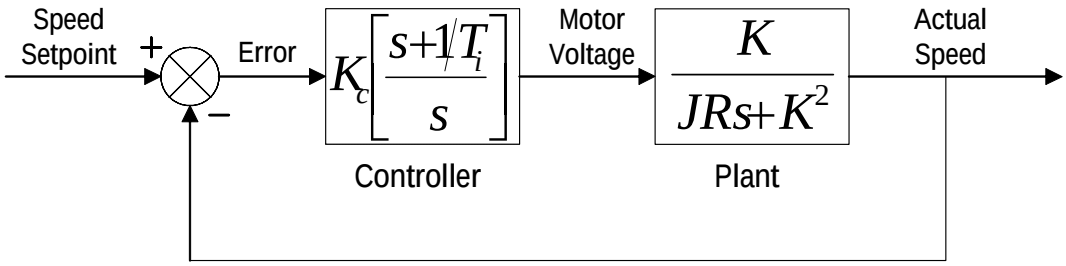

Figure 6: Updated Controller and Plant System

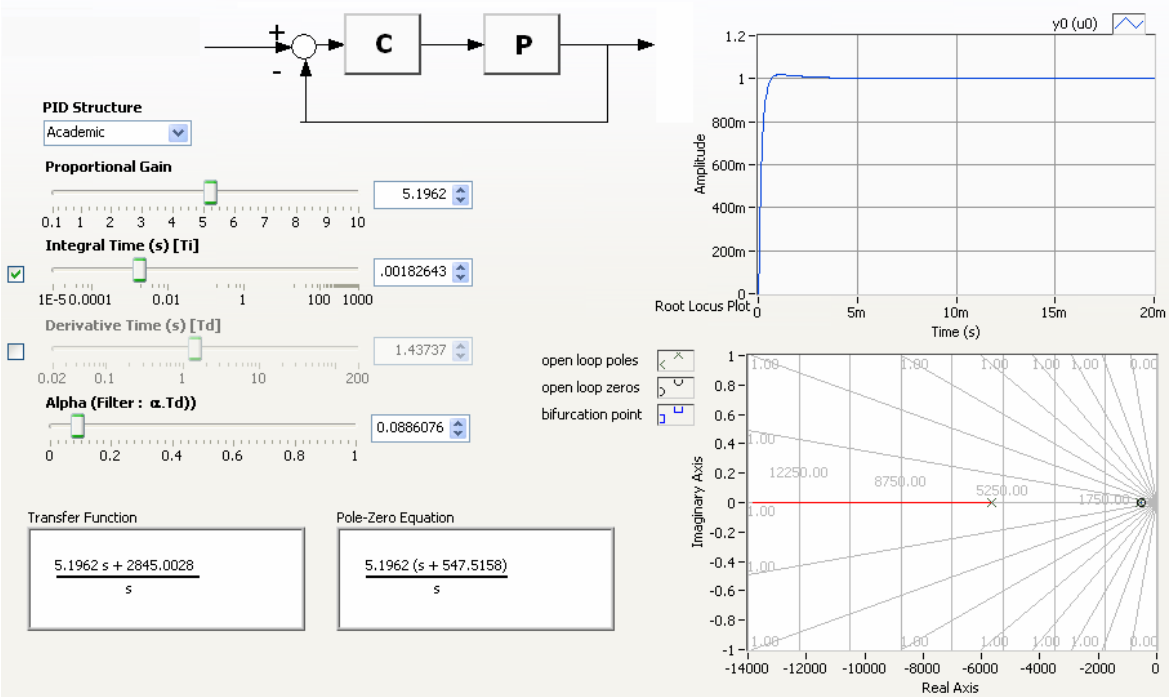

Figure 7: PI Controller design

5.4. Controller-Plant System Analysis and Simulation:

Next, using the modeled controller (PI controller) and motor (plant), the student starts the analysis and simulation process of the complete controller-plant system. Different responses of the electric motor and controller to different simulated input signals (square, saw tooth, and sine wave) are evaluated. At any moment, the student can go back to the PI Control design module to modify the parameters (gains) of the controller, and immediately see the results in the Simulation module. For example, in order to see a simulation of the controller responding to a square wave (Figure 8), the student switches to the Signal Type option in the application, selects the corresponding wave (square wave), and look at the results. The student can once again modify the Kc parameter in the PI controller module to a different value, switching back and forth between the simulation module and the PI controller module as needed. In this

example, it will be noted that with the new gain the controller doesn't track the speed setpoint that well. When a sawtooth wave is selected, the student can definitely see the deviation. At the end, the student is able to define the best values for the PI Controller proportional and integral gains.

