ASEE AMERICAN SOCIETY FOR ENGINEERING EDUCATION

Paper ID #35141

Integrating Computational Thinking in an Interdisciplinary Programming Course for Engineering Undergraduates

Dr. Prabha Sundaravadivel, The University of Texas at Tyler

Dr. Sundaravadivel is an Assistant Professor in the Department of Electrical Engineering, at the University of Texas at Tyler. She received her Ph.D. in Computer Science and Engineering from the University of North Texas, Denton in 2018. She earned her Masters of Technology (M.Tech) in VLSI design from VIT University, India, in 2015 and Bachelors of Technology (B. Tech) in Electronics and Communication from SRM University, Chennai, India, in 2011.

Currently, she is working with a diverse multi-disciplinary research group of Graduates, Undergraduates, and High Schoolers as the Director of the Intelligent Systems Laboratory (ISL) at UT Tyler. As a Faculty at the University of Texas at Tyler, she has been involved in outreach activities in East Texas to broaden participation in STEM.

Integrating Computational Thinking in an Interdisciplinary Programming Course for Engineering Undergraduates

Prabha Sundaravadivel
Assistant Professor
Department of Electrical Engineering
The University of Texas at Tyler, Tyler, Texas.

Abstract

With current technology advancements, the primary focus on today's engineering problems is in automating the systems. This can start from a simple automated temperature monitoring system in a home environment to advance Industrial automation using drones and robots in real-time. These advancements have been a major drive for job opportunities and have opened doors for interdisciplinary research across all domains of Engineering. Due to the huge scope of the problems in the current modern world, Engineering solutions are becoming increasingly software-driven. Hence there is a high demand for Engineers who can drive their solutions using software tools and various programming languages. To help our Electrical and Mechanical Engineering Undergraduates with necessary programming skillsets, the EENG 2301 Programming Languages for Design course was designed in Spring 2020. This course included hands-on coursework designed to teach the high-level programming languages and modern Engineering tools required for modeling, analyzing and designing projects. The curriculum was centered around integrating Computational Thinking (CT) through project-based learning approach. CT is a problem-solving learning process that can improve the thought process of the learners. As introduced by the International Society for Technology in Education (ISTE), CT involves 4 main cornerstones where students break down complex problems into smaller, simpler problems and derive solutions with the required details. In this paper, the effectiveness of incorporating CT in the early stages of programming is discussed as a strategy to improve the confidence and engagement of students. This course was taught as an in-person and online course at the University of Texas at Tyler in the Spring of 2020.

Introduction

Industry 4.0 refers to a new phase in the industrial revolution that focuses heavily on interconnectivity, automation, machine learning and real-time data[1]. Due to this revolution, programming the physical devices for scheduling, maintenance and data extraction has become part of everyday job duties. Learning to code and mastering in any one of the widely-used programming languages has become a key component in a student's curriculum vitae. With the drive towards this programming trend, recently kids as young as 7 years of age have started to learn programming basics through several development tools that achieve the learning outcomes through Block-based programming[2]. In such tools, learners are taught about the data structures, loops and programming style by helping them arrange simple colored blocks one after another.

In Undergraduate curriculum, most of the students who opt for engineering degrees, do not have the programming background due to their career interests. But the increasing trend to learn to code, has made it challenging for such students who are not enrolled in departments such as Computer Science or Computer Engineering. In Electrical Engineering, learning to code in C or C++ language has been helpful for engineers to program their microcontrollers and perform some analysis on circuits and devices where the theoretical work is quite advanced. Mechanical Engineers on the other hand use programming if they are interested to work in control systems, robotics, or mechatronics[3].

To help our Electrical and Mechanical engineering students with necessary programming skillsets and be better prepared for the ever-changing job market, a new course EENG 2301 Programming Languages for Design was developed and taught in Spring 2020 at the University of Texas at Tyler. In this course, hands-on coursework was designed to teach the high-level programming languages and modern Engineering tools required for modeling, analyzing and designing projects. This paper discusses the active learning innovations included in this project-based course which had 107 students enrolled in Spring 2020. The programming languages identified as most relevant for both the electrical and mechanical engineers were MATLAB, Python, and Arduino programming. Though the semester started as an in-person class, due to COVID-19 the course was taught as a hybrid course.

