Internet of Things 14 (2021) 100387

Contents lists available at ScienceDirect

Internet of Things

journal homepage: www.elsevier.com/locate/iot

# An IoT Raspberry Pi-based parking management system for smart campus ☆

Waheb A. Jabbar*, Chong Wen Wei, Nur Atiqah Ainaa M. Azmi, Nur Aiman Haironnazli

*Faculty of Electrical and Electronics Engineering Technology, Universiti Malaysia Pahang, 26600 Pekan, Pahang, Malaysia*

ARTICLE INFO

*Article history:*
Received 17 September 2020
Revised 12 February 2021
Accepted 5 March 2021
Available online 31 March 2021

*Keywords:*
Blynk
IoT
Parking Management
Pi camera
Raspberry Pi
Smart Campus
Ultrasonic Sensor

ABSTRACT

Parking slots have become a widespread problem in urban development. In this context, the growth of vehicles inside the university's campus is rapidly outpacing the available parking spots for students and staff as well. This issue can be mitigated by the introduction of parking management for the smart campus which targets to assist individually match drivers to vacant parking slots, saving time, enhance parking space utilization, decrease management costs, and alleviate traffic congestion. This paper develops an IoT Raspberry Pi-based parking management system (IoT-PiPMS) to help staff/students to easily find available parking spots with real-time vision and GPS coordinates, all by means of a smartphone application. Our system composes of Raspberry Pi 4 *B+* (RPi) embedded computer, Pi camera module, GPS sensor, and ultrasonic sensors. In the IoT-PiPMS, RPi 4 *B+* is used to gather and process data input from the sensors/camera, and the data is uploaded via Wi-Fi to the Blynk IoT server. Ultrasonic sensors and LEDs are exploited to detect the occupancy of the parking spots with the support of the Pi camera to ensure data accuracy. Besides, the GPS module is installed in the system to guide drivers to locate parking areas through the Blynk App. that discovers parking spaces availability over the Internet. The system prototype is fabricated and tested practically to prove its functionality and applicability. According to the results, the IoT-PiPMS can effectively monitor the occupancy of outdoor parking spaces in the smart campus environment, and its potency in terms of updating the data to the IoT server in real-time is also validated..



## 1. Introduction

The growing number of automobiles is rapidly outpacing the parking lots availability in urban areas; thus, parking has become a widespread challenge in metropolises development [1]. This problem can be observed clearly in cities, especially centers, shopping malls, open markets, government offices, hospitals, schools, and universities [2]. There are two main types of parking lots, indoor and outdoor. In conventional parking management systems, the information about parking spots cannot be efficiently collected and updated to the management platform [3,4]. In such systems, many parking spaces exist as local data silos, but their parking spot information cannot be shared remotely with drivers [5]. Additionally, data of isolated parking cannot be transmitted to a unified platform for city parking management.

The code (and data) in this article has been certified as Reproducible by Code Ocean: (https://codeocean.com/). More information on the Reproducibility Badge Initiative is available at https://www.elsevier.com/physical-sciences-and-engineering/computer-science/journals.

☆ This work was supported by Universiti Malaysia Pahang (UMP), under the grant scheme number RDU190304.

* Corresponding author.
*E-mail address:* waheb@ieee.org (W.A. Jabbar).

https://doi.org/10.1016/j.iot.2021.100387

A smart parking concept has been introduced to mitigate the difficulties of parking spot management in a smart environment like the smart city [3,6,7]. Smart parking systems have been given considerable attention in the literature to assist individually match drivers to parking slots, enhance the utilization of parking spaces, reduce the cost of management, and alleviate congestion of traffic [8–10]. However, the existing smart parking systems still possess many constraints related to architecture, connectivity, and deployment. Existing systems that are deployed for smart parking use different types of wireless networks for communication between sensor nodes and a management platform server [11]. This requires additional gateways, thus increases system complexity and makes it hard to maintain and impedes the scalability of smart parking. This severely constrains the deployments of smart parking systems in most areas when necessary. The foliage effect on path loss, shadowing, and multipath dispersion.

Due to the increasing number of students/staff who drive to the university campus, the number of vehicles inside the campus also grows. Therefore, parking spaces are getting more occupied than ever. This problem poses a daily challenge to both students and staff alike, as they have to make many rounds just to find a vacant parking space. As a result, drivers would waste their precious time and energy where they might be late for their work or classes. The parking management system based on the Internet of Things (IoT) technology has great potential to overcome the stated issues faced by the conventional systems of parking management. Owing to the characteristics of the university campus with regard to size, the Internet infrastructure, and the existence of open spaces as outdoor parking lots, Raspberry Pi-based IoT system is a potential candidate to enable parking management for the smart campus. Recently, universities including our university (Universiti Malaysia Pahang) have devoted more focus to the smart campus paradigm in particular with the emergence of the Fourth Industrial Revolution (IR 4.0). Several initiatives have been started under the smart campus project, smart parking management is one of them. Thus, discovering parking spots around the university campus in real-time is essential for the convenience of staff, students, administrations, and even visitors. Furthermore, not only for convenience purposes, but such a smart system can help vehicles owner within the campus in time-saving, pollution reduction, and efficient resource utilization. In this context, the use of IoT networks will allow sharing of data about the outdoor parking slots of the campus over the Internet; thus, helps both drivers and security management.

Since parking monitoring needs detection of vehicle existence, multiple sensors should be deployed in parking spaces. For the sake of system accuracy, different sensor types should be combined depending on the desired data to be measured, such as distance, coordinates, and vision. Three sensors are exploited, which are ultrasonic, GPS, and Pi camera for video streaming as an add-on system function. One ultrasonic sensor can be placed per each parking spot, while one GPS sensor and one Pi camera can cover several parking slots, depending on components features and the measured parameters. Raspberry Pi as a single board computer would provide both higher efficiency and lower cost, especially for large-scale smart campus deployments. The Raspberry Pi with the embedded Wi-Fi interface can be used to easily connect to widespread hotspot internet coverage inside the campus and performs as the gateway of the IoT system. The collected data from various sensors will be uploaded to the Blynk IoT server and can be accessed via mobile apps.

In this work, an IoT Raspberry Pi-based parking management system (IoT-PiPMS) for smart campus or any similar environments is developed as a solution for monitoring and managing outdoor parking slots in real-time. We have also proposed and integrated three algorithms that gather the required measurements from the sensors to detect the occupancy of the parking spots. The developed IoT-PiPMSA system is attached to a parking space prototype that has been designed and fabricated for testing and validation purposes. The IoT-PiPMS is applicable for outdoor parking facilities including, but not limited to universities, schools, colleges, markets, hospitals, and shopping malls. The key benefits of the proposed system include detection accuracy enhancement of the system with an ultrasound sensor and Raspberry Pi Camera, simplicity of microcontroller system, can be operated in areas within Wi-Fi coverage, and the GPS support enables ease of locating parking facility. The number of available parking slots will be detected by placing an HCSR04 ultrasonic sensor on each parking slot and also using the Pi camera for visual detection. The available parking slot will be updated to the Blynk IoT server and can be accessed by both users and the management office. The major contribution of the current study includes four aspects:

i) We have designed an IoT-based parking management system using Raspberry Pi 4 *B+* (RPi) to detect the occupancy of the outdoor parking space and update the information to an IoT platform for the smart campus use case.
ii) We have fabricated the developed system as a prototype to emulate the real situation of outdoor parking lots for testing and validation purposes by implementing several sensing nodes with one Pi camera and a GPS sensor.
iii) We have integrated three Python algorithms to gather data from the deployed sensors and send them to the Blynk IoT server in real-time. The status of parking slots will be displayed in a mobile-based IoT dashboard to be used by staff, students, visitors, and management as well.
iv) We have reported the results of the field trial experiments conducted using a real environment with end devices. The obtained results provide an insight into the system functionality and applicability in a smart campus.

The remaining part of this paper is organized as follows: Section II presents background and related works. Section III describes system design and fabrication. Section IV presents system implementation. System functionality testing and results are explained in Section V. The conclusions are drawn up in Section VI.

## 2. Background and related works

With the development of the suburban area, the spaces around the cities and rural areas are being used up to build commercial zones and skyscrapers. However, even with the increase in the size of the city, the available parking slots are always scarcity. Because of this development, many issues were raised in terms of transportations in the city, such as traffic congestion, limited car parking facilities, and road safety. Due to these problems, many smart parking systems are established in the market.

Most of the parking systems over the past few years in the literature provided solutions to the design and development of parking availability information system, parking reservation system, occupancy detection, and parking lot management. However, few works emphasize on the authentication of the vehicle's clearance to access the parking space in the first place. The idea of creating parking management for the smart campus is becoming possible with the emergence of the IoT paradigm. In the last two decades, modern communication technologies and micro controllers have promoted a smart living style that looked practically inconceivable. IoT is one of the main drivers of future smart spaces. The term IoT has been more inclusive, covering a wide range of applications like healthcare, utilities, transport, and so on [12]. The IoT enables new operational technologies of different applications and also offers environmental benefits. With IoT, spaces are evolving from being just 'smart' to become intelligently connected. IoT can be applied in many ways, such as smart campus, traffic control, smart vehicles, smart cities, home automation, and healthcare monitoring.

