Advances in Brain-Computer Interfaces and Neural Integration ISSN: 3142-9319

PRIME OPEN ACCESS

Volume 2, Issue 2

Research Article

Date of Submission: 03 July, 2026
Date of Acceptance: 29 July, 2026
Date of Publication: 05 Aug, 2026

Conceptual Design of IoT-Enabled Parking Lot Monitoring System Using PIR Motion and Ultrasonic Sensors for Real-Time Occupancy Detection

Tunji John Erinle[1*], Olalekan Victor Akerele[1] and Akeem Abiodun Rasheed[2]

[1]Department of Mechanical Engineering, School of Industrial Engineering Technology, Federal Polytechnic, Ado-Ekiti, Nigeria

[2]Department of Industrial and Production Engineering, School of Infrastructural, Minerals, and Manufacturing Engineering, Federal University of Technology, Akure, Nigeria

***Corresponding Author:**
Tunji John Erinle, Department of Mechatronics Engineering, School of Industrial Engineering Technology, Federal Polytechnic, Ado-Ekiti, Nigeria.

Citation: Erinle, T. J., Akerele, V. A., Rasheed, A. A. (2026). Conceptual Design of IoT-Enabled Parking Lot Monitoring System Using PIR Motion and Ultrasonic Sensors for Real-Time Occupancy Detection. *Adv Brain-Computer Interfaces Neural Integr,* 2(2), 01-12.

Abstract
The rapid growth of urban populations has intensified the demand for efficient parking management, making smart solutions an essential part of modern city infrastructure. This paper presents the design of an Internet of Things (IoT)-enabled parking lot monitoring device that integrates a Passive Infrared (PIR) motion sensor and an Ultrasonic sensor to achieve real-time occupancy detection. The proposed system leverages the Passive Infrared motion sensor to detect vehicular movement, while the Ultrasonic sensor measures distance to confirm the presence or absence of a vehicle in a given slot. Data from these sensors is processed and transmitted through an Internet of Things framework, allowing real-time monitoring via a cloud platform and mobile application interface. The combined use of Passive Infrared motion and Ultrasonic sensing enhances detection accuracy, reducing false positives commonly associated with single-sensor systems. By providing real-time updates on parking slot availability, the device aims to minimise the time drivers spend searching for parking, thereby reducing traffic congestion, fuel consumption, and associated carbon emissions. The system contributes to the broader vision of sustainable smart cities by optimising space utilization and supporting eco-friendly mobility solutions. The design demonstrates promising results; challenges such as sensor calibration, environmental variability, and network scalability must be addressed for large-scale deployment. The integration of the Internet of Things with low-cost sensors highlights the potential of intelligent infrastructure in addressing urban mobility challenges, offering a practical pathway toward smarter, greener, and more efficient cities.

Keywords: Internet of Things, Occupancy Detection, Smart Parking, Smart Cities, Sustainable Mobility

Introduction
Urbanisation has resulted in an increase in car usage, creating issues for effective parking management. In densely populated locations, a lack of real-time information causes traffic congestion, fuel waste, and driver irritation. According to Al-Turjman and Malekloo (2019), inefficient parking arrangements lead to traffic congestion, air pollution, and greenhouse gas emissions [1]. Innovative methods are required to enhance city parking resource management. Traditional parking management systems are inefficient because they are manual, reactive, resource-intensive, and do not provide real-time adaptation. Drivers frequently rely on physical inspections or antiquated signboards, resulting in waste, environmental deterioration, and wasteful vehicle idling. The demand for intelligent, automated, and linked

systems is critical [2].

The Internet of Things (IoT) is a game-changing technology that may reduce parking inefficiencies by providing real-time monitoring, data-driven decision-making, and user convenience. This integration is consistent with smart cities' objective of sustainability, efficiency, and increased urban living quality, hence increasing the total efficiency and effectiveness of parking infrastructure [3].

Accurate occupancy identification is an essential requirement in IoT-based parking systems, but existing solutions frequently rely on single sensors such as ultrasonic or infrared, which can be constrained by external circumstances. A hybrid strategy that includes both technologies may improve system efficiency and robustness by delivering more reliable detection [4].

The main objective of the present research is to design an IoT-enabled parking lot monitoring system that uses Passive Infrared (PIR) motion and ultrasonic sensors to identify occupancy accurately in real time. The system's goal is to reduce detection mistakes, driver search time, and promote sustainable urban travel. The study assesses scalability, cost-effectiveness, and integration with smart city frameworks.

Overview

Over the last two decades, parking management systems have progressed from manual to more automated technology. Traditional techniques, such as attendants and barrier gates, are inefficient in urban contexts because they need human interaction and have limited scalability. With an expanding car population, traditional tactics are no longer viable, emphasising the need for smart parking solutions [5,6]. IoT-based solutions are rapidly being investigated for real-time data sharing between physical objects and digital platforms [7]. They have demonstrated potential for managing parking systems by combining wireless sensor networks, cloud platforms, and mobile apps. Low-power wireless protocols such as Zigbee, LoRaWAN, and Wi-Fi send sensor data to central servers, which reduces search time and traffic congestion. However, system performance is dependent on vehicle detecting sensors [8].

Diverse sensor methods have been investigated for occupancy detection, each having advantages and disadvantages. Ultrasonic sensors assess distance, although they can be affected by background noise and weather conditions [9]. PIR sensors detect motion and heat signatures but may produce false positives [10]. Camera-based systems are accurate, but they are expensive, have limited privacy, and need complex image processing [11].

Recent research demonstrates that hybrid sensor techniques could enhance accuracy, reliability, and resilience. Hybrid PIR-ultrasonic systems could discriminate between genuine cars and false detections caused by ambient conditions [4]. Integrating IoT with cloud-based data analytics improves decision-making and enables predictive parking demand analysis. These advancements are consistent with smart transportation systems' efficiency, sustainability, and user ease [12].

