A Freshmen Programming Course for Mechanical Engineers Using Mechatronics Applications

Joseph C. Musto, John H. Lumkes Jr., and William Carnell[1]
Mechanical Engineering Department
Milwaukee School of Engineering

Abstract

A new freshmen-level course has been developed to teach programming applications to Mechanical Engineering students at Milwaukee School of Engineering (MSOE). Rather than focusing on typical introductory programming applications (like sorting, numerical methods, etc.), the course is focused on the development of algorithms that include electromechanical hardware in the loop. While the objectives of the course are similar to those of a traditional programming course (flowcharting, algorithm development, loops, logical structures, and data arrays), the use of mechatronics applications provides two advantages over traditional programming courses:

- Students are provided with immediate, concrete, visual feedback on their algorithms and programming implementations.
- Student interest in multidisciplinary problems, and the importance of programming to Mechanical Engineering students, are implicitly developed in the course.

The approach is loosely based on a similar course developed at Northeastern University by McKnight, et. al. [1]. The course has been developed using *Matlab* as the primary programming platform. A low-cost USB interface device is used to connect mechatronic hardware to student laptop computers. Experiments including LEDs, temperature sensors, distance transducers, light sensors, solar cells, DC motors, and stepper motors, as well positioning tables and servo-controlled robots, have been developed. The course culminates in a creative design project, in which teams of students combine the various types of hardware used in the laboratory into a new application of their choosing. Based on both student and instructor feedback, the initial implementation of the course has been overwhelmingly positive.

I. Introduction

In the fall of 1999, Milwaukee School of Engineering (MSOE) introduced a Technology Package Program for all incoming students. As part of the initiative, each new student is provided with a notebook computer and a standard suite of software. In conjunction with this program, a revised Mechanical Engineering curriculum was launched, designed to exploit the availability of

1 W. Carnell is currently with the Mechanical Engineering Department at the University of Connecticut

notebook computing technology [2]. As part of this curriculum redesign, a two-course sequence in computer programming has been implemented for freshmen students studying Mechanical Engineering.

It is well-known among engineering educators that while traditional introductory programming courses are an important foundation for future study, the typical tasks assigned in introductory programming courses (sorting, "Hello World" output, etc.) fail to excite and motivate many freshmen engineering students [1]. This is especially true of many students in Mechanical Engineering, who are drawn to the discipline by a strong desire for "hands-on" applications. In an effort to motivate the use of programming techniques, and to stimulate interest among the students, a programming course designed around the use of computing in mechatronics and control applications was developed. This course is based on the approach taken by McKnight, et.al., at Northeastern University, where a "High Tech Tools and Toys" Laboratory is used to integrate control applications within an introductory programming course [1].

In this course—implemented as the second course in a two-quarter course sequence in introductory programming—laptop computers running *Matlab* and equipped with a low-cost USB interfacing device are used in a variety of mechatronics applications, from simple switching of LEDs to servo control of a robotic manipulator. The laboratory exercises reinforce concepts developed in the first programming course, including the use of *for* loops, *while* loops, and *if-elseif-else* logical constructs. However, unlike the "number crunching" algorithms developed in a typical programming course, students are provided with immediate, concrete, visual feedback on their programs; successful implementations result in a functioning electromechanical system, while errors in loop structures or programming logic result in a malfunctioning system. In addition to the benefits that such feedback has on programming skills, the mechatronics applications have the additional benefit of adding an exciting, "hands-on" component to material considered by many Mechanical Engineering students to be both confusing and dull; the impact on student *interest* in programming is enormous.

In Section II of this paper, the details of the course structure will be presented. The weekly course goals, and the laboratory exercises developed to attain these goals, will be presented. In Section III, a description of the hardware and software that comprise the laboratory workstations will be developed. Section IV of the paper will offer some general conclusions based on the first implementation of this course, as well as present some future direction for the course.

