ASEE'S VIRTUAL CONFERENCE
At Home with Engineering Education
JUNE 22 - 26, 2020 #ASEEVC
Paper ID #31009

Design Course in a Mechanical Engineering Curriculum

Dr. Jamie Szwalek, University of Illinois at Chicago

Dr. Jamie Szwalek is currently a Clinical Assistant Professor at University of Illinois at Chicago in Mechanical and Industrial Engineering.

Dr. Yeow Siow, The University of Illinois at Chicago

Dr. Yeow Siow has over fifteen years of combined experience as an engineering educator and practitioner. He received his B.S., M.S., and Ph.D. from Michigan Technological University where he began his teaching career. He then joined Navistar's thermal-fluids system group as a senior engineer, and later brought his real-world expertise back into the classroom at Purdue University Calumet. He is currently a Clinical Associate Professor at the University of Illinois at Chicago where he enjoys success in teaching and education research.

Miss Jaqueline Oxmara Rojas Robles, University of Illinois at Chicago

PhD student in Mechanical Engineering at University of Illinois at Chicago

Pitfalls and Lessons Learned in Integrating Arduino into Introductory Design Course in a Mechanical Engineering Curriculum

Abstract

Responding to the shifting world economies, significant changes have been made to Introduction to Engineering Design, a first-year course in the mechanical engineering curriculum at University of Illinois at Chicago since Fall 2018. In particular, "electrification" of student projects and learning outcomes has been front and center in the department's latest strategic planning. Leveraging recent literature and faculty expertise, an increasingly deeper integration of Arduino has since taken place, while attempting to maintain the core of team-based mechanical design using morphological methods. The focus of this paper is to identify the challenges and pitfalls in such an endeavor by reflecting on the process of change over three semesters of implementation, including the deployment of both top-down and bottom-up approaches. In particular, this paper will examine course content development, teaching staff management and training, student learning assessments, inventory and experiential enhancement, and communication improvement. Data from student self-assessment, instructor evaluations, final project demonstration results, and teaching assistant and instructor observations are also reported. One of the roadblocks experienced included hardware choices (e.g., unexpected interference among ultrasonic sensors) and the administrative team needed to quickly identify and implement alternatives without sacrificing student outcomes. Another major issue was student preparedness in coding prior to enrolling in this course and it became apparent that specially crafted exercises were essential for each student to build a successful electromechanical device. Additionally, the quality and quantity of the support staff, in particular undergraduate teaching assistants, were found to be more crucial than anticipated and a robust recruitment process became necessary. The high-stake design project in ME 250 changes each semester to prevent students from obtaining a set of solutions or project reports from prior terms, so teaching assistant training is continuous. The specifics of each problem encountered will be described in the paper, along with lesson learned on how best to handle each situation and create a structure where continuous improvement can be made sustainable.

Keywords: first year design, mechanical engineering, Arduino, project-based engineering

1 Introduction

ME 250 is a first-year design course offered at University of Illinois at Chicago, a public university in an urban setting, and it is taught as a design studio which is both project- and team-based. It is a required class for Mechanical, Civil, and Industrial engineering students and it is open to other engineering majors as an elective. There are typically forty students per class section, and in recent years four to six sections of the class are offered in each term, taught by two to four faculty members. The total enrollment has been growing and is close to 400 for the current academic year. Despite being offered as a first-year course, ME 250's enrollment typically includes students from all years. Students are placed in teams of four to complete two or three projects, including the final project which is a high-stake design-build-test whose theme varies from term to term. This paper describes three semesters of the course: Term 1 is Fall 2018, Term 2 is Spring 2019, and Term 3 is Fall 2019. The course currently underway is Spring 2020 and referenced as Term 4.

Students are tasked with a design-build-test of a mechanical device for the end-of-term “competition” to showcase their high-stake design project. This class employs a team of 20 undergraduate teaching assistants (TAs) to help facilitate various aspects of the course and to staff the laboratory around the clock during business hours. Two to three graduate TAs are also assigned to the course, to help manage the workflow, staffing, and training of undergraduate TAs, and to assist the instructors in fine-tuning student projects prior to implementation.

One of the challenges of teaching first-year design is that students may not have a background in coding, computer aided design (CAD), machine design, and technical writing. ME 250 only has a pre-requisite of a basic English composition course. Many students, including direct admits and transfers, enter the mechanical engineering program with little to no CAD and coding skills. For most students taking ME 250, they are immediately faced with the challenge of learning new skills in an unfamiliar team-based environment.

The Mechanical Engineering department at University of Illinois at Chicago has recently undergone an assessment describing the need for a curriculum change based on key desired skills from industry: agility, mechatronics, coding, data science, entrepreneurship, effective communication, and creativity [1]. Prior to Term 1, ME 250 required limited electromechanical elements for student design projects. The course now serves as a platform in which to incorporate mechatronics to the curriculum, with the goal of introducing undergraduate students to robotics and coding early in their education, and with learning outcomes that include retention and a meaningful appreciation of robotics principles. These new objectives are in addition to the existing course learning goals of effective communication via report writing and oral presentations, effective teamwork and project management, and demonstration of creativity (in aesthetics and functionality) using the morphological design process [16].

Using Arduino in an undergraduate mechanical engineering design course is not new [2-9]. Many higher institutions have chosen to incorporate it into a sophomore-level design classes where most students have already gained design, CAD, or coding skills prior to registration. In these courses, students typically enjoy a single hands-on project while learning about manufacturing. Students have generally responded positively to adding Arduino into the curriculum, with high levels of self-efficacy in basic mechatronics projects [2]. When introduced early in the curriculum, students gain confidence in using the platform and are more likely to use it later [7]. The hands-on project work has a strong positive impact on the student engagement and learning [9]. Many redesigned courses have created assignments and activities that are related to the course projects [4].

With intentionality, the high-stake design project in ME 250 changes each semester to prevent students from obtaining a set of solutions or project reports from prior terms. This paper focuses on how this strategy can be practically implemented and how it requires continued growth of the instruction team to constantly and creatively devise solutions to unexpected issues that may arise during the course of the projects.

Each student in the class is provided an Elegoo EL-KIT-003 UNO Project Super Starter Kit † based on the Arduino Uno. The Elegoo‡ kit was selected based on cost (Amazon, $35) and compatibility with the Arduino software [10]. The kits are fully paid for by laboratory fees, and are therefore owned by the students. The kits are intended to be reused in other courses as the student progresses through the curriculum. Students use the kits to complete coding and circuit exercises to develop the skills needed for the project.

