

Board 322: Integrating Internet of Things into Mechatronics to Prepare Mechanical Engineering Students for Industry 4.0

Dr. Hakan Gurocak, Washington State University, Vancouver

Prof. Gurocak is the founding director of the Professional and Corporate Education program at Washington State University Vancouver. His research interests include haptics, robotics and automation.

Dr. Xinghui Zhao, Washington State University, Vancouver

Dr. Xinghui Zhao is the Director of the School of Engineering and Computer Science, and Associate Professor of Computer Science at Washington State University Vancouver. She received her Ph.D. from Department of Computer Science at the University of Saskatchewan in 2012. She previously received an M.Sc. from the same university, and a B.Sc. from Department of Computer Science, Nanjing University. Dr. Zhao's research interests lie in the general areas of parallel and distributed systems, big data computing, cloud computing, and machine learning. Dr. Zhao is a member of IEEE, ACM, ASEE, and IEEE Women in Engineering, and has been actively contributing to the professional community. She served as the general chair for the 15th IEEE/ACM International Conference on Utility and Cloud Computing (UCC2022) and the 9th IEEE/ACM International Conference on Big Data Computing, Applications and Technologies (BDCAT2022). She also served as the local arrangement chair for IEEE CLUSTER 2021. She was the guest editor for Special Issue on Integration of Cloud, IoT and Big Data Analytics, Software: Practice and Experience (Wiley Press). In addition, she has served on the technical program committee for a number of conferences, and as reviewer for various journals.

Dr. Kristin Lesseig

Integrating Internet of Things into Mechatronics to Prepare Mechanical Engineering Students for Industry 4.0

Abstract

The Internet of Things (IoT) technologies can enable products to become smarter through sensing their environment, analyzing lots of data (big data), and connecting to the Internet to allow for the exchange of data. As smart products become ubiquitous, they provide enormous opportunities for scientists and engineers to invent new products and build interconnected systems of vast scale. As a result, the STEM workforce demands are shifting rapidly. Mechanical engineers will play a significant role in innovating and designing smart products and manufacturing systems of the Industry 4.0 revolution. However, the current mechanical engineering curriculum has not kept pace. In this paper, we present an overview of a new curriculum along with the design of an inexpensive smart flowerpot device that was used as an instructional tool throughout the curriculum. We provide details about how two curriculum modules were implemented in the first offering of the course. Preliminary assessment results from the first offering of the course are discussed.

1 Introduction

Smart products can sense their environment, analyze lots of data (big data), and connect to the Internet and other smart products over a network to allow exchanging data. Today, there are many consumer smart products in our lives such as smart door locks, bike locks, smart kitchen appliances, irrigation controllers, smart thermostats (e.g. Nest), and Amazon Echo just to name a few. Such physical objects (things) connected to the Internet is called the Internet of Things (IoT) [1].

Smart products are becoming ubiquitous and STEM workforce demands are shifting rapidly, but the current mechanical engineering curriculum at Washington State University Vancouver and elsewhere has not kept pace. A recent job search [2] showed that as the products become smarter, employers are looking for engineers with additional skills. Here are some excerpts from the job ads that came up in our search: knowledge and experience in integration of IoT into vehicle monitoring systems; understanding of the software development process; programming in Python; ability to integrate electromechanical systems; design, build, test smart product prototypes; participate in cross-functional development teams; and experience in using cloud-based platforms.

Mechanical engineers will play a significant role in innovating and designing smart products and manufacturing systems that are driving the Industry 4.0 revolution for smart factories [3–5]. The mechanical engineer of the future needs the same foundation of technical skills and ability to solve problems as always. But additional skills are needed to participate in the IoT revolution. Thus, preparing mechanical engineering students to contribute in this new field is a pressing educational need.

To meet this need, we developed a new modernized mechatronics course that focuses on the IoT technologies, and incorporates project-based learning (PjBL) as well as software engineering methods from computer science. Our overarching goal is to integrate skills from computer science and mechanical engineering, and bridge the gap in mechanical engineering curriculum to better prepare future students for the Industry 4.0 revolution.

