2010 American Control Conference
Marriott Waterfront, Baltimore, MD, USA
June 30-July 02, 2010

Redesign of an Undergraduate Controls Laboratory with an Eye toward Accommodating Future Upgrades

Jacob Wieneke, Dale Schinstock, Warren N. White, and Guoqiang Hu

***Abstract*—This paper describes a recent and very advantageous upgrade of the undergraduate controls laboratory in the Mechanical and Nuclear Engineering Department at Kansas State University. The current lab has been in use for about a decade. Details regarding the original hardware fundamental to the laboratory activities, including the embedded digital signal processor (DSP) and the brushless DC motor stand (called the "Motorlab"), are presented and a summary of each of the fourteen weekly lab exercises is included. The DSP control program and PC user interface (UI) were each written in C. In order to accommodate future lab needs and to ease the maintenance burden, the embedded DSP was replaced by a PC running Real-Time LabVIEW from National Instruments (NI) and a NI data acquisition card. The original lab software was replaced with programs produced graphically in LabVIEW, known as Virtual Instruments (VIs). The VIs implement the controller on the real-time PC as well as the user interface on a host PC. Both the embedded DSP and the LabVIEW controllers are able to close the loop on the laboratory equipment with a 10 kHz, hard real-time, sample rate.**

I. Introduction

A laboratory component attached to an undergraduate controls lecture is an excellent way of both reinforcing lecture material and providing hands-on experience to the students. Applying lecture principles to actual hardware, the theoretical issues and tools are demonstrated to be practical.

The subject of creation and organization of control laboratories is one that has been addressed in technical literature for several decades. An example of some early work is the University of Minnesota digital control lab reported by Bailey and Waltz (1975). Over time, the direction of control laboratory development has broadened from traditional "lab bench and hardware" environment to include remote experimentation through the Internet and virtual laboratories. Ramakrishnan et al. (2000) describe a web based means of controlling the liquid level in a tank. Valera et al. (2005) present virtual process control and virtual robotic control laboratories. It is easy to appreciate the motivation behind these new development directions in light of limited resources, limited space, and providing large classes with the ability to perform exercises within a flexible schedule. However convenient such laboratories might be, the original "lab bench and hardware" environment still has merit.

The implementation of real-time control in an undergraduate laboratory follows several familiar paths, such as using an embedded DSP in a host computer, running real-time control under Windows, or using an external computer for real-time control. A popular approach is to use Real-time Workshop from MATLAB to create and download control software for external computers and DSP devices. An example of such an application is presented by Spong (1999). An example of using an external computer for data acquisition and real-time control is Kapila et al. (2000), where the Wincon software and the Multi-Q hardware were used for control.

LabVIEW Real-Time has also been utilized for control. Astilean and Folea (2006) used this software to program an embedded processor used in manufacturing applications. Mrad et al. (2000) used real-time LabVIEW on an embedded processor board to control an inverted pendulum cart. Searches of controls literature has not yielded applications of a laboratory based on a PC running LabVIEW RT connected to a Windows machine hosting the user interface.

The introductory undergraduate controls class in the Mechanical and Nuclear Engineering (**MNE**) Department at Kansas State University has had a laboratory component for nearly two decades. During that time, there have been significant changes in both computer software and hardware within the lab. In the current version of the laboratory organization, the students meet weekly and perform a series of fourteen laboratory exercises that closely parallel the lecture. The laboratories utilize a brushless DC motor stand that had been previously controlled by an embedded DSP on a motion control card sitting in the PCI bus of a PC. The DSP provided hard real-time control with a 10 kHz sample rate. This rate was chosen because it is appropriately faster than the motor, amplifier, and sensor dynamics. The controller was written in C and the compiled code downloaded from the PC. The user interface running on the host PC was written in C++. Communication with the

Manuscript received September 22, 2009. This work was supported in part by the Mechanical and Nuclear Engineering Department of Kansas State University. The authors gratefully acknowledge the assistance.

J. Wieneke is a graduate student in Mechanical and Nuclear Engineering at Kansas State University, Manhattan, KS 66506 USA (e-mail: jwieneke@ksu.edu).