Background and Literature Review

College of Engineering at UT Tyler is home to 5 departments including Electrical, Mechanical, Civil, and Chemical Engineering, and Construction Management. Located in the central-east Texas, UT Tyler aspires to be a primary educational and economic driver of East Texas and UT Tyler's strategic plan includes supporting student involvement, promoting discovery that benefits East Texas, and provide workshops to guide students through the research cycle[4]. The main *challenge* in designing such an interdisciplinary programming course material for electrical and mechanical engineering students is their level of familiarity with programming. Because of the use of C, C++ in the Electrical engineering courses, the Undergraduate curriculum for Electrical Engineering degree, requires students to take programming courses from Computer Science Department in the Freshmen year. These courses include Programming fundamentals and Object oriented programming. However, mechanical engineering students had little to no programming background. In such an interdisciplinary environment, the main challenge was to introduce the basics of programming such as data structures and algorithms in a comprehensive class. Figure 1 shows the student's response on the familiarity of MATLAB programming in the first day of the class. In this figure, X axis represents number of students and Y axis represents the questions asked in the survey. Out of 105 students who took this survey, 49 of them said that they are hearing "MATLAB" for the first time and 36 of them said that they are in beginner level.

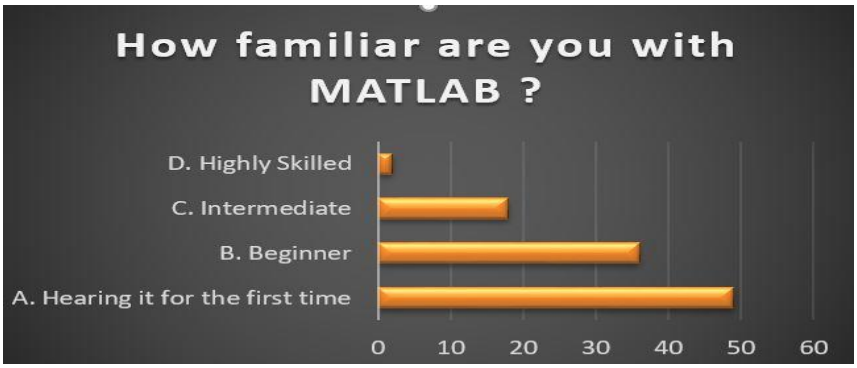

Fig. 1: Introductory Survey conducted in EENG 2301

To overcome this challenge and design a learning material that can help students conceptualize the programming fundamentals beyond a programming language, *Computational Thinking through project-based learning* approach was considered. As introduced by the International Society for Technology in Education (ISTE), CT is the problem-solving process that involves 4-main cornerstones such as **decomposition**, that helps students to break down complex problems into smaller, simpler problems; **Pattern recognition** that guides students to make connections between similar problems and experience; **Abstraction** that invites students to identify information while ignoring irrelevant details; and use **algorithms** to approach the design through simple steps[5]. Figure 2 shows the four cornerstones of the CT project. Incorporating CT right from the beginner's level curriculum, can help in improving the confidence of the learner and help them to conceptualize the solutions logically and creatively[6]. Though CT is commonly used in computer science related courses, *it is not very common to explicitly use the CT techniques for computer engineering courses.* Project-based learning (PBL) is a student-centered teaching method that can help teachers to engage students by providing real-world examples or problems. Through PBL, students investigate on a problem for extended period i.e. few weeks to a semester, which would help them to respond to a challenging authentic problem by creating a public product for real