The current state of knowledge on IoT-based parking solutions lacks comprehensive integration of hardware and communication infrastructure, as well as scalability and cost-effectiveness. Many people concentrate on sensor design or software implementation, ignoring the significance of comprehensive integration of hardware and communication infrastructure. A hybrid PIR and ultrasonic sensor setup, connected with IoT platforms, is required for dependable and practical parking monitoring [2].

Potential Applications and its Benefits

- **Smart Parking Systems:** Guide drivers to available parking spots, reducing congestion and improving parking efficiency.
- **Parking Lot Management:** Monitor parking lot occupancy in real-time, enabling more efficient management and optimization of parking resources.
- **Security:** Detect unauthorized access or suspicious activity in the car park.

Materials and Methods

Materials

Design of an IoT-enabled parking lot monitoring system employs a hybrid sensing approach that contains sensors, which detect motion and presence of objects (vehicles or pedestrians) in the car park using Passive Infrared (PIR) motion sensor and also measure the distance of objects from the sensor, allowing to determine if a parking spot is occupied or vacant by employing ultrasonic sensor to reliably identify vehicle presence and occupant. As well as a microcontroller to process data and a communication module to send the data to a remote platform. The design also includes power management and control. Table 1 provides a thorough list of electronic and electrical components, as well as additional accessories, necessary for the design of an IoT-enabled parking lot monitoring system.

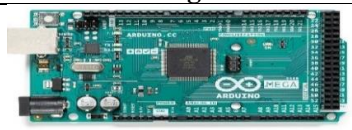
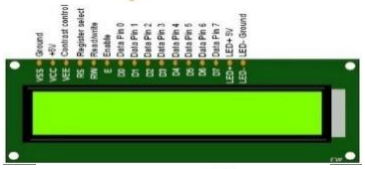
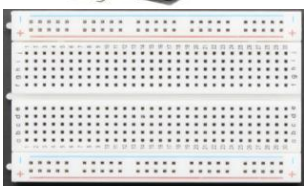
Component	Description	Images
Microcontroller	Arduino Uno It serves as the central processor, handling sensor inputs and decision logic.	
Passive Infrared (PIR) Motion Sensor	A passive infrared sensor (PIR) is an electronic device that detects infrared (IR) light generated by objects in its range of vision (HC-SR501).	
Ultrasonic Sensor	Ultrasonic sensors measure distance by sending and receiving ultrasonic signals, which are sound waves that exceed 20 kHz (HC-SR04).	
Data Communication Modules	Wi-Fi Module (ESP8266) It transmits sensor data to a mobile app in real-time.	
Display Unit	LCD Display (16 x 2) It provides visual feedback for local farmers to view sensor readings.	
Power Supply Unit (Solar Panel and Battery)	12V Power Supply Module It powers the system to support off-grid operation.	
Storage Device	Micro SD Card Module with Card	
Real-Time Clock (RTC)	RTC Module A Real-Time Clock (RTC) module is an electrical circuit that keeps track of the current time and date even when the system's main power supply is switched off.	
Connectors and Cables	Jumper Cables for Connection	
Buzzer	For local alerts	
Breadboard	Breadboard The PVC breadboard is a versatile instrument used in electronic prototype circuits, allowing for quick changes to component arrangement and connections.	

Table 1: Materials for the Design of the System with Images

System Design and Conceptual Framework

The conceptual framework for parking systems emphasises dependability, cost-effectiveness, and scalability. It employs reliability for backup, inexpensive sensors, open-source microcontrollers for cost-effectiveness, and a modular architecture for scalability [13]. This concept overcomes the problems of existing parking systems while also serving as a future-ready component of smart city infrastructure. Its hybrid sensing, IoT connection, and cloud-based data management make it applicable to a variety of environments.

The proposed IoT-enabled parking lot monitoring system employs a hybrid sensing strategy that combines a Passive Infrared (PIR) motion sensor and an Ultrasonic sensor to reliably identify vehicle presence. The system employs sensors at each parking space to collect real-time occupancy data, reducing false positives and increasing dependability, guaranteeing that users receive correct occupancy information according to Hassan et al. (2020) and Elfaki et al. (2023) [4,14].

The microcontroller collects and processes information from PIR and ultrasonic sensors to determine the condition of each parking space. The data is then sent through an IoT connection module to a centralised cloud server, ensuring scalability and real-time data retrieval. This design connects several parking lots to a single monitoring platform.

The software layer of a system converts sensor data into usable information for end users. A microcontroller's integrated firmware guarantees that sensors and IoT modules communicate properly. Cloud servers run algorithms that filter noise and validate occupancy data. Administrators may access real-time updates, slot reservations, and occupancy metrics via a user interface that is either mobile or online-based.

Sensor design is crucial for accuracy, with ultrasonic and PIR sensors positioned at appropriate angles and distances to accommodate vehicle heights and sizes. Power issues are also extremely important, particularly in big parking lots with hundreds of sensors [15]. Solar-powered modules and energy-efficient microcontrollers can boost system sustainability while lowering operational expenses. Figures 1 and 2 depict a flow chart and block diagram of the designed IoT-enabled parking lot monitoring system.

