Paper ID #19397

# Incorporating the Raspberry Pi into laboratory experiments in an introductory MATLAB course

**Dr. Naji S Husseini, Biomedical Engineering at NCSU and UNC-CH**

Naji Husseini is a lecturer in the Joint Department of Biomedical Engineering at the University of North Carolina at Chapel Hill and North Carolina State University. He received his B.S. and M.Eng. in Engineering Physics from Cornell University and his M.S. in Electrical Engineering and Ph.D. in Applied Physics from the University of Michigan, Ann Arbor. He teaches classes in materials science, biomaterials, MATLAB programming, and biomechanics for undergraduate and graduate students at both UNC and NCSU. His primary interests are in engineering education, materials physics, and x-ray imaging.

**Mr. Ian Kaszubski, North Carolina State University**

# Incorporating the Raspberry Pi into laboratory experiments in an introductory MATLAB course


## Abstract

Many introductory computer-programming classes in engineering fields stress coding skills, but they often do not incorporate data acquisition or output until later classes. Interfacing computers with laboratory equipment is an excellent way to engage and excite students who may not otherwise see the importance of programming. Additionally, they collect and process data directly instead of receiving artificially or simulated data. Our laboratory curriculum integrates the Raspberry Pi, an inexpensive and versatile single-board computer, with a standard computer to provide dynamic and engaging biomedical-engineering-related programming activities in an introductory MATLAB course, all without complicated MATLAB commands or routines. The total cost for each station is only about $75 or less. Concurrently, more complicated engineering concepts are introduced at this early level to pique interest in biomedical engineering and improve learning in later classes when these concepts are described in more detail. Students report enjoying programming more and seeing its quintessential role in engineering experiments while being better prepared for later computer-based courses and job opportunities.


## Introduction

Biomedical engineering (BME) blends several engineering fields towards biological applications. BME projects from x-ray imaging to gait analysis to electromyography to pacemaker design all require both data collection and analysis, often done with either specialized software packages or homegrown programs. In our department's sophomore-level introductory programming class, we teach basic programming via MATLAB (The MathWorks, Inc.) with specific applications for later classes such as signal analysis, instrumental interfacing, and multidimensional image reconstruction. In the past, as with many programming classes, our weekly two-hour lab sessions involved processing simulated data from textbooks or experiments given to but crucially not collected by students. While students were adequately prepared for their later classes with their MATLAB skills in these artificial situations, they were unprepared to acquire or process real-world data with devices. Moreover, feedback received from students on labs was tepid and often mentioned labs were little different than in-class homework assignments from a generic engineering textbook with the "bio" prefix added.

Recently, we incorporated a series of six laboratory activities involving the Raspberry Pi (RPi, Raspberry Pi Foundation), a small single-board computer. While unable to run MATLAB alone, the RPi can easily interface with MATLAB for data input/output (I/O) using additional basic electronics components, all at less than about $75 per station. The two-hour partner labs and some additional outside coding reinforced and extended the fundamental MATLAB concepts from lecture via data collection and external-device triggering. Unlike some of the elegant single-board computer experiments available online or previously reported [1-4], ours focused mostly on teaching MATLAB rather than involving advanced circuitry or programming in the Unix-based native operating system. This project-based learning approach has already had success in other engineering disciplines' programming classes [5-7]. The ultimate goal was to create an engaging and enjoyable learning environment that stimulated students' interest while

simultaneously learning to collect and analyze their own data and developing relevant BME concepts critical for modern biomedical engineers [8].

The laboratory curriculum has been in place since the fall of 2015 and taught now for three semesters (one being a ten-person summer session), naturally with revisions after each term. Student feedback has been very positive, as demonstrated in course evaluations and questionnaires. Last year's students have said that the RPi-based labs prepared them for their later instrumentation classes and even summer internships working with RPis.

Each week's lab had a particular focus that tied into the BME curriculum while being simple enough for sophomores to understand. The concepts tested were familiar from daily life, such as bouncing balls and musical-pitch recognition, so students could use their intuition when writing their code and interpreting their results. This paper presents the basics about the RPi and its versatility, then describes the setup and objectives for each experiment. While designed for a BME curriculum taught in MATLAB, many labs have relevance to other engineering fields and can be easily adapted to other programming languages. As important, though, was that using the RPi to teach data acquisition and output while simultaneously engaging students makes programming interesting and relevant for engineers.

