Integrating Image, Video, and Machine Learning into an IoT Learning Environment

Dr. David Hicks, Texas A&M University-Kingsville

David Hicks is an Associate Professor in the Electrical Engineering and Computer Science Department at Texas A&M University-Kingsville. Before joining TAMUK he served as Associate Professor and Department Head at Aalborg University in Esbjerg, Denmark. He has also held positions in research labs in the U.S. as well as Europe, and spent time as a research scientist in the software industry.

Dr. Lifford McLauchlan, Texas A&M University - Kingsville

Dr. Lifford McLauchlan is an Associate Professor in the Electrical Engineering and Computer Science Department at Texas A&M University - Kingsville, and has also worked for Raytheon, Microvision, AT&T Bell Labs, and as an ONR Distinguished Summer Faculty at SPAWAR San Diego, CA. He has over 55 publications covering areas such as adaptive and intelligent controls, robotics, an ocean wave energy converter, green technology, education, wireless sensor networks and image processing. He is a co-inventor on 3 US patents related to control systems. Dr. McLauchlan is a member of ASEE and was the 2012-2014 Chair of the Ocean and Marine Engineering Division. He is also a member of IEEE (senior member), SPIE, Eta Kappa Nu, ACES and Tau Beta Pi, and has served on the IEEE Corpus Christi Section Board in various capacities such as Chair, Vice Chair, Secretary and Membership Development Officer. Dr. McLauchlan has received the Dean's Distinguished Service Award twice and the Dean's Outstanding Teaching Award once for the College of Engineering at Texas A&M University-Kingsville.

Dr. Mehrube Mehrubeoglu, Texas A&M University - Corpus Christi

Dr. Mehrubeoglu received her B.S. degree in Electrical Engineering from The University of Texas at Austin. She earned an M.S. degree in Bioengineering and Ph.D. degree in Electrical Engineering from Texas A&M University. She is currently an associate prof

Integrating Image, Video, and Machine Learning into an IoT Learning Environment

Keywords: Internet of Things, IoT, Engaged Remote Student Learning, Image Recognition, Machine Learning

Introduction

Internet of Things (IoT) technology continues to enable the design and development of a variety of new types of applications and systems [3]. The advanced capabilities found in a modern smart home provide numerous examples including the remote and automated management and control of thermostats, lighting systems, and security devices [12]. More broadly, the data gathering and control capabilities of IoT technology is also helping to fuel the development of significant improvements of existing larger scale infrastructure related systems for important tasks such as traffic management (smart signals) and power distribution (smart grids). As a result, the importance of teaching IoT related concepts and technology to students in computer science, electrical engineering, computer engineering and other relevant STEM education programs continues to increase. As graduates from these programs enter the workforce they will require knowledge of sensing devices, communication technologies, and control techniques to successfully meet an ever-increasing demand for the design and support of IoT related systems [1, 4, 15].

An ongoing project at Texas A&M University-Kingsville and Texas A&M University-Corpus Christi, both Hispanic Serving Institutions, has focused on enhancing support for teaching IoT related concepts and technology. The project has especially emphasized expanding support for engaged remote student learning to accommodate learners that do not have ready access to lab facilities, such as those with family or work obligations that make it difficult to attend class at a time when labs are typically scheduled [2, 6, 13]. A learning toolkit approach has been adopted in which a collection of basic IoT equipment and components has been assembled that can be utilized by remotely learning students to complete IoT related exercises and assignments [7, 10, 16]. Students are provided with a learning toolkit at the start of a class and can then utilize it throughout the semester to study IoT related materials and complete projects at a time and place that fits their schedule.