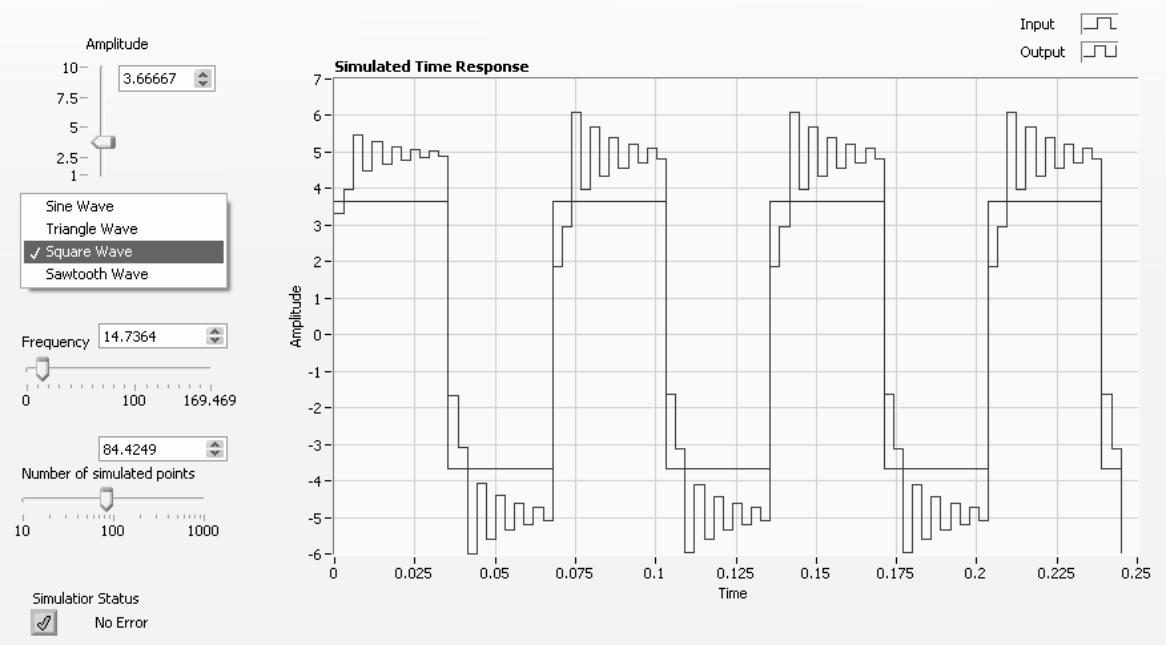

Figure 8: Time analysis of the motor and PI controller (Square wave)

Once the student has fully simulated the controller and the plant, he/she can now continue with the experiment and prototype the system with real-world hardware.

5.5. System Prototyping and Testing:

In this stage, the student can create a prototype of the electric motor controller with the real-world motor. This is an important part of the active learning process because he/she can actually see the controller they designed being applied to a real DC motor. In order to build a prototype of the system, an optical encoder (or photogate), a DC motor, a power supply and some resistors are required. The motor is to be controlled by a computer (PC) where the controller virtual instrument will be executed. As an alternative, a programmable automation controller (PAC), a stand-alone real-time controller with input/output modules can be used. A PCI-bus data acquisition board (DAQ) is used for handling the I/O signals (channels ACH4+, ACH4- are used) to/from the motor and the optical encoder (Figure 9).