D. Schinstock is with the Mechanical and Nuclear Engineering Department, Kansas State University, Manhattan, KS 66506 USA (e-mail: dales@ksu.edu).

W. N. White is with the Mechanical and Nuclear Engineering Department, Kansas State University, Manhattan, KS 66506 USA (phone: 785-532-2615, fax: 785-532-7057, e-mail: wnw@ksu.edu).

G. Hu is with the Mechanical and Nuclear Engineering Dept., Kansas State University, Manhattan, KS 66506 USA (e-mail: gqhu@ksu.edu).

embedded DSP was implemented using a set of drivers provided by the manufacturer which are no longer standard products and not currently supported. The C programs were written and the motor stands built in 2002 and have required little maintenance since. However, the continued use of unsupported drivers in new operating systems, the expense and maintenance of a cross compiler for the DSP, and the availability of real-time solutions in more general purpose software packages that are commonly available in the academic environment (such as LabVIEW) provided motivation to consider alternative architectures.

The constraints for a new controller system were:

1) A controller that was flexible and easy to maintain from one operating system version to the next.
2) An economical controller implementation.
3) A system capable of 10 kHz hard real-time sample rate.

The task of modifying and debugging C code each time it is necessary to improve and/or maintain the controller and the controller interface is bothersome. The software for the embedded DSP was written "in house", and as a result met specific needs. A software tool that generates the controller code might not produce efficient code resulting in benchmarks such as a desired sample rate not being met. Modifying the code produced by the software tool to achieve the specific benchmarks might not be straightforward. Several options were examined before selecting the new lab development direction.

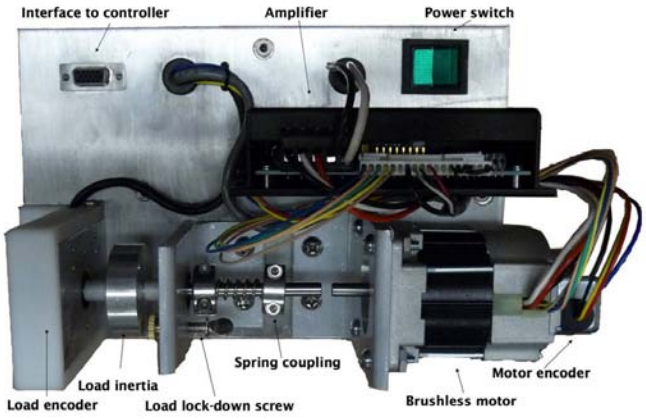

Fig. 1. Annotated picture of Motorlab hardware

An honors student project in the fall of 2007 showed that the same control provided by the DSP could be achieved using a PC running real-time LabVIEW from National Instruments. A significant benefit of this approach is that both the controller program and the user interface are created through the same graphical programming means. Since the submission of that honors project, the software has been perfected and sufficient hardware purchased to incorporate the real-time controller in each of the eight stations of the controls lab. These software and hardware revisions are being implemented in our laboratory during the spring 2010 semester. The real-time PC uses an operating system that is similar to the original PC DOS. The real-time PC communicates asynchronously with the user PC directly through Ethernet. The user interface was created with NI LabVIEW to resemble the original C++ interface. Because the real-time PC does not need to support Windows, a monitor, a mouse, or a keyboard, the hardware requirements for the real-time PC are minimal. In fact, the PCs used in the lab prior to a regularly scheduled update of the computers are used as the real-time PCs.

II. Original Motorlab System

The original Motorlab system was constructed in the summer of 2002 and the laboratory was equipped with 10 identical systems. While it is not the focus of this paper, the new system closely models it and utilizes much of the original hardware. The major components of this system are:

1) Brushless motor: LA052-040E from Shinano Kenshi
2) Amplifier: Model 503 DC brushless servo amplifier from Copley Controls
3) Load encoder: RCML15 low profile encoder from Renco Encoders, Inc.
4) 24 Volt DC power supply
5) Motion control card: MC4000 DSP Motion from Precision MicroDynamics, Inc. (PMDi)

All of these parts are still currently available at a relatively low price with the exception of the power supply and the motion control card. The specific manufacturer of the power supply is unimportant, and in fact a very inexpensive surplus power supply was used in the original system. The most expensive piece of the system, the MC4000 motion control card, is still available, but it is