**Raspberry Pi**

The RPi is a single-board computer originally developed by the Raspberry Pi Foundation in 2011 to promote the teaching of computer science in schools with an inexpensive yet versatile device. Though all work discussed in this paper was done with a Raspberry Pi 2+, the latest iteration, the Raspberry Pi 3, contains Wi-Fi, Bluetooth, USB ports, Ethernet, camera board, and HDMI connections. It features a bank of 19 general-purpose input/output (GPIO) pins suitable for controlling anything from simple circuits to complex serial interface buses (five have predefined functions that can be overridden). Control of the RPi and its peripherals is through its native Linux-based operating system either from a conventional desktop environment or via a headless command-line only configuration. The Raspberry Pi Zero is a cheaper and more stripped down iteration that theoretically could handle most of the lab, though we have not tested it yet.

While not powerful enough to run MATLAB itself (though it can run Simulink, another programming language developed by MathWorks), the MATLAB support package allows real-time interface between a host computer running MATLAB and the RPi to control hardware such as the GPIO pins, camera board, or serial buses. The connection can be either wireless or via Ethernet. Built-in function calls in the support package control the GPIO pins and camera board, allowing real-time I/O in a single MATLAB session. When no built-in functions exist in the support packages, such as with USB microphones, then MATLAB can directly access the Linux shell to run UNIX commands for recording and transferring audio files. This level of control is one reason for choosing the RPi over other similar devices.

**Laboratory Assignments**

Weekly labs were two hours long with sixteen students working in pairs. Since this was the initial programming experience for most students, the first labs acquainted students with the

programming environment by working on early homework assignments with ample supervision from the teaching assistants (TAs). The RPi-based lab curriculum began after students were comfortable with basic concepts like variables, functions, arrays, conditionals, and loops.

*Components*

Each station contains the components listed in Table 1. Not all are required in each lab period, and none are consumables unless damaged in use. The total cost for these parts is approximately $75, excluding the cost of a host computer.

**Table 1**: Minimum number of components required for RPi lab sequence.

| | |
|---|---|
| 1 Raspberry Pi | 1 buzzer |
| 1 Ethernet cable | Resistors (as necessary) |
| 1 breadboard | 20 male-female jumper cables |
| 2 red LEDs | 10 male-male jumper cables |
| 2 yellow LEDs | 1 USB microphone |
| 1 green LED | 1 Raspberry Pi camera board (or USB/laptop camera) |
| 16 LEDs (any color) | |

*Lab 1: Introduction to Raspberry Pi*

Objective: Learn how to wire light-emitting diodes (LEDs) and buzzers to GPIO pins. Practice conditionals and loops.

Summary: Students wire a red, yellow, and green LED to the RPi's GPIO pins, along with a buzzer. The lab consists of four parts: 1) test light and buzzer functionality; 2) create a traffic light (Fig. 1); 3) write a Morse code translator that reads a text file and illuminates the red LED and triggers the buzzer appropriately; and 4) make a trivia game that prompts a user for inputs to simple questions with red/green LED outputs for correct/incorrect answers.

Assessment: Nothing is submitted for this lab. The students work through as many parts as they can in the two-hour session in a low-stress environment. Most students can finish the entire assignment and enjoy testing the TA with their trivia game.

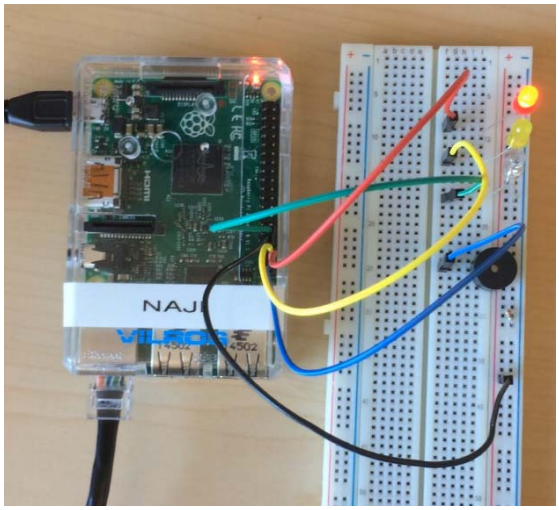

**Fig. 1**: RPi wired for Lab 1 with the red LED illuminated (STOP!).

*Lab 2: 2D puzzles*

Objective: Learn about 2D arrays and associated concepts, such as indexing and logical operators. Additional practice with more complicated circuit design.