The most recent iteration of the evolving IoT learning toolkit incorporates a camera module in order to expand its capabilities and provide support for IoT projects that involve image and video capture. This paper reports on the latest version of the toolkit along with a series of exercises developed to teach IoT concepts and technology, especially those involving image and video capture, to engaged remote student learners. The following section provides a brief look at the evolution of the IoT learning kit during the course of the project. The focus then shifts to the latest version of the kit, first with a look at the components it contains followed by a detailed description of the new exercises and support materials developed to accompany it. The next section discusses deployment plans for the kit along with its adaptability for students of differing backgrounds. The paper concludes with a brief look at future plans for the IoT learning kit.

Background

The initial IoT learning toolkit designed during this project was a very basic one. It consisted of a Raspberry Pi processor board [14] and a collection of sensors, actuators, resistors, and LEDs. A set of exercises was developed to guide students through the process of utilizing jumper wires and a breadboard to connect the necessary components of the toolkit to establish and test basic IoT solutions [11]. Subsequently a second more extensive toolkit was designed based on an IoT learning platform. A second series of exercises was also developed to introduce students to the components and capabilities of the learning platform and how they can be utilized in designing IoT solutions [8]. As described in [9], there are advantages and disadvantages associated with both versions of the IoT toolkit, and both a student's technical background as well as the complexity of an IoT exercise or assignment to be performed are significant factors in choosing between them.

Support materials were created to accompany the exercises developed for both versions of the IoT toolkit. These materials provide students with examples and demonstrate the IoT concepts being taught and how to incorporate the underlying technology into a programming solution. The C programming language was utilized in creating materials to support the first set of exercises, and students were also instructed to code their exercise solutions in C. Support materials for the second set of exercises were based on the Python programming language and students were instructed to use Python in coding their solutions. As discussed in [9], it is anticipated that switching to the Python language will accommodate a broader variety of student backgrounds and expand the potential use of the toolkit to additional engineering disciplines.

Augmenting the IoT Toolkit

More recently a new version of the IoT learning toolkit has been designed. In the development of the new IoT toolkit the set of basic components that comprise the initial learning kit was used as a starting point and augmented to incorporate a camera module. The Raspberry Pi board is again utilized as the processor for the augmented learning kit. Its relative affordability and variety of interfaces and connection capabilities, including one that can readily accommodate a camera module, make it well suited for the task. The processor board for each kit will have an updated version of the Raspberry Pi operating system preinstalled on its SD card.

A similar collection of sensors, actuators, resistors, and LEDs is also included in the augmented kit along with a breadboard and jumper wires to connect components. Of course, the specific sensors and actuators included can be adjusted as necessary to meet the needs of a particular class or topic being taught. The camera module included in the new kit consists of the camera itself along with a connector that is compatible with the processor board utilized in the kit (the Raspberry Pi 4). The intent is to develop a budget friendly IoT learning toolkit that can provide students with experience in a wider variety of IoT applications including those that utilize image and video input.

IoT Exercises

A set of exercises has been developed to accompany the new toolkit and teach students how images and video can be utilized in IoT solutions. The exercises will guide students in learning how to integrate the camera module into an IoT solution and capture images and video for local storage or to be sent to a cloud-based facility for storage and analysis. Students will also learn about strategies for exploiting image recognition and other machine learning techniques in the design of an IoT solution. The Python language was used in developing support materials for the new exercises in order to broaden their applicability as described above, and students will be instructed to utilize Python in coding their solutions.