II. The Structure of the Mechatronics-Based Programming Course

The mechatronics-based programming course, entitled *Computer Applications in Engineering II*, is a required freshmen-level course for all Mechanical Engineering students at MSOE. The course is offered in the third (Spring) quarter, and is the culmination of the three-term Freshmen Sequence in Mechanical Engineering. The first two courses in the sequence can be summarized as follows:

- ***Introduction to Mechanical Engineering and Design:*** A three-credit course that combines an introduction to the profession, to the design process, and to solid modeling and graphics.
- ***Computer Applications in Engineering I:*** A three-credit introductory programming course. Flowcharting, algorithm development, loops, logical structures, and data arrays are stressed throughout the course, and programs are implemented using *Matlab*. Numerical examples from various branches of engineering are used to motivate the programming assignments. An introduction to spreadsheets is also presented.

Ideally, at the conclusion of the *Computer Applications in Engineering I* course, students should be proficient *Matlab* programmers. However, in the first three years that this course ran (as a stand-alone introductory *Matlab* programming course), some common observations surfaced:

- More than half of the freshmen Mechanical Engineering students had not taken a programming course before entering college. For students with little or no previous programming experience, a single ten-week course was not sufficient for learning a programming language.
- Specific weaknesses in the use of loops and the use of data arrays could be seen in a significant number of students.
- Despite faculty efforts to motivate students with examples from engineering, many students still found the material in the programming course to be very "dry", and many could not see (or did not believe) that programming would be an important and integral skill for both their academic and professional careers.
- Many students had little interest in flowcharting their algorithms prior to coding, or in debugging their code once written; there was little downside to incorrect algorithms or improperly coded algorithms, other than a grading penalty for producing the wrong answer. For students with little inherent interest or motivation to learn programming, they were willing to accept partial credit for partially working code!

It was with these observations in mind that the mechantronics-based *course* was developed. The main goal was to provide additional reinforcement of structured programming concepts, while stimulating student interest and motivation for learning computer programming. After reviewing the work of McKnight, et.al. [1], a mechatronics-based course seemed to address the concerns resulting from the standalone *Matlab* programming course.

A ten-week course was developed to build on the introductory programming experience obtained in the first course of the sequence. The course was designed with a one-hour lecture session and a two-hour laboratory session each week. Weekly topics involved the introduction of new mechatronic hardware, and the development of the software to control the newly-introduced hardware. The course was designed to culminate in a creative project, where students could combine the hardware and software modules developed during the course into a new configuration for an application or their choice. The remainder of this section will be devoted to a week-by-week description of a typical course offering.

- ***Week 1: Digital I/O***
  In the first week of the course, students interfaced LEDs, a Hall-effect latching switch, and a push-button switch to their laptop computers via a USB interfacing device. Students were required to perform the following programming tasks:
    - Develop a program to sequentially light the four LEDs through a user-specified number of cycles.
    - Develop a program to monitor the status (open or closed) of the two switches, and display the status to the screen.
    - Combine the basic elements of the previous two programs to develop an "interactive" LED light show, where the status of the switches is used to modify the flashing sequence of the LEDs.

  In addition to providing an introduction to the concept of interfacing, this exercise reinforces the use of *for* loops (for LED sequencing) and *if-elseif-else* logical constructs (for monitoring and decision-making involving switch status).
- ***Week 2: Analog I/O***
  In this exercise, students interface a number of analog sensors to their laptops (a temperature sensor, a pressure sensor, an infrared range sensor, a solar cell, and a photocell), as well as two small DC motors. The students are assigned three programming challenges:
    - Develop a program to continuously monitor the voltage outputs of the five analog sensors for ten seconds, store the results, and plot the data to the screen at the conclusion of the program.
    - Develop a program to allow the user to activate one of the motors, interactively change the speed, and stop the motor based on a specific keyboard input.
    - Combine the basic elements of the previous two programs, allowing the user to control the speed of the motors based on the output value of one of the analog sensors. For example, the light level impinging on the solar cell could be used to control the speed of the motor.

  This laboratory allows for an introduction to the concepts of analog-to-digital conversion (resolution, etc.), as well as the concept of sensor calibration. Programming concepts reinforced in this exercise include the use of *for* loops (for monitoring of sensor values), data arrays (for storage and plotting of sensor outputs), and *while* loops (to exit motor control programs based on a specific keyboard input).