Summary: Students wire a 4x4 array of LEDs to the GPIO pins. The lab consists of two games. The first is a classic memory game, where the user sees a 2D array of illuminated lights for a few seconds, then must recreate it by entering row and column numbers in the command line. The second game is a puzzle, where the user is given a random array of lights and must extinguish all of them. Each selected light, however, changes the state of the nearest neighbors (similar to the classic Game of Life). For the memory game, we provide the students with a rough outline of the code beforehand.

Assessment: Nothing is submitted for this lab. All students finish the first assignment; most finish the second. They again enjoy watching their TA attempt to solve the two puzzles.

*Lab 3: Scrolling marquee*

Objective: Continues array concepts from previous lab.

Summary: Same setup as Lab 2 (Fig. 2). Letters and numbers are defined as 4x4 logical arrays in a separate lookup-table function. The program reads a separate text file, then scrolls the text across the 4x4 LED array.

Assessment: Nothing is submitted for this lab.

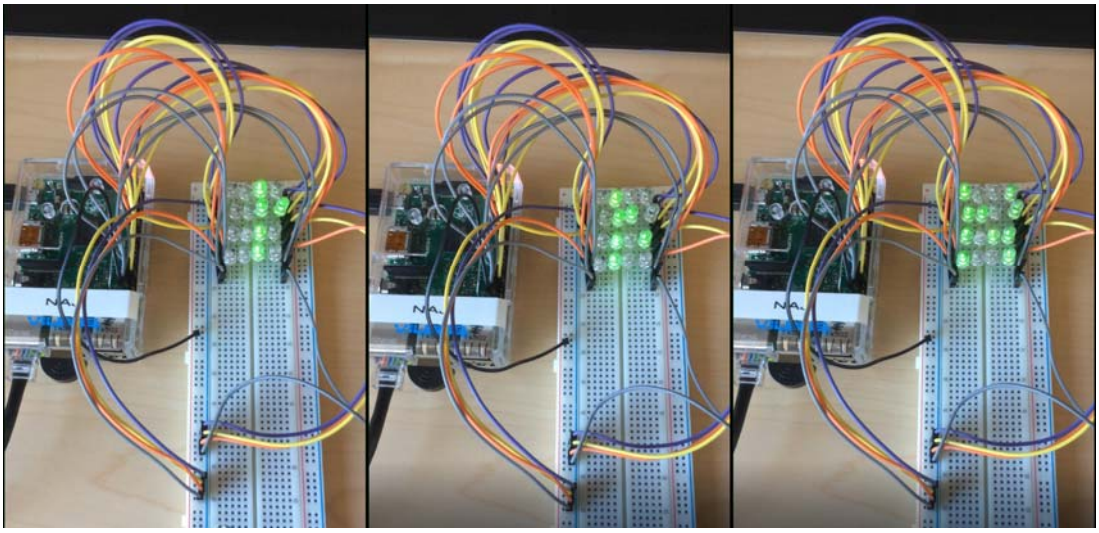

**Fig. 2**: RPi wired for Lab 3 showing the letter "N" moving across the LEDs. This figure does not demonstrate a proper wiring convention or elegant wiring.

*Lab 4: Chromatic tuner*

Objective: Apply basic signal processing from lecture. As students are sophomores, they have not formally learned about Fourier transforms or filtering. This lab helps students understand these concepts in an audible way using a topic about which they are familiar – musical pitches.

Summary: Students wire two red, two yellow, and one green LED, and plug in a USB microphone directly into the RPi (Fig. 3) to mimic a chromatic tuner. They are given a custom function for a fast Fourier transform (FFT) that takes a signal and returns the frequency and power spectra. Their code reads an audio signal from the RPi, runs a simple filter to clean up the data, uses the provided FFT function to find the dominant frequency, then determines to what musical note it is closest. The output is displayed to both the MATLAB command window and visually using the LEDs to indicate proximity to the proper pitch. This lab introduces students to signal processing, which they will use for later BME concepts like electromyography or hearing-aid function.

Assessment: During the lab session, students write code to take in a signal and determine the dominant frequency. As homework over the next week, they write code to calculate a frequency's proximity to a musical note in addition to the code to output results to the RPi. The TAs test the functionality at the beginning of the following lab session using a frequency generator and get the students to see whether they can whistle an A.