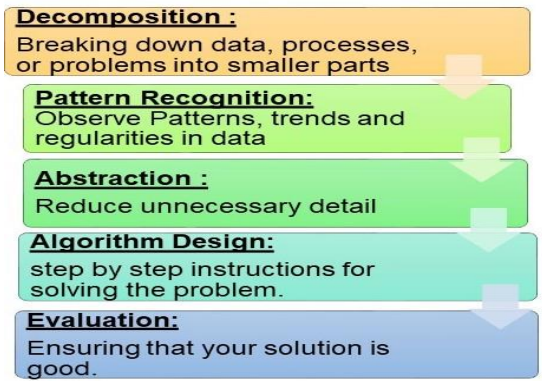

Fig. 2: Computational Thinking cornerstones as defined by ISTE

audience[7]. Integrating the strategies of Computational Thinking in a PBL framework can help students to creatively solve the real-world problems and apply their knowledge to various scenarios[8].

Techniques included in the Curriculum

To promote Computational Thinking through project-based learning, the curriculum was designed with three mini-projects, four hands-on assignments and a final group project. Following are the techniques involved in the classroom to engage students:

- Strategy-1: ConcepTests: Short surveys and quizzes were conducted in every class to analyze student's preparedness for the topic. This helped in adjusting the teacher's pace accordingly to improve the learning outcomes.

- Strategy-2: Think-Pair Share: Since this was a class of 107 students with Sophomores and Juniors from Electrical and Mechanical Engineering, initially they weren't familiar with their peers. Hence, students were randomly assigned into 27 Groups of 3-4 members to work on Mini Projects. These mini projects were given every alternate week for MATLAB and towards the end of 4-weeks of Python. By the frequency of these mini-projects, students were able to work as a team and establish collaborations that was helpful for their final project. Mini project assignments included,
  - Developing a Graphical User Interface using MATLAB
  - Using MATLAB as a calculator for simple budget, and temperature prediction. These were implemented using customized functions.
  - Developing a simple Tic-Tac-Toe game using Python.

- Strategy-3: Problem-solving through demonstrations, proofs, and stories: Since this course involved teaching and learning 3 programming languages, MATLAB, Python, and Arduino, specific examples and assignments were given in such a way that, students can conceptualize the application of the respective programming languages in system-level modeling. For the final project, students were encouraged to build a microcontroller-based project, that they can deploy it in real-time. They were encouraged to come up with creative ideas and use sensors and robotics concepts to build their final project.

Impact of Online class on active learning

This course was offered for the first time in Spring 2020. Due to the pandemic, the course was offered as a fully online course after the spring break. This was particularly challenging for implementing the strategy 2 and 3 for teaching Arduino programming. In the available 6-week time frame, students were encouraged to build a microcontroller-based project, that can be deployed in real-time. The recorded online lectures served as modules for each design phase of the

real-time microcontroller-based project. The challenges, changes and the impact on students measured attributes such as project demonstrations, and surveys are detailed below:

Challenges: 1. Until before the spring break, in the face-to-face classroom setting, during the MATLAB and Python lectures, students were encouraged to implement simple programs as part of the "hands-on" session. This helped the Instructor and Teaching Assistant to help students in debugging any errors. In the online sessions, it was difficult to address all the issues arising in Arduino, as this was a class of 107 students.
2. The project groups were unable to get the desired list of hardware components due to the delay in shipping because of COVID-19.
3. The Arduino IDE tool could not be used for teaching, as many students did not have the hardware i.e. Arduino Uno board.

Changes in the Instructional Material: The challenges related to the hardware availability was assessed in the first week of online class through a short survey. Based on which the following changes were incorporated in the course delivery:

1. A "flipped-classroom" technique was implemented for the Arduino modules. By the week 2 of the online class, the Instructor had recorded short lectures which can help students design the microcontroller-based project through "divide and conquer" approach. The 3-hour class meeting time was dedicated to address any of the debugging issues or questions/concerns that the students had. This helped students to review the lectures at their own pace and spend enough time in debugging during the following weeks.
2. As the hardware components were not available for many students, an online Arduino emulator, TINKERCAD, was used for live demonstrations. This tool helped in emulating the sensors and Arduino board along with the resistors, LCDs and LEDs, that can be used for implementing real-time monitoring frameworks.
3. Some example monitoring systems were designed using the TINKERCAD tool in the lecture videos to help students understand and implement the same.