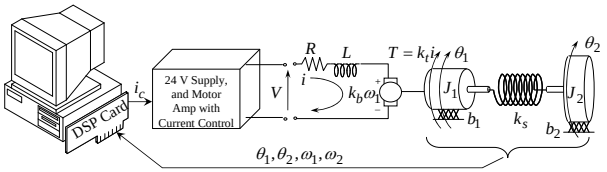

Fig. 2. Closed Loop Control Schematic of original Motorlab System

currently sold with proprietary software for the Analog Devices SHARC DSP already loaded. In the original system, custom control software was written at KSU using a cross compiler from Analog Devices and utilities provided by PMDi were used to download the programs to the card and interface with the host computer through dual-port memory; these utilities served their purpose, but they are not standard products of PMDi. The authors greatly appreciate the support of PMDi during the development of the original Motorlab system.

The Motorlab system includes two position sensors on the apparatus. The position of the motor shaft is measured using the encoder supplied with the motor and the position of the load inertia is measured using the load encoder. In addition, the velocities of the two inertias were measured using hardware on the MC4000 motion control card that

accurately measures the time between pulses coming from the encoders, a capability that is difficult to match in general purpose computer interfacing hardware. The motor amplifier has an analog control loop that measures and controls the electric current in the motor windings. This results in what is commonly known as a "torque controlled" motor. The bandwidth of this loop is approximately 400 Hz. While it would be possible to tune the amplifier to obtain a higher bandwidth it was left low intentionally to enable

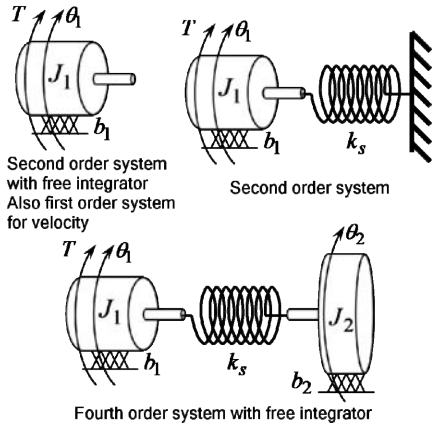

Fig. 3. Variations on the mechanical model

observation of the loop dynamics with a 10 kHz sample frequency. The sample frequency was chosen so that the limitations of the various sensors (i.e. time delays) and components (i.e. bandwidth) could be demonstrated without the sample rate being a limitation. The DSP motion control card was interfaced to the motor amplifier through a ±10V analog signal from a digital to analog converter (**DAC**) on the card. By varying the magnitude of this voltage from the DAC, the current in the motor is varied. This voltage, which is proportional to the controlled current, serves as a current command for the current control loop in the amplifier. An additional sensor, not shown in the schematic, is the current sensor in the amplifier. This sensor was also read by the DSP card, using an analog to digital converter (**ADC**). Although this signal is not used in the control loops on the DSP card, it is recorded for data analysis.

Several different configurations of the system can be utilized in experiments. Either the motor or load encoder can be used for control loop feedback, with a selection in the user interface. The motor encoder is a collocated sensor while the load encoder is a non-collocated sensor. Additionally, the mechanical system can be changed with the lock down screw and spring coupling. Also, a choice can be made between velocity control and position control by selecting the appropriate control program. The mechanical models shown in Fig. 3 can be realized with this system and further dynamic variations achieved through choice of position control, velocity control, or open loop control, and through choice of the feedback sensor, providing many different variations for the exercises.

In the original software, there were three different programs used to control the Motorlab hardware. Each program consisted of a GUI interface that ran on the host PC and a low-level control program that ran in the DSP. The host software was written in the Borland C++ Builder environment and the DSP software was written with a cross compiler for the SHARC DSP from Analog Devices. An example (the position control program) of the host GUI is shown in Fig. 4. The PC's processor and the DSP communicate over the PCI bus in the host computer using dual-port memory. A PID controller was used in the two programs that implemented closed loop control. Additionally, the user has the option of including feedforward velocity and acceleration gains. In the open loop program the feedback sensors are not used for closed loop control. Also, the DAC output from the motion control card to the motor amplifier was determined directly by the wave command (square, triangle, etc.) buttons and the jog buttons on the user interface. In the position (velocity) control program the feedback sensors were used to close the position (velocity) control loop. The DAC output from the motion control card to the motor amplifier was determined by the controller algorithm.