In fact, a smart campus concept came out with the same principles of smart city applications. The smart campus is developed on the basis of digital campuses. Compared with the traditional digital campus, the smart campus provides services in a timely manner, reduces effort, and cuts operational costs. Smart campus implies that the institution will adopt advanced technologies to automatically control and monitor facilities on campus and provide high-quality services to the campus community [13]. This application of smart campus will lead to an increase in efficiency and responsiveness of the campus; thus, better space utilization, decision making, and campus community experience. In the past decade, many initiatives have been placed on improving campus services. For instance, many universities worldwide invested a lot of money in campus development as part of a smart campus scheme involving multiple sites and structures to saving energy consumption, reducing CO2 emissions, and increasing sustainability and interaction.

Basically, searching for parking space more painful than ever for drivers and prompts to develop traffic congestion and excess consumption of fuel [14]. Smart parking solutions to provide information about the available parking lot have attracted considerable attention from both academia and industry. These systems work by collecting real-time data on the available slots in a parking area using the sensors scattered in the parking lot. That information will then be updated continuously, and the result will be displayed in the software application regarding the availability status of the parking lot and the location of it. For example, there are 3 parking lot. Sensors are deployed in the parking lot to scan the vehicle that enters the parking lot. The total amount of vehicles in the parking lot is revealed on the board. Next, the information will be processed by the system that provides information to the users about the available parking lot [15]. In this section, we review some recently published relevant studies on IoT-based smart parking systems.

In [16], an automatic smart parking system based on IoT is proposed. An 8-bit PIC16F73 CMOS FLASH-based microcontroller is used to read the data and send it to the IoT web page through Wi-Fi. An Espressif Systems Smart Connectivity Platform (ESCP) Wi-Fi module is used to connect the microcontroller to the Internet. An IR proximity sensor is used to detect parking occupancy. The microcontroller complexity is the main limitation. It needs an extra module for Wi-Fi connection. Moreover, a limited number of sensors can be connected to the controller. Ref. [17] proposed an IoT-based smart parking system using ultrasonic sensors that are connected wirelessly to an RPi via ESP8266 Wi-Fi chip. The RPi is used for processing sensor data and transmitting it to the IBM MQTT Server. The server maintains information including parking entry time, parking duration, payment required, and payment mode so that users could access the information through their mobiles. However, the system relies on one sensor type which may affect data accuracy. Besides, it requires a subscription to the IBM MQTT server.

In [18], the authors proposed a smart parking prototype by utilizing ultrasonic distance sensors and a Pi camera that is connected to Raspberry Pi. The system aimed to identify plate numbers of vehicles, and the details are sent to the central parking system mediator. A smart parking mobile client application is used to fetch details over the cloud. The system is locally operated and used for management only where the users cannot access the information about available parking. Another smart parking for occupancy monitoring and visualization was suggested by [19]. The authors used Arduino Uno as a microcontroller to collect data from ultrasonic sensors that were equipped with XBee modules for internet connectivity and deployed at each parking slot. The sensors were wirelessly connected to a local XBee gateway that provided with an Internet connection and allowed the sensors to report their status to the remote MySQL database.

The authors of [20] built a prototype that allows users authentication for a valid booking parking slot. An infrared IR sensor was used to detect the vehicle while Raspberry Pi was utilized as an embedded controller. The Amazon web server was exploited as the IoT server to maintain the database. An Android-based mobile application was developed for booking parking and a DC gear motor was used to control the operation of the parking gate according to the entered booking ID. LEDs were installed as in the conventional indoor parking on top of the parking slots to indicate whether the parking slot is available. The system was not practically implemented or tested. A similar study in [21] allows guaranteed reservation by using QR Code authentication. The system supported the optimal allocation concept, where it allocates the nearest park-

ing slot to the user on the mobile. IR sensor and LED are used to detect the presence of the vehicle and to indicate the availability of a slot, respectively. A Raspberry Pi Model 2B is used to sending parking slot data to the NoSQL database.

In [22], a parking system with an authentication feature for reserving parking slots by integrating RFID and IEEE 802.15.4-based WSN technologies was introduced. The system also allows the payment of parking fees through NFC from the user's smartphone. The system could be used by traffic policemen who may be alerted to improper use of reserved parking space or the expiration of purchased time via an Android app. Ref. [23] proposed a smart parking system by combining the Cloud-based computation and Raspberry Pi. The system used Raspberry-Pi to communicate with the NoSQL database hosted on MongoDB. An RFID module was used to identify the user's information and to map a user to a certain parking slot. An Arduino Uno R3 acts as a bridge between the RFID module and Raspberry Pi. An ultrasonic sensor was used to detect the vehicle in a parking slot and the Google Distance Matrix API was used to get the distance between the user's location and all parking slots by sorting according to distance. Although the system has improved accuracy, the utilization of an additional Arduino microcontroller for each node will increase the system implementation cost.

Ref. [24] proposed an IoT-based E-parking system for smart cities. The proposed system aimed to detect improper parking and automatically collect parking charges using an integrated component called 'Parking Meter'. It consists of an ultrasonic sensor node, LED, camera module to snap the plate number, alarm module for warning in case of wrong parking, and a Wi-Fi module for communicating with the local parking management server, which calculates the parking charges and sends SMS with payment option to the user if payment is due. The authors of [25] constructed a prototype of an automatic smart parking system using IoT. The system is Raspberry Pi-based and contains a Pi camera for sensing the parking lot. The administrator can add/remove the number of parking slots on the central server. The system also integrates a navigation system that guides the user to the exact location of the parking slot.

For the sake of brevity, smart parking-related studies have been summarized and compared in Table 1. The proposed IoT-PiPMS system is also included in the comparison to emphasize its main features with respect to the existing systems. As it is presented in the table, the proposed system aims at overcoming the limitations of the existing systems and is supported with a real implementation scenario for validation purposes. Overall, most of the available smart parking system consists of a microcontroller, a sensing unit with communication capability, and a server. An on-site deployment sensing unit is used to monitor and signalize the state of availability of each single parking space. In some systems, a mobile application is also provided that allows an end-user to check the availability of parking space and book a parking slot accordingly. From the literature, most of the existing smart parking systems had limitations in terms of the full utilization of the new IoT technology. Even the concept of wireless sensor networks is applied, the integration of the Internet to allow data updating to the cloud is not yet considered. Thus, most of the existing smart parking systems are locally operates. Other than that, several studies focused on applying the IoT concept for parking monitoring and management. However, the existing systems have some limitations in terms of connectivity range, microcontroller type, sensing capabilities, system complexity, energy consumption, cost, and accuracy. As for the improvement, this paper proposes the use of Raspberry Pi as a single board computer with multiple sensors to improve processing capabilities. The utilization of a single microcontroller will result in reducing the overall power consumption comparing to other systems that used multiple controllers, as shown in Table 1. The Wi-Fi support of our IoT-PiPMS system comes to overcome the short communication range of some systems that use Bluetooth, NFC, and reduce the complexity of systems that are mainly based on wired networks. Our system also combines sensors and camera to increase data accuracy and improve the overall system performance

## 3. System design and modeling

In this section, the adopted methodology, research materials, system design, and fabrication are described. In the first phase, the problem statement and limitations of the existing systems were identified. After that components selection phase starts by reviewing the utilized materials in previous studies such as microcontrollers, sensors, actuators, and IoT platforms and comparing their pros and cons. Next, the system architecture is proposed and the parking system design and implementation are carried out. After that, prototype design by using Siemens NX 10 software is taken place prior to the fabrication process which followed by the functionality testing phase. System enhancement and optimization also conducted before the system finalization and performance validation. The concept that we came up with this paper is the integration of IoT-PiPMS with the concept of smart campus. Our priority in this paper is to ensure the system is functioning well both in the fabricated prototype and real deployment in the campus environment

### *3.1. Components selection*

The materials and components are chosen based on necessities and compatibility based on the conducted literature review. While conducting the review of related studies, we have surveyed multiple variations of designs and components used and also their respective function in smart parking systems. The hardware and software are decided during the component selection phase. The hardware components that are chosen are Raspberry Pi 4 Model B, Pi Camera v2.0, HSC-SR04 ultrasonic sensor, LED lights. The software tools that are chosen are Raspbian Operating System, Python, Fritzing, Siemens NX 10, Proteus Design Suite, Blynk IoT platform, and VNC. Once the component selection has been completed, the system architecture is figured out. In this subsection, an overview of the selected components is introduced.