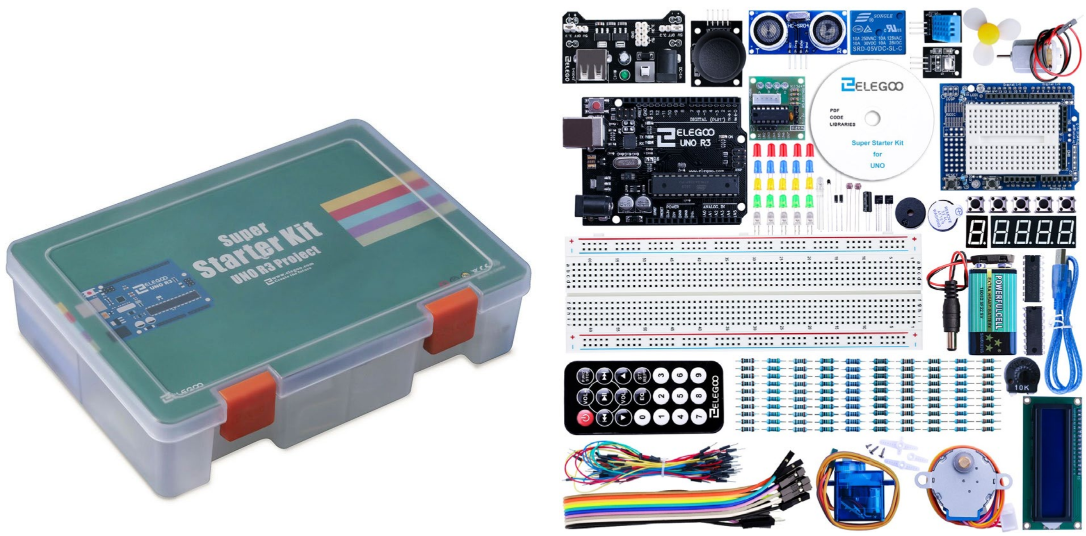

Figure 1. Microcontroller kit chosen for the students [10].

2 Methods

We adopted implementation and assessment methods that have been validated by many others in literature. Some of the activities, including using TinkerCad and creating smaller coding assignments, has been used successfully in a first-year design class [8,13]. Some of the mini-projects used at other universities, like Blinking LED, are similar to the coding assignments we have used [8,17]. In Term 4, we added two in class activities to teach students how to troubleshoot their hardware using digital multimeters and this has been used previously by others [6]. However, we had to develop many new processes because of the structure of the class, number of teaching assistants, and new projects. Each project required some new skills that students need to be successful and changing projects each semester presents a number of other challenges which has not been discussed deeply in literature.

This paper examines course content development, teaching staff management and training, student learning assessments, and communication improvements. Data from student self-

† The authors declare no financial interests

‡ For convenience, the two brand names, Arduino and Elegoo, are used interchangeably in this paper

assessment, instructor evaluations, final project demonstration results, and teaching assistant and instructor observations are also reported.

3 Different Approaches to Teaching Coding

We have tried different approaches to teach coding, including lectures and in class activities. In Term 1, the first coding activity was to download the Arduino software and show them how to write code and print text or variables to a screen (called the serial monitor in Arduino). This activity teaches students how to use and define variables, void setup, void loop, and baud rate. Students were given about ten minutes as a team to come up with a code that printed "Hello World" to the serial monitor. The second activity is to create a variable which increments the number of times "Hello World" is printed to the screen and the delay feature is introduced so the print to the screen is not unreadable text.

We approached these initial class activities as a "Google it" adventure with teaching assistants walking around and helping students. The rationale behind this approach is that coding requires a lot of self-learning and we want to promote resourcefulness. There is no way we can teach them everything they will need in one week when doing a four week project; in Term 1, they needed to have working code in the second week. Additionally, Arduino has the benefit of having a large amount of online reference materials available and we want to encourage to practice looking up relevant information on their own and applying it. However, the students viewed it negatively, as an increased workload and wanted more available resources.

We quickly learned that a "Google it" approach overwhelmed the students with limited coding experience, and sometimes students may find solutions on Google that may not be what we had intended them to use. Also, the students were not "Googling it right" and there is an advantage to learning it in the classroom and directing students to the appropriate resources [12]. Students can be introduced to the Arduino community and made aware of its open and available resources.

In Terms 2 and 3, we moved away from the "Google it" approach and created separate activities to do during class and more structured assignments for outside of class. In Term 4, we are setting aside an hour each week in class to do coding in class and use that dedicated time to have them start assignments. The rationale is that if students can start the work and get some questions answered right away, they will hopefully be more likely to follow through and complete the assignment later.

4 Coding Assignments

To ensure that students are successful in developing coding skills, we created new course materials for in class activities, homework, and some team-based assignments. The purpose of these assignments is to build skills and gain familiarity with components which are needed for the project. ME 250 is not a computer programming course, so the intent is to use Arduino as a tool, either collecting data or controlling motors or reading sensors.

We prepared smaller coding assignments where parts of the code can be used to build into a code for the design device. We do not provide the students with a starter code for the projects and they are required to innovate. Students must draw upon the skills learned in the activities and piece them together in a logical way to create a working project. This may mean that one assignment taught them how to operate a motor, but the project requires them to extend this to running two motors. In the assignment, we ask them how they will use two motors (in theory), but do not require them to code it yet.

In Term 1, we created a few short assignments which could be solved with a few lines of code if the students found the appropriate online reference material. Some students submitted code using functions we did not intend them to find and this often led to complex and lengthy code. One student mentioned that he/she had looked online for several hours and could not find anything and because of no prior coding experience, he/she just did not know where to start. Also, many syntax issues (semicolon in particular) were mentioned as a common issue that students struggled with.

In Term 2, we redesigned the assignments and instead started with TinkerCad for the first two assignments. TinkerCad is a website which simulates an Arduino and IDE and it is an excellent tool for beginners. Students can begin coding in Arduino (without downloading any software) and use hardware, such as LED or sensors, without having to wire the circuit. This allows students to simulate using a microcontroller and everything is easy to visualize. This has been used by other institutions for self-paced and self-guided activities prior to replicating the system using physical hardware [13]. This has been shown to be effective and recommended for similar courses and can help to give students a solid understanding of programmatic logic without the need to spend large amounts of time teaching syntax [8].

