Introducing STEM Students to Advanced IoT Design Concepts

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

David Hicks is an Associate Professor and Associate Chair 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 85 publications covering areas such as IoT, 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 senior member of ASEE and is the current Program Chair for the Ocean and Marine Engineering Division (OMED); in prior years, he was an officer for the OMED as well, including 2012-2014 Chair, 2014-2016 Past Chair and 2023-2025 Secretary/Treasurer. He is also a senior member of IEEE, and a member of 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, Treasurer 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 a professor of Electrical Engineering at Texas A&M University-Corpus Christi currently serving as the Program Coordinator for BS in Electrical Engineering, and MS in Engineering programs. Her research interests are multidisciplinary in nature and entail applications of imaging modalities (hyperspectral (visible, NIR, SWIR), thermal, color) for image segmentation and object classification/identification using classical image processing and machine learning/deep learning models. She is also interested in engaged student learning using engineering pedagogies, advanced technologies such as IoT and the cloud, with consideration of sustainable design and professional skills.

Introducing STEM Students to Advanced IoT Design Concepts

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

Introduction

The proliferation of Internet of Things (IoT) applications continues to increase the importance of effectively teaching engineering and computer science students about this technology [1]. In support of this goal, an ongoing project at Texas A&M University-Kingsville and Texas A&M University-Corpus Christi, both Hispanic Serving Institutions, has focused on enhancing the teaching of IoT concepts and technology. Throughout the project a particular emphasis has been placed on accommodating engaged remote student learners who might not have ready access to lab facilities, such as those with work or family obligations. A hands-on approach has been adopted in which students are tasked with completing lab exercises to learn about IoT concepts and technology. Students are provided with an IoT learning toolkit that contains a processor board along with sensors, actuators, resistors, LEDs, and the other elements needed to design and implement an IoT solution for each exercise. This enables students to work on exercises at a time and place that fits their schedule [2-7].

The exercises developed during the project have been organized into exercise sets, each focusing on a particular aspect of IoT technology. The first exercise set that was developed defined a series of tasks designed to introduce and familiarize students with basic IoT elements and how they can be combined to form an IoT solution [8]. Subsequent exercise sets have expanded on this foundation to provide students with experience in designing IoT solutions that incorporate additional components and capabilities. This paper focuses on a new set of exercises designed to introduce students utilizing the IoT learning toolkit to more advanced IoT design concepts including the use of artificial intelligence (AI) and edge computing techniques in IoT solutions.

The following section provides additional background on the IoT learning toolkit and its evolution during the course of the project along with an overview of the accompanying exercise sets. A description of the advanced IoT concepts that are introduced by the latest exercise set follows. The next section describes the specific exercises developed to introduce students to the more advanced IoT concepts and technology and their use in the design of IoT solutions. A discussion section follows that includes a description of the deployment plan for the new exercise set. The last section concludes the paper including a brief look at future research plans.

Project Background

The original IoT learning toolkit was a very basic one consisting of a simple Raspberry Pi 4 processor board [9] along with the sensors, LEDs, resistors, actuators, jumper wires, and other elements needed to complete introductory IoT exercises. As the project continued, the IoT toolkit was expanded and modified as needed to support students' learning of additional IoT concepts and technology.

The latest version of the toolkit adds the additional components needed to support introducing students to more advanced IoT concepts. In particular, the processor board has been upgraded to a Raspberry Pi 5 [10] to support the design of IoT solutions that require more extensive local computation and information processing. The kit also includes a compatible camera and connector cable for image and video capture (Raspberry Pi Camera Module 3 [11]). A motion detecting sensor is included to accommodate students in completing the latest exercise set.