**Table 1**
Summary and comparison of related studies.

| Sources | Multiple Sensors | Vision Monitoring | Microcontroller | Communication Interface | Local Indicator | Remote Indicator | Support Payment | Localization Support | IoT Platform |
|---|---|---|---|---|---|---|---|---|---|
| [24] | YES | YES | Arduino MEGA | Wi-Fi | YES | NO | YES | NO | NO |
| [26] | NO | NO | PIC16F73 CMOS FLASH based 8-bit microcontroller | Wi-Fi | YES | NO | NO | NO | NO |
| [17] | YES | NO | Raspberry-Pi Board | Wi-Fi | NO | YES | NO | NO | IBM MQTT Server |
| [18] | NO | YES | Raspberry-Pi Board | Wi-Fi | NO | YES | NO | NO | Restful Web Service |
| [27] | YES | YES | Raspberry-Pi Board and Arduino | Wi-Fi | NO | YES | NO | NO | NO |
| [20] | NO | NO | Raspberry-Pi Model 3B | Wi-Fi | YES | YES | NO | NO | Amazon Web Server |
| [28] | NO | NO | Raspberry-Pi Model 2B | Wi-Fi | YES | NO | NO | NO | NO |
| [29] | YES | NO | Raspberry- Pi Board | ZigBee& Wi-Fi & NFC & GPRS | YES | NO | YES | YES | NO |
| [23] | NO | NO | Raspberry-Pi Model 2B+, Arduino Uno | Wi-Fi & RFID | NO | YES | NO | YES | Amazon Web Service |
| [30] | NO | NO | Raspberry-Pi, Arduino Uno | Wi-Fi & RFID & sub-GHz CC1101 transceiver | NO | YES | YES | NO | NO |
| [31] | YES | NO | Raspberry Pi 2 Model B | ZigBee& Wi-Fi & NFC & GPRS | YES | YES | YES | NO | NO |
| [32] | NO | NO | Raspberry Pi *B+* | Wi-Fi | NO | YES | NO | YES | NO |
| [33] | NO | YES | Raspberry Pi 2 Model B | Wi-Fi | NO | NO | NO | NO | NO |
| [34] | YES | NO | Raspberry Pi | Wi-Fi | NO | YES | NO | YES | ThingSpeak |
| [15] | YERS | YES | N/A | RFID | NO | YES | NO | NO | Cloud Application, Management centre |
| [35] | YES | NO | NodeMCU | Wi-Fi | NO | YES | NO | YES | Amazon Web, Services, MQTT |
| [36] | NO | YES | Raspberry Pi | Wi-Fi | NO | NO | NO | NO | Amazon Web Services |
| [37] | NO | NO | NodeMCU, Raspberry Pi 3, Scanning barcode | Wi-Fi | NO | YES | NO | NO | Online database |
| [38] | NO | YES | Computer | Wired | NO | NO | NO | NO | NO |
| [39] | NO | YES | Raspberry Pi 3 | Wi-Fi | NO | NO | NO | NO | NO |
| IoT-PiPMS | YES | YES | Raspberry Pi 4 | Wi-Fi | YES | YES | Can be extended | YES | Blynk IoT server |

*3.1.1. Raspberry Pi 4 model B*

Raspberry Pi 4 Model B has been chosen for this project mainly due to its powerful processing capabilities as a single-board embedded computer and its ability to connect to the Internet either via Ethernet port or wirelessly using Wi-Fi or Bluetooth. Ease to connect to the Internet is one of the RPi advantages over Arduino, and it also supports a huge variety of programming languages such as Python, Java, C, *C++*, Perl, Ruby, BASIC; whereas Arduino, however, only accepts either Arduino or C/*C++*. Raspberry Pi 4 Model B is a direct upgrade over the Raspberry Pi 3 Model *B+*. It has Broadcom BCM2711 1.5 GHz quad-core 64-bit CPU with a Cortex-A72 processor, which is more efficient and much more powerful than Pi 3′s 1.4 GHz processor. The GPU of Pi 4 is capable of running comfortably compare to Pi 3 due to Pi 4 improved clock speed: 500 MHz compared to Pi 3′s 400 MHz. The RPi 4 Model B has a better CPU and GPU compared to Model 3 B, which is essential when we have decided on a visual-based detection method using the Pi camera. The Pi camera requires significant processing speed to be able to capture and process the images. Therefore, Raspberry Pi 4 is overall more suitable for our system and also able to give better performance in terms of video capturing and streaming.

*3.1.2. Pi camera*

The pi camera is selected for the proposed system since it is a camera that is built specifically for Raspberry Pi. It is very suitable to use in a situation whose light is low because of its NoIR filter. Users could just plug in the camera onto the Pi board, run a few commands then the camera is active and ready to use. The Pi camera connects directly to the GPU, which is capable of 1080p30 HD video encoding. It is also capable of snapping pictures of 5MP resolution. Due to its attachment to the GPU, it does not draw any resources from the CPU, leaving it available for other processes. USB Webcams, on the other hand, drain resources from the CPU, lowering the performance of the entire system. As for the prices, this camera only costs around 25 USD, which is a great deal in terms of features and performances it provides.

*3.1.3. HSC-SR04 ultrasonic sensor*

The ultrasonic sensor is a sensor that transmits sound waves between 25 and 50 kHz to the surrounding by detecting transmitted energy, which is reflected back to the sensor. The reflection of the ultrasonic wave, together with a signal processing module, the wave will be analyzed to detect the presence or absence of an object in the surrounding every 60 milliseconds. This sensor can be utilized by detecting vehicles and assessing the occupancy of the parking space. The ultrasonic sensor is easy to install without the need for facility closure. In the parking management system, a huge number of distance sensors are required (one sensor for each slot). The ultrasonic sensor is chosen due to its cost-effectiveness when considering the number of the required sensors. It only costs about 1 USD each, and it is still able to provide reliable results compared to other proximity sensors. In the deployed prototype, we have used four ultrasonic sensors. This sensor can utilize for detecting vehicles and assessing the occupancy of the parking space. The ultrasonic sensor is easy to install without the need for facility closure.

*3.1.4. NEO-6 M GPS module*

NEO-6 M GPS Module is a GPS module capable of supporting microcontrollers. It is user-friendly and uses up only a little space on board. It is a very useful device for locating device location. This module is chosen to allow the positioning of the parking lot location.

*3.1.5. Green LEDs*

LEDs are very efficient in energy usage; it consumes only 10% of the power of what a normal incandescent bulb would normally consume. This, in turn, reduces the power costs due to low operating power. The LED is used in this project to indicate whether a parking spot is occupied or vacant by switching its light: ON green for vacant and OFF for occupied. One LED is used for each parking slot.

*3.1.6. Raspbian*

Raspbian is a free operating system that is used for modifying or program all models of Raspberry Pi. Raspbian is pre-installed with other applications for general use, programming, and educational purposes. It supports language for Python, Java, Scratch, and more. One would install Raspbian onto a PC or Micro SD and plug it into RPi, then connect a monitor to the Pi board, allowing the Pi board to run like a normal PC. Raspbian lets users install plenty of software from its open-source software repository for free. In this study, Python is used to develop the required coding for gathering data from various sensors to the RPi and upload it to the IoT cloud.

*3.1.7. Blynk*

Blynk is an IoT platform that enables the development and implementation of smart IoT devices with ease and speed. Tree major components that make Blynk a perfect candidate for our project: apps, server, and library. The apps offered by Blynk allows user to customize the widgets shown on the interface. Once the interface is done designed, the apps would then connect to an open-source Blynk server. The server acts as a centralized cloud service that would allow communications between the devices. Other than that, Blynk provides a variety of libraries that supports multiple hardware devices such as RPi, Arduino, and ESP8266. The server allows communication through Wi-Fi, Bluetooth, BLE, USB, and GSM.

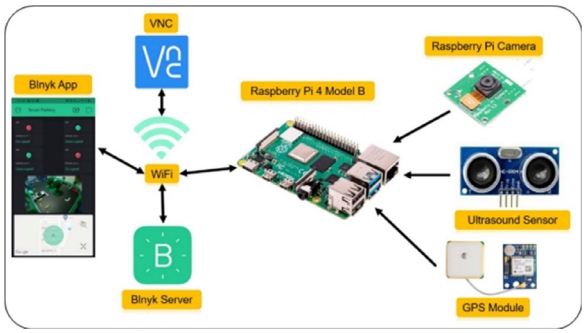


**Fig. 1.** Raspberry Pi Based Smart Parking Management System architecture.

*3.1.8. Proteus Design Suite*

Proteus is software that allows design engineers and technicians to create a sheet of electronic prints and schematics. Other than that, it also allows simulation of microcontrollers such as Atmel AVR, Arduino, or Microchip Technologies. Since there is no hardware required to use the simulation, it is convenient as a teaching tool or designing for mini-projects. We have simulated our system using Proteus to confirm the correctness of our circuit design before the real implementation.

*3.1.9. Siemens NX 10*

NX 10 is software designed for CAD/CAM/CAE. It is mainly used for designing prototypes by direct solid/surface modeling, perform engineering analysis like stress testing the prototype and also manufacture the finished design using included machining modules. The NX 10 is a direct competitor against Catia, SolidWorks, Autodesk Inventor, and Creo. The NX 10 is chosen because it has its own module for CAD, CAM, and CAE; users could transfer their project between the modules immediately with no data loss.