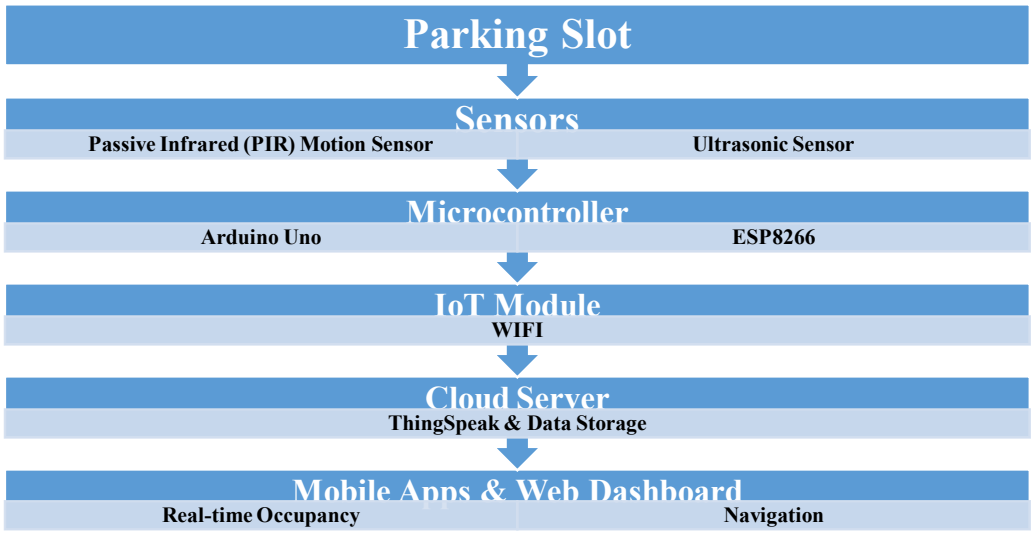


Figure 1: Flow Chart of the Designed IoT-Enabled Parking Lot Monitoring System

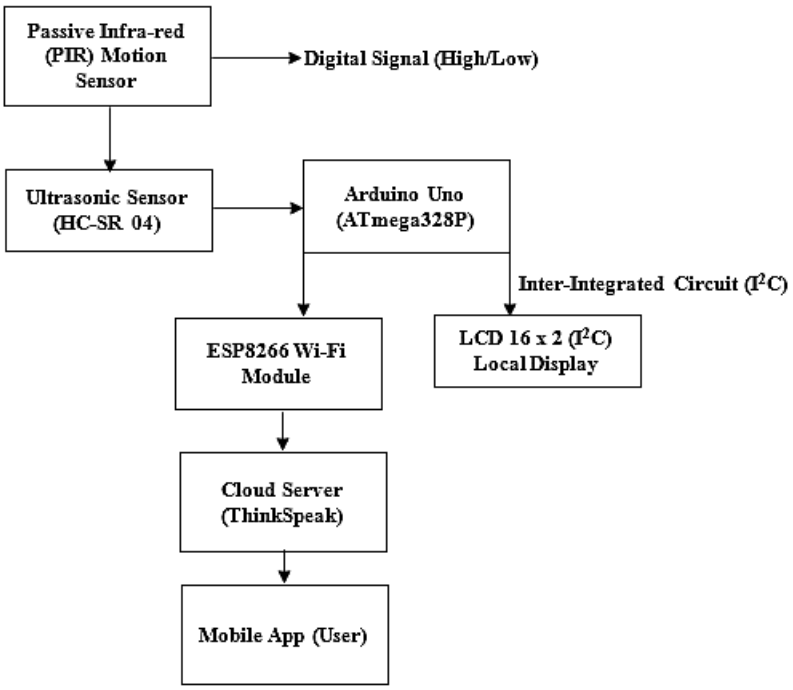


Figure 2: Block Diagram of the Designed IoT-Enabled Parking Lot Monitoring System

Methods

The integration of the two sensors with an Arduino Uno and an Inter-Integrated Circuit (I2C) liquid crystal display (LCD) represents a practical demonstration of sensor-based data acquisition and real-time feedback in embedded systems. The sensors would be placed in various areas of the parking slot and linked to an Arduino Uno. The sensors are devices that detect the object's movement in the parking slot by placing them in the surroundings. The microprocessor receives analogue and digital input from the Passive Infrared (PIR) sensor used for detecting human presence or motion, and an ultrasonic sensor for detecting distances by sending and receiving ultrasonic waves. The ESP8266 Wi-Fi module connects the Arduino to a remote mobile app, allowing for real-time data visualisation. A smartphone application was created to notify users of occupant presence. The circuit diagrams describe how the circuit elements connect, as shown in Figures 3–5.

The HC-SR04 ultrasonic sensor was interfaced with the Arduino by connecting its TRIG pin to digital pin D 10 and the ECHO pin to digital pin D 13. It is responsible for detecting distances by sending and receiving ultrasonic waves for the distance measurement.

The human sensor's OUT pin was connected to digital pin D2 of the Arduino Uno. This allows the microcontroller to receive real-time binary signals such as HIGH (1) when motion is detected ("Human Detected") and LOW (0) when no movement is present ("No Human Detected").

The liquid crystal display (LCD) for data visualisation was interfaced through the Inter-Integrated Circuit (I2C) protocol with serial data (SDA) connected to A4 and serial clock (SCL) connected to A5 of the Arduino Uno. Both modules shared a common ground to maintain circuit stability. This configuration ensured a smooth flow of signals between the sensing module, the microcontroller, and the display interface. Both the sensors and LCD are powered using a regulated +5V line from the Arduino Uno. The Voltage Common Collector (VCC) and ground (GND) of the two sensors were connected to +5V and ground, respectively. Figure 6 shows the schematic circuit diagram of the designed IoT-enabled parking lot monitoring system.