- ***Week 3: Stepper Motor Control***
  This exercise introduces the students to a stepper motor-controlled X-Y positioning table, as shown in Figure 1. Each axis is controlled by a single stepper motor and belt drive, and end-of-travel limit switches are provided on each end of each axis. The table is interfaced to the laptop using the USB interface device. This hardware platform is used in multiple laboratory sessions; in this first introduction, only some very simple control and calibration activities are performed.

  - A program is developed to move each independent axis through 100 steps in each direction. A correlation between stepper rotation and linear travel is developed.
  - A program is developed to "home" the X-Y table at the center position along each axis. This is done by sequentially moving each axis to the end of travel (as indicated by monitoring the limit switches), and computing the number of steps required to return to center.

  Once a correlation between rotary and linear steps has been established, and a basic "homing" routine has been established, students are prepared for more complex control applications involving the X-Y table. Both *for* and *while* loops are used extensively in this application.

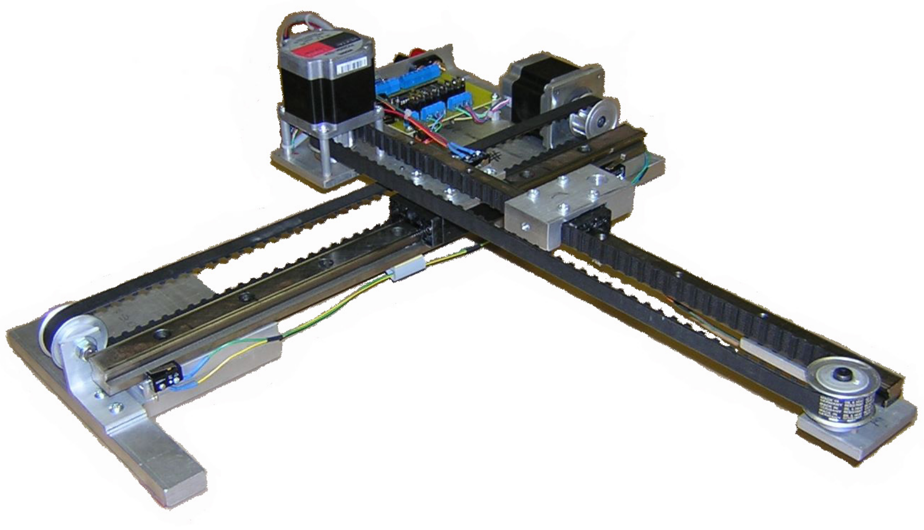

Figure 1: Stepper-Controlled X-Y Tables

- ***Week 4: Development of a Stepper-Controlled Pen Plotter***

  In this exercise, the X-Y tables introduced in Week 3 are augmented with a pen attachment. The students are required to move from single axis control (as used in Week 3) to coordinated X-Y control. The application they are assigned is to use the pen plotter to generate a drawing of their choice. An example output is shown in Figure 2.