*3.1.10. VNC Viewer*

VNC Viewer is a client system that simply allows simultaneously access to many devices from one screen using the IP address. The VNC provides home computing environments access from everywhere on the public Web server. Thus, providing application sharing on the computer. It is usually used with RPi to access its desktop from other screens to increase the portability and mobility of Raspberry Pi. The VNC viewer is installed by default within Raspberry Pi allows users to see and operate the Raspberry Pi board desktop on their own laptop as if it is on the display monitor. Users just have to connect the Raspberry Pi and the laptop onto the same network and login to VNC on both devices.

*3.2. System Architecture*

We have decided about the layout of the system architecture as shown in Fig. 1 based on the selected components. The signal flow and the interlinks between various components are suggested. In the developed system, Raspberry Pi is the main controller that collects data from the implemented sensing units and Pi camera. The GPS module provides the location for the parking area. Then the ultrasonic sensors, which sense the vehicle presence inside the parking spot where will trigger the data of the parking slot status to the Raspberry Pi, which uploads data to the IoT server with the embedded Wi-Fi module. Thus, drivers with Internetconnected devices such as smartphones and tablets can access the information about parking slots occupancy. The users can observe and monitor the parking area through the Pi- camera that provides visual monitoring via video streaming as an add-on function when required. Alternatively, the Pi camera is acting as a security surveillance monitor to the parking area to increase the security of the users. A real-time condition of the parking area is streaming to the cloud for monitoring, and it can be used by the management office for security purposes.

*3.3. IoT-PiPMS Design*

Once the system architecture is confirmed, the electrical circuit diagram of the designed system is initiated using the Fritzing tool as shown in Fig. 2. Raspberry Pi GPIO pins are connected to the input/output pins of the three sensors (ultrasonic sensor, GPS sensor, and Pi camera. In this design, only one ultrasonic is shown. However, in the testing prototype, more sensors are used. The HSC-SR04 is used to detect the presence of a car in the slot and accordingly, the RPi will change the LED status ON/OFF that indicates the parking slot is already occupied or not. The components are installed on the breadboard to ensure that the system works successfully before transferring the component onto the solder circuit panel. The Pi camera will trigger as the parking slot is occupied or not via visual streaming. The suggested design is then simulated using Proteus software to check the compatibility of the components. Fig. 3 shows the stimulation of RPi with GPS module, Pi camera, and ultrasonic sensor in the Proteus software. After the electronics stimulation in Proteus simulation, all the electrical and electronic components were assembled and connected to the microcontroller and programmed using Python in the

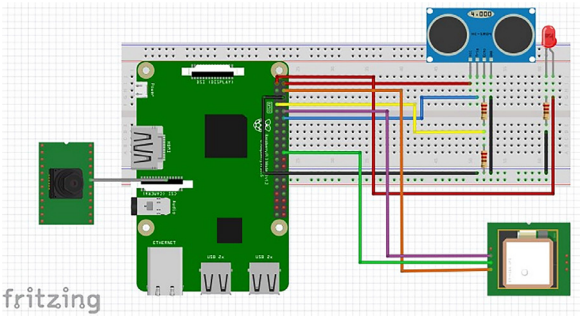


**Fig. 2.** Component circuit diagram.

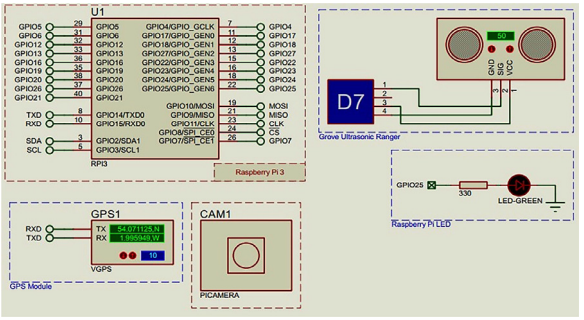


**Fig. 3.** Proteus simulation of the control circuit.

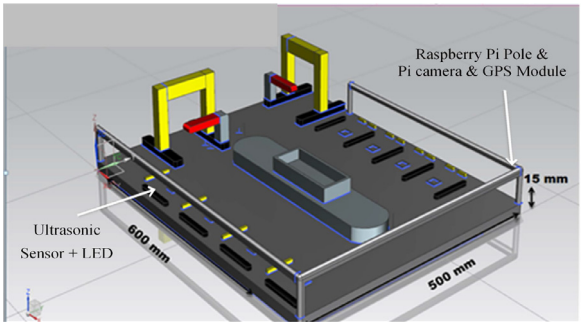


**Fig. 4.** Design and dimensions of the prototype model.

Raspbian OS. However, the Proteus software not fully utilized because some of the component libraries were not found in the software, as well as on the Internet. Therefore, the sensors and components had to be tested experimentally using RPi.

### *3.4. Model Design*

In order to emulate the functionality of the developed system, we need to design and fabricate a proper outdoor parking space model. Siemens NX 10 software is utilized to design the prototype with a measurement of 500 × 600 mm and a thickness of 15 mm, as shown in Fig. 4. The reference of measurement is taken from the basic size of a square table. The purpose of this design is to visualize how the prototype will be created and where to put all the components. This model is designed with the exit and entry gate along with 8 parking slots to make it a clearer view on how the parking slot is. Among the 8-parking slots, there are four parallel parking slots and four perpendicular parking slots. As for the Pi camera and GPS Module, it will be placed at a higher place which is at the pole that is located at the corner of the parking space. Thus, one Pi camera can cover a wider space and streams video about the situation of the covered parking lot. Also, the GPS sensor will be placed on the same pole close to the microcontroller; thus, mitigating the required wiring. On the other hand, the ultrasonic sensors will be placed facing in front of the parking slot so it can immediately detect the car as the distance between the sensor and the car becomes closer.

The RPi support pole is designed as shown in Fig. 5. We measured every edge and corner of RPi to make the box fit to it. During the design process, we left some space around the designated box to provide space for some of the wiring parts. The

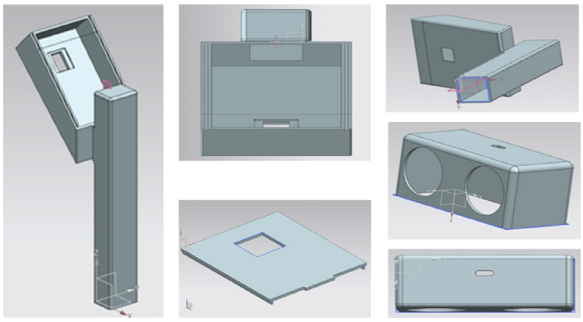

**Fig. 5.** Design of components for system model.


**Fig. 6.** The fabricated system model.

designed RPi box also has a hole for the Pi camera. For the space for the wiring part, the pole was designed with the hole inside the RPi box and along with it. In order to place the GPS module on the same support, the RPi box cover is designed together with the GPS Module hole. The function of the GPS module installed there is to help drivers locate unoccupied parking slots. This cover box will close the upper side of the RPi box to protect the components that are put inside the box. Since we also use ultrasonic sensors to detect the presence of the car, the ultrasonic sensor box is also being designed with an LED hole. The ultrasonic sensor boxes have been specifically designed for the HC-SR04 sensor. The box has two holes at the front of the box for the transmitter and receiver of the sensor and one hole on top of the box for placing LED. The function of the LED light is to indicate the presence of the car.

### *3.5. Model Fabrication*

Once the simulation is completed, the fabrication process is beginning. During the manufacturing process, the materials are assembling according to the simulated design. First of all, the process of measuring and cutting plywood takes place. Plywood is used as the base of the car parking model to strengthen the design. Next, the canvas board was used as the next base of the design. It was glued on the plywood. Both of them act as the base of the design. The function of the canvas board was to make it easier to put the hardware components in it without causing any scratch to the components. After both of the bases were glued, the other components were assembled. The next step is to create space for the car parking slots. To realize our model, the Styrofoam board and plastic board are used. To make the parking looks livelier and realistic, some fake trees and carpet grass were added. A hole was made at each parking slot for the wire to go through under the prototype. 3D printing is used to fabricate the part components of the smart parking model by using materials of PLA Filament. We successfully fabricate the model's components, including the Raspberry Pi pole, ultrasonic sensor box, RPi box cover, entry and exit parking signs using 3D printing. The fabricated model of the proposed system is shown in Fig. 6.

