"Smart Parking Navigator": An IoT Fog-Based System for Realtime Car Parking Detection

Emma Qumsiyeh
Department of Computer Science and Information Technology
Al-Quds University
Jerusalem, Palestine
emma.qumsiyeh@hotmail.com

Isam Ishaq
Department of Computer Engineering
Al-Quds University
Jerusalem, Palestine
isam@itce.alquds.edu

***Abstract*—Obtaining a parking spot is a significant obstacle for many citizens. With the massive increase in the number of vehicles in crowded cities, finding an unoccupied parking slot may consume time and effort and result in more traffic jams. Travelers often utilize conventional methods to locate a parking space near their destination. They navigate the entire area to locate the nearest open space. The mission is indeed aggravating and difficult. Due to these issues, "Smart Parking Navigator," an IoT fog-based smart parking system, is developed to facilitate searching available roadside parking spaces. The system monitors a predetermined parking space and provides users with real-time data about the occupancy of slots. The system employs image processing techniques, object detection algorithms, and OpenCV to determine whether a parking space is occupied or vacant. A Raspberry Pi camera connected to a Raspberry Pi 4 computer is used for real-time detection. Each parking space's status is stored in a Firebase Realtime Database. A mobile application for Android has been developed to assist clients in locating available parking. The application is connected to the database to retrieve the real-time status of each position. Once the user launches the application, the interface will visualize the condition of each parking slot. The solution is a cost-effective alternative to specialized and costly systems that has the potential for future commercial development.**

Keywords— IoT, Raspberry Pi 4, Raspberry Pi camera, OpenCV, Android, Object Detection Algorithms, Firebase Realtime Database.

I. INTRODUCTION

In contemporary times, there has been a significant surge in the number of personal automobiles globally, leading to insufficient parking spaces for the populace. According to CEIC Data [1], as of December 2019, the State of Palestine (West Bank and Gaza) reported 268,338 registered vehicles. In 2018, an increase of 15,227 units was observed from the preceding number. In rush hours, vehicle owners struggle to find unoccupied parking areas with busy traffic flow. This issue is frequently observed in Bethlehem City, defined as a small urban area where households may possess multiple personal vehicles. When an individual intends to venture outside their residence with their vehicle, the primary consideration that arises pertains to the location of parking.

Individuals frequently navigate their intended location multiple times for an available parking space. Despite the occasional availability of parking spaces, drivers often encounter significant delays in locating the closest unoccupied parking spot near their intended destination. This lack of knowledge is a big hassle and time-consuming. Not only do drivers waste time, but they also waste fuel, resulting in increased traffic congestion that can have a negative impact on the environment [2].

Using Internet of Things (IoT) applications is a crucial approach for implementing intelligent parking systems, as it has been observed to experience significant growth in this domain. The Internet of Things (IoT) infrastructure facilitates the interconnection of various devices, including sensors, cameras, and controllers, through the Internet to exchange data and execute necessary operations. It also provides real-time information gathering and analysis using accurate sensors and seamless connectivity. IoT technology in smart cities has yielded numerous benefits, including the facilitation of parking spot identification for drivers, enabling them to make informed and effective decisions [3]. Unfortunately, these options are costly and rely on intricate machine learning algorithms. In addition, some solutions need the installation of different devices in the parking areas or even in roadside parking. Therefore, cameras and image processing seem the most cost-effective solution, as one installed top-viewed camera can detect empty parking lots for a wide area. This solution is deemed the most promising and appropriate approach to enhance the present circumstances.

In this research, we implemented a "Smart Parking Navigator" system to locate the availability of roadside parking and display the results to end-users using a mobile application. Using a Raspberry Pi camera in conjunction with a Raspberry Pi 4 computer enables immediate notifications about the status of parking spaces, specifically regarding their occupancy or availability. Utilization of fog-based digital image processing and OpenCV is employed to analyze the image through the Raspberry Pi. The Firebase Realtime Database is utilized to store and maintain the status of each parking. The mobile application for the Android platform retrieves the current status of parking spaces from a database and presents the information in real-time on the application's user interface.

The paper is organized as follows. Section II is dedicated to presenting the relevant literature. Section III covers the system architecture and its components, while Section IV pertains to the implementation phase. The outcomes of the study are clarified in Section V, while Section VI provides the paper's conclusion.