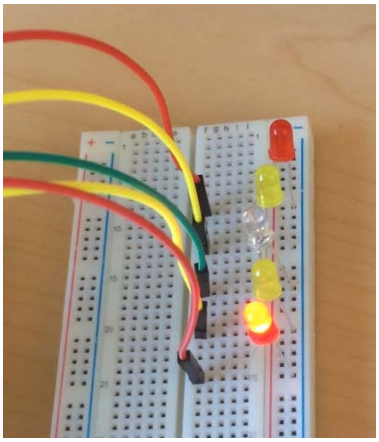

**Fig. 3**: RPi wired for Lab 4 when whistling a frequency of 1846 Hz. This is closest to an A#/Bb which has a nominal frequency of 1865 Hz. The whistled pitch is 20 cents flat and triggers the flat red LED.

*Lab 5: Motion detector*

Objective: Use basic image processing to detect motion using the RPi's camera. This practices basic image acquisition commands and develops the concept of segmentation for grayscale images from class.

Summary: This lab requires the RPi camera board (or a USB webcam plugged directly into a computer) and a buzzer wired to a GPIO pin. The program converts the current color image to grayscale, then subtracts it from a grayscale version of the previous image. Based on a user-defined threshold, the subtracted image is segmented. If a certain percent of the pixels are white, then the buzzer is triggered and "ALARM" is written on the figure (Fig. 4).

Assessment: Nothing is submitted for this lab. If a group finishes the assignment (and many do), then they are encouraged to create a color-sensitive motion sensor that only detects their TA.


**Fig. 4**: MATLAB figure for motion detector. The original image converted to grayscale, subtracted from the previous image, then thresholded according to the sensitivity level. If a defined fraction of pixels are triggered, then the alarm is activated. Note that the author moved prior to the image shown. (Face covered with white box for review process only)

*Lab 6: Coefficient of restitution*

Objective: Track the centroid of a bouncing ball using the RPi's camera and determine its centroid using image processing for colors instead of grayscale. Determine the ball's kinematic parameters using calculus, then calculate the coefficient of restitution (CoR) of the ball.

Summary: Continuing the previous lab's image processing concepts with centroid calculation, this lab integrates numerical calculus to find the velocity and acceleration of a bouncing ball. Motion tracking is used in later labs and courses for joint-motion analysis and particle tracking.

Assessment: During the lab session, students are expected to finish tracking the centroid position of the ball. Over the next week, they calculate the kinematic parameters and CoR as homework then submit their code for grading.

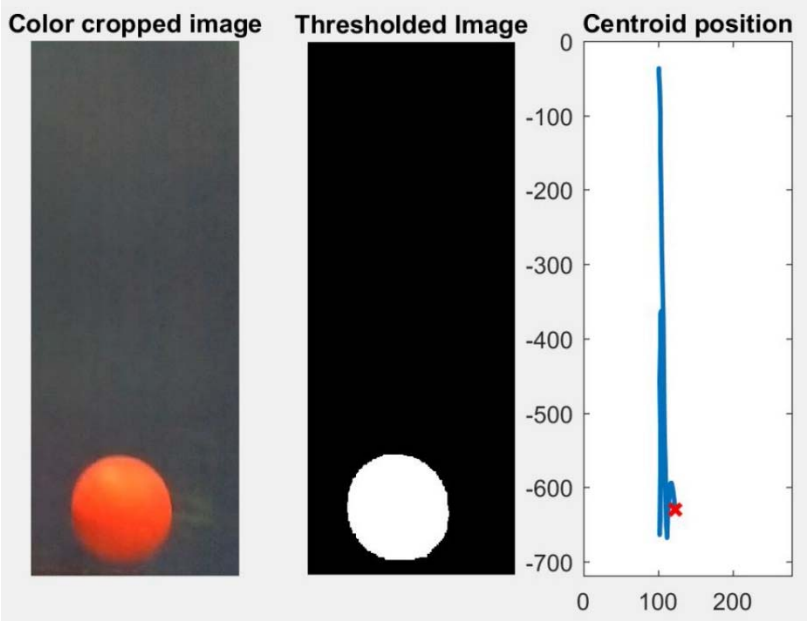


**Fig. 5:** MATLAB figure for the bouncing-ball lab. The ball is set against a contrasting background and then thresholded. The centroid position is tracked as the ball bounces to rest.