We are building on prior work by others using active learning [6–11], PjBL [12–17], worked examples [18–20], Jupyter notebooks [21, 22], agile software development methods [23–25], as well as existing IoT course materials [26–44]. However, the existing mechatronics course materials with IoT tend to target Electrical Engineering (EE) and Computer Science (CS) students and the *creation* of underlying IoT technologies, especially low-level software. Mechanical engineers need to develop smart products and systems for Industry 4.0 through *integration* of the IoT technologies *not* creation of them. Thus, we kept this important distinction front and center in our curriculum. Another unique feature is the use of a formal software engineering methodology by Mechanical Engineering (ME) students to develop high quality code.

In this paper, we present an overview of the curriculum for the new course. We provide details of two of the course modules along with preliminary assessment results. In addition, we share the design of an inexpensive smart flowerpot device that is used as an instructional tool throughout the curriculum.

2 Overview of the new curriculum

Our mechanical engineering program at WSU Vancouver has a senior-level elective course on microcontrollers. This course is part of a 3-course sequence in the mechatronics option track. It is a 3-credit semester course with two 75-minute lectures per week. The course attracts students from mechanical and electrical engineering programs with a typical enrollment of 25-30 students. In Spring 2023, the new curriculum was offered in this course to ultimately replace the existing course with a new IoT course in the mechatronics track.

The curriculum contains 10 weeks of instructional material organized into five modules. The last 5 weeks constitute the class project phase where student teams develop smart products they proposed.

Module 1: Overview of Python - (3 weeks) This is an introductory review of Python programming language. This module reviews data types, strings, lists, dictionaries, loops, conditionals and functions.

Module 2: Data Collection - (2 weeks) This module examines interfacing sensors and actuators to the microcontroller (Raspberry Pi) with the aim of explaining how a typical mechatronic system

is designed. Circuit diagrams are presented for each type of device and code segments are given for hands-on demonstrations.

Module 3: Data Transmission and Processing - (2 weeks) This module starts with an overview of cloud computing. Then, programming details on how to retrieve weather forecast data from the National Oceanic and Atmospheric Administration (NOAA) servers are presented. Finally, two introductory machine learning techniques are discussed for data processing.

Module 4: Data Transmission and User Interfaces - (2 weeks) This module starts with an overview of MQTT protocol for network communications. Then, programming details of how to build a remote user interface with gauges, digital displays, and buttons are presented for real-time display of data transmitted over the Internet from a smart device.

Module 5: Software Engineering - (1 week) This module starts with an overview of the software process models. Agile software development method is introduced. The module concludes with version control technologies.

Final class project - (5 weeks) Students work in small teams and propose a smart product to build as their class project. The project requires interfacing sensors and actuators to the Raspberry Pi and code development following the agile software development method. At the end, student teams present their project to the class.

2.1 Design elements of the course

Instructional design - The instructional design of the course incorporates (1) Active learning through live coding with Jupyter notebooks, (2) Worked-examples, and (3) Agile software engineering methodology for robust code development.

Active learning increases student success in STEM [6] and leads to increased student retention and engagement [7–11]. We use *Jupyter notebooks*[45] to implement active learning. A Jupyter notebook is a free, open-source, web application that allows students to create and share documents containing live code, equations, visualizations and narrative text.

Worked examples lead to significant learning improvements compared to problem-solving with no guidance [18–20]. Our learning modules follow this design approach with lessons incorporating step-by-step worked examples to improve learning.

Developing high quality code is challenging, especially for non-CS majors [24]. The mechanical engineering students tend to use an ad hoc approach in code development. The Agile method is systematic and used often by the rapidly growing and volatile Internet software industry [25]. Student teams incorporate this approach throughout the class project in the last 5 weeks of the course.

Project-based learning to frame the curriculum and instruction - In project-based learning (PjBL), students learn the course material from completing a project, which contains and frames the curriculum and instruction [12]. PjBL has been shown to be significantly more effective in engineering education and in mechatronics courses [13–17]. We chose a *smart flower pot* to be



Figure 1: Smart flower pot. It can connect to a cloud service to retrieve 5-day weather forecast for the location of the pot, measure various things using its sensors and adjust its actions based on the forecast to periodically rotate the plant and deliver just the right amount of water to keep it alive.

used as the PjBL platform to deliver the instructional content (modules 1-4) of the new course.