The first assignment tasked students with lighting up an LED and getting it to blink, so they learned about digital pins, digital input/output, and delays. This is similar to the mini-projects and assignments used at other universities [2, 17]. We have continued to use this assignment and in Term 4, we start the assignment in class to ensure students can get TinkerCad working and have a little success with it before they leave class.

We created a second TinkerCad assignment that builds upon the first and introduced other sensors, like a photoresistor. Syntax errors were still mentioned as problems, but not as much as in the first assignment. The third assignment introduced the physical hardware and Arduino IDE, and the same assignments were repeated using physical hardware instead of TinkerCad. Wiring components is a new skill and different experience, so many students had wiring difficulty. We included wiring diagrams in the assignments for the student's reference. Many students mentioned that they needed more of an understanding of how to code and that more time for programming could have helped them avoid some of the basic syntax and hardware issues they encountered. Additional coding assignments were more focused on skills or hardware needed for the project that semester, including a type motor that would be used. Due to time constraints, we had to cut some assignments which taught the differences in the motors and how to use each one to an assignment that taught them how to use one type of motor.

In Term 2, we created a webpage with the assignments for easy integration of links to videos/links as resources and where we could upload gifs showing key information like sample wiring and sample outputs for the students to compare their assignments with. When designing coding assignments, one lesson learned is to break up the assignment into smaller exercises that gradually build upon each other. Our teaching assistants have noticed that students are good at noticing small variations and it is a good idea to keep the goal of the exercise to one objective. Each exercise includes a list of basic steps to complete that exercise. At the end of each assignment, there are a series of short answer questions to address the student's need to get a better understanding of coding/hardware, which was expressed as an issue in Term 1. In addition to the assignments, we created additional in class activities to help students with the specific project. Our teaching assistants did the assignments before they were posted and added in hints where he/she believed students would need extra help and this allowed students to figure out solution faster. Also, we added links to online Lynda tutorial videos (free) which were helpful for students.

One of the challenges for students is creating motivation to complete coding assignments. If it does not appear useful for the project, the students have less motivation to complete it. We added in some comments at the top of the assignments about how these skills will be needed to complete a successful project. Also, when announcing assignments in class, it was important to tell students that the assignments build upon each other and how they need to do them all. When designing coding assignments, they should teach the minimum information needed to complete the projects and not over reach. Students use a submission template to turn in their work which tells students what information to include, gives short questions to answer, and reflect on what they have learned. For submission, students took screenshots of the IDE and serial monitor and answered conceptual questions about what was happening.

Another challenge faced was that coding took much longer than anticipated for students to learn. They frequently had to go back to previous assignments to remember how to code some part since the exercises in the assignments were short tutorials where we outlined the general steps needed. This ties in observations that students tend to copy and paste code from tutorials without really understanding how to write that code from scratch [4].

Other challenges observed from Terms 1-2 is that if students get off to a bad start with coding, they may never attempt it. Some students took a hands off approach to learning coding and refused to engage in it the entire semester. This happened frequently in Term 1 and less so the following semesters, though this still occurs. We learned that using class time is important to get students started with coding and have many teaching assistants walking around to assist. We generally try to have 3-4 (more is better) undergraduate teaching assistants helping with a class of 40 students when working on coding. In Term 4, we dedicated an hour each week to give the student's time to get started on coding and ask coding questions and we have found that this method has employed more success in motivating students.

5 Creating Teams for Projects

For a class filled with students at different levels of experience, one question is how teams should be created for the projects. ME 250 uses teams of four students in the projects. First, we created a survey for students to ask them about their coding experiences and to rate themselves as novice, some coding experience (self-taught or short coding class), intermediate (completed a coding class), and expert (can use multiple languages). We used this form to create teams with mixed skills to ensure that all teams could be successful with the design project. In Term 1, the mixed teams gave varied success due to time constraints and it was typical that one team member with good coding skills took over the entire coding for the project. For Term 2, teams were created using members with similar skill levels, so they can work together at the same level and this limited one experienced coding member from taking over the project. We found that similar level teams worked well and continued this in the following terms. It also puts more responsibility on all team members to gain skills needed for the project by doing the assignments as we emphasized that students can be successful if they learned these skills.

6 Design Projects using Arduinos

Providing a hands-on design experience has been shown to be an important recruitment tool to keep students engaged and interested in studying engineering. As we created new projects for each semester, here are some challenges we have faced and lessons learned for each project.

6.1 Design Project: Rush Hour using UltraSonic Sensors

In Term 1, we did two Arduino based projects. The first Arduino project, Rush Hour, concerns data collection, analysis, and interpretation to define the "rush hour" for a specific location on campus. This project is done in a span of four weeks and students are required to use the ultrasonic sensor from their kit [10]. Teams are asked to collect foot traffic data at a specific location on campus over a period of time, ideally 9am-6pm for a typical week. Teams are tasked to develop a code for the Arduino microcontroller, set up a pair of ultrasonic sensors with a microcontroller, and prepare the hardware for foot traffic data collection. The teams then handled and interpreted the large amount of data to identify patterns and statistics and define what "rush hour" is for that location on campus. Data science or data analytics is a topic in industrial engineering, and automation is a topic in electrical, computer, and mechanical engineering.

With forty teams in the class, this meant that teams had to use the same locations on campus. For efficiency, five locations were selected so we built a tower to mount multiple team's sensors. The sensors were aligned vertically in a tower which is placed near the room entrance to the side of the door. Anyone entering/exiting the room must be close enough to the frame to be detected, which is difficult if the doorframe is large and students were walking farther away so as to avoid hitting into the frame. Also, since this was for multiple teams, the teaching assistants were responsible for placing the towers in the mornings and collecting them at the end of the day to ensure their safety.

Teaching students from a diverse background of coding expertise in only four weeks is a huge challenge. Students commonly struggle with syntax errors or hardware errors. Many of our students have never coded before and/or worked with a breadboard and basic wiring. Therefore, we felt it would be too daunting to have students work right away with the code in the short time frame. So, our approach was to introduce flow and logic using a flowchart algorithm that would allow students to think about the steps and flow that their code should have to meet the goals of the challenge. Also, having students make a flow chart, would help us pinpoint flaws in their logic and help us understand concepts that were not clear to the students.