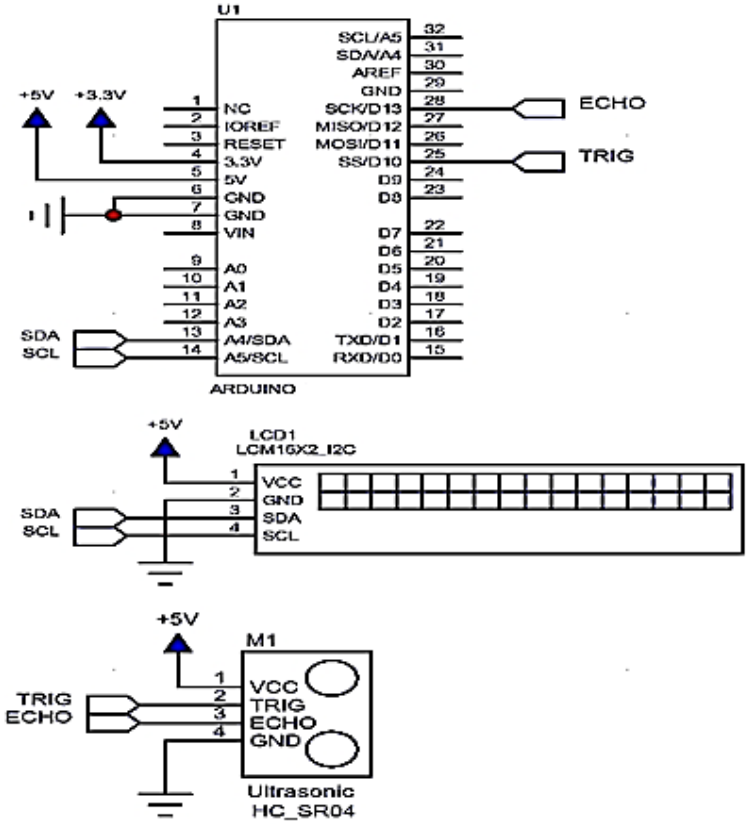


Figure 3: Circuit Diagram for Ultrasonic Sensor Connection

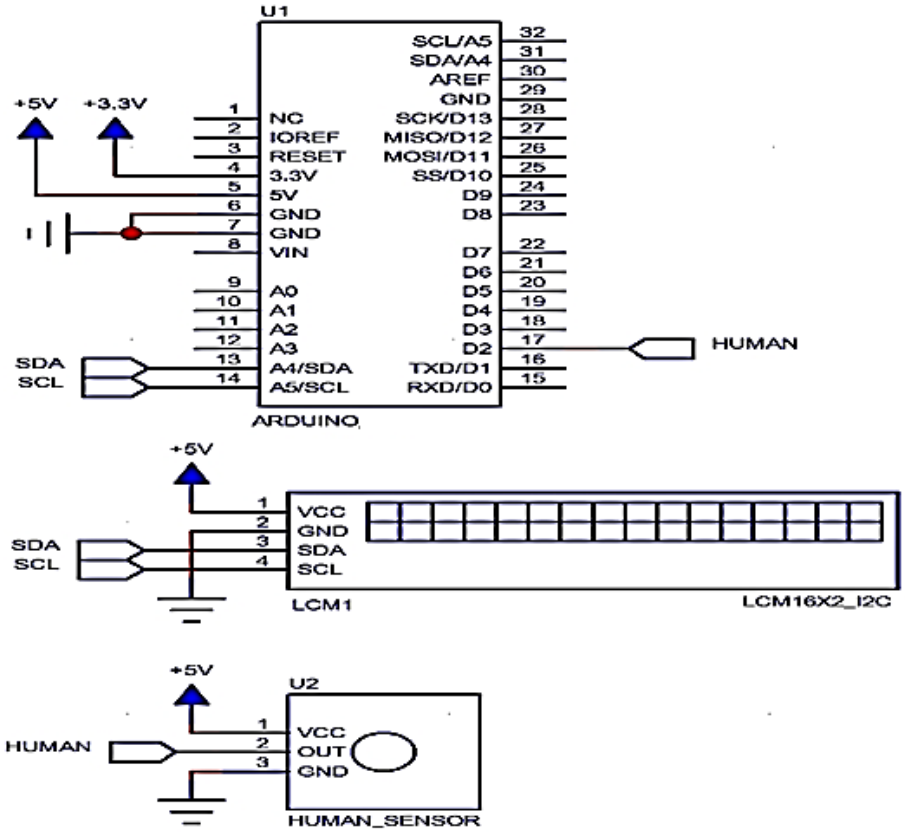


Figure 4: Circuit Diagram for Passive Infra-red (PIR) Motion Sensor Connection

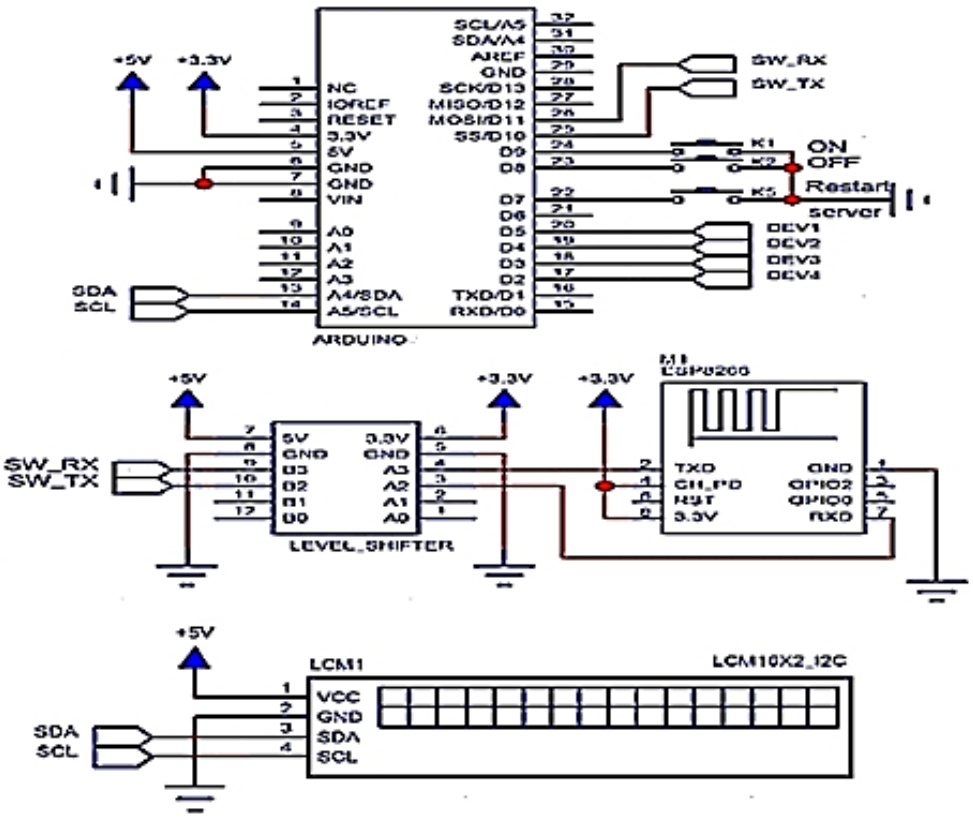


Figure 5: Circuit Diagram for ESP8266 Wi-Fi Module Connection

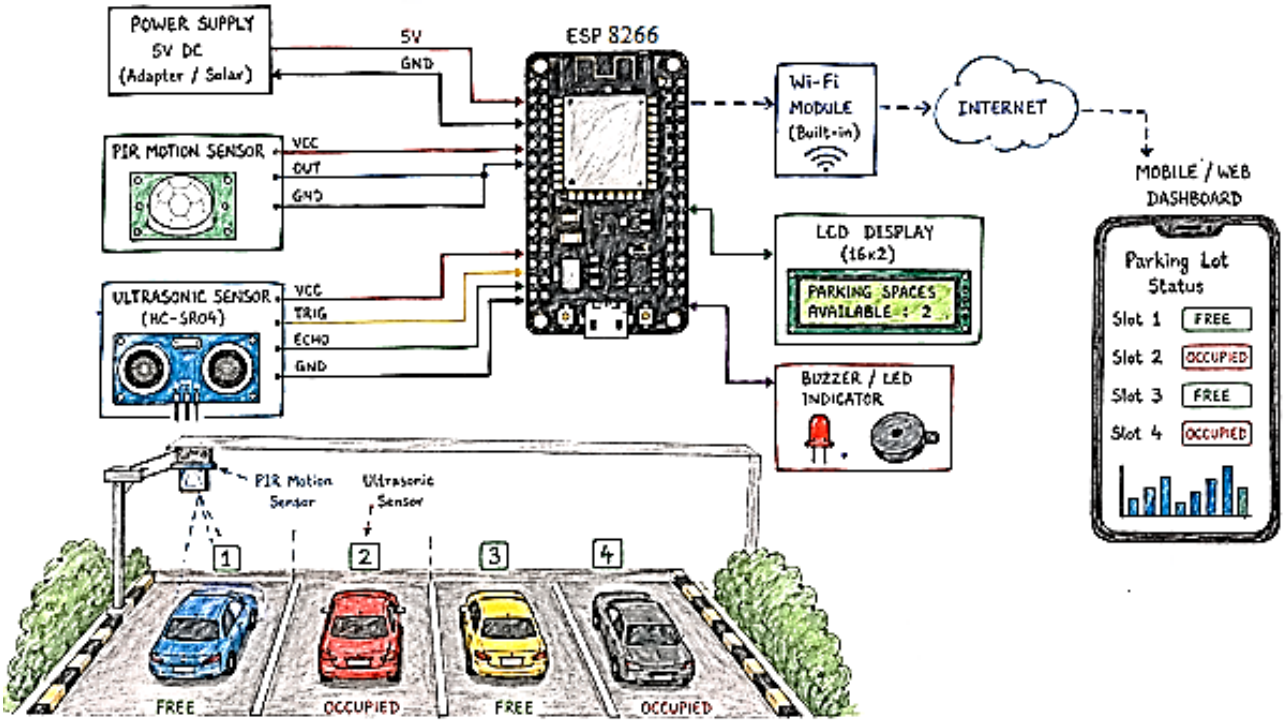


Figure 6: Schematic Circuit Diagram of the Designed IoT-Enabled Parking Lot Monitoring System

Results and Discussion

The IoT-enabled parking lot monitoring system was designed. A hybrid sensor technique combining PIR motion and ultrasonic sensors was adopted to increase detection reliability under a variety of situations. The design step entailed putting together hardware components to produce a functioning system. The ultrasonic sensor calculated the distance between barriers, while the PIR sensor sensed motion and heat signatures. The microcontroller with ESP8266 was chosen because it can function as both a processor unit and a Wi-Fi communication module. A power management approach was adopted, which involved employing low-power modes when inactive. Rapid prototyping involved the use of breadboards and jumper wires.

Programming and Signal Processing

Programming and signal Processing of the system include setting hardware, writing code to communicate with sensors, and creating software for data collecting, processing, and perhaps control. Figure 7 shows the Arduino Integrated Development Environment (IDE), which is used to program the microcontroller in C++.

Figure 7: Programming of the Code Interface for the Monitoring of the Parameters