Smart flower pot - The system consists of a flower pot on a motorized rotating base platform (Figure 1). The clear plastic bottom section of the pot is a water reservoir with a submersed pump. The white plastic top part is where a plant can be placed. The smart flower pot contains a light sensor to measure the amount of light the plant receives. It also has sensors to measure the soil moisture, water level in the reservoir and temperature, and humidity sensors for ambient air. All of the electronic components, wiring, and a Raspberry Pi are housed under the metal pan at the base of the flower pot. Each flowerpot is connected to a monitor, keyboard, and mouse to construct a workstation in the computer lab. The smart flower pot was custom designed and built. Excluding the Raspberry Pi , the rest of the hardware costs about \$200 per pot.

The smart flower pot can connect to a cloud service to retrieve 5-day weather forecast for the location of the pot, measure various things using its sensors, and adjust its actions based on the forecast to periodically rotate the plant and deliver just the right amount of water to keep it alive. Its functions can be monitored over the Internet using a remote dashboard with gauges, digital displays and trend charts.

3 First offering of the course

As mentioned before, we piloted these materials in an elective course for electrical and mechanical engineers. In Spring 2023, the course consisted of 20 students (17 seniors and 3

juniors). None of the students had prior experience with Python, but all had some programming experience. At the time of this writing, students have finished modules 1 and 2.

Module 1 - Lectures for the first module for overview of Python were held at a regular department computer lab. Jupyter Notebook was used to ultimately provide lecture notes to the students. In each lecture, students started from an *initial notebook* that contained just text explaining concepts as shown in Figure 2a. There were no code examples in this initial notebook. Each student was sitting at a computer with this notebook open on their screen. The instructor's notebook was shown on the projector screen. As the instructor explained concepts, code examples were added to the notebook as shown in Figure 2b. Students were typing these examples into their own notebooks along with the instructor and running them. If there were any mistakes, they got immediate feedback from the Jupyter notebook. The active engagement in the lecture generated lots of questions from the students.

Each notebook also contained several sections called "Your Turn" with questions for the students to work on (Figure 2a). When the lecture reached a Your Turn section, the instructor paused the lecture for a while and allowed the students to work on the problem on their own. Students were entering Python code directly into the notebook. Again, lots of interaction took place with the instructor and among the students. Then, the instructor showed the solution to the class and explained the details. The lecture resumed with the next topic in the notebook following the same approach. At the end of each notebook was a section called "IoT Example." This section had problems to show how the programming concepts they just learned are applied to real-life IoT programming situations. Again, the students first worked on these problems for a short while on their own, then the solutions were explained. After the lecture was over, the instructor posted a complete notebook with text, sample code segments, and solutions for the Your Turn and IoT Examples sections. This gave the students a complete set of lecture notes after each lecture.

Module 2 - This module examined interfacing sensors and actuators to the Raspberry Pi to explore how a typical mechatronic system was designed. Starting with this module, the class moved to another lab where 10 stations with the smart flower pots were available. During each lecture, two students shared one station (Figure 3).

Jupyter Notebook can run on the Raspberry Pi, however it is relatively slow. Therefore, these lectures used a Python App for coding that came with the Raspberry Pi instead. The lectures were presented using PowerPoint slides that contain the "Your Turn" sections. Same approach as in Module 1 was used to allow the students to work on these problems on their own for a while. But this time, they were using the smart flower pot hardware to test their codes. While the students were working on the code, the instructor walked around the classroom to interact with the students at their stations and helped them as needed. The lecture resumed with the next topic in the slides. Once again, a very active lecture environment was generated with this approach. After the lecture, the instructor posted complete PowerPoint slides and files containing Python code.


4 Preliminary Results

Practice problems were assigned weekly, but were not graded. Instead, students were provided with solution files as well as recommendations for how to use the problems to enhance their

Dictionaries

A dictionary is a collection of items. Each item is a key:value pair. Unlike a variable that can hold one data value (e.g. x=5), a dictionary can contain many different types of key:value pairs.