III. New Motorlab Controller

In order to implement a system that could be easily upgraded and maintained with operating system upgrades, a solution was created based on National Instruments LabVIEW and LabVIEW Real-Time. LabVIEW Real-Time is a hard real-time operating system that can be installed on many targets, including properly outfitted desktop PCs. National Instruments regularly updates LabVIEW, which ensures the laboratory control program can be kept up-to-date with computer hardware and operating system upgrades. By replacing the DSP card with a desktop computer, a cost-effective alternative to purchasing specialty systems or commercial devices has been created that satisfies the development constraints.

The new system offers the benefit that in the event the host program needs to be modified, the MNE Department maintains a current site license for LabVIEW allowing the host program to be easily modified and re-deployed; previously, modifying the user program would have required the department to find an older copy of Borland or purchase a new license for a program that would be used for a single purpose. From the students' point of view, there is very little change from the system that uses the DSP card and the system that utilizes a dedicated real-time PC. Due to the nature of programming in LabVIEW, the existing user interface was easily replicated and improved as shown in Fig. 5. The button layout was changed to incorporate the three control scenarios into one user interface, instead of three separate programs. To allow the students to change the PID gains directly (instead of opening a menu) the buttons were rearranged.

Also, functionality was added to make data collection of step responses easier for the students. Previously, while

running a square wave students had to turn on 'Store Data' and then click 'Save Data' to capture the pertinent data. In the new system, students need only click 'Auto-Save

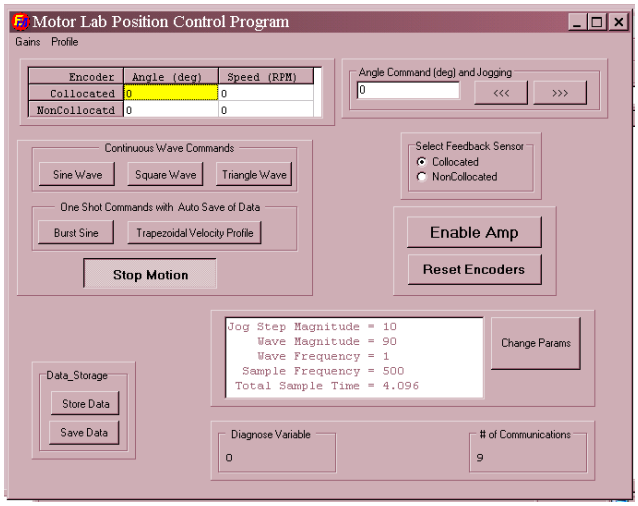

Fig. 4. Host position control GUI for the original lab setup.

Square' and the program calculates when the data buffer has reached capacity and saves the data on the host machine automatically, without further input from the user. Another data-saving simplification implemented is the 'Log Data' button, replacing the 'Store Data' and 'Save Data' buttons.

The laboratory was recently remodeled and student workstations were upgraded. As a result of this upgrade, the previous desktop computers were made available to use as

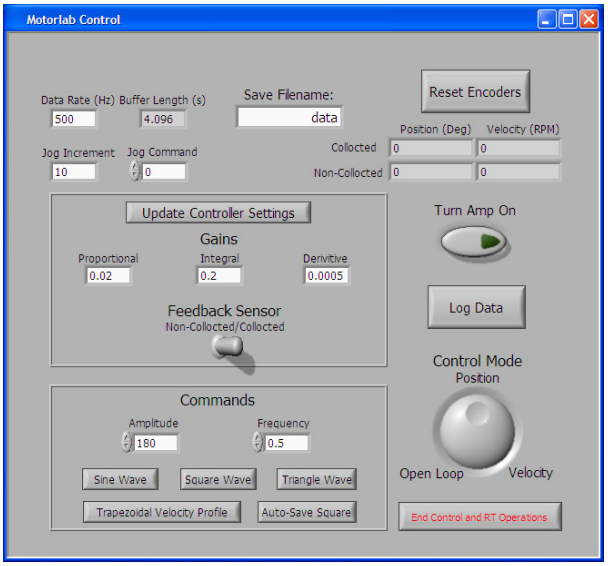

Fig. 5. Host GUI for new Motorlab control program.