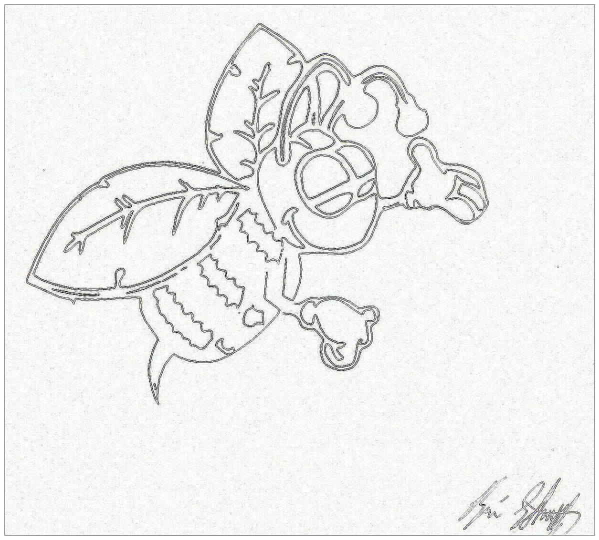

Figure 2: Pen Plotter Output

- ***Weeks 5 and 6: Development of an Infrared Scanning Device***
  In this exercise, the X-Y tables used in Weeks 3 and 4 are fitted with the infrared range sensor used in Week 2. An object is placed in the workspace of the X-Y table and by programming the X-Y table to raster scan the workspace and collecting the output of the infrared sensor, a 3-D map of the workspace can be obtained. By plotting this 3-D map, a scan of the object can be plotted to the screen. This is similar to the main exercise used in the work of McKnight, et. al. [1]. An example output is shown in Figure 3.

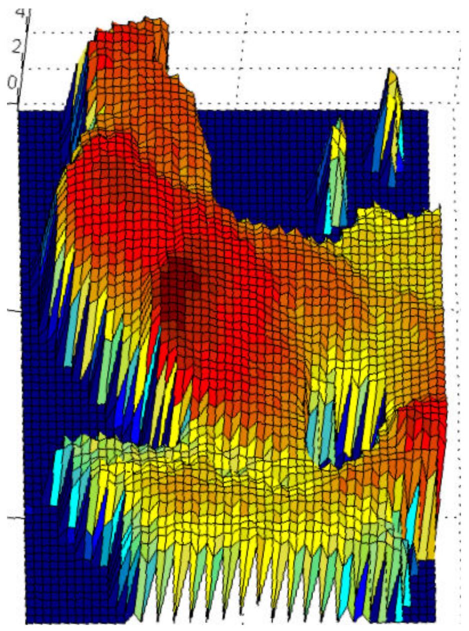

Figure 3: A 3-D Map of an Object

- ***Week 7: Servo Control of Robots***
  In this exercise, a small robotic manipulator is introduced. The small robot is an educational "toy" robot, which uses standard R/C servos as the joint actuators (as shown in Figure 4). The robots were connected to student laptops via a serial connection. Students were provided with *Matlab* functions (developed by the instructors) to perform primitive joint moves. Using the functions within their programs, students were asked to develop a Matlab program to perform a pick-and-place operation.

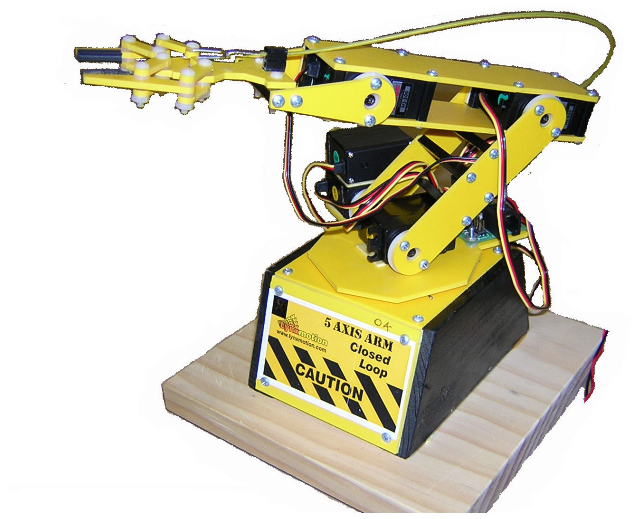

Figure 4: An R/C Servo-Controlled Robot

- ***Week 8: Straight-Line Motion Generation for a Robotic Manipulator***
  At the conclusion of the Week 7 laboratory exercise, students were familiar with the kinematics of and the methods for programming the control of the R/C servo motors. In this exercise, the students were asked to develop a program that accepted Cartesian start and end points, and compute and generate a straight-line path in between the points. This required coordinated moves among the axes, rather than single axis control.
- ***Weeks 9 and 10: Creative Projects***
  At the conclusion of the first eight weeks of the term students were proficient in developing software for a wide variety of mechatronic hardware:
  - LEDs
  - Switches
  - Various analog transducers
  - DC motors
  - Stepper motors