- A collection of items
- Each item contains a key:value pair
- Each key must be unique (cannot be repeated)
- A dictionary is defined by using { }
- Key:value pairs are separated by commas
- A key can be a string or a number

 **Your Turn**

Define a dictionary of items for 4 fruits and their colors. Print the dictionary.

[]:

(a) Initial notebook given to students without any Python code.

Dictionaries

A dictionary is a collection of items. Each item is a key:value pair. Unlike a variable that can hold one data value (e.g. x=5), a dictionary can contain many different types of key:value pairs.

- A collection of items
- Each item contains a key:value pair
- Each key must be unique (cannot be repeated)
- A dictionary is defined by using { }
- Key:value pairs are separated by commas
- A key can be a string or a number

```
[1]: dict = {"name":"John", "lastname":"Smith", "age":35}
print(dict)
{'name': 'John', 'lastname': 'Smith', 'age': 35}
```

In the above example, "name", "lastname" and "age" are the keys. All keys in this example are strings. Below is an example where the keys are numbers. Also, when a dictionary is created, the data type for it will be a dictionary (dict) object:

```
[2]: cars = {1:"Ford", 2:"Toyota", 3:"Honda"}
type(cars)
[2]: dict
```

```
[3]: car = {
    "make" : "Toyota",
    "model" : "RAV4",
    "year" : 2011,
    "cyl" : 6,
    "colors": ["red", "white", "black"] # a List is used as a value for the "colors" key
}
```

(b) Python code typed into the notebook during the lecture to complete it.

Figure 2: Example Jupyter notebook for Module 1.



Figure 3: Lab stations with smart flower pot. Each station is shared by two student during the lectures for modules 2-4.

learning and confidence in Python programming. At the end of each course module, students completed a quiz containing exercises similar to the assigned practice problems and a survey to gather students' perceptions of their own learning and feedback on the course delivery.

4.1 Module quizzes

At the time of this writing, students had just finished the quiz on Module 1. Students were given 1 hour to complete the quiz, which contained 12 programming questions that spanned skills from all major topics in the module leading to a maximum score of 36. Students with the lowest three scores indicated they ran out of time. As shown in Figure 4, if we consider all scores, the students did well overall (median = 33, st. dev. 4.8). If the lowest three scores are excluded, the median improves even more, as expected (median = 34.5, st. dev. 3.6).

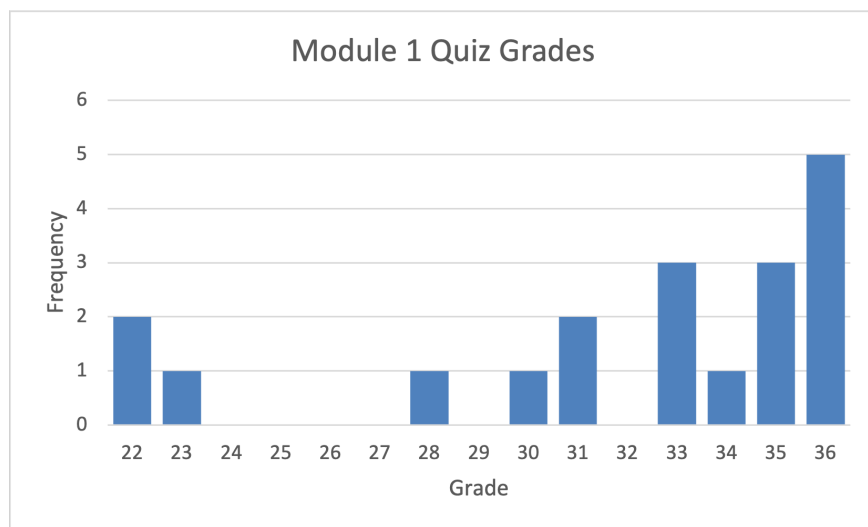


Figure 4: Distribution of Quiz 1 grades after completing module 1. Maximum possible score was 36.