## 4. System Development and Implementation

### *4.1. System Setup*

In this stage of research, setup and initialization of the system components (controller, sensors, actuators) are conducted. Each component has been locally tested and configured to work with the microcontroller. Raspberry Pi 4 Model B is a single board computer, meaning it does not have the plug, program, and play abilities like the Arduino board. It has to go through

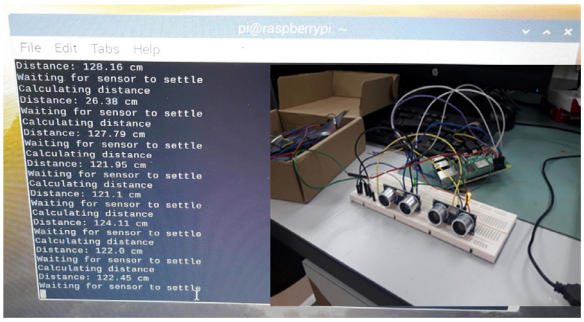


**Fig. 7.** Ultrasonic sensor configuration.

the setup and initialization phase in order to install the operating system into the Raspberry Pi. The RPi runs on an OS called Raspbian, which is a version of Linux specifically tweaked and modified for use on Raspberry Pi. Since Raspberry Pi is a computer board, it requires an output monitor screen to see the operations running inside it and also requires a storage device to save its OS and other software included within the Raspbian itself. Nonetheless, it also requires a USB mouse, keyboard, and also HDMI cable, which would output the content onto an HDMI compatible screen. All of the components above are needed to perform the initialization phase successfully. The Internet connection can be obtained either through Wi-Fi or using an Ethernet cable. It is as simple as connecting to the Wi-Fi hotspot on a desktop computer: simply chooses a hotspot and key in the hotspot password. VNC is used for operating it without a display monitor is required. We could perform our coding even using a laptop through VNC viewer into the Raspberry Pi desktop without ever needing an external display monitor. The RPi is configured by identifying its IP address so that it would automatically connect to the hotspot every time it is booted up, and we directly access the RPi through VNC Viewer.

The next step in developing the smart parking system is to test out the HC-SR04 ultrasonic sensor with the Raspberry Pi. The ultrasonic sensor is connected to the breadboard, which then is connected to the GPIO pins of the RPi. Since Python is a powerful and new programming language, it has been selected to develop the required coding to build our system. The RPi GPIO pins and sensor pins are defined according to the coding performed. In this case, Pin 3 is the Trigger, Pin 5 is the Echo, and Pin 7 controls the state of the LED. The VCC and GND pin of ultrasonic sensors are connected to Pin 2 and Pin 6, respectively. We have developed an algorithm using Python to enable the RPi to collect data from ultrasonic sensors. The algorithm measures the time taken for the ping to bounce back from the object in front back to the echo chamber, then it calculates the distance based on the time taken and multiplies it with the speed of sound. The result is the value of the distance between the object, and the HC-SR04 sensor will be displayed continuously on the terminal of the Raspberry Pi board. A predefined time interval delay between measurement updates. This is to allow the pulse from the previous measurement to dissipate before the next measurement is taken so that the results stay consistent and do not get interference from the previous measurement. The sensor is configured so that if the measured distance is more than 50 cm, it means the parking slot is empty, then the green LED will be ON; otherwise, the LED will be OFF. The condition is later changed to 8 cm to accommodate the size of the prototype model. This testing step is repeated for each of the ultrasonic sensors to ensure all of the sensors are working and gives accurate result. The testing is further expanded by connecting multiple ultrasonic sensors to the RPi. Thus, a new script has been written to accommodate multiple ultrasonic sensors and able to display the distance measured from each sensor and update to the RPi terminal. Fig. 7 depicts a side of the conducted tests for ultrasonic sensors with RPi and the displayed results on the screen.

After the testing phase of ultrasonic sensors has been completed, we conduct the testing of the GPS module. The GPS module used in this project is the Ublox Neo 6 M GPS Module. The GPS module allows the user to track down the location of the parking facilities. When the GPS module is connected to the Raspberry Pi, it requires a script to display and send its data to the terminal. Thus, we have used a Python-based algorithm to perform this task. The collected data is the longitude and latitude of the module's location. Fig. 8 displays the connection of the GPS module to the RPi and the output results that show the location longitude & latitude. The python script involved would periodically check and display those data onto the terminal.

The next step is the setup and configuration of the Pi camera. The purpose of the Pi camera is used to detect the occupancy of the parking slot. The camera will check if it is a vehicle that is parked at the parking slot. If any other object is occupying the parking slot, then the camera would update the status to the RPi. The camera will be detecting objects using an algorithm called TensorFlow Algorithm. The TensorFlow Object Detection API has a database of samples for its algorithm to compare so that any general vehicle detected can be classified as a vehicle and then further classified as either a car, a motorcycle, a truck, etc. Both the TensorFlow and Open CV are installed and setup into the Raspberry Pi. The Pi camera has its own port and can be plugged into the RPi directly. We have to enable the option for displaying the Pi camera in the Raspberry Pi configuration and run Open CV to see the live stream of the footage captured by the RPi. The Pi camera can detect the vehicles and identify their type of cars/motorcycles; the range is severely limited underground. Thus, a pole is


**Fig. 8.** GPS Module configuration.

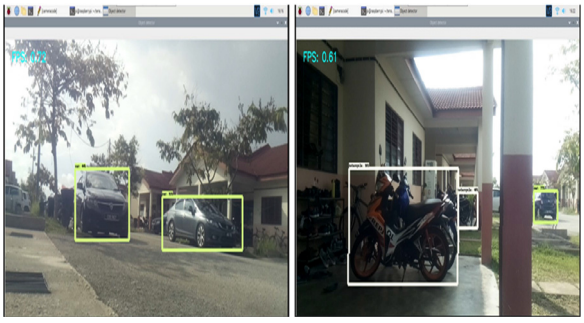


**Fig. 9.** Pi camera objects detection and classification.

designed and fabricated to allow the camera to capture footage from a height. The obtained outputs on the RPi are shown in Fig. 9.

In order to update the collected data from the sensor to the cloud, and IoT platform is required. Blynk platform is selected in our system, which is a user-friendly freemium IoT platform with multiple widgets to allow users to do and test all kinds of interesting features. The Blynk has been successfully installed into the RPi, and its features are tested. The virtual pins of the Blynk platform are configured to receive the data from the RPi. Blynk libraries for Python are used to enable information exchange between the developed system and the IoT platform, and the data will be displayed on a mobile App. Widgets. Multiple display widgets can be added for each sensor. Authorization token from Blynk App. is included in the developed Python script. In this project, we utilize the ability of Blynk App. to change its Widget's properties to indicate whether the parking slot is occupied or not. In the early phase, the widgets are shown to display an LED and the distance measured from the ultrasonic sensor. The green indicator on the dashboard would turn OFF when the distance measured is below the threshold value and turn ON when it is above it. This is first used to test out the Python script and its compatibility with Blynk. When the return value of the Python script has been successfully sent to the Blynk App., the widget will update the status according to the received value. Besides, the information received from the GPS module is received by the Blynk virtual pin, and the map widget displays location data as a dot on the map widget. This would allow the users to locate the parking slot when they are driving towards the location. Blynk App. could also stream the footage of the camera, but it does not include the TensorFlow Algorithm in its live stream. Therefore, the live streaming on the Blynk App. would act as a CCTV instead of detecting and identifying vehicle types. However, on the management server, the output of the TensorFlow Algorithm can be accessed on the RPi screen by using the VNC viewer for mobile devices or desktops.

### *4.2. System Implementation*

During the design phase, a draft design of the overall circuit is simulated using the Proteus software until the desired outcome is achieved. In the previous section, all components have been configured and tested locally by using the RPi, and no data is sent to the IoT server. In this phase, all components are composed to represent the proposed system which is implemented by connecting ultrasonic sensors, Pi camera vision detection, and GPS module to Raspberry Pi to send and/or receive data and update it in real-time to the IoT server. Firstly, these components are connected on a breadboard, as shown in Fig. 10, and then implemented in the developed model when the circuit worked properly.