Impact on students: In the first week of the online class, students were concerned with the final project, as it was contributing to 30% of the total grade. With the help of recorded lectures and the TINKERCAD tools, students were enthusiastic towards developing creative final projects, as the learning outcomes was directly related to the project implementation. As part of the final demonstration, student groups wrote interesting blogs with videos and tutorials in LinkedIn and their personal websites, which showed their enthusiasm and engagement in learning arduino programming. The details of the project challenge statement and example projects are explained in the following subsection.

Assessments and Evaluations

The active learning strategies were integrated as part of all the assessments included in the entire semester. The active learning strategies integrated as part of the curriculum is listed as strategies in the previous section. The following results are based on the assessments included as part of each strategy.

Strategy-1: ConcepTests: In the very first class of this semester, a survey was conducted to analyze the familiarity of the programming knowledge of the students. Figure 1 shows the results of this survey. To improve the learning outcomes, students were given quiz every week in the beginning of the semester for MATLAB lecture materials. This helped them to stay updated on the lecture content delivered in the previous class. For Python and Arduino, the quizzes were placed towards the end of week 3 of the respective programming language. Figure 3 shows the trend in quizzes.

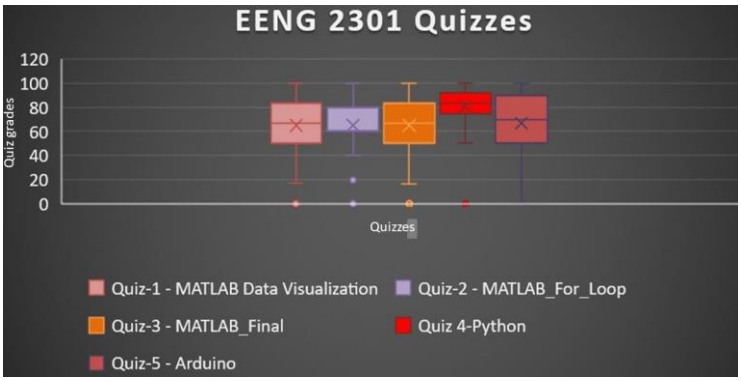

Fig. 3: Box and Whisker Plot of EENG 2301 Quizzes

As seen in this box and whisker plot, compared to the first quiz (Quiz-1 MATLAB Data Visualization), the overall median in the class was improved towards the MATLAB final quiz. In this figure, X axis represents the quizzes and Y axis represents the student's quiz grades. Comparing the final quiz of MATLAB, Python and Arduino, it can be noticed that the overall class average improved and there were more students towards 60% and above.

Note: To be consistent in the analysis, this plot includes grades of students who decided to drop or stopped attending after the COVID situation.

Strategy-2: Think-Pair Share: In the mini projects, students were assigned into smaller groups. Though the initial group assignment was random, efforts were taken to divide them into groups such as students with less familiarity in programming were partnered with students with intermediate or highly skilled students. Every alternate class, students were given 45 minutes towards end of the class to discuss with their groups and work on the projects. Many students consistently performed well in the projects and some of them remained in the same group for their final projects. Figure 4 shows a trend in scores for mini projects. In this figure, X axis represents the mini projects and Y axis represents the mini project grades. It can be observed that students were able to achieve the design objectives and specifications for each project and the overall class average improved towards the third mini project with Python.