979-8-3503-3761-7/23/$31.00 ©2023 IEEE

II. LITERATURE REVIEW

A wide variety of Internet of Things (IoT) solutions exist for parking systems, each with distinct advantages and limitations. Various sensor-based methodologies have been employed, including magnetic sensors [4], microwave radar [5], and ultrasonic sensors [6]. The installation expenses of implementing such solutions in off-street parking areas are substantial.

Loong and colleagues [7] developed an Internet of Things (IoT) smart parking system that utilizes camera images and machine vision to monitor parking spaces. The system is capable of receiving the current parking status in real time. A 5-megapixel camera was utilized in conjunction with a Raspberry Pi 3. The software was developed using the Python 2 programming language and the OpenCV computer vision library. The data corresponding to the subject matter was uploaded onto the Ubidots platform for monitoring. A mobile application for Android was developed to enable users to access real-time data on parking information.

Khanna and colleagues [3] devised an intelligent parking system to facilitate locating a parking spot for users. The researchers utilized a Raspberry Pi 3, a Python Integrated Development Environment (IDE) for software development, an ultrasonic sensor, a Pi Camera, and a Firebase server for the purpose of implementation. Users utilize mobile phone applications to evaluate the availability of parking spaces and make payments through PayPal. A constraint of the system is that using online banking alternatives is not feasible, thereby restricting users from solely employing PayPal as a payment method.

Sobeslav and colleagues [8] conducted a pilot implementation of an Internet of Things (IoT) smart parking lot system utilizing a mini-PC platform, sensors, and IQRF technology. The implementation of a cost-effective solution enables the recording of the occupancy status of parking spaces and the storage of this data in a real-time database. The Geofencing service allows the driver to ascertain the number of available parking spaces through a mobile application on an Android device, manually or automatically. The operator can reserve a parking spot for 60 minutes.

Moreover, Hakim et al. in [9] used image processing with the Haar-Cascade method to implement Smart Parking System. The Raspberry Pi was utilized to execute image processing to detect available parking spaces. This was achieved through its interface with the Firebase cloud platform. The data is transmitted to the Firebase channel by the system. The driver can visualize the data through the phone of the car operator. The system's efficacy was evaluated in a basic parking lot and yielded satisfactory results. The omission of the gap distance between vehicles has resulted in the camera detecting a mismatch due to the perceived proximity of the cars. Moreover, the presence of shadows underneath the vehicles resulted in inaccuracies in detection. Furthermore, it should be noted that there has been no development of a mobile application.

Our proposed system exhibits superior performance compared to prior methodologies due to its simplicity and lack of necessity for complicated configurations. Employing a singular Raspberry Pi camera with an eagle view positioned atop the parking lot is a financially efficient resolution. The camera is placed on the highest point available to detect outdoor roadside parking for a wide area. Furthermore, the Raspberry Pi 4 module is utilized for image processing as it detects stationary objects. Monitoring a low-traffic environment, such as when a car park in or leaves its place, is considered a slow-moving object environment. Thus, the computational power of the Raspberry Pi 4 is sufficient for the aforementioned solutions.

Hence, the video collected by the camera connected to the Raspberry Pi is analyzed within the Pi itself using an object detection algorithm, OpenCV, and image processing techniques. Real-time video processing can be achieved without the requirement of connecting to a centralized cloud server, thereby mitigating potential delays. The fog model is utilized to bring down the computation capabilities of the conventional cloud-based system to the local network. The transmission of raw data over network connections is minimized, reducing network data burden [10]. The proposed solution does not involve the storage of videos captured by the camera, as they are promptly deleted. Therefore, the concept of privacy is understood as the absence of any storage of confidential information. Our primary objective is to ascertain the status of individual parking slots without any accompanying aggregate data. The Firebase real-time Database is utilized solely to store the status of individual slots. Upon a client's utilization of the mobile application to inquire about parking availability, the application retrieves the relevant information from the database. The resulting analysis will provide information regarding the total parking spaces and their corresponding occupancy statuses. Hence, the utilization of the fog computing model is favored in this approach, as it results in energy conservation and decreased computational expenses.

III. THE PROPOSED SYSTEM ARCHITECTURE AND COMPONENTS

The "Smart Parking Navigator" system consists of four main components, as illustrated in Fig. 1. The components are the hardware components, the software, the database, and finally, the Android mobile application.