Each of the exercise sets developed during the project includes learning materials to provide students utilizing the learning toolkit with explanations of IoT concepts and examples of their use in an IoT solution. Students are instructed to complete the exercises within each exercise set sequentially as part of scaffolding pedagogy since these exercises progressively build upon each other to introduce and teach the associated IoT concepts. As described by Hicks *et al.* [12], the C computer language was originally used in developing examples for the earliest exercise set, and students were also asked to code their exercise solutions in C. Support materials for subsequent exercise sets have been based on the Python language, and students are instructed to code their solutions in Python. The switch to Python was intended to widen the accessibility of the exercises and support materials making them appropriate for student learning from a wider set of disciplines [13] and a broader set of resources.

A Focus on Advanced IoT Concepts and Applications: Moving towards the edge

As reported by Hicks *et al.* [14], a previous exercise set developed during the project focused on providing students with hands-on experience in integrating image and video capture capabilities into IoT solutions. It also provided students with an initial experience in utilizing basic AI capabilities using a pre-trained AI model in an IoT solution. Specifically, students were given the task of utilizing a cloud-based AI service (Google Gemini [15]), accessed through an Application Programming Interface (API), to analyze an image captured by the camera.

The current paper reports on the new set of exercises developed to provide students with further experience utilizing image capture and AI technology in the design of IoT solutions. While the previous exercise set integrated cloud-based AI functionality into an application through an API, the new set focuses on deploying a machine learning (ML) model directly on the local processor board itself and utilizing it for object recognition. As described in more detail in the next section, the new exercise set also includes support materials and tutorials that introduce and explain relevant concepts, including the features and use of a specific pretrained machine learning model (YOLO11 [16]) that can support object detection and run on the processor board provided in their IoT learning toolkit.

As with previous exercise sets, after examining the support materials for an exercise, students are asked to apply the new material in the design and implementation of an IoT solution. Exercise descriptions include examples of how the associated topics covered relate to real world IoT scenarios and solutions. The exercises in the new set progressively build to include material on image capture, object recognition, and edge computing implementation strategies. The final

exercise in the set combines the use of these new concepts, tasking students with the design of a security application that utilizes a camera to capture images, analyzes each image using an ML model, and raises an alarm if a person is detected in an image. The use of object recognition in this application ensures the alarm will only be triggered by a person and not some other event, such as animal activity, thus improving its performance over a system that is based on a simple motion detector alone.

IoT Exercises

Table 1 provides an overview of the seven exercises that were developed for this exercise set. Like previous exercise sets, the first one introduces students to the IoT learning toolkit and describes how to assemble the primary components of the kit. In the next three exercises students learn how to install the camera module and access it from within an IoT application. These are similar to exercises from a previous exercise set in which the camera module was also used. However, in this set they have been updated to provide appropriate instructions specific to the upgraded processor board. The remaining three exercises focus on integrating image processing capabilities into their IoT development environment, including this technology in the design of their IoT solutions, and edge computing-based implementation strategies. The remainder of this section describes each of the exercises in more detail.

Table 1. The seven exercises of the new IoT exercise set

Exercise	Topics Covered
1. Overview and orientation	Unpack IoT learning kit, connect basic components (monitor, mouse, keyboard, and power supply), create and run HelloIoTWorld program.
2. Camera module setup and test	Connect camera module to processor board and utilize operating system level command to verify successful installation.
3. Programmatic camera access	Learn how to utilize new libraries to access camera device from within a Python program for image capture.
4. Multiple image capture	Strategies for multiple image capture including control of when capture events occur (at regular intervals and event driven approaches).
5. Enhancing IoT Deployment Environment	Installation of necessary components required to support machine learning model for image recognition on local processor (edge device).
6. Integrating AI/ML-based image analysis	Perform image capture when a designated event occurs and then analyze image on local processor.
7. An improved security system application	Use AI/ML-based image analysis on local processor to improve the capabilities of a basic security system application.

Exercise 1. Overview and orientation: In the first exercise students learn about the central component of the IoT learning kit: the processor board. Students are instructed to connect the monitor, mouse, keyboard, and power supply and then observe the boot process of a preloaded (Linux) operating system and ensure it successfully starts. Students are next asked to use a preinstalled Python Interactive Development Environment (IDE) to create a simple IoT version of the classic HelloWorld program. The purpose of the simple program is to perform the basic task of printing "Hello IoT World" to the console. This ensures that the IDE and Python are working correctly and will be available for completing later exercises. Another purpose of this exercise is to impress upon students the dual roles the processor board serves in the IoT learning toolkit as both a programming environment for development and a deployment platform. Figure 1 shows a Python code listing of a solution for Exercise 1.