4.2 Module survey

Analysis of the Module 1 survey revealed that the overwhelming majority of students felt very confident in their coding abilities and were generally satisfied with the course delivery to date. The survey contained 8 statements about specific Python coding skills such as “I know fundamental data types in Python” or “I can build functions with parameters and return values.” Students ranked their agreement with the statements on a 5-point Likert scale. Across all items, over 70% of students chose positive ratings, selecting either “somewhat agree” or “strongly agree,” and no students selected “strongly disagree.” The highest ratings were given to items describing general understandings of Jupyter notebooks or logic structures. For example, over 63% (12/19) students strongly agreed with the statement “I can construct conditional statements using IF... THEN structures.” This is perhaps not surprising given that these skills live outside of the particulars of Python, which was a new programming language for the majority of students. Skills that were more specific to the precise syntax and language of Python were ranked lower. For example, students expressed the least confidence in the statement “I can manipulate dictionaries to add/remove items, retrieve values,” with only 6 (of 19) students saying they strongly agree, 11 somewhat agree, and 2 somewhat disagree. The statement “I can manipulate strings using library methods” received almost identical ratings with 7, 11, 1 student in the respective categories.

The survey also provided an opportunity for students to reflect on their learning through a series of open response items. Students were asked to comment on how the method of instruction helped them remember key concepts as well as how the use of Jupyter Notebooks and practice assignments affected the way they learned or studied. Students appreciated the examples presented in class as well as the hands-on opportunities to try problems on their own. A number of students also commented on the flexibility of the practice assignments and appreciated the fact that they could work through problems at their own pace and complete them at home, versus in the computer lab. In short, the students felt the course was taught in a way that supported their learning. This positive sentiment is captured in these final comments by two of the students: “Having to spend time figuring out a method to solve a problem is my ideal way of learning, rather than following instructions” and “I can write my own code and run it immediately, then I can find out where my errors are.”

5 Conclusions

In this paper, a new curriculum for an Internet of Things (IoT) course was presented. The course was designed to bridge a gap in STEM education, specifically in mechanical engineering, to better prepare future students for the Industry 4.0 revolution and for smart product design. The new curriculum focuses on the IoT technologies and brings software engineering methods from computer science into mechanical engineering.

At the time of this writing, the course was being offered for the first time. A smart flower pot was custom designed as a platform to be used throughout the course so students could gain hands-on experience with the IoT technologies. The smart flowerpot can connect to a cloud service to

retrieve 5-day weather forecast for the location of the pot, measure various things using its sensors, and adjust its actions based on the forecast to periodically rotate the plant and deliver just the right amount of water to keep it alive. The flowerpot functions can be monitored over the Internet using a remote dashboard with gauges, digital displays and trend charts.

Results for the first module, an overview of Python programming language, are encouraging. Majority of the students could demonstrate their skills on a module quiz, which led to a grade distribution with median = 33, st. dev. 4.8 where maximum possible score was 36. Survey results indicate overall satisfaction with the use of the Jupyter notebooks, the active learning environment during the lectures, and the practice assignments that supplemented the in-class activities.

Many universities have mechatronics courses, which can be replaced by the new course. Coupled with the inexpensive hardware and the free open-source software, the products of this work can easily be transferred to other institutions increasing the potential for high impact in STEM education.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. DUE-IUSE-2116226. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

- [1] Autodesk Inc., “The Essentials of IoT for Modern Engineers,” <https://www.autodesk.com/industry/manufacturing/resources/mechanical-engineer/iot-internet-of-things-essentials-for-engineers>, 2016.
- [2] Indeed.com, “Engineering jobs search with IoT keyword,” January 2023.
- [3] Forbes.com, “What is Industry 4.0? Here’s A Super Easy Explanation For Anyone,” <https://www.forbes.com/sites/bernardmarr/2018/09/02/what-is-industry-4-0-heres-a-super-easy-explanation-for-anyone/#430ff4a59788>, September 2018.
- [4] D. Bradley, D. Russell, I. Ferguson, J. Isaacs, A. MacLeod, and R. White, “The internet of things – the future or the end of mechatronics,” *Mechatronics*, vol. 27, pp. 57 – 74, 2015.
- [5] P. Eichinger, B. Hofig, and C. Richter, “Education 4.0 for mechatronics – agile and smart,”