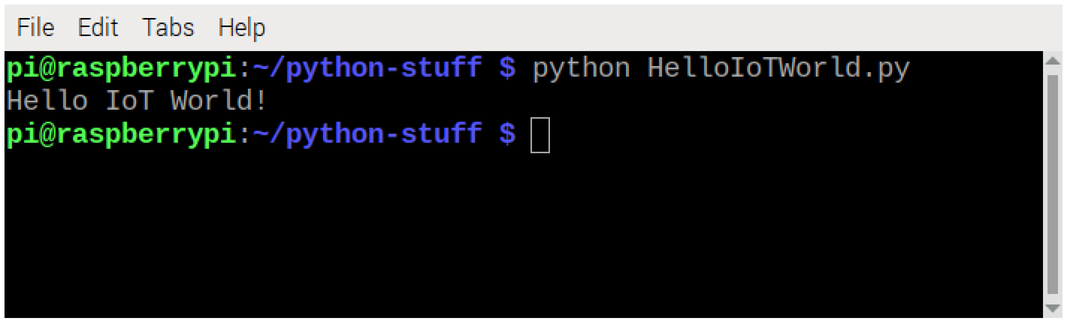
This section examines the tasks associated with the new exercises and how they will be used to teach students about the opportunities for incorporating image capture or video into an IoT solution. Table 1 summarizes the topics covered by each of the seven new exercises. They start with the basic tasks of assembling and checking the functionality of the toolkit components and progress incrementally to provide students with experience in utilizing image capture in an IoT application.

Table 1. Exercises for the Augmented IoT Learning Toolkit

Exercise	Topics Covered
1. Introduction and orientation	Unpack kit and connect components to processor board (monitor, keyboard, mouse, and power supply).
2. Camera module installation and test	Install camera module and utilize an operating system level command to verify installation.
3. Programmatic image capture	Learn how to utilize camera device from within a Python program including the new programming libraries required.
4. Periodic image capture	Include the camera module in an IoT solution to monitor a condition through periodic image capture.
5. Event driven image capture	Modify solution for previous exercise to perform image capture when a designated event occurs.
6. Cloud-based resource integration	Learn about integration of cloud-based IoT resources by sending captured images to a ThingSpeak account.
7. AI-based API integration	Integrate appropriate APIs needed to perform AI-based image recognition/analysis.

Exercise 1. Introduction and orientation: The first exercise begins by tasking students to unpack and connect the central component of the IoT toolkit: the processor board. They are instructed on how to attach a keyboard, mouse, monitor, and power supply to the board in order to establish an IoT development environment. Once the components are connected students power on the processor board to ensure the preloaded Linux [17] based operating system boots correctly. As

in the exercise sets for previous versions of the IoT toolkit, students are then instructed to create an IoT version of the classic HelloWorld program and ensure it runs correctly in their new programming environment. A preinstalled Python Integrated Development Environment (IDE) supports this task. Figure 1 shows the output produced by the HelloIoTWorld program after being successfully run in an operating system console shell.



```
pi@raspberrypi:~/python-stuff $ python HelloIoTWorld.py
Hello IoT World!
pi@raspberrypi:~/python-stuff $ 
```

Figure 1. Output of HelloIoTWorld program.

Exercise 2. Camera module installation and test: In the second exercise students are asked to install the camera into their IoT development environment and test its functionality. The camera included with the augmented toolkit is the Raspberry Pi Camera Module 3 [14]. It includes both the camera itself as well as a cable to connect to the processor board. Students are instructed to carefully connect the camera to the processor board using the supplied cable. The camera is then to be tested by ensuring it can capture an image correctly. To accomplish this students are asked to open an operating system shell and issue the "`rpicam-still -o test.jpg`" command. If the camera is installed correctly, invoking this command will open a window that displays the contents currently viewed by the camera. After a brief delay the image currently viewed by the camera will be automatically captured and stored in "`jpg`" format in a file named "`test.jpg`". Inspecting the contents of the file utilizing a file browser allows students to verify that the image was captured and the camera is working correctly. Testing the camera this way utilizing the "`rpicam-still`" operating system command enables the camera installation to be verified before attempting to use it programmatically in subsequent exercises. Figure 2 illustrates a processor board with the camera module correctly installed.