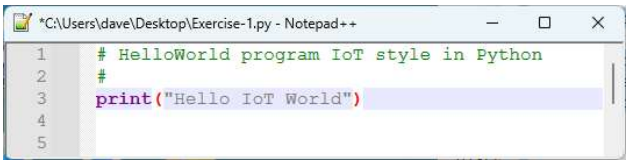

Figure 1. Python code listing of a solution for Exercise 1.

Exercise 2. Camera module setup and test: The topic of the second exercise is the integration of the camera module component provided in the IoT learning kit. Students are provided with instructions on how to utilize a specialized cable included with the kit to carefully connect the camera to the correct port on the processor board. Once connected, students are instructed to power up the processor board and open an operating system shell. They are then asked to utilize an operating system level command to verify that the camera module is correctly installed. If installed correctly, when the "`rpicam-still -o picture1.jpg`" command is issued in the operating system shell, the camera will activate and an additional camera preview window will open that displays the image it currently observes. After a brief (5 second) delay, the camera will take a picture, capturing the image it observes and saving it to a file with the specified name ("`picture1.jpg`"). The contents of the file can be inspected using a jpg viewer to verify the camera module is working correctly. Utilizing an operating system command to test the camera enables its functionality to be verified before attempting to access it programmatically in later exercises. Figure 2 illustrates a preview window displaying an image observed by a camera, in this case the mouse and keyboard of the student's IoT workstation.

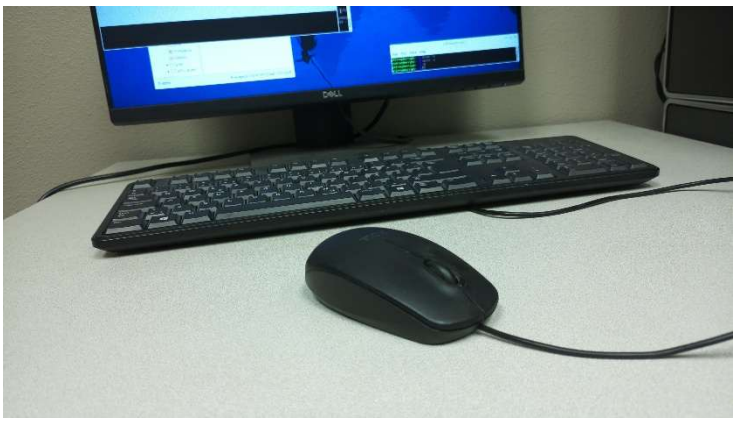

Figure 2. Camera preview window.

Exercise 3. Programmatic camera access: In this exercise students learn about accessing the camera from within a Python program. They first read supporting material that explains the concept and process of including external libraries within a Python program. Students are also given an explanation of the required camera-related libraries that should be

imported into their program to facilitate access to the camera. Descriptions are provided for the specific methods that enable common camera actions to be performed, such as configuring the camera, starting the camera, and capturing an image to be saved to a file.

An explanation of the "`time`" library and the "`sleep`" method is also included as these are often used when interacting with the camera programmatically. Students are then tasked with writing a Python program that will configure the camera, start the camera in preview mode, capture a single image after a 10 second delay, and then save the image to a file. After running the program, the students are instructed to examine the file and ensure the image was captured and stored correctly. Though a basic one, successfully completing this exercise introduces students to the process of accessing the camera programmatically and prepares them for later exercises.