- in *2017 International Conference on Research and Education in Mechatronics (REM)*, 2017, pp. 1–7.
- [6] S. Freeman, S. L. Eddy, M. McDonough, M. K. Smith, N. Okoroafor, H. Jordt, and M. P. Wenderoth, “Active learning increases student performance in science, engineering, and mathematics,” *Proceedings of the National Academy of Sciences*, vol. 111, no. 23, pp. 8410–8415, 2014.
- [7] J. Gao and J. Hargis, “Promoting technology-assisted active learning in computer science education,” *The Journal of Effective Teaching*, vol. 10, no. 2, 2010.
- [8] C. Martínez and M. Muñoz, “ADPT: An active learning method for a programming lab course.” *Proceedings of the 10th International CDIO Conference*, Universitat Politècnica de Catalunya, Barcelona, Spain., 2014.
- [9] P. Seeling, “Active learning moves programming students from novice to skilled,” <https://www.pearsoned.com/active-learning-programming-skilled/>, 2016.
- [10] O. Mironova, I. Amitan, J. Vilipõld, and M. Saar, “Active learning methods in programming for non-IT students.” *International Conference e-Learning*, 2016.
- [11] B. Gottfried, “Teaching computer programming effectively using active learning.” *ASEE Annual Conference and Exposition*, 1997.
- [12] J. Larmer and J. Mergendoller, “The main course, not dessert,” Buck Institute (www.bie.org), 2011.
- [13] Y. Wang, Y. Yu, H. Wiedmann, N. Xie, C. Xie, W. Jiang, and X. Feng, “Project-based learning in mechatronics education in close collaboration with industrial: Methodologies, examples and experiences,” *Mechatronics*, vol. 22, pp. 862–869, 2012.
- [14] N. L. Toner and G. B. King, “Restructuring an undergraduate mechatronic systems curriculum around the flipped classroom, projects, LabVIEW, and the myRIO.” *Boston: American Control Conference (ACC)*, July 6-8, 2016.
- [15] S. Chandrasekaran and J. M. Long and M. A. Joordens, “Evaluation of student learning outcomes in fourth year engineering mechatronics through design based learning curriculum.” *IEEE Frontiers in Education Conference (FIE)*, 2015.
- [16] J. Mynderse and J. Shelton, “Assessment of an improved problem-based learning implementation in a senior/graduate mechatronic design course.” *ASEE Annual Conference and Exposition*, 2015.
- [17] K. Meah, “First-time experience of teaching a project-based mechatronics course.” *ASEE Annual Conference and Exposition*, 2016.
- [18] T. van Gog, L. Kester, and F. Paas, “Effects of worked examples, example-problem, and problem-example pairs on novices’ learning,” *Contemporary Educational Psychology*, vol. 36, no. 3, pp. 212 – 218, 2011.
- [19] O. Chen, S. Kalyuga, and J. Sweller, “The worked example effect, the generation effect, and