**Evaluation**

The success of the new RPi-based laboratory sequence was assessed in two ways: numerical course-evaluation scores at the end of the semester and questions asking specifically about the new labs. Anecdotal evidence from questionnaires given to students in their later instrumentation classes suggests that students' familiarity with computer interfacing has improved, and they are better prepared and quicker to learn new material taught both in MATLAB and LabView (National Instruments Corporation); however, currently no quantitative data exist at this point in the semester.

Student evaluation data from available semesters show a major change after the RPis were introduced (Fig. 6). In 2015, the first year the new lab sequence was implemented, students appreciated the instructor's effort (black data) but found the labs were too difficult (magenta data) due to inadequate guidance in the assignment page, insufficient pre-lab introductions by the TAs, and some minor hardware and software complications. By 2016, after improvements to the labs based on student feedback, the scores in all non-instructor metrics increased dramatically, though the labs were still considered challenging.

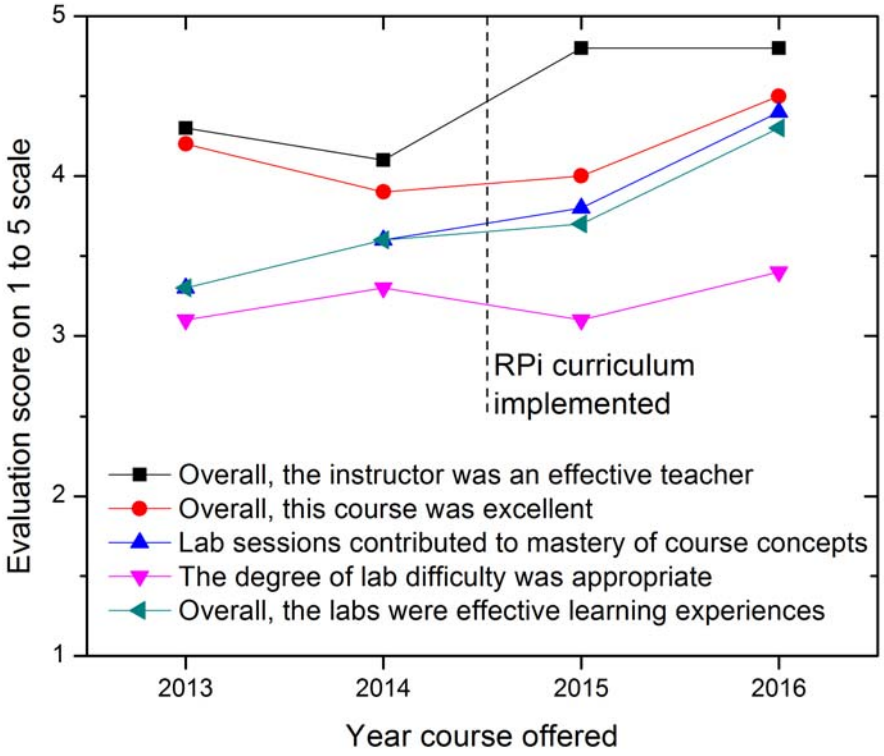


**Fig. 6:** Student course evaluations in the fall semesters over the past four years. Exact question wording is shown in the legend. RPi curriculum was implemented in the fall of 2015, marked by the dotted line, and refined for the fall of 2016.

Student feedback to questions about the labs were generally positive and enthusiastic, even in 2015 with some technical troubles. When asked to comment on the weaknesses of the labs, a typical comment was "*I would like to use the RPis, as I think they are a great real world application of the class, but they need work.*" After the improvements in 2016, the comments improved drastically. Representative comments from students who had just completed the class included the following:

- "*Raspberry Pis were awesome to use!!!! I bought one for myself!*"
- "*[RPis] were a huge step for me, because it showed me the true power of programming and how there really are not limits to how much you can do.*"
- "*[I want to] extend what [I] learned from the Raspberry Pi technology … to fit [my] own research and personal project needs.*"
- "*While some of the labs were very challenging, they were really interesting and really forced you to think out of the box.*"

When students from 2015 were asked to comment on how the RPi has so far affected their other classes, they were also positive:

- "*…very helpful in many of my other classes, as I know how to use MATLAB for engineering purposes.*"
- "*Dr. Husseini taught a course on how to apply computer programming to technology, not just a course on how to write MATLAB code.*"
- "*…I learned how to use MATLAB especially for my future courses. It has made [instrumentation] much easier than I was told it would be."*

Two students found summer internships after their sophomore year specifically for their RPi-based MATLAB experience, suggesting that companies are actively seeking students with this type of coursework.