real-time targets for controlling the motor stands. These machines possess hardware characteristics that are useful in real-time targets such as Intel processors, floppy disk drives (useful for LabVIEW Real Time software installation), and BIOS settings that allow the SATA hard drive to operate in 'legacy' mode. Network Interface Cards containing Intel chipsets and NI PCIe-6361 X-series Multifunction DAQs were purchased to complete the real-time motor stand controller. These purchases were necessary to meet the communication requirements of LabVIEW Real-Time and to give the PCs the ability to interface with the motor stands. To complete the system, NI PCB connector blocks were purchased and cables were fabricated to connect the X-series DAQ to the motor stand.

The real-time code is implemented in three primary sections: initialization, the control loop, and the communication loop. During initialization the real-time PC establishes default values for variables and commands and starts the tasks needed for the lab apparatus, such as setting the channels and parameters for the amplifier switch, motor command, and encoders. The dual-core processors in the real-time targets are able to be programmed independently with LabVIEW, allowing the two loops of the real-time code to each run on a dedicated core. The control loop produces the command waveform, performs the control algorithm, sends the command to the motor, and records the data from the encoders. It is worth noting that without the hardware timers available on the DSP, the new system required the implementation of an observer or a low pass filter to minimize velocity signal noise. All of these tasks must be accomplished at the specified rate of 10 kHz. Initially, there were concerns that the real-time system would not be able to achieve the control rate demonstrated by the DSP card. However, thorough testing of the system and streamlining the code by reducing unnecessary communications with the host PC, the control rate was measured to be 10 kHz, duplicating that used by the older lab system. Simultaneously, the communication loop operates at a slower rate and manages the data and command transfer from the UI to the control loop, which allows the control loop to be streamlined and run deterministically. All of the communication between the host PC and the real-time PC is conducted over a crossover Ethernet cable.

IV. Brief Description of Laboratory Experiments

There are fourteen laboratory assignments associated with the introductory level controls course in the MNE Department. These laboratories progress from the fundamentals of modeling dynamic systems with differential equations and transfer functions to detailed analysis and design of closed loop control systems using methods from the time-domain and frequency domain. This progression is coincident with the topics in the lecture portion of the course. All of the labs make use of data from physical systems, solidifying difficult theoretical concepts for the students.

Intro Laboratory: This is an introduction to the Motorlab apparatus and software and to MATLAB. The students become familiar with both in this lab as they are used throughout the semester.

Laboratory 1: Using constant motor current inputs to create constant velocities, the students experimentally determine an approximation for the viscous friction coefficient of the brushless motor system shown in the upper

left hand corner of Fig. 3. Then the students use this coefficient together with other motor parameters to predict the system response to an initial condition (initial velocity) in MATLAB and compare to the measured response.

Laboratory 2: The students are required to experimentally determine the coefficients for the plant model of the motor-and-spring system shown in the upper right hand corner of Fig. 3. They are able to determine the inertia from the motor specification sheet and estimate other model parameters using experimental data such as the steady state deflection obtained from a constant motor current/torque. The students also compare the computed step response of their model to experimental data from the actual system.

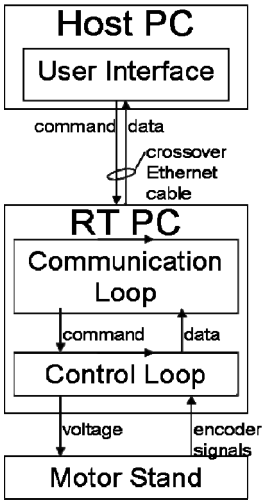

Fig. 6. Diagram of new Motorlab control flow.

Laboratory 3: In Lab 1, the students find a linear estimate of viscous friction in the motor. It is obvious from the data that the friction also has nonlinear effects. In this lab the students explore these nonlinear effects through simulation in SIMULINK. They use a model which includes a high coefficient of friction at low velocity and a lower coefficient at high velocity to simulate the nonlinear friction. Using this simulation, the students are able to generate a nonlinear initial condition response for the velocity of the motor that closely matches the experimental data in Lab 1.