Exercise 4. Multiple image capture: The focus of this exercise is to familiarize students with techniques to support multiple image capture within an IoT application. Tutorials for the exercise describe example IoT application areas in which capturing multiple images is necessary, such as in an agricultural setting in which an IoT system is deployed for monitoring crop health or soil conditions. Strategies are explained for controlling when image capture should occur. These include techniques such as periodic image capture at regular intervals, which could be appropriate in a monitoring type application, as well as event driven image capture, which could be appropriate in a security type application. Strategies are discussed for naming and storing captured image files in an organized way, including the utilization of cloud-based IoT resources. After examining the support materials for this exercise, students are tasked with extending their solution for Exercise 3 to capture images repeatedly at a 10 second interval, store the images in the local filesystem, and label the files according to the time that an image was captured.

Exercise 5. Enhancing the IoT development and deployment environment: In this exercise the focus is on preparing a student's IoT development and deployment environment to accommodate machine learning tasks. The tutorials for the exercise first describe how a task such as image processing can be done through the use of a networked resource, such as the Google Gemini API [15]. This is contrasted with the approach of running a machine learning model directly on a local processor to perform image processing. The tradeoffs of each approach are examined along with an introduction to the concept of edge computing in which as much information processing as possible is performed on the periphery (edge) devices of a network rather than being sent to a centralized server [17-19]. Image processing and object recognition are used as examples in the explanation of these important concepts.

After examining the support materials, students are tasked with preparing their specific IoT environment for processing images directly on the local processor. The students are first instructed to open an operating system command shell and then enter the following commands:

```
"python3 -m venv -system-site-packages venv"
"source venv/bin/activate"
```

These commands create and activate a Python virtual environment named "`venv`". This virtual environment will serve to compartmentalize the customizations being made so they do not interfere with other Python users or programs. The next step involves installing additional libraries needed to support the local processing of images. Students issue the following command from the same command shell to install the needed libraries from Ultralytics [20] and NCNN [21], to support a lightweight open-source neural network inference framework:

```
"pip install ultralytics ncnn"
```

Students are next instructed to issue the following command:

```
"yolo detect predict model=yolo11n.pt"
```

This command downloads the YOLO11 ML model to be used for object detection and recognition. This model has been trained on the Common Objects in Context (COCO) dataset to detect and recognize 80 common objects [22].

The final step in preparing the IoT environment to support local image processing is testing to verify that it functions correctly. Students are provided with a Python test program that will perform several functions including loading the installed ML model, running inference on an image, parsing the results, and displaying bounding boxes around detected objects. When the program is started it will open a window on the monitor and continually display the results of processing the video stream being received from the attached camera in real time. Figure 3 shows a screenshot of the test program output window as it is being used to verify an installation. As seen in the figure, the camera is viewing the same objects from the student's IoT workstation as in Figure 2. The view has been zoomed in slightly, and both the keyboard and mouse objects are detected, surrounded by bounding boxes, and correctly labeled by the verification program.