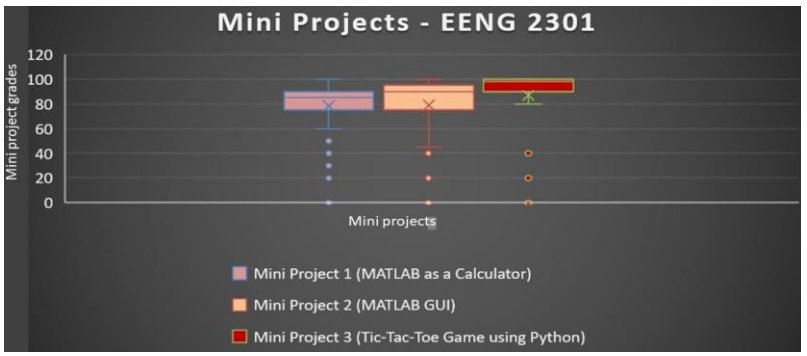

Fig. 4: Mini-Projects Analysis

Strategy-3: Problem-solving through demonstrations, proofs, and stories: The final group project that involved designing an Internet of Things (IoT)-based monitoring framework by programming an Arduino Uno R3 board, included computational thinking cornerstones in each design phase. Figure 5 shows the steps involved in designing the final group project using the cornerstones in computational thinking.

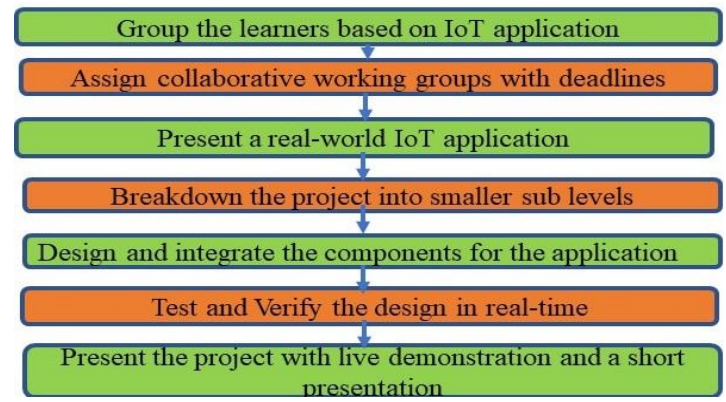

Fig. 5: Computational Thinking steps integrated as part of the Arduino-based IoT project

The details of the final project implementation are given as follows:

Challenge statement: Smart security systems help in establishing a surveillance around the area of interest, where the surveillance footage can be remotely monitored using a mobile application or user interface. Such security systems come with professional installations and a subscription fee, where a team of professionals will monitor the alerts triggered by the sensors. As an engineer, your challenge is to design a customized security system with the following design specifications. Use the concepts taught in the "IoT and Robotics" lecture to build a smart security system.

Device Specifications: The smart security system should meet the following specifications: 1. The device can be a stationary system with multiple subsystems or a moving robot. 2. The device should have at least one Arduino board integrated with required sensors 3. In case of security breach, the system should be able to alert the user within 2 minutes. 4. The data collected from the

device should be shown to the user through a LCD screen or a user interface such as mobile application or webpage.

Results: Students came up with creative systems based on Arduino Uno and used sensors such as temperature sensor, distance sensor, light intensity sensor, etc., and servo motors (for robotics). Due to the pandemic, students were allowed to implement the project on their own or form groups of up to 5 students. The class of 107 students, divided themselves into 42 project groups with group members ranging from 1-5 students. The sample projects included: Automatic pet feeder, digital piano, sea-roomba, robotic car, automatic coffee maker, Morse code detector, Burglar alarm, music carousel, automatic MP3 player, and so on. As part of this final project, they were asked to showcase their projects in LinkedIn through a Blog, video or post. Figure 6 shows the example projects implemented. Figure 6 a. shows the final prototype of granted access only doorway system, and Figure 6 c. shows the prototype of simple robot with echolocation capability.

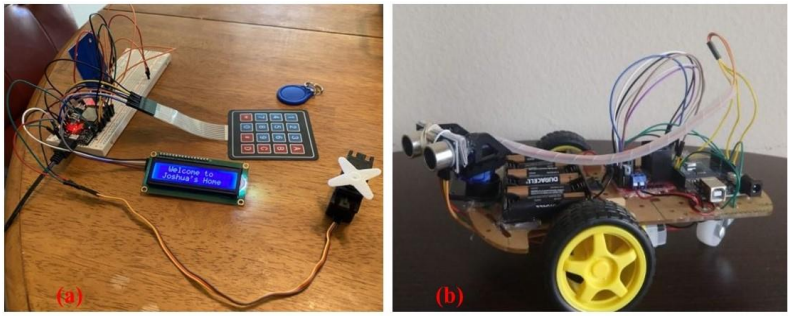

Fig. 6: a. Smart lock with limited granted access[9], b. simple robot with echolocation capability[10].