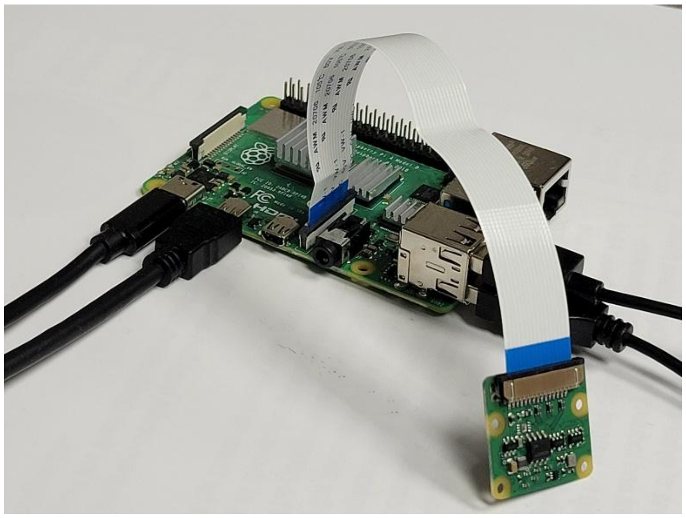
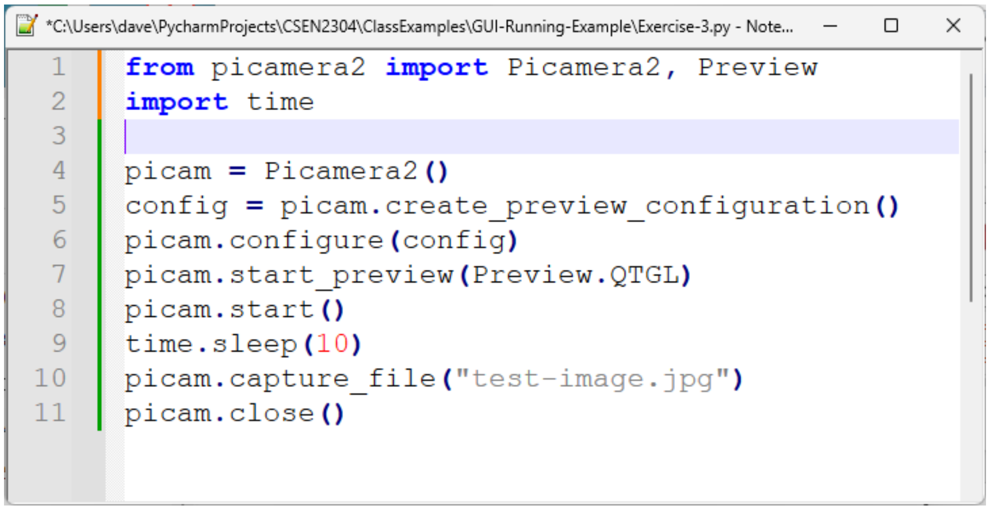


Figure 2. Processor board with camera module installed.

Exercise 3. Programmatic image capture: The topic of the third exercise is for students to learn how to access and utilize the camera from within a Python program. Support materials for this exercise explain to the student the concept of importing libraries into a Python program, and that the "`Picamera2`" and "`Preview`" libraries are needed to access and control the camera for this exercise, and that the "`time`" library is also required. Students are then tasked with writing a basic Python program that, when started, will initialize the camera and place it into preview mode so that a window is opened that shows the image currently viewed by the camera. After a brief time interval has elapsed the program should capture the image currently viewed by the camera and save it into a designated file. After completing and running the program students are asked to view the contents of the designated file to ensure the image that was viewed by the camera was successfully captured. The code listing in Figure 3 illustrates a potential Python solution for this exercise. Though this exercise is also a basic one with a brief solution, importantly, it does cover the steps that are required to incorporate the camera device programmatically as part of an IoT solution.