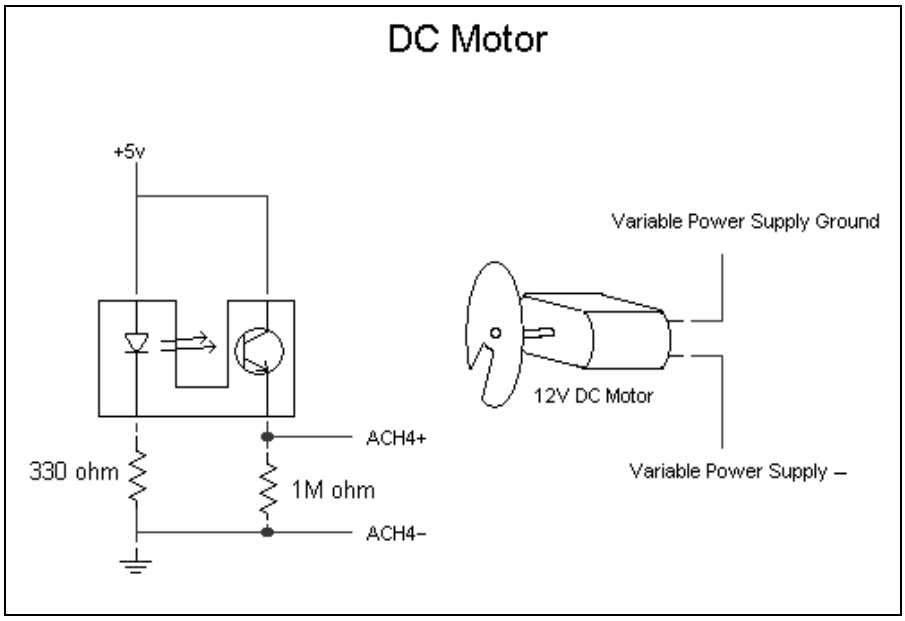

Figure 9: Motor and optical encoder

The setup can be built on a prototyping board which will be connected to the data acquisition board. The DC motor needs a voltage differential to run so the Variable Power Supply ground and negative leads are used to supply power to the motor (Figures 9 and 10). In order to measure the speed of the motor, a notched circular disk is used (Figures 9 and 10). When the DC motor rotates, the disk generates a pulse train in the optical encoder that is captured with the data acquisition board. This information is then used by the application to calculate the RPM of the motor.

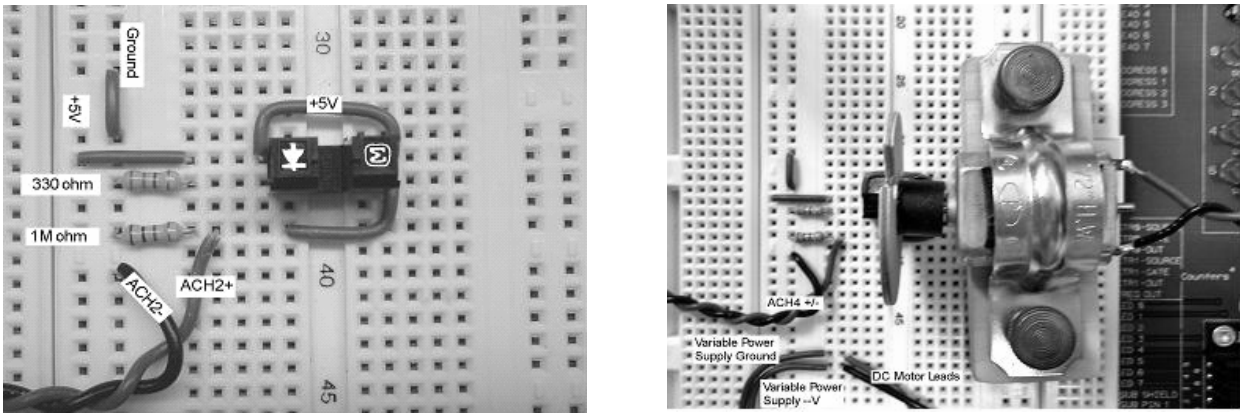

Figure 10: Building a prototype of the system

Since the controller built by the student is a PI controller, he/she could easily implement the final system with a different controller such as a PID controller. In order to have a PID controller, the student could easily modify the controller (Figure 7) and add a derivative component. When a PI controller is selected, the derivative value (D) value is set to zero. At this point, the complete physical system, using a PI controller, can be represented as shown in Figure 11.