- R/C servos
- X-Y tables
- Robotic manipulators

In the last two weeks of the term, students were allowed to develop an application of their own using the hardware and programming techniques introduced throughout the term, and develop and demonstrate their solution. Students worked in groups of two to four students. Example projects included:

- A sorting system, where objects were detected by color and sorted into bins by the robotic manipulator (see Figure 5).
- Using flexible resistive sensors connected to arms, fingers, and torso to control two robotic arms. The human subject provided the inputs by moving their body, the robots then followed the motions.
- The development of a Matlab algorithm to accept CNC code as the input to the X-Y table and pen.
- The adaptation of a numeric keypad used to control the motion of the robots.
- The implementation of a micro-PLC used as a stand-alone controller for an X-Y table and pen.

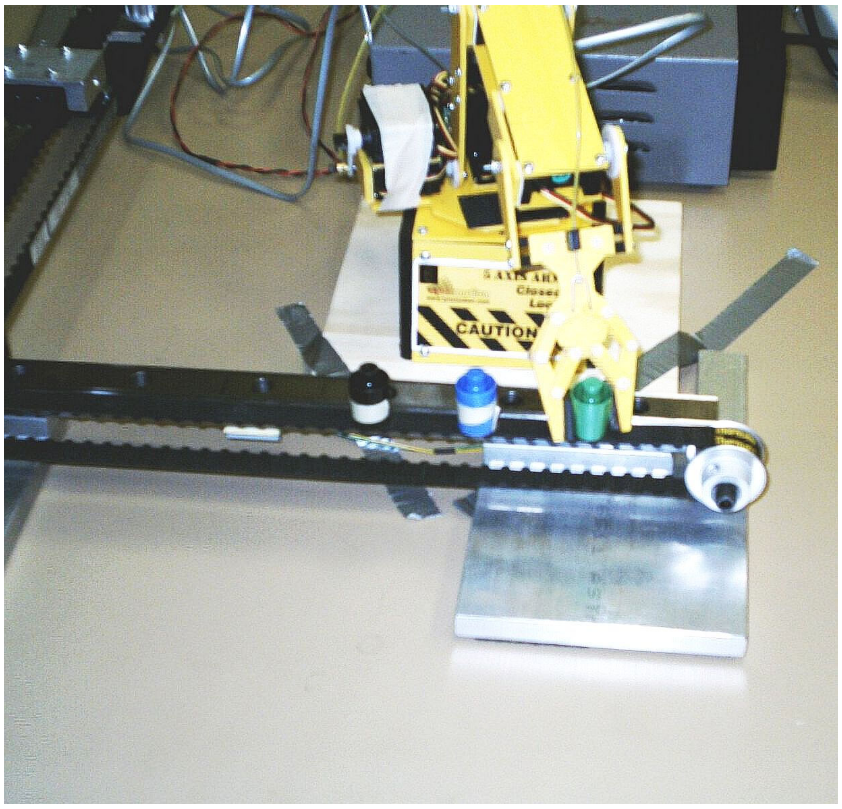

Figure 5: Computer-Controlled Sorting System

III. Laboratory Development

In the selection of hardware for the laboratory, it was first determined that Matlab would be the unifying software platform used in the course; therefore, interfacing hardware was selected with this in mind. The interface hardware chosen was the LabJack U12, a USB device available from LabJack Corporation (see Figure 6). The availability of *Matlab* drivers for this device, as well as the relatively low price ($119 retail), make it well-suited for this laboratory [3].

The robotic manipulators chosen for this laboratory was the Lynx 5, available in kit form from Lynxmotion, Inc.. While no standard Matlab drivers were available, the use of standard R/C servos made it possible to develop Matlab functions to drive the manipulator via a serial interface. The low cost of this robot ($247.97 retail) made it well-suited for use in a freshman course, where experimentation with the equipment was encouraged [4].

The X-Y tables used in the laboratory, as well as the stepper driver circuits, were designed and built in-house at MSOE.