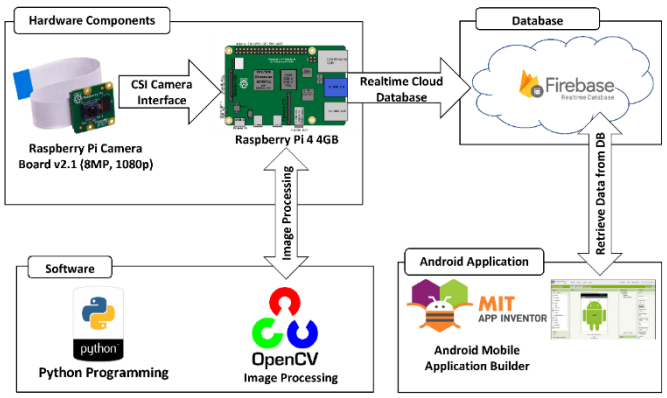


Fig. 1. The overall system architecture

A. *The hardware components*

The hardware configuration comprises two primary devices: the Raspberry Pi camera and the Raspberry Pi 4 module. The Raspberry Pi camera is a sensor for image processing and

machine learning. It is connected to the Raspberry Pi 4 module through the CSI interface [11]. The camera is situated in an elevated position to provide a bird's-eye view of the parking spaces, thereby ensuring a fixed orientation during video recording. The system is designed to maintain continuous surveillance over the parking slot area through a camera, which is programmed to automatically detect the availability of parking spaces.

The Raspberry PI is a compact computing device that operates on the Linux operating system, providing a comprehensive computing experience. It is an inexpensive card-sized computer. The Raspberry Pi project is designed to replace an ever-more-complex "closed box" station [12]. The Raspberry Pi is equipped with a processor, system memory, network interfaces, a memory card slot, and various ports for connecting sensors, peripherals, and other devices. According to reference [13], it offers a favorable balance between expenses, dimensions, energy consumption efficacy, development promptness, and computational capacity. This platform is highly recommended for the implementation of Internet of Things solutions. The Raspberry Pi 4 is energized through micro-USB utilizing a conventional power supply rated at 5.1V and 3.5A in our study. A wireless local area network (WIFI) facilitates data transfer into the Firebase database, which operates in real-time and is designed for the Internet of Things (IoT).

B. The software

The utilization of the Raspberry Pi camera enables the real-time monitoring of parking slots. Consequently, the current video data is subjected to processing and analysis through object detection algorithms, image processing techniques, and OpenCV's pre-existing libraries. The algorithm for processing system images has been implemented using the Python 3.9 programming language on the Raspberry Pi 4 module.

C. The Firebase Realtime Database

The Firebase Realtime Database is a service offered by Google. The database is hosted on the cloud and is available at no cost. The status of each parking slot is stored utilizing this system. The information is stored in the JavaScript Object Notation (JSON) format and updated in real-time for every user. End-users can access the data through various application platforms, including Apple, Android, and JavaScript SDKs. Users can access the database through their Android application, enabling them to receive automatic updates on the status of parking slots [14]. Utilizing the Firebase database with Wi-Fi technology is a proficient and dependable wireless communication methodology for data transmission. Fig. 2 represents the Firebase Realtime Database for the "Smart Parking Navigator" system.

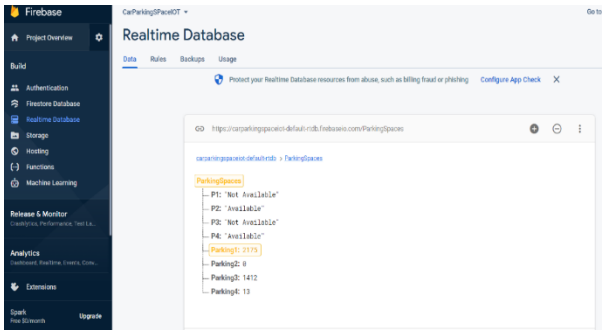


Fig. 2. Firebase Realtime database for "Smart Parking Navigator" system

D. The Android app "Smart Parking Navigator"

Using MIT App Inventor, we created a simple Android smartphone application that shows the status of the four parking spaces. The MIT App Inventor is a free drag-and-drop visual block-based programming tool for creating fully complete mobile apps for Android [16]. There are several available components that one can include in the phone screen interface. Some are visible such as text boxes and drawing canvas, whereas others are non-visible items, such as the camera and database in our system [15].

Part of the code blocks for the mobile app is shown in Fig. 3. For each of the four parking spaces, we defined four variables whose values vary depending on whether the space is available or occupied.