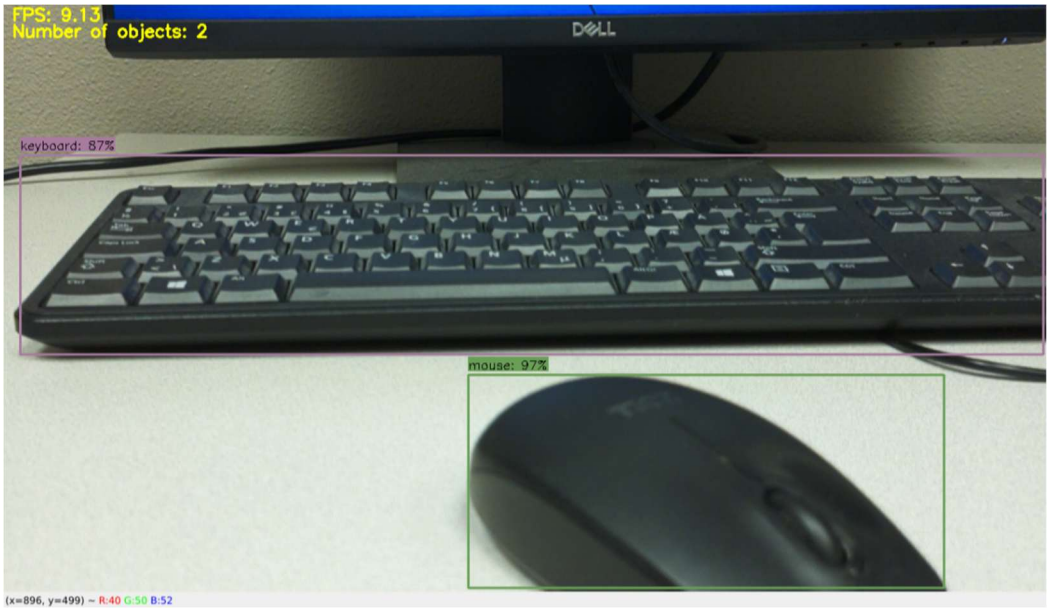

Figure 3. Output of object detection verification program.

Exercise 6. Integrating AI/ML based image analysis: In this exercise students learn about how to access and utilize the newly installed ML model programmatically. In the tutorials for this exercise, students are guided to import the libraries needed to use the image analysis model from within their programs. They also learn about the methods that enable them to initialize and then communicate with the model. These methods include coverage of the "`predict()`" method that can be used to request the model to analyze a specified image supplied as an argument. A description is also provided of the format in which the "`predict()`" method returns results and how those results can be interpreted programmatically.

After reading the support materials students are tasked with writing a Python program to initialize an instance of the ML model and request it to analyze an image. The results of the analysis should be processed to determine if a specific object is contained in the image. A message should be printed to indicate whether the specified object was found to be present in the image. Figure 4 illustrates a code listing of a solution for Exercise 6.

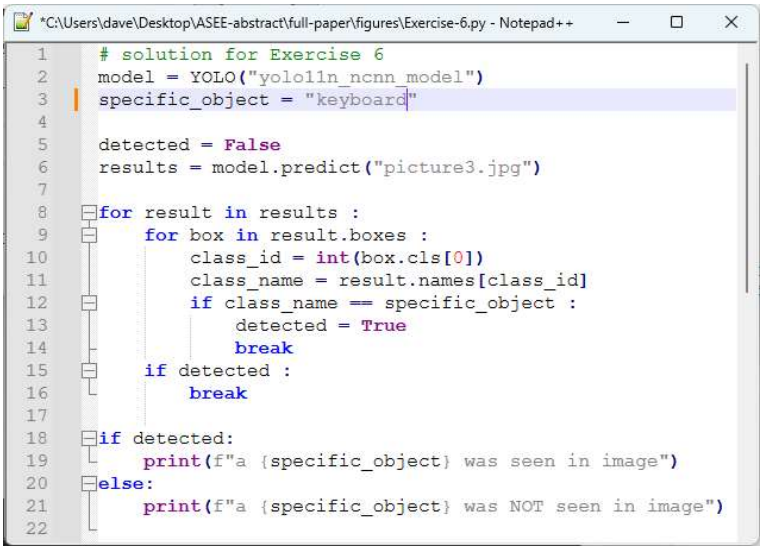

Figure 4. Code listing of a Python solution for Exercise 6.

Exercise 7. An advanced security system application: This exercise provides students with an example of how machine learning and edge computing can be utilized in the design of an IoT application. In this case, performing object recognition on a local processor enables the capabilities of a basic security application to be improved. The support materials for this exercise describe the design of a basic IoT application that monitors and records the activity at the entrance to a building. The system includes a motion detecting sensor. When the sensor detects motion around the entrance, an image is captured and stored. The support materials discuss the limitations of this basic system design, including that the motion detector could be triggered by an event other than a person entering or exiting the building, such as animal activity around the entrance. Though capable of monitoring and recording activity around the building entrance, this system could not reliably be used to alert authorities of attempted unauthorized access since it could easily generate a false alarm.