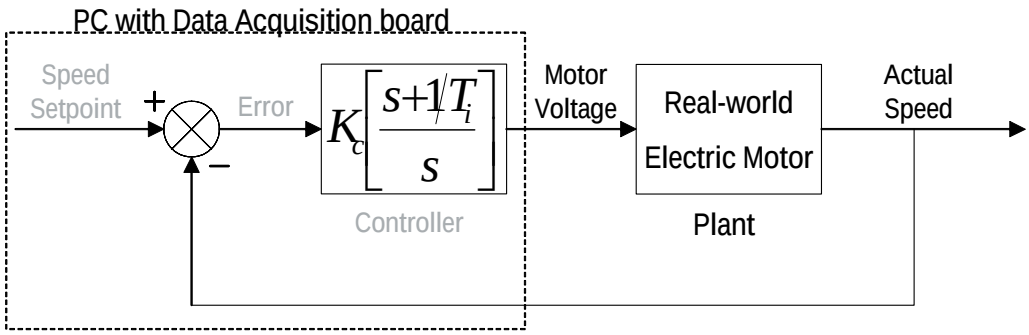

Figure 11: Final Controller-Plant System (Deployment)

As shown in Figure 11, the controller has evolved into a PC equipped with the application program (PI controller) and a data acquisition board with analog and digital inputs/outputs. Also, as shown in the same figure, the plant model has been replaced by a real-world DC motor. The experiment has been completed and executed in all its phases, from design to test. Other curriculum topics could be added to this experiment using the same virtual instrument (Disturbance Rejection, Tracking Control and Regulation, Lead / Lag Compensation, Frequency Analysis, Nyquist Stability, System Identification, etc.).

6. Conclusions

A Virtual Instrumentation application has been built with a PC equipped with an application program (front panel and function block diagram), a data acquisition board, a motor, an optical encoder, a power supply and some other basic electronic components (resistors, etc.). Virtual Instruments based on graphical programming languages not only allow you to model and simulate control (linear or nonlinear, continuous or discrete) systems, but also to prototype and test them in real-time without the need of an intermediate language such as C, C++ or VB. Using a simple yet powerful, intuitive, easy to learn, graphical programming language, the user of such a system can go through all the phases involved in the design of a controller-plant system. In the first phase, the user or student works with a software-based model of the controller and the plant. In the second phase, the student applies the software model of the controller to a physical model of the plant (actual motor), validating and testing the system. Rapid prototyping and Hardware-in-the-loop testing are easily implemented. Using this method, the learning-teaching process becomes an active one, in which the student not only learned and studied the theory behind DC motor control design, but also had the opportunity to build a real-world working model of the system in which the controller he/she designed was actually controlling an electric DC motor. By easily prototyping the system with real-world hardware, the user or student can test out the control algorithm with real-world dynamics. For instance, if a sensor has a rather long acquisition time/delay, this can't be accounted for in the simulated world. Real-world prototyping helps identify these problems before the control engineering student spends too much time in targeting the application (generating the source code, compiling it and downloading it to a specific target such as a microcontroller in order to test it). Finally, the fact that the student can validate the theory behind control design with a working prototype of the controller-plant system using flexible virtual instruments helps to enhance the education process and help to make engineering education more engaging, enjoyable, fun and effective. In short, virtual instrumentation improves the quality of education students receive while they enjoy to learn, to experiment, to create. All the students that participated in this research at different universities in the LATAM region have shown a new interest in the Control Design subject, more engagement and better academic performance, demonstrating the positive effect of the active-learning approach.

References

Alleyne Andrew, Brennan Sean, Rasmussen Bryan, Zhang Rong, and Zhang Yisheng, Control and

Experiments: Lessons Learned, IEEE Control Systems Magazine, October 2003
Bernstein Dennis S. and Apkarian Jacob, Experiments for Control Research, IEEE Control Magazine, October 2003
Geen F. Franklin, J. David Powell, Abbas Emami-Naeini, Feedback Control Dynamic Systems, Addison Wesley, 1986
Schwartz Trudy L. and Dunkin Bradley, Facilitating Interdisciplinary Hands-on Learning using LabVIEW, The International Journal of ENGINEERING EDUCATION, 2000
National Science Board, Science and Engineering Indicators 1998, Arlington, VA: National Science Foundation, NSB 98±1 (1998)
Andre T., Minds-on and hands-on activity: improving instruction in science for all students, Mid-Western Educational Researcher, 10, 2 (1997)
Astrom Karl J., Hagglund Tore, "PID Control" in The Control Handbook, ed. William S. Levine, CRC Press, 1996
LabVIEW User Manual, National Instruments Corp., 2003
NI ELVIS User Manual, National Instruments Corp., 2003
Ahrends Stephan and Vento Tony, Control Design 101, National Instruments Corp., 2003
Academic Resources CD, National Instruments Corp.., 2003

Authorization and Disclaimer