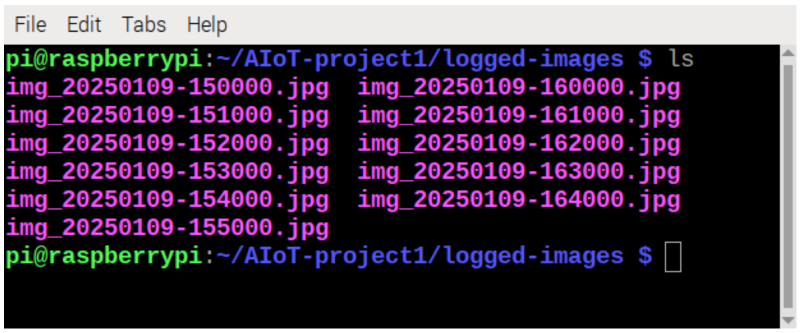
*C:\Users\dave\PycharmProjects\CSEN2304\ClassExamples\GUI-Running-Example\Exercise-3.py - Note...

```python
from picamera2 import Picamera2, Preview
import time

picam = Picamera2()
config = picam.create_preview_configuration()
picam.configure(config)
picam.start_preview(Preview.QTGL)
picam.start()
time.sleep(10)
picam.capture_file("test-image.jpg")
picam.close()
```

Figure 3. Code listing of a Python solution for Exercise 3.

Exercise 4: Periodic image capture: The fourth exercise builds upon the solution to the third one. It asks students to extend their basic solution to that exercise into a program that can monitor a condition or situation visually by periodically capturing and logging images over a specified time period. The logged image files are to be stored in a specific directory and named according to the time and date at which they were captured. Figure 4 depicts a directory containing images captured periodically and named appropriately.



```
pi@raspberrypi:~/AIoT-project1/logged-images $ ls
img_20250109-150000.jpg  img_20250109-160000.jpg
img_20250109-151000.jpg  img_20250109-161000.jpg
img_20250109-152000.jpg  img_20250109-162000.jpg
img_20250109-153000.jpg  img_20250109-163000.jpg
img_20250109-154000.jpg  img_20250109-164000.jpg
img_20250109-155000.jpg
pi@raspberrypi:~/AIoT-project1/logged-images $ 
```

Figure 4. Directory containing images captured periodically.

The support materials for this exercise describe an IoT application scenario in which this functionality is important such as monitoring the status of crops in an agricultural setting. Depending on the class and department in which the IoT Toolkit is being utilized, if necessary, supplemental materials can also be provided to students covering the Python programming constructs required to complete the exercise.

Exercise 5: Event driven image capture: In the fifth exercise students are presented with the task of creating a Python program that utilizes the camera to observe a condition or situation and capture images in response to some predefined event. The support materials for the exercise describe a security-related scenario in which a camera should observe a building entrance and log entries and exits by capturing images as people enter or leave. To simulate this scenario as an exercise students are asked to create a Python program that integrates input from the camera along with input from an appropriate sensor device, such as an IR sensor or motion sensor. The program should continually monitor the sensor input. When the program detects that a designated event has occurred, such as the presence of a person in front of the sensor, it should use the camera to capture an image. The program should then reset and continue monitoring the sensor and capturing images whenever the designated event occurs. The captured images should be stored in a specific directory and inspected after the program has run to ensure that images are correctly captured.

Exercise 6. Cloud-based resource integration: The topic of the sixth exercise is to integrate cloud-based analytic resources into an IoT solution. Students will be instructed about how to establish an educational account on a cloud-based platform such as ThingSpeak [18]. They will then be asked to modify their solution code for Exercise 5 so that in addition to storing captured images in the local file system a copy is also sent to their cloud-based ThingSpeak account. To verify the images are being received and that the program is working correctly, students will also be instructed to have their ThingSpeak channel display the most recently received captured image onto a local window of their IoT development environment. It should be updated each time the designated event occurs and a new image is captured. The support materials for this exercise will provide students with an overview of the array of IoT related services typically available through cloud-based analytic platforms.