The student task for this exercise is to design an improved security IoT application that also observes activity around the entrance to a building. Like the basic security system, the improved one should include a motion detector and capture an image any time activity is detected around the entrance. However, in the improved design, after an image has been captured, the system should utilize a ML model to perform an analysis of the image on the local processor. The

results of the analysis should then be processed to see if a person is present in the image. If a person *is* present in the image, the image should be stored to the local file system and the message "alert: attempted unauthorized access" should be written to the console output. If a person is not present in the image, it should be discarded and the message "person not present: false alarm" should be written to the console output. Figure 5 shows the console output of a solution for this exercise in which a person was present in some of the captured images and not present in others.

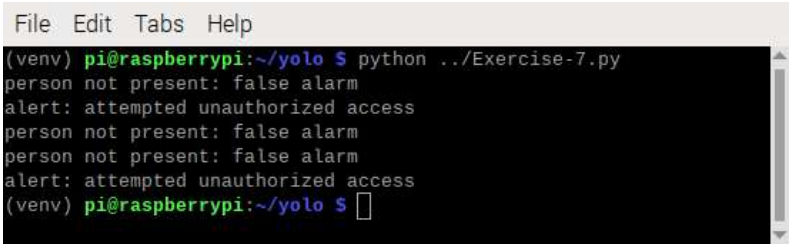

Figure 5. Console output from improved IoT application.

Discussion

The tasks that make up the new exercise set are designed to expand students' knowledge through IoT in important ways. Utilizing a ML model running on the local processor board rather than accessing a cloud-based service through an API will introduce students to the important concept of edge computing. In the edge-based approach of designing applications, as much processing as possible is to be done as close to a data source as possible, offering important advantages in an IoT solution. Additionally, students will also learn about utilizing tools such as OpenCV (Open Source Computer Vision [23]) and Numpy (Numerical Python [24]) to facilitate interacting with a locally running ML model directly rather than accessing a cloud-based AI service. This experience will prove valuable for students when designing more advanced and robust IoT solutions utilizing an edge-based approach.

The deployment environment for the new IoT toolkit and exercise set will be upper-level engineering and computer science courses at both of the participating universities. A particular target will be senior level capstone project courses in which students often integrate various software and hardware components into their design projects. Pre- and post-course surveys will allow assessment of the impact of the toolkit and exercises as well as their effectiveness at increasing student knowledge and awareness of IoT concepts and technology. Of particular importance and focus will be the assessment of the ML and edge computing design approach, and the potential important role these concepts can play in the design of an IoT solution.

Conclusion and Future Work

The latest set of exercises developed for the project described in this paper focuses on teaching students about advanced concepts that can be used in the design of an IoT application through hands-on scaffolding learning. In particular, students learn about how machine learning, object detection, and edge-based implementation strategies can enhance the design of an IoT solution. Deployment of the exercises in engineering and computer science courses at the participating universities will support assessment of their effectiveness at teaching these concepts to students.

A future version of the IoT learning toolkit is planned that will include enhanced support for computationally intensive applications. The addition of components such as the AI HAT+ [25] will provide improved performance for applications utilizing ML capabilities. Exercises will be designed to teach students how to further exploit the power of this technology in the edge computing-based design of IoT solutions.

References

[1] Ghashim, Ibrahim Ahmed, and Muhammad Arshad. 2023. "Internet of Things (IoT)-Based Teaching and Learning: Modern Trends and Open Challenges" Sustainability 15, no. 21: 15656. https://doi.org/10.3390/su152115656

[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] Biju Bajracharya, Vamsi Gondi, and David Hua, "IoT Education using Learning Kits of IoT Devices". In: Information Systems Education Journal (ISEDJ), Volume 19, No. 6, December 2021.

[4] Attapan Chan-in and Arnan Sipitakiat. 2021. Designing an Educational Internet of Things Toolbox: A case-study of a learner-centric tool design. In Proceedings of the FabLearn 2020 – 9th Annual Conference on Maker Education (FabLearn '20). Association of Computing Machinery, New York, NY, USA, 90-93. https:://doi.org/10.1145/3386201.3386203.