The Arduino was programmed using the Arduino IDE with a structured algorithm for triggering the ultrasonic sensor by sending a 10-microsecond (μs) HIGH pulse through the TRIG pin. Receiving the reflected signal on the ECHO pin and measuring the time taken for the signal to return. Using the time difference to calculate distance, as given in Equation (1). The factor 2 accounts for the signal's round-trip. Displaying the calculated distance to the object in centimeters on the LCD screen.

$$Distance = \frac{Time\ (T) \times Speed\ of\ Sound (C)}{2} \quad (1)$$

The implemented prototype successfully measured distances using the HC-SR04 ultrasonic sensor interfaced with an Arduino Uno, and displayed the results on a 16 x 2 LCD screen in real time. As shown in Figure 8, in the experimental setup, the LCDs are a distance of 10.20 cm, corresponding to the space between the sensor and the placed object. The system was tested under different environmental conditions to ensure reliable measurements. The ultrasonic sensor was placed in front of objects at varying distances (10 cm – 200 cm), and the values displayed on the LCD were compared with manual measurements using a ruler. Any discrepancies were noted, and calibration adjustments were made in the Arduino code to improve accuracy. The system consistently measured distances within the sensor's reliable range (2 cm to 400 cm). At short distances (less than 50 cm), the readings were highly accurate, with deviations of less than ± 0.5 cm compared to manual ruler measurements. This demonstrates the effectiveness of the ultrasonic sensor's time-of-flight principle in calculating object distances. At longer ranges, slight fluctuations were observed, primarily due to environmental factors such as surface reflectivity, angle of incidence, and ambient noise. The system's repeatability during multiple trials indicated a high level of reliability for close-to mid-range measurements.