Exercise 7. AI-based API integration: The seventh exercise will provide students with experience in integrating basic Artificial Intelligence (AI) functionality into an application and how it could serve as a building block in designing an IoT solution. The primary task of the exercise will be the analysis of an image that is captured by the camera. The approach utilized in the exercise requires configuration of a student's IoT development environment in order to utilize an AI-based API (Application Programming Interface). Students from some disciplines will likely be capable and comfortable in customizing their development environment, and benefit from the experience. For students from other backgrounds the process might be more challenging. Instructors can decide how much or little of the configuration process they would like for students to perform and take care of any pre-configuration required as appropriate.

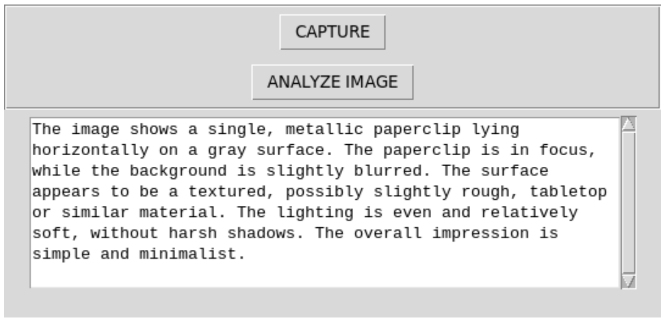
The primary development environment customization required is the installation of the components necessary to integrate AI functionality into an IoT software solution. In this exercise the Gemini APIs from Google will be utilized for image analysis [5]. This requires the installation of the "`google-generativeai`" package into the development environment. To do so students are instructed to open an operating system command shell and issue the following commands:

```
"python -m venv ./my-environment/"
"source my-environment/bin/activate"
"pip install -q -U google-generativeai"
```

The first command creates a Python virtual environment into which the student can install the required package. The second command activates the new Python virtual environment. The third command performs the installation of the required package into the student's virtual environment, making it available for use within an application program.

Once the required development environment customization has been completed, students are asked to create a Python program that will present a basic interface to the user consisting of one window to display the image currently being viewed by the camera and another window that consists of two buttons and an output field. The first button should allow the user to capture the image currently being viewed by the camera. The second button should allow the user to request that the captured image be analyzed and the results displayed in the output field of the user interface. Once completed students should test their code to ensure the captured image is successfully submitted for analysis and that the returned results are reasonable.

Figure 5a and Figure 5b illustrate a potential solution for this exercise. The interface providing the two buttons and the output field is shown in Figure 5a. Figure 5b shows a window that displays the image currently being viewed by the camera. In this case the "Capture" button was pressed while the camera was observing a paper clip. The output text field below the buttons in Figure 5a shows the result of the image being analyzed by the Google Gemini facility [5].



```
The image shows a single, metallic paperclip lying
horizontally on a gray surface. The paperclip is in focus,
while the background is slightly blurred. The surface
appears to be a textured, possibly slightly rough, tabletop
or similar material. The lighting is even and relatively
soft, without harsh shadows. The overall impression is
simple and minimalist.
```

Figure 5a. User interface for an Exercise 7 solution.

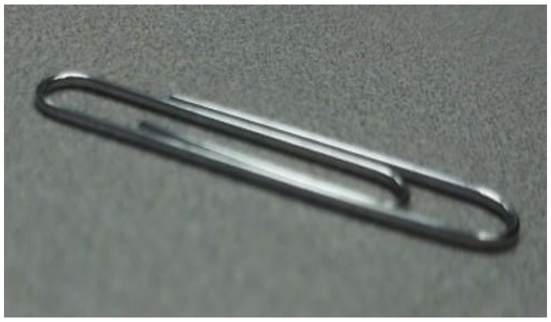


Figure 5b. Image observed by camera (a paperclip).

The support materials for the exercise describe the parts of the "`google.generativeai`" API that are used to perform the image analysis, especially the "`.GenerativeModel()`" method which allows for the specification of the model to be used for the analysis task and the "`.generate_content()`" method to perform the image analysis. The "`tkinter`" library is also described as a facility that can be used to create the graphical user interface for the program [19].