- element interactivity,” *Journal of Educational Psychology*, vol. 107, no. 3, pp. 689–704, 2015.
- [20] F. Paas and T. van Gog, “Optimized worked example instruction: Different ways to increase germane cognitive load,” *Learning and Instruction*, vol. 16, pp. 87–91, 2006.
- [21] R. W. Krauss and A. Ali, “Teaching dynamic systems and control without dynamics.” ASEE Annual Conference and Exposition, 2017.
- [22] M. Borowczak and A. C. Burrows, “Interactive web notebooks using the cloud to enable CS in K-16+ Classrooms and PDs.” ASEE Annual Conference and Exposition, 2017.
- [23] R. C. Martin, *Agile Software Development: Principles, Patterns, and Practices*. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2003.
- [24] M. Guzdial and A. Forte, “Design Process for a Non-majors Computing Course,” in *Proceedings of the 36th SIGCSE Technical Symposium on Computer Science Education*, ser. SIGCSE '05, 2005, pp. 361–365.
- [25] P. Abrahamsson, O. Salo, J. Ronkainen, and J. Warsta, “Agile Software Development Methods: Review and Analysis,” *CoRR*, vol. abs/1709.08439, 2017.
- [26] J. He, D. C.-T. Lo, Y. Xie, and J. Lartigue, “Integrating Internet of Things (IoT) into STEM undergraduate education: Case study of a modern technology infused courseware for embedded system course.” IEEE Frontiers in Education Conference (FIE), 2016.
- [27] D. V. Gadre, R. S. Gaonkar, S. N. Ved, and N. Prasannakumar, “Embedded systems and Internet of Things (IoTs) - challenges in teaching the ARM controller in the classroom.” ASEE Annual Conference and Exposition, 2017.
- [28] J. O. Hamblen and G. M. E. van Bekkum, “An embedded systems laboratory to support rapid prototyping of robotics and the Internet of Things,” *IEEE Transactions on Education*, vol. 56, no. 1, 2013.
- [29] X. Zhong and Y. Liang, “Raspberry Pi: An effective vehicle in teaching the Internet of Things in computer science and engineering,” *Electronics*, vol. 5, no. 56, 2016.
- [30] V. Galluzzi, C. A. Berry, and Y. Shibberu, “A multidisciplinary pilot course on the Internet of Things: Curriculum development using lean startup principles.” ASEE Annual Conference and Exposition, 2017.
- [31] L. O. Kehinde, O. T. Ayodele, O. O. Akintade, and K. O. Olawale, “Development of a module to teach basic concepts of interfacing and connectivity in Internet of Things.” ASEE Annual Conference and Exposition, 2016.
- [32] S. Abraham and A. Miguel, “Creation of an Internet of Things (IoT)-based innovation lab.” ASEE Annual Conference and Exposition, 2017.
- [33] P. Bisták, R. Moezzi, F. Semeraro, and G. Nicassio, “Teaching IoT Using Raspberry Pi Based RC-Car,” 2020, pp. 1–6.

- [34] S. J. Dickerson, "Preparing undergraduate engineering students for the Internet of Things." ASEE Annual Conference and Exposition, 2016.
- [35] D. Guo and A. Koufakou, "How cloud computing is addressed for software development in computer science education." Human-Computer Interaction. User Interface Design, Development and Multimodality, 2017.
- [36] C. Rennick, C. Hulls, D. Wright, A. J. B. Milne, E. Li, and S. Bedi, "Engineering design days: Engaging students with authentic problem-solving in an academic hackathon." ASEE Annual Conference and Exposition, 2018.
- [37] D. J. Orser, K. Bazargan, and J. Sartori, "Harnessing state-of-the-art Internet of Things labs to motivate first-year electrical and computer engineering students." ASEE Annual Conference and Exposition, 2018.
- [38] S. J. Dickerson, "Introducing the Internet-of-Things to the next generation of engineers." ASEE Annual Conference and Exposition, 2017.
- [39] S. Artis and G. N. Washington, "Design, code, build, test: Development of an experiential learning summer engineering and computer science outreach program for high school students (evaluation)." ASEE Annual Conference and Exposition, 2017.
- [40] J. R. Porter, J. A. Morgan, and M. Johnson, "Building automation and IoT as a platform for introducing STEM education in K-12." ASEE Annual Conference and Exposition, 2017.
- [41] J. A. Morgan, J. R. Porter, and M. Johnson, "IoT-based building automation and energy management." ASEE Annual Conference and Exposition, 2018.
- [42] D. Turgut, L. Massi, S. S. Bacanli, and N. H. Bidoki, "Multidisciplinary undergraduate research experience in the Internet of Things: Student outcomes, faculty perceptions, and lessons learned." ASEE Annual Conference and Exposition, 2017.
- [43] J. Frochte, M. Lemmen, and M. Schmidt, "Seamless integration of machine learning contents in mechatronics curricula," in *2018 19th International Conference on Research and Education in Mechatronics (REM)*, 2018, pp. 75–80.
- [44] W. Yu, O. Farook, J. P. Agrawal, and A. Ahmed, "Software-hardware integration of system design discipline." ASEE Annual Conference and Exposition, 2018.
- [45] Jupyter.org, "Jupyter Notebook," <https://jupyter.org/index.html>, June 2019.