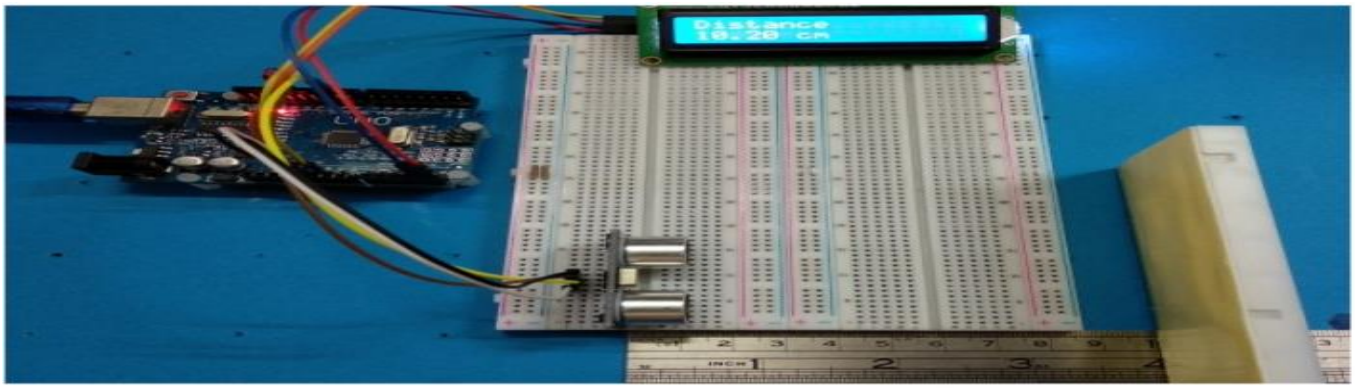


Figure 8: Ultrasonic Sensor with Connection to Microcontroller

The Arduino was programmed with an algorithm to continuously monitor the sensor's output by reading the digital signal from the PIR sensor. If the signal is HIGH, interpret it as "Human Detected". If the signal is LOW, interpret it as "No Human Detected", as shown in Figure 9. The corresponding message on the Inter-integrated circuit (I2C) liquid crystal display (LCD) was displayed. This straightforward logic ensures immediate feedback for any detected movement. The system was tested by placing the sensor in different environments, such as indoors with minimal movement and outdoors with more frequent human activity. The detection range and response time of the PIR sensor were observed. Adjustments were made in the code (such as adding a small delay or debouncing logic) to filter out false triggers caused by sudden temperature changes or electrical noise. The movement creates a small alternating current (AC) signal, around 1 mVpp, that is superimposed on a varying direct current (DC) signal.

Detection Range /Height: This is to calculate the height at which a sensor should be placed to cover a certain range, as given in Equation (2). The angle is half of the sensor's total detection angle

$$tan\ (Total\ Detention\ Angle\ (\theta)/2) = \frac{detection\ raduis\ on\ \ ground}{mounting\ height} \quad (2)$$

Detention Area: This is to find the total circular area covered by the sensor, as given in Equation (3).

$$\text{Total Circular Area } (A) = \pi r^2 \quad (3)$$

Signal Filtering (RC Circuits): The resistor-capacitor (RC) filter is used to remove the unwanted noise and condition the weak signal from the PIR sensor. Equation (4) helps to determine the values for the resistor and capacitor needed to create a bandpass filter, which allows the desired alternating current (AC) signal frequencies to pass through while blocking others, as R = 10 kΩ, C = 1 µF, with the fc ≈ 16 Hz.

$$Cutoff\ Frequency\ (f_C) = \frac{1}{(2\pi RC)} \quad (4)$$

Power Consumption: This is used to calculate the power draw, as given in Equation (5).

$$Power\ (W) = Voltage\ (V) \times Current\ (I) \quad (5)$$

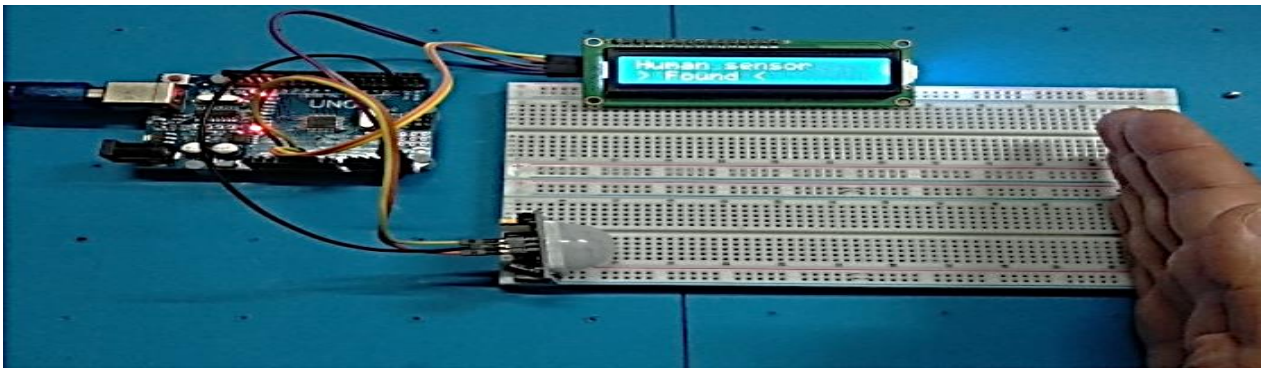


Figure 9: Passive Infrared (PIR) Motion Sensor with Connection to Microcontroller

The ESP8266 Wi-Fi Module acts as a web server. An Android App accesses the web server to control the real-time monitoring of parking of cars and occupant detection for data visualisation, as shown in Figure 10. Figure 11 shows the schematic diagram for the ESP8266 Wi-Fi connection to the Mobile App. The programming code for the microcontroller controls the ESP8266 Wi-Fi to transmit the information from the sensors through the Mobile App.