The entire system is tested where the Raspberry Pi collects the data from ultrasonic sensors, Pi camera, and GPS sensor and uploads it to the Blynk IoT server; thus, we manage to access the data on Blynk App. using smartphones. Three Python scripts are developed to receive the data from the sensor simultaneously. In the first algorithm, the ultrasonic sensor detects the distance between the sensor and vehicle in the parking spot. According to Algorithm 1, whenever the trigger becomes

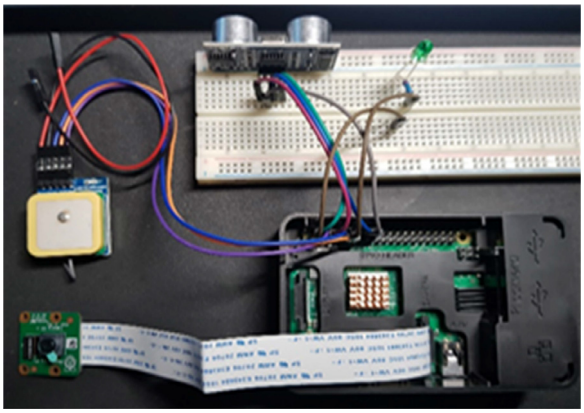

**Fig. 10.** Connection of the sensor, GPS module, and Pi camera on the breadboard.

**Algorithm 1**
Ultrasonic sensing algorithm for Raspberry Pi-based smart parking system.

Require: Real-time monitoring to identify parking slot status

Ensure: Distance to vehicles less than a predefined threshold
1: define HC-SR04 library
2: include Blynk libraries
3: define RPI.GPOI for HC-SR04 ⊃ Ultrasonic sensor pins (Trigger, Echo, $V_{cc,}$ and GND)
4: define RPI.GPOI for LED outputs
5: set $D_{th}$ ⊃ The threshold distance to identify vehicle existence
6: $D(t) \leftarrow$ *Distance measured by the ultrasonic sensor at a time, t*
7: initialize HC-SR04 ⊃ at $t = 0$
8: acquire the state (Trigger, Echo) of HC-SR04 ⊃ Start and Stop state
9: TimeElapsed = stopTime – startTime ⊃Time difference between Trigg. and Echo
10: *D(t) = (TimeElapsed * 34,300) / 2* ⊃ Where 34,300 is sonic speed
11: for each round do
12: get *D(t)*
13: if *D(t)* >= $D_{th}$ then
14: switch ON Green LED
15 update the LED status in Blynk to Green "unoccupied"
16: else
17: switch OFF Green LED
18 update the LED status in Blynk to Red "occupied"
19 return distance
20: end if
21: Send data to *Blynk Server over the Internet of RPi*
22: Retrieve data in *Blynk App* using Smartphone
23 end for

LOW, it will send a pulse from the open front of the ultrasonic sensor. When the pulse is being bounced back by the object, it will be received by the Echo receiver, which will make the Echo pin on the ultrasonic sensor to go HIGH. The distance between the object and the ultrasonic sensor is calculated by multiplying the time taken for the pulse to shoot out of the sensor and bounced back to be received by the speed of sound. A threshold distance value is defined to identify how close the vehicle is from the sensor, and accordingly, the status of the LED will be updated. The obtained data from these sensors can be monitored in real-time using the Blynk App.

The TensorFlow algorithm is used for the real-time Pi camera to display the captured video on OpenCV. This algorithm is included as a part of the developed Algorithm 2 to retrieve the visual information from the Pi camera to the RPi to be upload to the IoT cloud. The Pi camera is set up to one frame per second to avoid overheating to the Raspberry Pi's GPU, which will cause a shutdown to the system. The same video is streamed to the Blynk App. as well. We need to ensure identical information about the parking slots occupancy from both the ultrasonic sensor and Pi camera. Another Python-based algorithm (Algorithm 3) is used to enable the GPS module to send data to the RPi. A Pynmea2 library is installed to the Raspberry Pi for parsing the NMEA protocol for transmitting and receiving the signal from the satellite to access the location data. From the Pynmea2, the specific location such as longitude and latitude data is obtained. Then, the data obtained will be updated through the Raspberry Pi to the IoT server.

Once the implemented system on the breadboard is functioning as proposed, we integrate the IoT-PiPMS to the fabricated model, as shown in Fig. 11. The GPS module and Pi camera are installed into the microcontroller box for protection and tidy purpose where the GPS module is mounted onto the cover. The Pi camera also is attached with a fisheye lens for a wide

**Algorithm 2**
. Visual monitoring algorithm for Pi camera with Raspberry Pi.

Require: Vehicle detection and video streaming of parking lot

Ensure: Real-time vision monitoring and surrounding conditions
1: define Pi camera module for in RPi
2: import the required packages (TensorFlow, OpenCV, Blynk) into RPi
3: Initialize Pi camera
4: get the vision of the covered area
5: set camera resolution and framerate
6: Initialize TensorFlow model
7: For each round (do)
8: identify the object in the parking slot
9: classify the object type
10: decide about the slots occupancy (occupied/unoccupied)
11: visualize the results
12: Live Stream of the captured vision to the RPi
13: Update camera vision on OpenCV, Blynk, and VNC
14: end for

**Algorithm 3**
Location data acquisition algorithm for NEO-6 M GPS Module and Raspberry Pi.

Require: Providing drivers with the location of the parking lot via Blynk

Ensure: Real-time update of GPS coordinates for the parking area
1: define Raspberry Pi GPIO for the GPS module
2: include the required libraries ⊃ (Blynk, Pynmea)
3: for each round (do)
4: read the positioning data from NMEA ⊃ using (NEO-6 M GPS Module)
5: *GPSdata = pynmea2.NMEAStreamReader()*
6: get Latitude and Longitude information
7: write the location data to the Blynk server
8: Update the location of the parking lot into the Blynk App. widget
9: end for

range view of the camera. Wires to connect the deployed ultrasonic sensors to the microcontroller box are installed as well. All the wiring connections have been checked by doing testing with a digital multimeter for each connection. The wire has been specific and renames to easily verify the type of wire connection. The ultrasonic sensors would be assembled and connected underneath the prototype. model. The wires for each sensor are connected and ran from the sensors into a hole opening on the top left. The holes on the top left lead into the inside of the pole and then in turn lead the wires onto the Raspberry Pi board on top. Each ultrasonic sensor shares the same Pin of 5 V $V_{CC}$ and GND, so all of the $V_{CC}$ pins and GND pins are soldered together so that they would form a parallel connection. Each of the trigger pins for all ultrasonic sensors is soldered onto the donut board and labeled accordingly so that later on it would be easier to connect the wires onto the Raspberry Pi. A final testing phase of the developed system starts after the implementation of the developed system into the finalized prototype. If there are errors while testing the prototype, further enhancements and optimization are carried out. This phase is important as it helps to improve system performance and allows error detection.

### *4.3. System Workflow*

The general workflow of the developed smart parking system is described in the flowchart of Fig. 12. The Raspberry Pi should be connected to the Wi-Fi network when the power of the microcontroller is turned on. The Raspberry Pi's Wi-Fi connection can be checked through the system monitor or cloud system network. The user must access the Internet through a smart portable device or laptop to discover and access the vacant parking slot in the smart parking system. The system starts working by checking and updating the detection of the ultrasonic sensor, Pi camera, and GPS module. The detection reading by the ultrasonic sensor sometimes fluctuates due to surrounding presence. The Pi camera will then monitor the parking slot to detect any physical presence that appeared in the parking slot. The GPS module updates the location of the parking area to the cloud system. Users can run the Blynk App. on their smartphones and get updates about the outdoor parking lot status. The notification can be seen whether the parking slot is occupied or still vacant. When the parking slot status meets the condition from the ultrasonic sensor and Pi camera, the result condition is uploaded to the cloud server to be displayed into the server application. The user can also view the parking area's real-time condition through server applications

**Fig. 11.** Raspberry Pi with sensors embedded to the prototype.

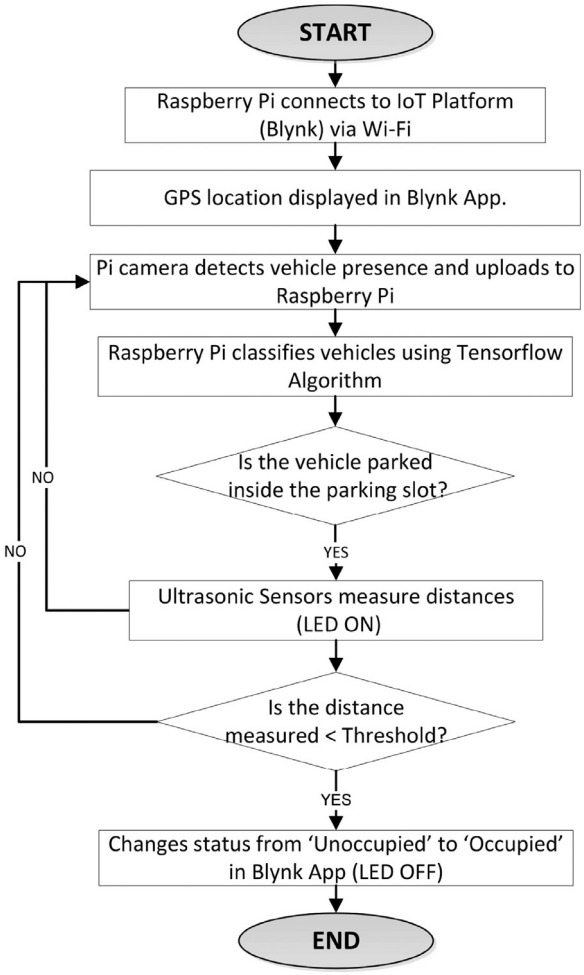


**Fig. 12.** Flowchart of the system.

**Fig. 13.** Finalized Prototype.

## 5. Experimental Results

In this section, we present the system validation and discuss the results. As a result, we have successfully implemented the IoT-based smart parking system into the prototype model. The finalized prototype can be seen in Fig. 13. Our smart parking system consists of four ultrasonic sensors, four green LEDs, one GPS module, one Pi camera, and one Raspberry Pi 4 board. The ultrasonic sensors would sense the presence of the vehicle in front of it, with a green LED mounted on top of it to indicate the occupancy of the parking slot. The GPS module is used to locate the parking facility, and the Pi camera is used to detect and classify objects detected using TensorFlow Algorithm. Each ultrasonic sensor has its dedicated number according to the slot number. The numbering helps tracking down the corresponding number of sensors and also its location. The mini car model is used to simulate a real car in the parking facility. The Raspberry Pi board is placed inside the designated box on top of the pole. A Pi camera is placed facing towards the prototype to cover the view of the whole model. The GPS module is mounted on top of the box cover to broadcast the location data so that users can track the location of the parking facility. The wires connecting the sensors with the Raspberry Pi board runs underneath the prototype model and are covered up using plastic pipes. The wires then run through the hollow content of the pole then reach the Raspberry Pi board inside. When the system is turned on, the operation of the IoT-PiPMS will start eventually. The Blynk automatically connected through Wi-Fi and received data from Raspberry Pi to Blynk cloud and updated it in the Blynk App. Graphical User Interface (GUI) as shown in Fig. 14. The GPS module will automatically detect the location of the parking lot and shown on the map as shown in the figure. Next, the Pi camera stream the video of the parking area and displays it in the live streaming widget of the Blynk App dashboard.