IV. Conclusions

The mechatronics-based *Computer Applications in Engineering II* course was taught for the first time in the Spring of 2003. The student and faculty feedback from the initial offering of this course was overwhelmingly positive. From the student standpoint:

- The students felt that the immediate visual feedback they received as to the "correctness" of their algorithms truly reinforced the basic programming concepts they were exposed to in the first programming course. One student, upon watching his X-Y table successfully operating under computer control, was heard to say: "*Now* I understand what a *for* loop does!". It is this type of feedback that many students need before they can grasp the intricacies of computer programming.
- The hardware-based laboratory exercises stimulated student interest in both programming and mechatronics applications; many students inquired about purchasing their own LabJack hardware for their personal use at the end of the term.

The main advantage of this course from the faculty standpoint was that students were inspired to test and debug their own code, and were extremely motivated to get their system to function as required. Unlike the previous programming course, where students were willing to accept partial credit for partially working code, students were excited to see their mechatronic systems functioning as designed.

The main challenge in the implementation of this course was in keeping all of the hardware functioning. Approximately 150 students took the class in the Spring of 2003; the eleven workstations developed for the course were in nearly constant use throughout the term. The students were encouraged to set up their own hardware and to experiment; as a result, hardware often needed to be repaired or replaced. Surprisingly, however, no laptop computers were damaged during the term.

With the new Freshmen Sequence in place, mechanical engineering freshmen at MSOE are completing their freshmen year with a firm understanding of both programming concepts and an ability to interface with and control mechatronic systems. Integration of these skills into the upper-division courses is the next curricular challenge.

Acknowledgements: The laboratory described in the paper was supported through a generous grant from the *John Deere Foundation*. As part of the team that taught the original version of the *Computer Applications in Engineering II* course during the Spring 2003 term, the contributions of Dr. John Pakkala, Dr. William Farrow, and Mr. Richard Phillips of the MSOE Mechanical Engineering Department are gratefully acknowledged.

Bibliography

1. McKnight, S.W., W. Cole, G. Tadmor, E.C. Everbach, and M. Ruane, "Teaching computing to engineering freshmen through a 'High Tech Tools and Toys' Laboratory", *Proceedings of the 2001 American Society of Engineering Education Annual Conference & Exposition*, Albuquerque, NM, June 2001.
2. Musto, J.C. and W.E.Howard, "Integration of laptop computers into a freshman mechanical engineering curriculum", *Proceedings of the 2001 American Society of Engineering Education Annual Conference & Exposition*, Albuquerque, NM, June 2001.
3. URL: http://www.labjack.com, December 2003.
4. URL: http://www.lynxmotion.com, December 2003.

JOSEPH C. MUSTO
Dr. Musto is an Associate Professor and Mechanical Engineering Program Director at Milwaukee School of Engineering. He holds a B.S. degree from Clarkson University (Potsdam, NY), and an M.Eng. and Ph.D. from Rensselaer Polytechnic Institute (Troy, NY), all in Mechanical Engineering. His industrial experience includes engineering positions with Eastman Kodak Company (Rochester, NY) and Brady Corporation (Milwaukee, WI). He is a registered Professional Engineer in the state of Wisconsin.

JOHN H. LUMKES, JR.
Dr. Lumkes is an Associate Professor in the Mechanical Engineering Department at Milwaukee School of Engineering. He holds a B.S. degree from Calvin College (Grand Rapids, MI), an M.S.E. from the University of Michigan (Ann Arbor, MI), and a Ph.D. from the University of Wisconsin—Madison (Madison, WI). His recent activities include advising SAE Aero and Formula Car design teams, teaching classes and seminars related to vehicle, aero, and control system design, and authoring a textbook in the field of control system design. He is a registered Professional Engineer in the State of Wisconsin.

WILLIAM CARNELL
Mr. Carnell is a doctoral student in the Mechanical Engineering Department at the University of Connecticut. He holds a B.S. degree from Milwaukee School of Engineering and an M.S. degree from Purdue University. He served as a lecturer in the Mechanical Engineering Department at Milwaukee School of Engineering during the 2002-2003 academic year.