A mobile platform must be created to simplify the use of our service and make it simpler for consumers to read unoccupied parking spaces instantaneously. Fig. 4 displays the "Smart Parking Navigator" program's interface. The four parking labels on the interface are green when slots are available. If not, they are highlighted in red. When the application is online, the status of each slot is automatically updated. The program uses Firebase Realtime Database to get all the data in real time.

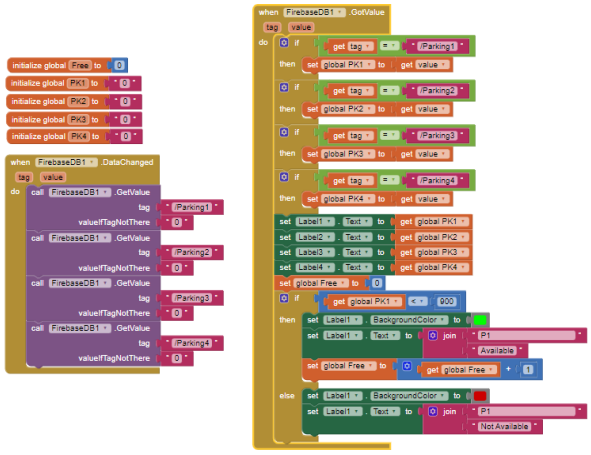


Fig. 3. A part of the mobile app program block.

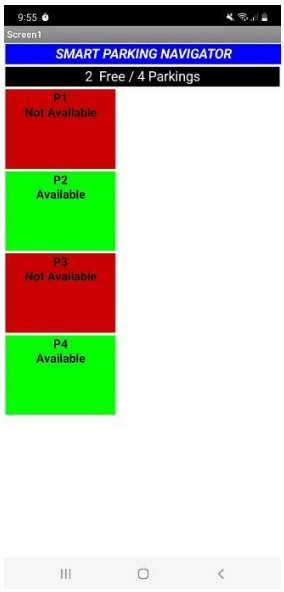


Fig.4. Mobile interface

IV. THE PROTOTYPE IMPLEMENTATION

We built a prototype that can detect the occupancy of four parking slots, P#1, P#2, P#3, and P#4, as shown in Fig. 5. The Raspberry Pi camera is positioned in an eagle view above the slots. It is fixed in a static position to determine the parking area boundary for each slot, as any camera movement will cause errors and need to be reconfigured again. The camera is connected to the Raspberry Pi 4 module to analyze real-time videos. The Raspberry Pi is connected to the Internet using Wi-Fi to stream the status of the parking slots into the Firebase Realtime Database.

Fig.5. Smart Parking Navigator Prototype

A. *The Data Flow diagram*

The system's data flow diagram is shown in Fig. 6. We begin by launching the software on the Raspberry Pi 4 module. The camera is then placed above the parking lot and turned on. The live video has now loaded the designated parking space. The predefined parking spaces are specified once in the initial setup, where we mark our parking areas in a rectangular shape as a boundary for each slot. Then, to make the detecting process more accessible, we crop each parking space into a single image. The object recognition and image processing methods used in our suggested methodology are then presented.