The author's current upper-level biomechanics course has students who learned program prior to the RPi introduction. They learned about image processing with simple examples but without interactivity that current MATLAB students get from the bouncing-ball lab (Fig. 5). One biomechanics lab involves gait analysis using a video camera to track joint motion, then segmentation in MATLAB to determine each joint's position as the subject walks (Fig. 7a). They can derive all necessary kinematic parameters, solve the kinetics problem, and extract forces and moments at each joint while walking (Fig. 7b). In lieu of a force plate, the ground reaction forces come from normal-gait force data [9]. Next year, when students from the redesigned MATLAB laboratory sequence perform the same experiment, we will have a more quantitative measure of how their programming skills have improved.

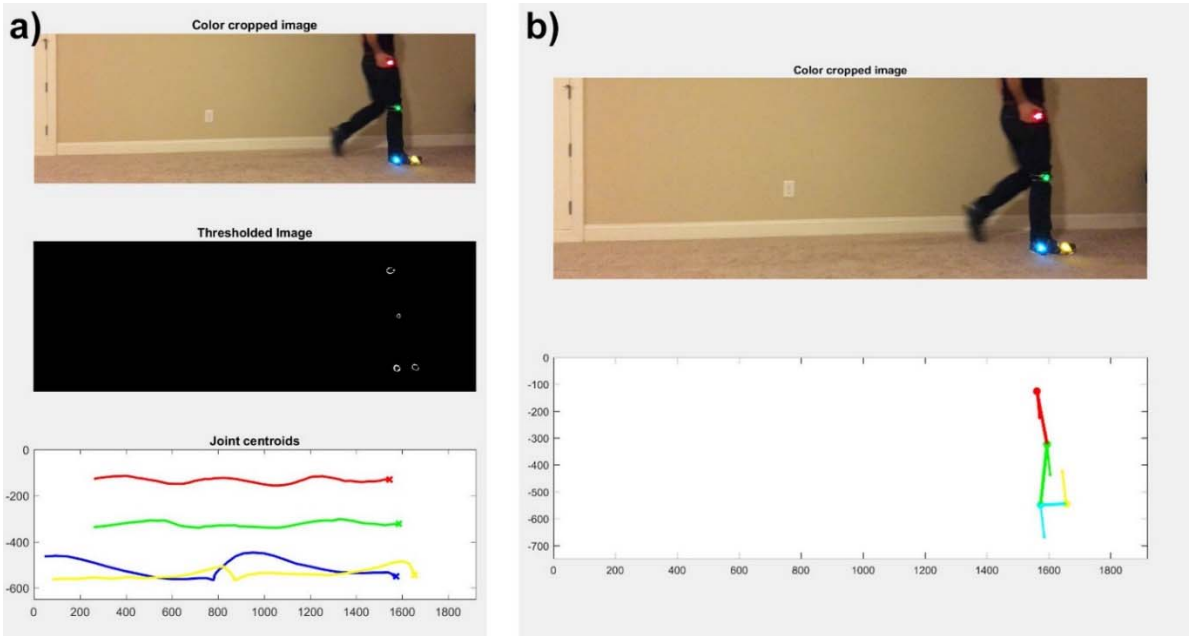


**Fig. 7**: Gait analysis. (a) Joint positions during video. (b) Schematic representation of gait with qualitative force vectors at each joint.

## Discussion

The relatively stagnant evaluation scores during the lab changeover between 2014 and 2015 were due mostly to technical problems and underprepared TAs. Unanticipated version conflicts between the MATLAB version on the students' laptops and the RPi firmware occurred and were difficult to debug due to unhelpful MATLAB error messages. Also, some of the labs were too complicated to finish in the two-hour lab session without better guidance from the instructors at the beginning of the lab session. These shortcomings were addressed in 2016 with clearer directions and better prelab introductions. All evaluation scores rose as a result. Some students still found some labs challenging, and we will continue to address this concern in future years. The problem remains that while most TAs are familiar with MATLAB and basic electronics, not all can troubleshoot problems with an RPi, necessitating proper TA training beforehand. For example, if the micro-SD card, which contains the operating system and firmware, is not in the RPi, MATLAB will give a generic connection error. The instructor should know how the RPi's motherboard status ACT and PWR LEDs flash during normal operations or (more simply) to check the micro-SD card slot.