Laboratory 4: In this lab the students experiment with a proportional position controller, where the current command to the motor amplifier is proportional to the error between the actual position and the command. Also, they use a model of the closed-loop position control system to predict the system response. They compare the theoretical step response with the actual response obtained experimentally from the Motorlab for three different proportional controller gains. Then they make connections between pole locations and characteristics of the response such as the frequency and decay rate of oscillations in the closed loop response.

Laboratory 5: Continuing their work in the previous lab, the students use high values of the proportional gain, K_p. This lab illustrates that there are always un-modeled, higher frequency dynamics that will affect the response if they "turn up the gains" too much. The students begin to learn when it is appropriate to ignore these dynamics and also how to account for them in control design without an exact model.

For a low range of K_p values the students observe that the predicted and measured performances are similar. However, the students see that as the poles of the simple model move farther from the origin of the s-plane it is necessary to include dynamic effects from the amplifier to explain the response. They verify the following rule of thumb: "Open loop poles and zeros can be ignored when they are more than 10 times larger in magnitude than the closed loop poles that result from ignoring them." They also note exceptions to this rule such as lightly-damped open loop poles.

Laboratory 6: Motivated by the previous lab focusing on "higher frequency dynamics," this lab requires the students to model the closed-loop current control system implemented in the motor amplifier and to compare the response of their model to experimental data acquired from the system.

A schematic from the amplifier manufacturer is given in the lab assignments and homework providing a functional diagram of the amplifier for the brushless DC motor. It implements closed loop current control of the motor using operational amplifier circuits that the students analyze in a homework assignment. The students develop a model of the closed loop current control system using the schematic and rudimentary explanations of the power electronics. The step response of the model gives nearly an exact match to the step response obtained from experimentally from the system.

Laboratory 7: In this lab, the students experiment with the velocity control system of the Motorlab apparatus. It is modeled simply, ignoring all higher frequency effects beyond the first order model of the inertia and friction. At this stage in the semester the students are capable of performing many detailed calculations relating the closed loop system to the open loop system. This simple model provides an opportunity to reinforce many of these calculations with real data and without confusion.

Laboratory 8: The students examine the closed loop response of a fairly complex, higher order mechanical system, where there are many significant poles and zeros, and where there are substantially dominant poles and zeros with less dominant poles and zeros causing superimposed effects in the response. This mechanical system includes the inertia coupled by a spring, as shown in the bottom of Fig. 3. The students use full PID control with a position controller for this system. For most of the lab the students use the motor encoder for feedback, but they also use the load encoder for feedback, demonstrating the destabilizing effects of a non-collocated sensor. Independent of which sensor is used for feedback, data from both sensors is available from the experiments. Common to nearly all the labs, the students are required to develop models for the system, validate the model with experimental data, and make

connections in the theory from the lecture part of the course.

Laboratory 9: In labs 4 and 5 the students experimented with a proportional (P) controller for position control in the Motorlab apparatus and found that as they raise the gain of the P controller that better control of the system can be obtained, but this improvement is limited. The gain can only be raised so much before the response becomes very oscillatory. Furthermore, the settling time cannot be improved. In this lab the students compare the proportional controller to a proportional-derivative (PD) controller. The PD controller adds a zero to the open-loop TF, changing the shape of the root locus by pulling the poles into the left half plane. This is the students' first experience of adding dynamics to the controller to shape the root locus in the design process.

Laboratory 10: In this lab the students experimentally determine five data points for the frequency response of the motor-and-spring system of the Motorlab (upper right hand corner of Fig. 3). Then the students compare their experimental response data to the theoretical frequency response from a transfer function they develop, tweaking the parameters of the model to obtain a close match. Using sine wave inputs to the motor current, the students begin the lab by experimenting to find the resonant frequency of the system. While the students are searching for resonance, they begin to understand the basics of frequency response including amplitude ratios and phase shifts.