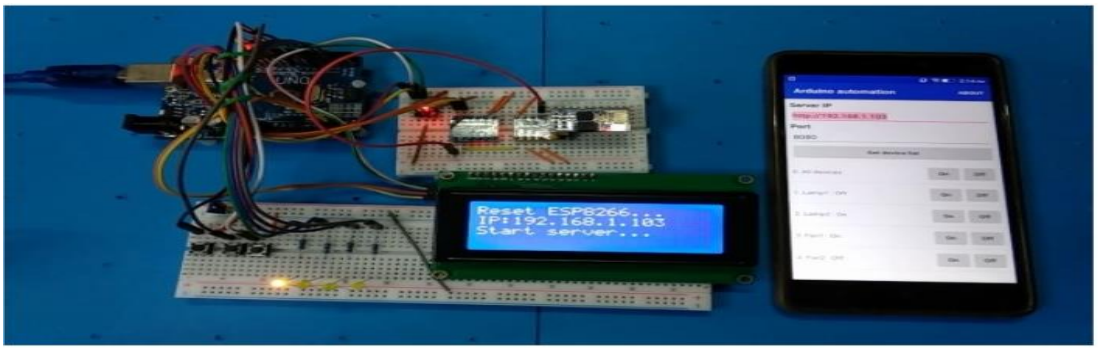


Figure 10: ESP8266 Wi-Fi Module Connection to Microcontroller with Mobile App

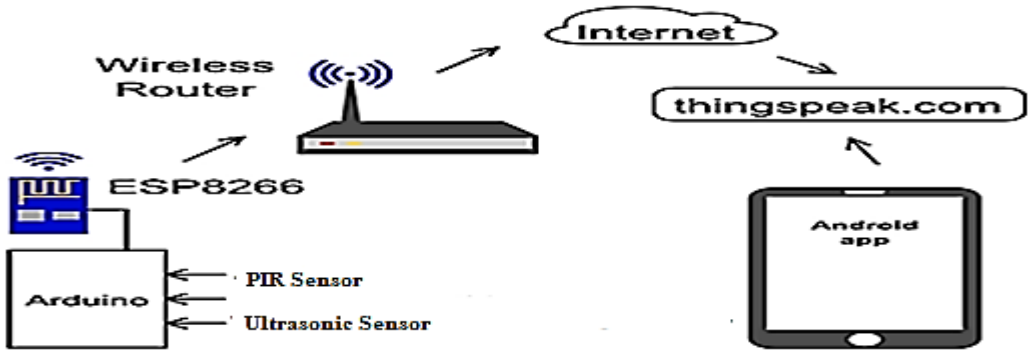


Figure 11: Schematic Diagram for ESP8266 Wi-Fi Connection to Mobile App/Cloud Server

Conclusion

The purpose of this study is to design an Internet of Things-enabled parking lot monitoring system that uses Passive Infrared (PIR) and ultrasonic sensors to identify occupancy in real time, solving urban concerns such as traffic congestion, fuel waste, and environmental degradation. The technology combines low-cost sensors with an IoT framework to provide accurate parking data and improve urban mobility.

The designed hybrid PIR-ultrasonic configuration has been proven viable in both the hardware implementation and testing phases, detecting object movement and measuring occupancy with high precision. Real-time data transfer via the ESP8266 Wi-Fi module enables smooth access to the cloud platform, while its user-friendly mobile interface would improve driving and parking experiences.

The study emphasises the socioeconomic and environmental benefits of a parking system, such as lower fuel usage, carbon emissions, and better air quality. It also supports worldwide smart city initiatives and traffic control. The system's modular and scalable architecture enables adaptation to a wide range of scenarios, from small parking lots to major urban facilities.

In conclusion, this research demonstrates that IoT-enabled smart parking solutions may improve urban settings by optimising space utilisation, lowering environmental impact, and increasing quality of life. The successful execution of this conceptual design lays out the foundation for future advances in smart transportation and sustainable urban planning. Routers may handle network concerns by capturing signals from network provider masks, and policymakers can create guidelines for efficient parking lots in urban areas.