Discussion

Like previous versions of the IoT toolkit developed during this project, the augmented toolkit and exercises described in this paper will be deployed in engineering and computer science capstone project classes at the two participating universities. The goal will be to teach IoT concepts to these students, with a focus on opportunities to include image or video capture, and increase their awareness of the important role this technology can serve in their class projects and in solutions to engineering problems in general. Pre- and post-course surveys will be utilized to assess student learning of IoT concepts and their awareness of opportunities to utilize this technology in engineering solutions. The surveys will also serve to evaluate the toolkit itself as well as the associated exercises and support materials.

As described previously, including the camera module in an IoT application does add some complexities to a student solution for an exercise, assignment, or project. In addition to physically installing the camera device, students must also configure their IoT development environment to ensure the availability of the new libraries required to access and interact with the camera programmatically. Further customization is also needed if AI functionality is to be included in an IoT solution, such as the creation of a virtual Python environment and installation of required libraries as discussed in the previous section. Although these additional complexities might prove challenging for some students, depending on their academic discipline or background, instructors can adjust descriptions provided in the exercise support materials and the amount of pre-configuration that is done for students accordingly.

Conclusion and Future Work

The latest version of the IoT learning toolkit has been targeted to assist students in learning about IoT concepts and technology with an emphasis on image capture techniques. A series of exercises have been developed to assist remotely learning students engage with this material and become familiar with how these capabilities can be utilized in IoT solutions. Students also learn about incorporating basic AI functionality in the final exercise of the series. Deployment of the IoT learning toolkit in engineering and computer science capstone courses will provide feedback for evaluation and improvement purposes for both the toolkit itself as well as the exercises and supporting materials.

An additional set of exercises is planned for the augmented toolkit. It will provide students with further experience in utilizing camera input as part of an IoT solution. This will include incorporating video capture as well as image capture and analysis into an application. Additional AI and Machine Learning (ML) content is also planned for the next iteration of exercises. Students will learn about AI models and the difference between accessing an external model through an API (Exercise 7) and the possibility of running a model locally on-device. With a continued emphasis on affordability lower cost hardware options will be explored for the toolkit such as the Raspberry Pi AI Kit [14].

Acknowledgement and Disclaimer

This material is based upon work supported by the National Science Foundation through Grant No. 2044255. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

[1] Farha Ali. 2015. Teaching The Internet of Things Concepts. In Proceedings of the WESE'15: Workshop on Embedded and Cyber-Physical Systems Education (WESE'15). Association for Computing Machinery, New York, NY, USA, Article 10, 1–6. https://doi.org/10.1145/2832920.2832930

[2] Anastasi, G.F., Musmarra, P. (2022). Teaching IoT in the Classroom and Remotely. In: Casalino, G., *et al.* Higher Education Learning Methodologies and Technologies Online. HELMeTO 2021. Communications in Computer and Information Science, vol 1542. Springer, Cham. https://doi.org/10.1007/978-3-030-96060-5_7

[3] Balaji, S., Nathani, K. & Santhakumar, R. IoT Technology, Applications and Challenges: A Contemporary Survey. *Wireless Pers Commun* **108**, 363–388 (2019). https://doi.org/10.1007/s11277-019-06407-w

[4] L. C. B. C. Ferreira, O. C. Branquinho, P. R. Chaves, P. Cardieri, F. Fruett and M. D. Yacoub, "A PBL-Based Methodology for IoT Teaching," in IEEE Communications Magazine, vol. 57, no. 11, pp. 20-26, November 2019, doi: 10.1109/MCOM.001.1900242.

[5] Gemini Developer API. [online]. Available at: https://ai.google.dev/gemini-api/docs. (Accessed: January 9, 2025).

[6] C. Grabowski, M. Rush, K. Ragen, V. Fayard, and Karen Watkins-Lewis, "Today's Non-Traditional Student: Challenges to Academic Success and Degree Completion," *Inquiries Journal*, Vol. 8, No. 03, pp. 1-2, 2016.