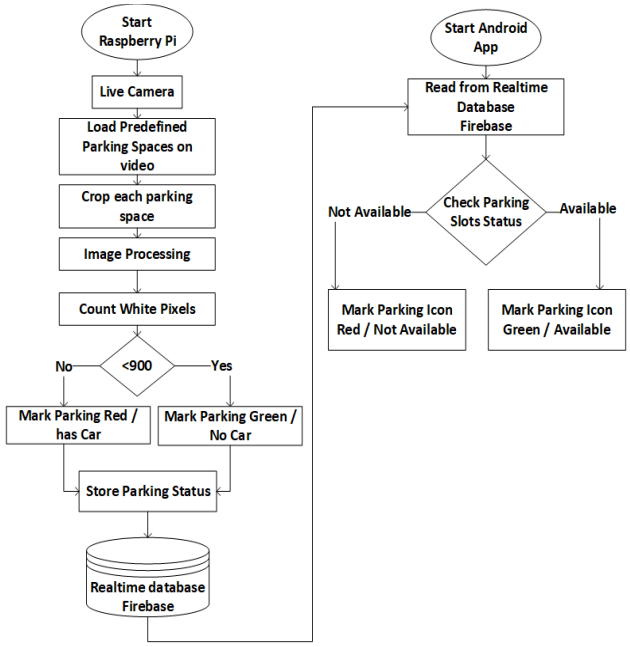


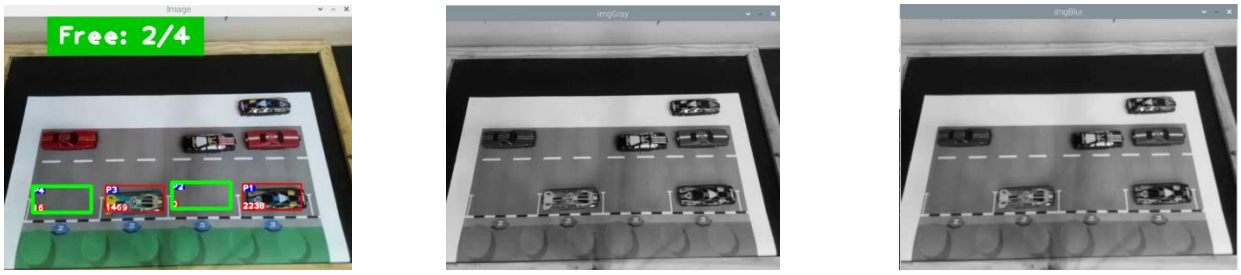
Fig. 6. The Data Flow diagram

B. *Object Detection using image processing*

Object detection algorithms [17] and image processing methods determine whether a parking space is occupied. The model identifies objects based on a determined threshold. The image is first converted to binary format, and then each image's coordinate points are determined. Below is a description of how object detection is implemented in our suggested system.

As we clarified, each parking slot is processed as a separate image. As a result, complexity and detection time are decreased. Evaluating the entire image would take time and money because the camera can detect more figures than just the cars in the parking lot. Our system's ideal approach is to crop the slots into independent photos after defining the boundaries with a rectangle shape. We preprocess each image to make the detection phase easier. We first convert the image to greyscale to reduce noise and then apply a blur effect. Each image is then subjected to a threshold to turn it into a binary "black and white" image. Once more, the median image function can remove any noise and extra white pixels from the image. To facilitate counting the pixels, a dilate function is applied to the thickness of the edges. We can now count the number of white pixels [17]. The number of 900 white pixels in each image within the rectangle is sufficient to distinguish between the two statuses of each slot, as determined by a series of tests and experiments. To ensure the detection of only huge objects, we considered the threshold of 900 white pixels. This means that moving humans, bicycles, or motors are not considered large objects in this solution. Our main aim is to reserve the slot only when cars are in it. Therefore, the parking space is considered available if the number of white pixels is lower than 900. Apart from that, it is occupied.

Each slot's state is automatically updated in the Firebase database. The state of each parking space is synchronized with each client when a user launches the application. As a result, the mobile interface will turn each parking space's color from red to green, depending on whether it is occupied or not. The image processing procedures for the parking lot are shown in Fig. 7.



a. Live Image

d. Gray Scale

c. Image Blur

b. Image Threshold

f. Image Median

e. Image Dilate

Fig.7. The image processing steps for the parking area

V. RESULTS

The following is an accurate assessment of the implementation of the system. The positioning of the vehicles within the slot is illustrated in Fig. 8(a). Three cars are positioned at P#1, P#2, and P#4. Whereas P#3 is empty. According to the data presented in Fig. 8(b), it can be observed that there is only one available slot.

Furthermore, it is worth noting that parking spaces P#1, P#2, and P#4 are marked by a red rectangle, indicating their current occupancy status. In contrast, a green rectangle bounds P#3 as it is free. The mobile application interface depicted in Fig. 8(c) displays P#3 in green to indicate its unoccupied status, while the other parking spots are shown in red to signify its occupied status. The three pictures are synchronized in time for each client using the Android app.

The implementation results of the "Smart Parking Navigator" system demonstrate its effectiveness in accurately detecting the occupancy of parking spaces and providing real-time updates to users via the mobile application interface. Using image processing and object detection techniques in the system enables the proficient and dependable detection of parking slots, thereby reducing complexity and detection duration. The system's precise identification of parking space availability can aid individuals in conserving time and energy while searching for and reserving parking spots. This can lead to the enhancement of parking management and user experience.

VI. CONCLUSION

A cost-effective "Smart Parking Navigator" system has been proposed, implemented, tested, and discussed in this paper. The IoT-based system detects the presence of a car in a designated roadside parking space using object detection algorithms and image processing techniques. On Firebase Realtime Database, each slot's state is made accessible online. The data can be accessed by users using the Android mobile application designed. Users may then immediately find out where the parking spaces are. Due to their efficiency and small weight, the Raspberry Pi 4 module and Raspberry Pi camera were used to construct the prototype hardware. Concerning data privacy, the statuses of each slot are stored in the Firebase database with reading/ write permissions. The data read permission is allowed for any client who uses our mobile application.