While algorithms are a valuable tool to help students understand logic in code, we found that for our students, it probably should not have been the first step. This concept was too abstract for many without prior coding experience and students could not see the connection of how a flowcharts can be used to generate code. While conditional statements were explained to the students in the flowchart, we found students did not understand completely how to code conditional statements. Most of the common issues mentioned by the students were syntax and hardware issues, which probably could have been avoided if we had given students more time to practice these skills before we introduced the project. We did not do flowcharts in Term 2 to the same extent and we quickly realized that we cannot teach this as a coding class in Term 1.

Some student comments from their Rush Hour project report include:

"Team five's code performed without flaws during individual testing. Evolving the algorithm into usable code was a challenge. Team Five faced many syntax errors and had to learn the best way to call and use the ultrasonic sensor values. After retyping and debugging, the team eventually developed a working code. At this point, the main problems were: the sensors failing, thus needing to be replaced and adjusting the delays. With persistence however, Team Five's code went on to work without flaws."

"Limitations on the project are to be considered. For one, the sensors that the team was using were not the best. They were not picking up everyone that came into the room... A lesson learned is that coding is very tedious and hard. It takes time to code devices to perform. Even though the objective was relatively simple and the code was not too long, the team had to debug the team's code many times to get it to work."

"The team, after much trial and error, was able to create a set of code along with successfully set up hardware, and was able to collect proper code and data. However, at first, it was a daunting task because the team has very little to no experience in C++ coding, which was required by the Arduino. Therefore, the team spent a few days learning the code by practicing loops, serial printing, and creating if statements. Through these lessons, not only did the team appreciate Google much more but were finally able to know enough code to successfully create the project...The problem was due to a hardware issue, once this was resolved the program worked well but it seemed to count too fast. To solve this problem the team messed with the code."

Due to time lag in ordering materials, it was not always possible to test all hardware and pinpoint common issues before the project starts. The first hardware issue we did not foresee was that many of the student's ultrasonic sensors stopped working. It could be due to the quality of the sensors, but more likely it was many of the students' first time wiring and there were many wiring problems. Many students accidentally mixed up their ground and their power connections and this caused the ultrasonic sensors to fry. In the end, most teams were able to get their circuit and code working properly. However, we faced additional problems with large interference issues when we placed several circuits in the tower next to each other, same shelf and above/below. This was unexpected since it was our first time using the ultrasonic sensors and more hardware testing could have revealed the problem sooner.

Creating specific exercises to troubleshoot issues with the circuit can be quite helpful for students and even for teaching assistants to help out. Many times, students are unsure if their ultrasonic sensor is working. We used an online tutorial for a buzzer (piezo speaker) and taught them how to use the buzzer. Then, students used the buzzer to sound if the ultrasonic sensor detected an interference. Students were able to wave their hand over the ultrasonic sensor and hear a buzz to ensure their sensor worked. It was simple enough that students were generally able to implement it on their own. Some teams figured out how to replace the buzzers with LEDS as indicators if the ultrasonic sensors were reading values or not. This is not something students would think of to do their own, so it does require some guidance from instructors and teaching assistants.

6.2 Design Project: Autonomous Pet Feeder using Motors

The second Arduino project for Term 1 was the design-build-test of an Autonomous Pet Feeder and this project started right after the Rush Hour project. The Autonomous Pet Feeder project is longer and spans nine weeks due to manufacturing a prototype. The Pet Feeder must sense the weight of the pet, calculate the correct amount of pet food needed, and dispense it into a cup. Students are allowed to use any components from their Arduino kits and we purchased stronger servo motors since the ones in the kit did not provide enough power after completing some initial testing.

The Autonomous Pet Feeder project is a good mechanical and coding design challenge for the students. The students not only had to figure out how to use a motor to dispense the pet food, but also had to make a well-designed mechanical transporting mechanism so the food did not clog and could be dispensed quickly and accurately. The design space was large and there were many creative solutions presented.

In their reports, students provided comments about the Autonomous Pet Feeder project. Despite the project being a challenge, students generally expressed positive/beneficial attitudes about the challenge.

"The project was very beneficial for learning the development or improvement of a unique or a current product. It would be impossible to reach this point without the tremendous help and mentoring of the instructor and technical assistance staff. The importance of each step was well taught."

"In theory, most ideas work; however, it's the testing that really matters. There are ways the team could have improved the design and made the gear work better with the loading, but there just was not enough time for new ideas."

"Frustration was one common feeling among all the team members being that the amount of work was overwhelming because none of the group members have had experience in such a project before, and the amount of time allotted didn't match the workload. Nonetheless,

excitement became the major feeling when the device was completed and the experimental testing proved to be a success."

For many students, the first challenge was figuring out the code. However, even after most students created working code, when they tried to integrate this with their built device, they came across unforeseen issues. Some teams had issues with the pet food getting stuck or falling out uncontrollably even after the motor stopped. For example, some teams learned from testing that they had to redesign their transporting mechanism (auger, for example) or the angle (straight vs. slanted) of the shoot from which the food was dispensed. Students often found the issues late in the project and some required significant changes to their built devices. The size of the pet feeder was not limited; many students printed large pieces or the entire device with 3-D printers which caused a backlog in 3-D printing. While students were waiting for 3-D printed parts to complete their device, they lost valuable time to troubleshoot and experiment, so many teams did not get to the experimental testing phase. However, those that did were able to troubleshoot key issues. It is important to allow more time to do more experimental testing and address the mechanical and coding issues, which may mean moving forward the deadline for the prototype to be built. Providing more lab sessions for the project to allow students the time for deeper investigation of electronics and programming could boost their understanding and confidence in these areas [7].

Again, we found that it is important to test all hardware components and ensure that they are properly chosen. Hardware issues were a common barrier for teams. Students were limited to the stepper motor in their kits (sg90 motors) and often it was too weak to handle the weight of the pet food. Earlier testing of the limitations/constraints of the motors could have indicated a need to purchase stronger motors and would alleviate some of the challenges.