Laboratory 11: The students experiment with frequency response design for the velocity control system using a PI controller. They simply adjust "the gain," not moving the zero of the PI controller. Then using their data and models they make connections between closed loop bandwidth and open loop crossover, between bandwidth and the dominant closed loop poles and the speed of the step response, and between the root locus and frequency response design techniques.

Laboratory 12: This is an investigation of the importance of low-frequency gain on the tracking capability of a control system. The students use two PID controllers in the position control system of the Motorlab apparatus, both with the same crossover frequency and bandwidth, but with different open-loop low-frequency gain. The students compare the tracking capabilities of the two closed-loop systems using motions generated from trapezoidal velocity profiles. This is not only an opportunity to reinforce an important frequency response design goal, but also to introduce command shaping, which is important in industry.

Laboratory 13: The students tune a PI controller for the velocity control system of the Motorlab apparatus. The nominal dynamics of the plant, G_m, are known to be first order and therefore a PI controller works well. To tune the controller the students pretend to 1) know the structure of the nominal dynamics, 2) not know specific numbers for the model (just the structure), and 3) not know the higher frequency (limiting) dynamics, as is often the case when tuning a controller. The students use numerical estimates of the system parameters obtained from previous labs, not to tune the controller, but to generate Bode plots from the models at selected points in the tuning process to understand what they are doing during tuning process.

Throughout the semester, as the laboratory experiences reinforce the concepts of the course, the instructors observe that students progress from a fear of differential equations in many cases, to capabilities in relating complex relationships to real engineering systems.

V. Conclusions

In this paper, an undergraduate control laboratory has been described that consists of a brushless DC motor stand and a PC running real-time LabVIEW using NI data acquisition equipment. This laboratory setting has undergone a significant upgrade using a cost effective design. The 10 kHz sample rate real-time PC plus data acquisition equipment can be realized for approximately $2K per station while the motor apparatus requires about $1K per station. These per station costs are very attractive compared to the costs necessary to use specialty control laboratory products. The software was developed with LabVIEW 9.0 and is not compatible with previous versions of LabVIEW. Inside the available zip file containing all of the LabVIEW programs, the project file is KSU_Controls_Lab.lcproj, the user interface is KSU_Motorlab_HostUI, and the real-time controller is KSU_Motorlab_RT. The zip files for this project can be downloaded from:
http://www.mne.ksu.edu/research/laboratories/dynamic-systems-controls-laboratory-1/motorlab

The authors gratefully acknowledge the contributions of Victor Salazar, Edgar Martinez, and Jose Chavira in the areas of drafting, part locating, and assembly; and also Nate McCormick's contribution of his original honors project.

VI. References

[1] Astilean, A. and S. Folea, "Design and Testing in Laboratory Environment of the Embedded Microsystem ECAM," *IEEE Int. Conf. on Automation, Quality, and* Testing**,** Robotics, pp. 442-447, 2006.

[2] Bailey, Fredric N. and Frederick M. Waltz, "A Hardware Digital Controller for Undergraduate DDC Experiments," **IEEE Trans**. **on Education**, Vol. 18, No. 4, pp 195-198, 1975.

[3] Kapila, V. M.S. de Queiroz, and A. Tzes, "A Multi-disciplinary Undergraduate Real-Time Experimental Control Laboratory," **Proc**. **American Control Conference**, pp. 3980-3984, 2000.

[4] Mrad, F., N. El-Hassan, S.E.-H. Mahmoud, B. Alawieh, F. Adlouni, "Real- time control of free-standing cart-mounted inverted pendulum using LabVIEW RT," **IEEE Industry Applications Conference**, 2000, pp. 1291-1298.

[5] Ramakrishnan, V., Y. Zhuang, S.Y. Hu, J.P. Chen, C.C. Ko, B.M. Chen, and K.C. Tan, "Development of a Web-Based Control Experiment for a Coupled Tank Apparatus," **Proc**. **American Control Conference**, pp 4409-4413, 2000.

[6] Spong, M.W. "Control Education Crossing Department Boundaries," **Proc**. **American Control Conference**, pp. 992-996, 1999.

[7] Valera, A. J.L. Diez, M. Valles, P. Albertos, "Virtual and Remote Control Laboratory Development**," IEEE Control Systems Magazine**, Vol. 25, No. 1, pp. 35-39, 25.