[7] K. Habib, E. Kai, M. Saad, A. Hussain, A. Ayob and A. Ahmad, "Internet of Things (IoT) Enhanced Educational Toolkit for Teaching & Learning of Science, Technology, Engineering and Mathematics (STEM)", *2021 IEEE 11th International Conference on System Engineering and Technology (ICSET)*, Shah Alam, Malaysia.

[8] D. Hicks, L. McLauchlan, M. Mehrubeoglu and H. K. R. Bhimavarpu, "Expanding Remote Student Learning-Internet of Things Applications and Exercises," 2023 IEEE Frontiers in Education Conference (FIE), College Station, TX, USA, 2023, pp. 1-8, doi: 10.1109/FIE58773.2023.10343065.

[9] Hicks, D., & McLauchlan, L., & Mehrubeoglu, M., & Bhimavarapu, H. K. R. (2024, June), *Expanding Support for Engaged Remote Student Learning of Internet of Things*

Concepts and Technology Paper presented at 2024 ASEE Annual Conference & Exposition, Portland, Oregon. 10.18260/1-2—47384.

[10] M. Kusmin, M. Saar and M. Laanpere, "Smart schoolhouse — designing IoT study kits for project-based learning in STEM subjects," 2018 IEEE Global Engineering Education Conference (EDUCON), Santa Cruz de Tenerife, Spain, 2018, pp. 1514-1517, doi: 10.1109/EDUCON.2018.8363412.

[11] L. McLauchlan, D. Hicks, M. Mehrubeoglu and B. Zimmer, "Work-in-Progress: Internet of Things Enabling Remote Student Learning," *2022 ASEE Annual Conference and Exposition*, Minneapolis, MN, USA, June 26-29, 2022.

[12] Dragos Mocrii, Yuxiang Chen, Petr Musilek, "IoT-based smart homes: A review of system architecture, software, communications, privacy and security", *Internet of Things*, Volumes 1–2, 2018, Pages 81-98, ISSN 2542-6605, https://doi.org/10.1016/j.iot.2018.08.009.

[13] National Center for Education Statistics, "Characteristics of Postsecondary Students," *The Condition of Higher Education 2019*, Ch. 2, pp. 1-5, May 2019. Online: https://nces.ed.gov/programs/coe/pdf/Indicator_CSB/coe_csb_2019_05.pdf (Accessed: 5/9/2023).

[14] Raspberry Pi Official Documentation Site, [online]. Available at: https://www.raspberrypi.com/documentation/. (Accessed: January 9, 2025).

[15] N. Silvis-Cividjian, "Teaching Internet of Things (IoT) Literacy: A Systems Engineering Approach," 2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET), Montreal, QC, Canada, 2019, pp. 50-61, doi: 10.1109/ICSE-SEET.2019.00014.

[16] Mariana Aki Tamashiro, Marie-Monique Schaper, Ane Jensen, Rune Heick, Brian Danielsen, Maarten Van Mechelen, Kasper Løvborg Jensen, Rachel Charlotte Smith, and Ole Sejer Iversen. 2023. Teaching technical and societal aspects of IoT - A case study using the Orbit IoT Kit. In Proceedings of the 2023 ACM Designing Interactive Systems Conference (DIS '23). Association for Computing Machinery, New York, NY, USA, 1236–1247. https://doi.org/10.1145/3563657.3596092

[17] The Linux Foundation. [online]. Available at: https://www.linuxfoundation.org/projects. (Accessed: January 9, 2025).

[18] ThingSpeak for IoT Projects, [online]. Available at: https://thingspeak.mathworks.com/. (Accessed: January 9, 2025).

[19] Tkinter – Python interface to Tcl/Tk. Available at: https://docs.python.org/3/library/tkinter.html. (Accessed: January 14, 2025).