References

1. Al-Turjman, F., & Malekloo, A. (2019). Smart parking in IoT-enabled cities: A survey. Sustainable Cities and Society, 49, 101608.
2. Idris, M. Y. I., Leng, Y. Y., Tamil, E. M., Noor, N. M., & Razak, Z. (2020). Smart parking system using wireless sensor networks. Journal of Network and Computer Applications, 143, 1–12.
3. Pawar, P., Patil, A., & Bhagat, A. (2021). IoT-enabled real-time smart parking system with mobile application. Materials Today: Proceedings, 47(5), 620–626.
4. Hassan, N. U., Yuen, C., & Zhang, W. (2020). Smart car parking system with IoT-based hybrid sensor integration. IEEE Internet of Things Journal, 7(5), 4115–4126.
5. Chou, Y. C., Lin, C. H., & Hsu, C. L. (2018). Development of a smart parking management system based on wireless sensor networks. Sensors, 18(5), 1603.
6. Enríquez, F. J., Mejía-Muñoz, J.-M., Bravo, G., & Cruz-Mejía, O. (2024). Smart parking: Enhancing urban mobility with fog computing and machine learning-based parking occupancy prediction. Applied System Innovations, 7(3), 52.
7. Jackson, W., & Marchant, R. (2025). IoT-Powered Smart Parking: A Real-Time Approach to Automated Monitoring and Payments. Unpublished research paper.
8. Shaikh, F. A., & Rajput, A. Q. (2021). IoT-based smart parking system for urban areas: Design and implementation. Journal of King Saud University – Computer and Information Sciences, 33(8), 964–972.
9. Choudhury, P., Maity, S., & Bera, R. (2019). Smart parking systems using IoT: A survey. International Journal of Advanced Computer Science and Applications, 10(5), 504–512.
10. Gupta, R., Arora, A., & Malhotra, S. (2020). Design and implementation of an IoT-based smart parking system using sensor fusion. International Journal of Electrical and Computer Engineering, 10(6), 6512–6521.
11. Gong, S., Kumar, R., & Kumutha, D. (2021). Design of lighting intelligent control system based on OpenCV image processing technology. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 29(Supp01), 119-139.
12. Ahmed, M. M., Rashid, R. A., Ali, A. A., Mohammed, M. T., Diblawe, A. M., Salim, M. R., ... & Afwah, A. A. (2024, December). Integrating IoT Technologies for Smart Parking Management: A Cost-Effective System with Telegram-Based Real-Time Alerts. In 2024 IEEE 22nd Student Conference on Research and Development (SCOReD) (pp. 670-674). IEEE.
13. Shoup, D. (2021). High cost of free parking. Routledge.
14. Elfaki, O., Messoudi, W., Bushnag, A., Abuzneid, S., & Alhmiedat, T. (2023). A smart real-time parking control and monitoring system. Sensors, 23(24), 9741.
15. Abdelrahman, O., Elfaki, W., Messoudi, W., Bushnag, A., Abuzneid, S., & Alhmiedat, T. (2023). A smart real-time parking control and monitoring system. Sensors, 23(24), 9741.

Arduino Code

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <ESP8266WiFi.h>
// Pin Definitions
#define TRIG_PIN 10
#define ECHO_PIN 13
#define PIR_PIN 2
#define BUZZER_PIN 8
// Constants
const int DISTANCE_THRESHOLD = 30;  // cm
const int PIR_PERSIST_TIME = 3000;  // milliseconds
const int SAMPLE_INTERVAL = 1000;   // milliseconds
// Global Variables
LiquidCrystal_I2C lcd(0x27, 16, 2);
unsigned long lastPirTriggerTime = 0;
bool pirTriggered = false;
// WiFi credentials
const char* ssid = "YOUR_SSID";
const char* password = "YOUR_PASSWORD";
const char* server = "api.thingspeak.com";
String apiKey = "YOUR_API_KEY";
```

```
void setup() {
  Serial.begin(115200);
  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
  pinMode(PIR_PIN, INPUT);
  pinMode(BUZZER_PIN, OUTPUT);
  lcd.init();
  lcd.backlight();
  lcd.setCursor(0, 0);
  lcd.print("Parking Monitor");
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("WiFi connected");
  delay(2000);
}
float measureDistance() {
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);
  long duration = pulseIn(ECHO_PIN, HIGH, 30000); // 30ms timeout
  if (duration == 0) return 999; // No echo
  float distance = duration * 0.034 / 2; // Convert to cm
  return distance;
}
bool checkPIR() {
  int pirValue = digitalRead(PIR_PIN);
  if (pirValue == HIGH) {
    lastPirTriggerTime = millis();
    pirTriggered = true;
    return true;
  }
  // Check persistence
  if (pirTriggered && (millis() - lastPirTriggerTime) < PIR_PERSIST_TIME) {
    return true;
  }
  pirTriggered = false;
  return false;
}
String determineSlotStatus(float distance, bool motion) {
  if (distance < DISTANCE_THRESHOLD && motion) {
    return "OCCUPIED (MOVING)";
  }
  else if (distance < DISTANCE_THRESHOLD && !motion) {
    return "OCCUPIED (STATIONARY)";
  }
  else if (distance > (DISTANCE_THRESHOLD + 10) && !motion) {
    return "VACANT";
  }
  else {
    return "UNCERTAIN";
  }
}
void sendToCloud(String status, float distance) {
  if (WiFi.status() == WL_CONNECTED) {
    WiFiClient client;
    if (client.connect(server, 80)) {
      String postStr = apiKey;
      postStr += "&field1=";
      postStr += String(distance);
```

```
      postStr += "&field2=";
      postStr += status;
      client.print("POST /update HTTP/1.1\n");
      client.print("Host: api.thingspeak.com\n");
      client.print("Connection: close\n");
      client.print("X-THINGSPEAKAPIKEY: " + apiKey + "\n");
      client.print("Content-Type: application/x-www-form-urlencoded\n");
      client.print("Content-Length: ");
      client.print(postStr.length());
      client.print("\n\n");
      client.print(postStr);
    }
    client.stop();
  }
}
void loop() {
  float distance = measureDistance();
  bool motion = checkPIR();
  String status = determineSlotStatus(distance, motion);
  // Display on LCD
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Dist: ");
  lcd.print(distance);
  lcd.print("cm");
  lcd.setCursor(0, 1);
  lcd.print(status.substring(0, 15));
  // Buzzer on occupancy change
  static String lastStatus = "";
  if (status != lastStatus && status != "UNCERTAIN") {
    digitalWrite(BUZZER_PIN, HIGH);
    delay(200);
    digitalWrite(BUZZER_PIN, LOW);
    lastStatus = status;
  }
  // Send to cloud every 5 cycles
  static int counter = 0;
  if (++counter >= 5) {
    sendToCloud(status, distance);
    counter = 0;
  }
  delay(SAMPLE_INTERVAL);
}
```