We gave students some calibration assignments to help create a successful prototype. The first assignment helps students learn more about manufacturing and it a CAD modeling assignment in which students designed a "horn" for their stepper motor and 3-D print it. Students practiced proper dimensioning, tolerancing, and 3-D printing. This assignment helped students to visualize how their stepper code and motor would work together and taught them to harness the movement of the motor to convert it to mechanical energy. Another calibration assignment was an Arduino scale calibration which was meant to help students calibrate how their pet feeder would be integrated with the pet scale at competition so the correct amount of pet food gets dispensed. We provided most of the code, so students only had to change a few values specific to the weight/design of their device. This freed up to students to test their devices and spend their time on other aspects needed for this project.

The Autonomous Pet Feeder project was done in Term 1 and we had not assigned many coding assignments in preparation of this project and none were specific to the project. Although one assignment gave them exposure to different kinds of motors, we knew most would use stepper motor, so more effort could have been made to make assignments related to this motor. This influenced many of the changes made in Term 2.

6.3 Design Project: Measuring Speed using Photoresistors

For Term 2, the class completed a three week long design project, Measuring Speed using Photoresistors. Students used their Arduino kits to complete an IE based project to measure the speed of a car moving a track using a microcontroller and photoresistor sensors. The students used two photoresistors placed a certain distance apart on the track, underneath the track, to detect the car's motion. Students learned about how photoresistors work and how to use threshold values to code for if/then statements. Student wired and coded the photoresistors, determined thresholds to detect the car overhead, calculated velocity, and printed the speed to an LED screen.

We had some challenges using the components from the Arduino kits and some of the photoresistors did not work. The thresholds for each photoresistor are different, especially, when in series with a resistor. If the photoresistors slide out of place, the car racing by overhead may not be detected. Students were not careful with the wiring and some of the wires popped out and many students had trouble figuring out what the problem was.

6.4 Design Project: Color Sorting Machine

The second Arduino project for Term 2 was to design-build-test a Color Sorting Machine in nine weeks. Students are able to find many examples online of color sorting machines and even how to build them in Arduino. However, we limit the types of materials that can be used for building and constrain the sensor used for color sorting to the same for the entire class. Students were tasked with sorting a given number of objects by color and placing them into bins. Three colors were identified as desired-all other colors (two colors) were placed in a discard bin. They could either place like color objects together in one bin or place one of each of the three colors into a bin.

We had some hardware problems using the Adafruit color sensor we ordered for the project (Amazon, $10) [14]. The sensor needed to be right next to the object to determine the color as part of the design. Another problem is that when the sensor was placed in a different lighting situation, the thresholds of RGB were different, so the code needed to be calibrated for the room. This was a problem as student's troubleshooted at home, worked on it in the classroom, and used another room for the design competition. Additionally, some of the balls had large variations in color and this made it difficult to set thresholds for the RBG value read on the color sensor.

Some comments from the report about coding and the color sensor include:

"The team found that a lot of adjustments needed to be made to the RGB values in the code to account for the amount of light the color sensor was exposed to. When the code was written at the start of the design process, the color sensor was exposed to all of the lighting in the room. In the device, the color sensor is exposed to minimal lighting."

"The code proved to be difficult to fine-tune, because the color sensor did not seem to sense much of a difference between the different colors of each ball."

6.5 Design Project: Autonomous Line Following Robot

The design-test-build project for Term 3 is to design an Autonomous Line Following Robot in eleven weeks. The students were required to use photoresistors to detect the line (up to eight photoresistors), use two DC motors with dual shaft gear box we provided (Amazon, ~$15 for 4 sets [15]). Teams were each given a rechargeable battery 5V pack or a 9V battery for power supply.

We experienced many hardware issues for the project. Many students were having an issue with one (or both) of the DC motors "lagging." The DC motors we selected operated in a 3-6 Volt range and we found that using one power supply was inadequate. We had originally selected a 5V rechargeable battery so that all devices could be untethered and operate remotely to follow the line. We used an H-bridge to control the motor and found that the voltage drop across the H-bridge was too large to power both motors due to the large start up current required. The motors could not be connected to the Arduino microcontroller as the power supply because it is also limited to 5 V. Students also had the option to use a breakout board in conjunction with the H-bridge to bypass the microcontroller to power the motors. In hindsight, we should have selected a 9V rechargeable battery. We provided students with adaptors, but the motors received more power than needed and some burned out.

Other challenges faced with this project include power losses due to inefficient wiring which was a common problem for students. Students also did not ensure all the wires were firmly in the pins. Additionally, we found that some of the wires shorted and we needed to change out wires (one by one) to see if that was a problem. Also, it was common that students declared pins in their code and did not use the same pins on the microcontroller. The photoresistors on the robot detected the path and sometimes these moved as the robot moved; we learned that if they are not stabilized, their movement can change the threshold value and cause the device to wander off the path. Prior to starting the project, one team member determined the maximum weight for the motors and this information was presented to the students as a recommendation. This was extremely helpful information to have available.

We had improved both the structure and timeline of the course in Term 3. We removed the second project (data science) to free up two additional weeks for manufacturing and testing for the high-stake design project. For Term 3, we introduced three design milestones in the last weeks of the project which encouraged the students to keep pace to ensure completion prior to the competition. Previously, teams would test their devices, but there was no deadline for it until the last day. The design milestones provided a structure to the class that worked quite well and the students pushed themselves to complete these milestones since it was part of their grade. For the first time, some of the teams finished the project before the final week and the number of successful designs increased. This is something we can continue to use in the class and would highly recommend. The milestones helped ensure success as noted in one team's comments from their Autonomous Line Following Robot report:

"The group performed extensive testing of the device….The project started with nothing but the group's Arduino kits and knowledge from previous coding assignments…. We improved upon

our existing code and components by varying photoresistor threshold, the turning speed, distance from the photoresistors to the ground. The final product of all of these iterations and improvements is an autonomous line-following vehicle, as described in the project statement."

By the final week of classes of Term 3, most teams had successfully completed all three milestones and had working devices. The last week was used to troubleshoot and ensure the robot can follow a practice track. We are finally seeing some real improvements in the quality of the projects, including the number of devices that worked in Term 3. We plan to continue using the milestones in the future.

7 Effect of Arduino-Based Projects on Design Space

Prior to introducing Arduino based projects in Term 1, ME 250 focused only on mechanical designs for the final project. Students use morphological methods to optimize their design by considering all relevant possibilities and then reduce the design space by considering the objectives and comparing design alternatives [16].