Users could check the occupancy of the parking slot on the Blynk App. on their smart devices. Other than that, they also could locate the location of the parking facility by looking at the map widget. The map widget displays the location data that it received from the GPS module. Users could also watch the camera footage live stream on the Blynk dashboard to confirm the capacity of the parking slot. When the ultrasonic sensor has detected a vehicle close to it, it would then update the user interface to show ‘Occupied’ on the Blynk App. The green LED will also be changed to a red LED on the Blynk dashboard. This will let the user know that the parking slot is currently occupied. On the real LED in the site, the green LED will turn OFF instead. When the vehicle starts leaving the parking slot, the ultrasonic sensor would no longer detect an object in front of it; thus, it would then update the user interface by changing the red LED back to green LED, and also the occupation status from ‘Occupied’ to ‘Unoccupied’. Each of the parking slots is tested by placing a mini car model in front of the ultrasonic sensor. We confirm that our system is functioning well and the data transmitted in real-time to update the parking slots status to the cloud and can be accessed from anywhere at any time over the Internet via the Blynk App. Our system functionality and applicability in the real scenario of outdoor parking are proved as well.

In the meantime, the TensorFlow algorithm runs to detect the parking slot availability and identify the existing vehicle to be updated to the VNC Viewer and the Blynk dashboard, as shown in Fig. 15. It identifies the occupied/unoccupied parking slots and displays the status as text with the number of the parking spot. This vision monitoring is used as a backup to support the ultrasonic sensor data that is displayed as LED indicators in the local parking and the Blynk dashboard as well. The programmed TensorFlow algorithm detects any object inside the rectangular shape, which indicates the parking slot. The TensorFlow was customized to specify the type of object to be detected, which in this case, the vehicle type (car, motorcycle, etc.). If a vehicle occupies a parking slot and is located inside the detection area, it will be detected and update the slot status to occupied. Any object other than vehicle type will not be detected for occupying the park. As in the figure, parking Slot 2 is occupied with a vehicle, while parking Slot 1 is unoccupied. The status of the ultrasonic sensor of Slot 2 is a red LED, and Slot 1 is a green LED, thus it is in agreement with the obtained information from the Pi camera. This double confirmed information increases our system reliability and it is among its advantages. In addition, this interface of the VNC Viewer can be accessed by the management office of the parking area. For the user, it is enough to get information from the ultrasonic sensors and access live streams of the parking area through the Blynk App. Besides, a GPS attached to the Raspberry Pi connected to the Blynk App. The ability of the TensorFlow algorithm with the Pi camera to classify the vehicle

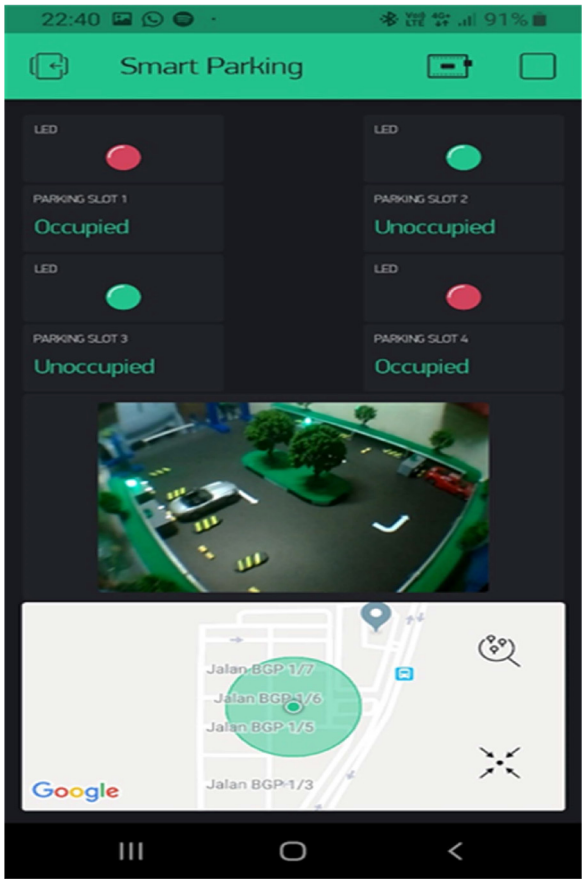


**Fig. 14.** IoT-PiPMS user interface on Blynk.

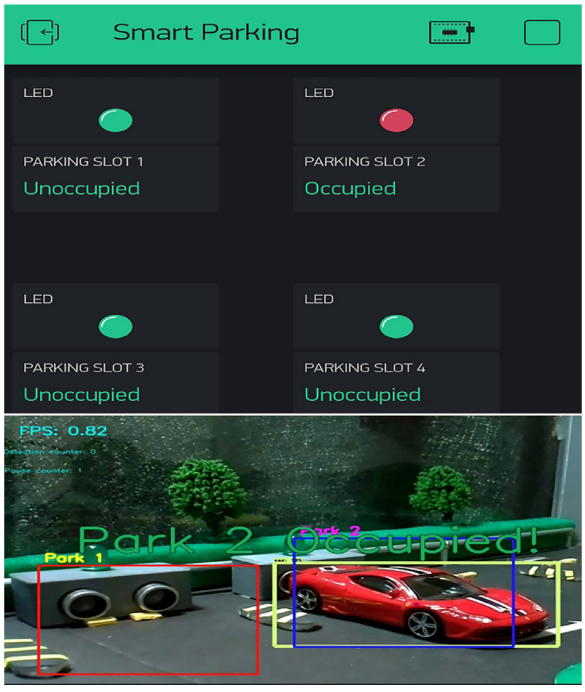


**Fig. 15.** Pi camera object detection vs. LED status.

type is proved via a practical test. When the detection from the Pi camera is updated, the Blynk will also update the data in the Blynk IoT server and mobile.

## 6. Conclusion

An IoT-based smart parking system using Raspberry Pi 4 *B+* has been designed and fabricated by utilizing ultrasonic sensors, Pi camera, GPS module, LEDs, and the Blynk IoT platform. The developed system has been tested and validated to be used in the smart campus environment or similar outdoor parking lot. The system's reliability has been improved by using multiple sensors to detect the existence of vehicles. Staff/students/visitors can easily monitor the parking lot around the campus via a GUI over the Blynk App. by accessing the system dashboard on their smartphones. The GPS sensor allows drivers to easily access the parking lot. The developed IoT-PiPMS provides accessibility, intelligence, comfortable, and improves the driver user experience. For practical implementation, our system can be extended to include multiple RPi and Pi cameras. The parking lot can be divided into several sections; each can be covered by one RPi and one Pi camera. The number of ultrasonic sensors will be increased to cover the entire parking area. The camera captions and video streaming function can be specified for management only rather than users to reduce data usage and improve system privacy and safety.

## Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## References


[1] C.G. Hoehne, M.V. Chester, A.M. Fraser, D.A. King, Valley of the sun-drenched parking space: the growth, extent, and implications of parking infrastructure in Phoenix, Cities 89 (2019) 186–198.
[2] J. Arellano-Verdejo, F. Alonso-Pecina, E. Alba, A. Guzmán Arenas, Optimal allocation of public parking spots in a smart city: problem characterisation and first algorithms, J. Exp. Theor. Artif. Intell. 31 (4) (2019) 575–597.
[3] F. Al-Turjman, A. Malekloo, Smart parking in IoT-enabled cities: a survey, Sustain. Cities Soc. (2019) 101608.
[4] R. Ke, Y. Zhuang, Z. Pu, and Y. Wang, "A smart, efficient, and reliable parking surveillance system with edge artificial intelligence on IoT devices," arXiv preprint arXiv:2001.00269, 2020.
[5] J. Liu, J. Wu, L. Sun, Control method of urban intelligent parking guidance system based on Internet of Things, Comput. Commun. 153 (2020) 279–285.
[6] F. Bock, S. Di Martino, A. Origlia, Smart parking: using a crowd of taxis to sense on-street parking space availability, IEEE Trans. Intell. Transp. Syst. (2019).
[7] J. Lin, S.-.Y. Chen, C.-.Y. Chang, G. Chen, SPA: smart parking algorithm based on driver behavior and parking traffic predictions, IEEE Access 7 (2019) 34275–34288.
[8] J.J. Barriga, et al., Smart parking: a literature review from the technological perspective, Appl. Sci. 9 (21) (2019) 4569.
[9] A. Mackey, P. Spachos, K.N. Plataniotis, Smart parking system based on bluetooth low energy beacons with particle filtering, IEEE Syst. J. (2020).
[10] P. Misra, A. Vasan, B. Krishnan, V. Raghavan, A. Sivasubramaniam, The future of smart parking systems with parking 4.0, GetMobile: Mobile Comput. Commun. 23 (1) (2019) 10–15.
[11] R. Iqbal, T. Maniak, C. Karyotis, Intelligent remote monitoring of parking spaces using licensed and unlicensed wireless technologies, IEEE Netw. 33 (4) (2019) 23–29.
[12] H. Sundmaeker, P. Guillemin, P. Friess, S.J.C.o.E.R.P.o.t.I.o.T. Woelfflé, European Commision, "Vision and challenges for realising the Internet of Things," 3 (3) (2010) 34–36.
[13] A. Abuarqoub, et al., A survey on internet of things enabled smart campus applications, in: Proceedings of the International Conference on Future Networks and Distributed Systems, ACM, 2017, p. 50.
[14] E.C. Thangam, M. Mohan, J. Ganesh, and C.J.I.J.o.A.E.R. Sukesh, "Internet of Things (IoT) based Smart Parking Reservation System using Raspberry-pi," vol. 13, no. 8, pp. 5759–5765, 2018.
[15] M.W. Sari, P.W. Ciptadi, R.H. Hardyanto, Study of smart campus development using internet of things technology, IOP Conference Series: Materials Science and Engineering, 190, IOP Publishing, 2017.
[16] A. Kianpisheh, N. Mustaffa, P. Limtrairut, P. Keikhosrokiani, Smart parking system (SPS) architecture using ultrasonic detector, Int. J. Softw. Eng. Appl. 6 (3) (2012) 55–58.
[17] A. Khanna, R. Anand, IoT based smart parking system, in: 2016 International Conference on Internet of Things and Applications (IOTA), IEEE, 2016, pp. 266–270.
[18] P. Ramaswamy, IoT smart parking system for reducing green house gas emission, in: 2016 International Conference on Recent Trends in Information Technology (ICRTIT), IEEE, 2016, pp. 1–6.
[19] R. Grodi, D.B. Rawat, and F. Rios-Gutierrez, "Smart parking: parking occupancy monitoring and visualization system for smart cities," in *SoutheastCon 2016*, 30 March-3 April 2016 2016, pp. 1–5, doi: 10.1109/SECON.2016.7506721.
[20] B. Mahendra, S. Sonoli, N. Bhat, T. Raghu, IoT based sensor enabled smart car parking for advanced driver assistance system, in: 2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), IEEE, 2017, pp. 2188–2193.
[21] F.I. Shaikh, P.N. Jadhav, S.P. Bandarkar, O.P. Kulkarni, and N.B.J.I.J.o.C.A. Shardoor, "Smart parking system based on embedded system and sensor network," vol. 140, no. 12, 2016.
[22] L. Mainetti, L. Palano, L. Patrono, M.L. Stefanizzi, R. Vergallo, Integration of RFID and WSN technologies in a Smart Parking System, in: SoftCOM, 2014, pp. 104–110.
[23] A. Gupta, P. Rastogi, S. Jain, Smart parking system using cloud based computation and Raspberry Pi, in: 2018 2nd International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC) I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC), 2018 2nd International Conference on, IEEE, 2018, pp. 94–99.
[24] P. Sadhukhan, An IoT-based E-parking system for smart cities, in: 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), IEEE, 2017, pp. 1062–1066.
[25] M.B.J.I.J.o.S. SR and R. Publications, "Automatic smart parking system using Internet of Things (IOT)," p. 628, 2015.
[26] F. Bock, S. Di Martino, A. Origlia, Smart parking: using a crowd of taxis to sense on-street parking space availability, IEEE Trans. Intell. Transp. Syst. 21 (2) (2019) 496–508.

[27] D. Kanteti, D. Srikar, T. Ramesh, Intelligent smart parking algorithm, in: 2017 International Conference On Smart Technologies For Smart Nation (SmartTechCon), IEEE, 2017, pp. 1018–1022.
[28] F.I. Shaikh, P.N. Jadhav, S.P. Bandarkar, O.P. Kulkarni, N.B. Shardoor, Smart parking system based on embedded system and sensor network, Int J Comput Appl 140 (12) (2016).
[29] L. Mainetti, L. Palano, L. Patrono, M.L. Stefanizzi, R. Vergallo, Integration of RFID and WSN technologies in a Smart Parking System, in: 2014 22nd International Conference on Software, Telecommunications and Computer Networks (SoftCOM), IEEE, 2014, pp. 104–110.
[30] O. Orrie, B. Silva, G.P. Hancke, A wireless smart parking system, in: IECON 2015-41st Annual Conference of the IEEE Industrial Electronics Society, IEEE, 2015, pp. 004110–004114.
[31] L. Mainetti, L. Patrono, M.L. Stefanizzi, R. Vergallo, A smart parking system based on IoT protocols and emerging enabling technologies, in: 2015 IEEE 2nd World Forum on Internet of Things (WF-IoT), IEEE, 2015, pp. 764–769.
[32] N.R.N. Zadeh, J.C.D. Cruz, Smart urban parking detection system, in: 2016 6th IEEE International Conference on Control System, Computing and Engineering (ICCSCE), IEEE, 2016, pp. 370–373.
[33] G. Amato, F. Carrara, F. Falchi, C. Gennaro, C. Vairo, Car parking occupancy detection using smart camera networks and deep learning, in: 2016 IEEE Symposium on Computers and Communication (ISCC), IEEE, 2016, pp. 1212–1217.
[34] S. Shinde, A. Patil, S. Chavan, S. Deshmukh, S. Ingleshwar, IoT based parking system using Google, in: 2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC), IEEE, 2017, pp. 634–636.
[35] H. Bandara, J. Jayalath, A. Rodrigo, A. Bandaranayake, Z. Maraikar, R. Ragel, Smart campus phase one: smart parking sensor network, in: 2016 Manufacturing & Industrial Engineering Symposium (MIES), IEEE, 2016, pp. 1–6.
[36] X. Ling, J. Sheng, O. Baiocchi, X. Liu, M.E. Tolentino, Identifying parking spaces & detecting occupancy using vision-based IoT devices, in: 2017 Global Internet of Things Summit (GIoTS), IEEE, 2017, pp. 1–6.
[37] D. Vakula, Y.K. Kolli, Low cost smart parking system for smart cities, in: 2017 International Conference on Intelligent Sustainable Systems (ICISS), IEEE, 2017, pp. 280–284.
[38] H. Al-Kharusi, I. Al-Bahadly, Intelligent parking management system based on image processing, World J. Eng. Technol. 2014 (2014).
[39] E.R. Buhus, D. Timis, A. Apatean, Automatic parking access using openalpr on raspberry pi3, Acta Technica Napocensis 57 (3) (2016) 10.

Waheb A. Jabbar, was born in Taiz, Yemen in 1978. He received the B.*Sc*. in Electrical Engineering from the University of Basrah, Iraq, in 2001, the M.Eng. in Communication & Computer and the Ph.D. in Electrical, Electronics, and System Engineering from Universiti Kebangsaan Malaysia (UKM), Bangi, Selangor, Malaysia, in 2011 and 2015 respectively. He is currently a Senior Lecturer in the Faculty of Electrical & Electronics Engineering Technology, Universiti Malaysia Pahang (UMP), Pekan, Pahang, Malaysia. His-research interests include Routing Protocols in Ad Hoc Networks, Mobile Communications, Wireless Networking, Advanced Electronics, and Automation. He also has a keen interest in the Internet of Things (IoT) applications and Smart City.

Chong Wen Wei studies Bachelor of Electrical and Electronics Engineering Technology at the College of Engineering Technology, University Malaysia Pahang, Malaysia. His-research interests include IoT and Control Systems

Nur Atiqah Ainaa B. M. Azmi studies Bachelor of Manufacturing and Mechatronic Engineering Technology at the College of Engineering Technology, University Malaysia Pahang, Malaysia. Her-research interests include Robotics, 3D Design, and Mechatronics.

Nur Aiman B. Haironnazli studies Bachelor of Electrical and Electronics Engineering Technology at the College of Engineering Technology, University Malaysia Pahang, Malaysia. His-research interests include embedded systems and image processing.