[5] 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

[6] 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 Systems Engineering and Technology (ICSET), Shah Alam, Malaysia.

[7] 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.

[8] McLauchlan, Lifford, Hicks, David, Mehrubeoglu, Mehrube, and Zimmer, Beate. "Work in Progress: Internet of Things Enabling Remote Student Learning". In: Proceedings of ASEE 2022 Annual Conference and Exposition, Minneapolis, MN, USA, June 26-29, 2022.

[9] Raspberry Pi official documentation site, [online]. Available at: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/. (Accessed: January 18, 2026).

[10] Raspberry Pi official documentation site, [online]. Available at: https://www.raspberrypi.com/products/raspberry-pi-5/. (Accessed: January 18, 2026).

[11] Raspberry Pi Camera Module official documentation site, [online]. Available at: https://www.raspberrypi.com/products/camera-module-v2/. (Accessed: January 18, 2026).

[12] Hicks, David; McLauchlan, Lifford, Mehrubeoglu, Mehrube, and Bhimavarapu, Hemanth. “Expanding Remote Student Learning – Internet of Things Applications and Exercises.” In: Proceedings of 2023 IEEE Frontiers in Education Conference (FIE 2023), College Station, TX, USA, Oct. 18-21, 2023.
[13] Hicks, David., McLauchlan, Lifford, Mehrubeoglu, Mehrube, and Bhimavarapu, Hemanth K. R. “Expanding Support for Engaged Remote Student Learning of Internet of Things Concepts and Technology.” In: Proceedings of 2024 ASEE Annual Conference & Exposition, Portland, Oregon. 10.18260/1-2-47384.
[14] Hicks, David; McLauchlan, Lifford; and Mehrubeoglu, Mehrube. “Integrating Image, Video, and Machine Learning into an IoT Learning Environment”, in: Proceedings of the ASEE 2025 Conference & Exposition, June 22 - 25, Montreal, Quebec, Canada.
[15] Gemini Developer API documentation site, [online]. Available at: https://ai.google.dev/gemini-api/docs. (Accessed: January 18, 2026).
[16] Khanam, Rahima, and Muhammad Hussain. "Yolov11: An overview of the key architectural enhancements." arXiv preprint arXiv:2410.17725 (2024).
[17] Kong, Linghe, Jinlin Tan, Junqin Huang, Guihai Chen, Shuaitian Wang, Xi Jin, Peng Zeng, Muhammad Khan, and Sajal K. Das. "Edge-computing-driven internet of things: A survey." ACM Computing Surveys 55, no. 8 (2022): 1-41.
[18] Hassan, Najmul, Saira Gillani, Ejaz Ahmed, Ibrar Yaqoob, and Muhammad Imran. "The role of edge computing in internet of things." IEEE communications magazine 56, no. 11 (2018): 110-115.
[19] Alnoman, Ali, Shree Krishna Sharma, Waleed Ejaz, and Alagan Anpalagan. "Emerging edge computing technologies for distributed IoT systems." IEEE Network 33, no. 6 (2019): 140-147.
[20] Ultralytics open-source project official documentation site, [online]. Available at: https://www.ultralytics.com/. (Accessed: January 18, 2026).
[21] NCNN open-source project documentation site, [online]. Available at: https://ncnn.readthedocs.io/en/latest/index.html. (Accessed: January 18, 2026).
[22] Caesar, Holger, Jasper Uijlings, and Vittorio Ferrari. "Coco-stuff: Thing and stuff classes in context." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1209-1218. 2018.
[23] Open-Source Computer Vision Project official documentation site, [online]. Available at: https://opencv.org. (Accessed: January 18, 2026).
[24] NumPy official documentation site, [online]. Available at: https://numpy.org/. (Accessed: January 18, 2026).
[25] Raspberry Pi official documentation site [online]. Available at: https://www.raspberrypi.com/products/ai-hat/. (Accessed: January 18, 2026).