To simplify ordering materials, students were required to use a particular sensor in the projects. In eliminating options for sensors, the design space actually becomes more limited. Instead of millions of design possibilities, the design space shrinks to the hundreds and many of the final designs end up being too similar once we put the constraints on the design. This is seen in the Line Following Robots project where we constrained the sensor to be a photoresistor and motors to be specific DC motors; most students used the two wheels with the dual gear shafts provided and all designs looked similar. For upcoming semesters, there is a push to bring that creativity back. In Term 4, we have thought more critically about where students can be creative and chose a project that will reward creativity and open up the design space.

8 Motivating Students to Learn and Perceived Student Workload

One of the challenges when introducing Arduino in ME 250 is overcoming the resistance to learn Arduino as many of the students had not been exposed to computer coding previously. The students experienced an increased workload during the shift to Arduino in Term 1 and this is documented in results from University of Illinois at Chicago's end-of-course teaching evaluations.

Data from the most recent seven terms has been analyzed. Besides Term 1 through Term 3 as defined earlier, we have included four previous semesters which precede Term 1 and referred to them as Term 1-1 through Term 1-4 and show them as control semesters. During these four terms, the class used limited electromechanical components. Specifically, in Term 1-1, a low-stake three week project was assigned where students were tasked with building a model truss bridge and lifting it using a servo motor and microcontroller. The training in coding and circuitry was provided on an ad hoc basis with only a short tutorial handout and lots of teaching assistants helping, and the learning outcome was limited to Level 1 of Bloom's taxonomy. Term 1-2 and earlier terms contained no coding or mechatronics whatsoever.

It is necessary to note that all seven terms share some similarities. Class size is capped to 44 students in each section; in fact, most sections have an average enrollment of 40. Student diversity and makeup have remained steady in recent years. Lastly, no significant changes have occurred in the Mechanical, Industrial and Civil Engineering curricula at University of Illinois at Chicago. Therefore, the course load and variety of courses undertaken by a student enrolled in ME 250 have been predictably stable across the terms.

The teaching evaluations use eight Likert-scale and two open-ended questions to survey student satisfaction and perception of their learning. We have examined the results from three Likert-scale questions in this section. Data from multiple sections, taught by the authors in parallel, in a particular term are synthesized. All sections are tightly coordinated, and variability in the type of data presented here is believed to be insignificant. The section mean is simply averaged across all sections, and the standard deviation is averaged by taking the square root of the average variance across sections.

Figure 2 shows the results of the teaching evaluations on how students rate the overall quality of the course. A rating of 1 means poor and 5 means excellent. There is a dip in the curve for Term 1, when Arduino was introduced in the class. The curve shows improvements that were made in Term 2 and Term 3 results in higher quality. By Term 3, the overall quality of the course is the same as the earliest control terms, Term 1-4 through Term 1-2.

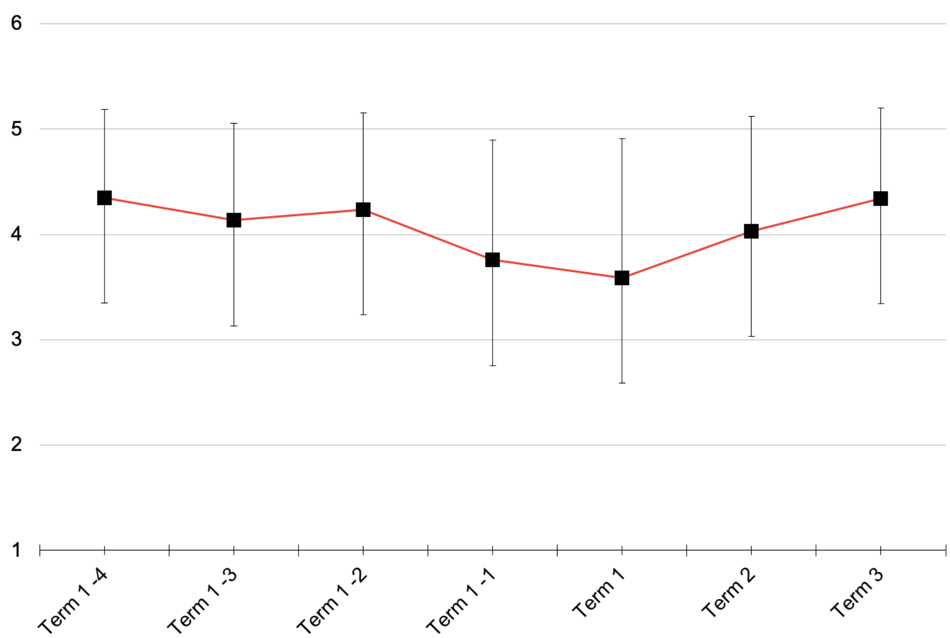

Figure 2. Results for rating the overall quality of the course: average and standard deviation, (1 = poor and 5 = excellent).

Figure 3 shows the results of the teaching evaluations on how students rate the course difficulty relative to other courses. A rating of 1 means very easy and 5 means very hard. The result shows a steady, upward tendency throughout the control terms, but the trend levels off beginning in Term 1 when Arduino was deeply integrated. This result is rather surprising at first glance, since students were expected to find it difficult to continuously learn and apply coding, electromechanics, CAD and teamwork skills – all within a semester. This result may be attributed to the careful and systematic approach in planning, implementing, and administering the revised projects, as well as the increased competency among the TAs, and increased frequency of communication between students and instructional team.