In our labs we run MATLAB on a computer and connect to the RPi via Ethernet; however, since the RPi can run Simulink natively or can connect to MATLAB via Wi-Fi, it could be used for off-site data collection or "Internet of Things" (IoT) tasks. The website *https://thingspeak.com* has many projects and acts as a streaming portal for many IoT applications, and prior work has shown promising results for using an RPi or Arduino for IoT applications [10]. Lab 5's motion detector, for example, could be mounted outside a door to detect visitors with either a Wi-Fi connection to a computer or running a Simulink script. Moreover, other single-board computers or microcontrollers, such as the Arduino (Arduino), BeagleBone (Texas Instruments), ODroid (Hardkernal Co., Ltd.), and NUC (Intel Co.), could also be used, as could another programming languages, such as Python with the NumPy package.

The general philosophy when designing labs is to look to the upper-level courses that process real-world data, then reduce the complexity to cater to students with just basic math, chemistry, and physics knowledge, all while using basic MATLAB commands that can be learned during a single-semester course. The goal is still primarily to teach programming but via BME experiments; as a result, complicated equations (e.g., Fourier transforms, linear acceleration from rotational kinematics) are given in the assignments. The program should have an interesting output, whether on the computer monitor or via the RPi's accessories. Incorporating experiments also may affect the assessed Accreditation Board for Engineering and Technology (ABET) student outcomes.

## Conclusions

Combining traditional lecture-based computer-science concepts with hands-on laboratory activities fully integrates theory with demonstrable effects, a disconnect common in many introductory engineering programming courses. The RPi is an inexpensive and versatile device for teaching data acquisition and device output without requiring substantial prior electronics experience. RPi-based labs using fun activities and relevant BME examples truly engage students and increase overall class performance, while simultaneously introducing complicated

concepts studied in detail in later courses. As our experiences and data demonstrate, any engineering program should strongly consider following a similar structure in their early programming classes.

**Acknowledgments**

The authors would like to thank Professor Warren Jasper for his help solving technical difficulties with the RPi and fruitful discussions for lab development. They would also like to thank Professor Gregory Sawicki for his help in the early stages of the course development, and TAs Zachary Congress, Patrick Erb, Claire Hall, Laksh Punith, and Henry Shin for their input in lab development and troubleshooting.

**References**

1. The MathWorks, Inc. MakerZone. Retrieved January 13, 2017, from http://makerzone.mathworks.com/.

2. Hopkins, M. A., Kibbe, A. M., & Asee. (2014, June). *Open-source hardware in controls education.* Paper presented at the 2014 ASEE Annual Conference & Exposition, Indianapolis, IN.

3. Zachariadou, K., Yiasemides, K., & Trougkakos, N. (2012). A low-cost computer-controlled Arduino-based educational laboratory system for teaching the fundamentals of photovoltaic cells. *European Journal of Physics, 33*(6), 1599-1610.

4. Recktenwald, G. W., & Hall, D. E. (2011, June). *Using Arduino as a platform for programming, design and measurement in a freshman engineering course* Paper presented at 2011 ASEE Annual Conference & Exposition, Vancouver, BC.

5. Pritchard, J. W., & Mina, M. (2012, June). *Hands-on, Discovery, Critical Thinking, and Freshman Engineering* Paper presented at 2012 ASEE Annual Conference & Exposition, San Antonio, Texas.

6. Steinmeyer, J. D. (2015, June). *Project-based Learning with Single-Board Computers* Paper presented at 2015 ASEE Annual Conference & Exposition, Seattle, Washington.

7. Wong, P., & Pejcinovic, B. (2015, June). *Teaching MATLAB and C Programming in First-year Electrical Engineering Courses Using a Data Acquisition Device* Paper presented at 2015 ASEE Annual Conference & Exposition, Seattle, Washington.

8. Mullett, G. J. (2015, June). *The creation of a Biomedical Engineering Technology Program for the 2020's* Paper presented at 2015 ASEE Annual Conference & Exposition, Seattle, Washington.

9. Winter, D. A. (2009). *Biomechanics and Motor Control of Human Movement* (4th ed.). Hoboken, NJ: Wiley.

10. Mullett, G. J. (2016, June). *Teaching the Internet of Things (IoT) Using Universally Available Raspberry Pi and Arduino Platforms* Paper presented at 2016 ASEE Annual Conference & Exposition, New Orleans, Louisiana.