The system's implementation has been planned for future commercial growth. The image processing method is regarded as quick and reliable. Any video camera can be used to evaluate images of many parking spaces, expanding the potential of this technology. Once the camera is positioned at a high point over any parking area, considering the measurement parameters, the system can detect the status of each predefined parking slot. Although the Firebase Database is reliable and secure, the Firebase Security Rules are advised for commercial development to enable more comprehensive and customizable data protection and to secure the entire system.

a. Placement of the cars in slots

b. The status of the slots

c. Mobile Application

Fig. 8. Testing environment

To make the system accessible to iPhone users, it would also be appropriate to create an iOS app.

Finally, the mobile application could be commercialized and made available in the Google Play store for free download and use, giving the public access to real-time parking status information.

REFERENCES

[1] CEIC. (2012). CEIC Database

[2] Rehman, M., & Shah, M. A. (2017, September). A smart parking system to minimize searching time, fuel consumption and CO2 emission. In 2017 23rd International Conference on Automation and Computing (ICAC) (pp. 1-6). IEEE.

[3] Khanna, A., & Anand, R. (2016, January). IoT based smart parking system. In 2016 International Conference on Internet of Things and Applications (IOTA) (pp. 266-270). IEEE.

[4] Nadzri, N., Hoe, C. K., Saari, M. M., Razali, S., Daud, M. R., & Ahmad, H. (2018). Vehicle Detection System using Tunnel Magnetoresistance Sensor. In Intelligent Manufacturing & Mechatronics (pp. 547-555). Springer, Singapore

[5] Prophet, Robert, Marcel Hoffmann, Martin Vossiek, Gang Li, and Christian Sturm. "Parking space detection from a radar based target list." In 2017 IEEE MTT-S International Conference on Microwaves for Intelligent Mobility (ICMIM), pp. 91-94. IEEE, 2017.

[6] Lee, C., Han, Y., Jeon, S., Seo, D., & Jung, I. (2016). Smart parking system using ultrasonic sensor and bluetooth communication in Internet of things. KIISE transactions on computing practices, 22(6), 268-277.

[7] Loong, D. N. C., Isaak, S., & Yusof, Y. (2019). Machine vision based smart parking system using Internet of Things. Telkomnika, 17(4), 2098-2106.

[8] Vladimir Sobeslav, Josef Horalek, "A Smart Parking System Based on Mini PC Platform and Mobile Application for Parking Space Detection", Mobile Information Systems, vol. 2020, Article ID 8875301, 15 pages, 2020. https://doi.org/10.1155/2020/8875301

[9] Hakim, I. M., Christover, D., & Marindra, A. M. J. (2019, April). Implementation of an image processing based smart parking system using Haar-Cascade method. In 2019 IEEE 9th Symposium on Computer Applications & Industrial Electronics (ISCAIE) (pp. 222-227). IEEE.

[10] Vaquero, L. M., & Rodero-Merino, L. (2014). Finding your way in the fog: Towards a comprehensive definition of fog computing. ACM SIGCOMM computer communication Review, 44(5), 27-32

[11] Raspberry Pi Foundation. Raspberry Pi Camera Module Documentation, 2016

[12] Vujović, V. and Maksimović, M. 2015. Raspberry Pi as a Sensor Web node for home automation. Computers & Electrical Engineering, 44, pp.153-171

[13] Mahmoud, Q.H., Qendri, D. and Lescisin, M. 2016, February. The Sensorian Shield: Transforming the Raspberry Pi into an IoT Platform. In Proceedings of the 47th ACM Technical Symposium on Computing Science Education, p.162. ACM

[14] Wu-Jeng Li, Chiaming Yen, You-Sheng Lin, Shu-Chu Tung, and ShihMiao Huang, "JustIoT internet of things based on the firebase realtime database". IEEE journal. 2018.

[15] Patton, E. W., Tissenbaum, M., & Harunani, F. (2019). MIT app inventor: Objectives, design, and development. In Computational thinking education (pp. 31-49). Springer, Singapore.

[16] "Blockly | Google Developers," Google. [Online]. Available: https://developers.google.com/blockly/.

[17] Tuncer, T., Avci, E., & Çöteli, R. (2015, May). A new method for object detection from binary images. In 2015 23nd Signal Processing and Communications Applications Conference (SIU) (pp. 1725-1728).