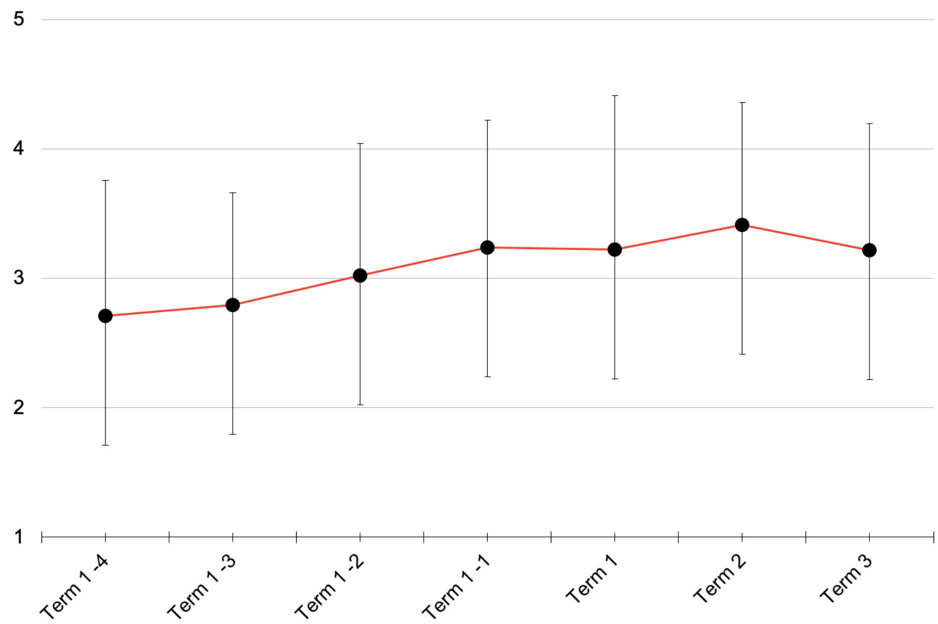

Figure 3. Results from rating the relative course difficulty: average and standard deviation, (1 = very easy and 5 = very hard).

The results of the teaching evaluations have been summarized and presented in Figure 4 for the student's rating of the course workload relative to other courses A rating of 1 means much lower workload and 5 means much more work. A generally increasing trend but diminishing slope is seen here, except for Term 3 where a 4% increase over Term 2 and Term 1-1 is observed. It is interesting to compare the trends across Terms 1 through 3 between Figures 3 and 4. In Term 2, students felt a slightly increased difficulty compared to Term 1, while workload remained essentially constant. Term 3 has the opposite effect when compared to Term 2, lowered difficulty but increased workload.

Student comments from the teaching evaluations have been analyzed across the seven terms. In summary, among the four control terms, students enjoyed learning CAD, hands-on and team-based nature, help from TAs and availability of the laboratory. In addition, for Terms 1 through 3, students also found coding and Arduino helpful. Some of the students, while initially challenged to learn coding in Term 1, took the initiative to learn how to code in Arduino and enjoyed it.

Critical comments from the four control terms include workload and inconsistent feedback from TAs and instructors. For Term 1, the main complaint was regarding how coding was taught (i.e., self-guided, research-based approach). In Terms 2 and 3, the coding experience seem to have improved greatly. For Term 4, we have introduced some mini-lectures to go over more of the material in class and assigned Lynda videos for Arduino too. Some students still felt overworked in Terms 1 through 3, but not to the extent of the control terms prior.

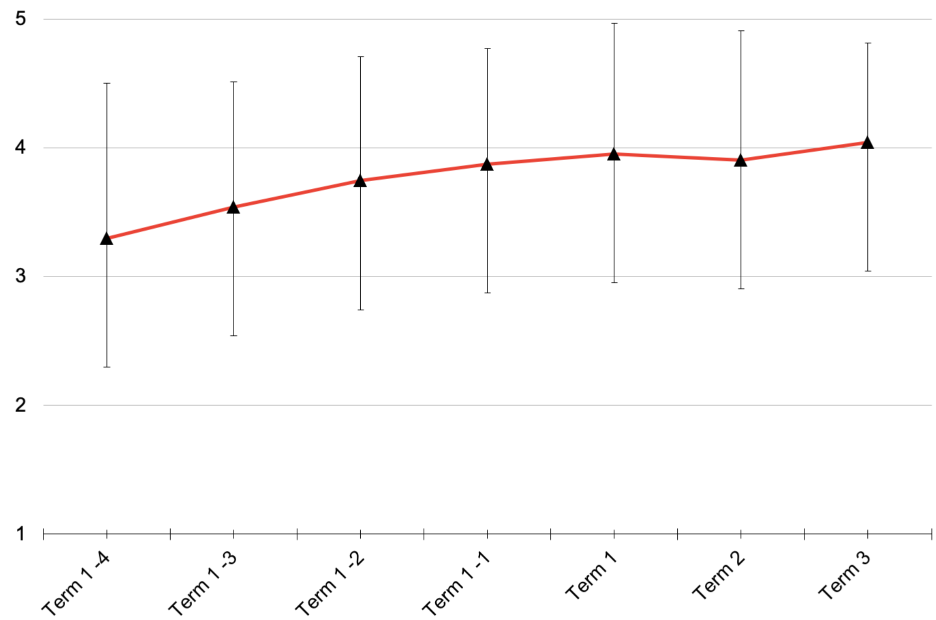

Figure 4. Rate the course workload relative to other courses results: average and standard deviation of the relative course workload, (5 = high load, 1 = much lower load).

9 Importance of Continuous Training and Having Outstanding Teaching Assistants

Using undergraduate teaching assistants to help manage the class is not new [18]. We value the peer learning cycle and students benefit from working in teams and with former students. We had a few key undergraduate teaching assistants who had prior experience with Arduino and could answer student questions quickly. But, many of our teaching assistants were hired earlier and had not taken the class after Arduino was introduced. These teaching assistants were unable to help with Arduino in Term 1, which resulted in increased frustration for students. This led us to design a mandated training program prior to Term 2 for teaching assistants.

The mandatory training program ran for three days prior to classes starting and teaching assistants had to work through all of the coding assignments and practice troubleshooting code and wiring. With having new projects each semester, the sensors we use may be new to the teaching assistants if they didn't learn them when he/she took the class. Graduate teaching assistants attend as well and help ensure that all training is completed.

One of the challenges encountered was how to quickly identify what was wrong with the student's project when it didn't work. It could be coding, wiring, hardware issues, or defective components. Our teaching assistants did not have a lot of experience using DC motors prior to Term 3 and this required some additional training sessions later in the semester. Although we gained experience, we learned it too late to be effective.

One of the lessons we have learned is how important it is to hire the right people. We want teaching assistants who can answer student questions and troubleshoot code, hardware, and manufacturing. The TAs need to be flexible and always ready to make changes if needed as new issues arise during the project. If they are not great in one aspect of the class, the TA must be willing to learn and further develop his/her skill and must also be good at working with other students. Our TA selection process has become more formal for Term 2 and we now find outstanding students and conduct an interview. Currently, we have a great diversity of teaching assistants. In Term 4, for example, we have ten female students and ten male students as undergraduate teaching assistants and they all have different majors (IE, Civil, ME).