Summary and Conclusions

Integrating Computational Thinking aspects in teaching a programming language helped students to think beyond a given programming language or a project. To explore different ways to promote active engagement in an interdisciplinary class, three active learning techniques were explored as strategies. It was observed that when students were given enough justification and presented with a real-world problem, they made an extra effort to learn the concepts. This was evident through the "Course certifications" they obtained in one of the three programming languages from any MOOC platform for extra-credit. 39 students, i.e. 36% of the class, pursued a "Course certification" for extra credit. Though there were some limitations in implementing the active learning techniques for the online class, with the help of projects and relevant online tools such as TINKERCAD, students were more enthusiastic towards the end of final project implementation. Future plans include conducting a mini conference as part of the final presentation to include faculties from the College of Engineering. This might be a preparation step for students before they go for senior design.

Acknowledgments

The course development was supported by the "Teaching and Learning" award provided by the Center of Excellence in Teaching and Learning (CETL) at UT Tyler. The author would like to acknowledge the support received from the Department of Electrical Engineering to successfully implement the proposed strategies.

References

1. "What is Industry 4.0 – the Industrial Internet of Things (IIoT)?", EPICOR : https://www.epicor.com/en-us/resource-center/articles/what-is-industry-4-0/ Last accessed on May 11, 2020.
2. Weintrop, D., and Wilensky, U., 2017, "Comparing block-based and text-based programming in high school computer science classrooms." ACM Transactions on Computing Education (TOCE) 18, no. 1 (2017): 1-25.
3. Kerns. J., "Do Mechanical Engineers Need Programming to Survive?", Blog, Machine Design, https://www.machinedesign.com/community/editorial-comment/article/21835239/do-mechanical-engineers-need-programming-to-survive.
4. UT Tyler Strategic Plan, 2020 - https://www.uttyler.edu/plan/ , Last accessed on August 3, 2020.
5. Czerkawski. B., 2015, "Instructional Design for Computational Thinking, 2015.
6. Valenzuela. J., 2020, "How to develop computational thinkers", https://www.iste.org/explore/Computational-Thinking/How-to-develop-computational-thinkers , Last accessed on August 3, 2020.
7. Forms of Project-Based Learning, 2020 - http://pdcenter.pdesas.org/CourseRendering/CourseContent/Render/01003303004901002200601100407800018604 6103068043, Last visited on August 3, 2020.
8. Yang, D., Swasnon, S. R., Bhaskar, B. C., Back, Y., 2018, "Work in Progress: Integrating Computational Thinking in STEM Education through a Project-based Learning Approach", Proceedings of the ASEE Annual Conference and Exposition, 2018.
9. Butler, J., 2021, "Smart lock with limited grant access", Project Showcase, Programming Languages Course, EE, UT Tyler, https://www.linkedin.com/pulse/granted-access-only-project-josh-butler/ , Last accessed on January 30, 2021.
10. Alvarez, M., 2021, "Simple robot with echolocation Capability, Programming Languages Course, EE, UT Tyler, https://www.linkedin.com/pulse/building-simple-robot-echolocation-capability-martin-morales-alvarez/?trackingId=UVgxqmk8TPyNGryAB8czXw%3D%3D Last accessed on January 30, 2021.

PRABHA SUNDARAVADIVEL
Dr. Sundaravadivel currently serves as Assistant Professor in the Electrical Engineering Department at the University of Texas at Tyler. She is is a certified team-based learning practitioner. Her research interests include developing consumer electronic systems for smart cities and smart healthcare, affective computing for healthcare, and reconfigurable systems for IoT applications.