It is important to hire the right students as teaching assistants, those who had success with Arduino and coding. They also need initiative and grit since introducing new projects every semester requires on-going training for all teaching assistants. The students who have been the teaching assistants do benefit from having this type of work experience on their resumes and tend to get multiple job offers when graduating.

10 Conclusion

There have been many challenges to modify the first year design course, ME 250, to include Arduino into the curriculum and there have been several lessons learned along the way. Each semester has tried to build upon what we learned previously and shown some improvement. We analyzed student comments from the teaching evaluations for seven terms. Teaching evaluations have shown that the first semester that Arduino is introduced may be perceived as a decrease in the quality of the course, but we have shown continuous improvement. In summary, among the four control terms, students enjoyed learning CAD, hands-on and team-based nature, help from TAs and availability of the laboratory. For Terms 1 through 3, students found coding and Arduino helpful.

There have been lessons learned so far, especially as we change the high-stakes design project each semester. Teaching assistants and instructors must always be ready to make changes if hardware or sensors fail, so it is important to start early and ensure all components will work well and even so, some issues will occur that are difficult to plan for. The best first exposure to coding is to use class time and many teaching assistants to support the student in getting started, either with TinkerCad or using the Arduino and wiring.

When creating course materials, we learned that we cannot teaching coding like a computer science class and the students just need the minimum skills to be able to complete the project. We provided many resources to the students and started dedicating time in class for coding activities. We started using TinkerCad prior to wiring a breadboard which made it easier for students to get started. Breaking up coding skills into small exercises that build upon each other and making design milestones also have helped our students stay on track. Term 3 seems to be the most successful class so far with Arduino, where students felt challenged by the project, but also supported with the correct resources and expertise of the teaching assistants. For the first time, we had teams who had a completely built device several days before it was due.

11 References

[1] Siow, Y., Szwalek, J., Komperda, J., Darabi, H., & Mashayek, F. (2019, March), *A Critical Look at Mechanical Engineering Curriculum: Assessing the Need*. Paper presented at 2019 ASEE Illinois-Indiana Section Conference, West Layfatte, Indiana.

[2] Bedillion, M. D., & Muci-Kuchler, K. H., & Nikshi, W. M. (2018, June), *An Arduino-Based Hardware Platform for a Mechanical Engineering Sophomore Design Course*. Paper presented at 2018 ASEE Annual Conference & Exposition , Salt Lake City, Utah. https://peer.asee.org/29774

[3] Walter, W. W., & Southerton, T. G. (2014, June), *Teaching Robotics by Building Autonomous Mobile Robots Using the Arduino*. Paper presented at 2014 ASEE Annual Conference & Exposition, Indianapolis, Indiana. https://peer.asee.org/23103

[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 Annual Conference & Exposition, Vancouver, BC. https://peer.asee.org/18720

[5] Metzler, S. A., & Lilly, B. W. (2015, June), *Implementation of a Novel Second-year Mechanical Engineering Course to the Curriculum.* Paper presented at 2015 ASEE Annual Conference & Exposition, Seattle, Washington.

[6] Mariappan, J. & Nissenson, P.M., & Shih, A.C., (2016, June). *A Hands-on First-year Mechanical Engineering Course*. Paper presented at ASEE Annual Conference and Exposition, New Orleans, LA.

[7] Tsuchiya, N., & Nissenson, P. M., & Jawaharlal, M. (2017, April), *A Student Assessment of the Value of a Redesigned First Year Mechanical Engineering Orientation Course.* Paper presented at 2017 Pacific Southwest Section Meeting, Tempe, Arizona. https://peer.asee.org/29200

[8] Sierzputowski, W., France, T., Katterle, K., Dazey, T., & Palmer, J. (2019, March 22-23). *Avoiding the Syntax: An Accessible Approach For Introducing First-Year Engineering Students To Microcontrollers*. Paper presented at the American Society for Engineering Education North Central Section Conference, Grand Rapids, MI.

[9] Recktenwald, G. W. (June, 2016). *Six Years of Living with the Lab: A Survey of Student Experience.* Paper presented at ASEE Annual Conference and Exposition, New Orleans, LA.

[10] Elegoo starter kit. https://www.amazon.com/EL-KIT-003-Project-Starter-Tutorial-Arduino/dp/B01D8KOZF4

[11] Belfadel, D., & Rodriguez, M. A., & Zabinski, M., & Munden, R. (2019, June), *Use of the Arduino Platform in Fundamentals of Engineering*. Paper presented at 2019 ASEE Annual Conference & Exposition , Tampa, Florida. https://peer.asee.org/33491

[12] Chancharaoen, R., Angee Sripakagorn, and Kintinee Maneeratana. (Dec 9-10, 2014). *An Arduino Kit for Learning Mechatronics and its Scalability in Semester Projects*. International Conference on Teaching, Assessment and Learning (TALE), Wellington, New Zealand.

[13] Lawrence Funke, J. Blake Hylton, and David Sawyers. (2019, June), *Work in Progress: Incorporating Microprocessors across the Mechanical Engineering Curriculum*. Paper presented at 2019 ASEE Annual Conference & Exposition, Tampa, Florida. https://peer.asee.org/33630

[14] Adafruit color sensor. https://www.amazon.com/Adafruit-Color-Sensor-Filter-White/dp/B00OKCRU5M

[15] Wheels and gearboxes. https://www.amazon.com/Electric-Magnetic-Gearbox-Plastic-Yeeco/dp/B07DQGX369/ref=sr_1_5?keywords=arduino+motor+wheel&qid=1571260922&s=digital-text&sr=1-5-catcorr

[16] Dym, C., Little, P., Orwin, E., and R.E. Spjut. Engineering Design: A Project-Based Introduction, 4th Edition. John Wiley & Sons, New York, 2014.

[17] Rosen, W., & Ertekin, Y., & Carr, M. E. (2014, June), An Autonomous Arduino-Based Racecar for First-Year Engineering Technology Students Paper presented at 2014 ASEE Annual Conference & Exposition, Indianapolis, Indiana. https://peer.asee.org/20044

[18] Metzler, S. A., & Lilly, B. W. (2015, June), Implementation of a Novel Second-year Mechanical Engineering Course to the Curriculum Paper presented at 2015 ASEE Annual Conference & Exposition, Seattle, Washington. 10.18260/p.24241