Designing an Automatic Parking System Using Arduino Uno and a Python-Based GUI Interface

Muhamad Faridzqi Suryadi[1], Muhammad Sahal Anwar Hadi[2],
Muhammad Saef Mubhaul Abbas[3], Shela Rahma Fitri[4], Novianti Ramadhani[5]
Bachelor of Informatics Engineering Program, Nahdlatul Ulama Indonesia University
Faculty of Engineering and Computer Science, Nahdlatul Ulama Indonesia University
E-mail: geryandgrey@gmail.com[1], sahalanwarhadi25@gmail.com[2], nahmubhan31@gmail.com[3], shela04rahma@gmail.com[4]
nvntiramadhani180@gmail.com[5]

Abstract— Conventional parking systems often cause problems such as long queues, delays in recording, and lack of efficiency in vehicle data management. Therefore, this study designed an automatic parking system using Arduino Uno combined with HC-SR04 ultrasonic sensors, servo motors, input buttons (*push buttons*), and a *Python Graphical User Interface* (GUI). When a vehicle is detected by the sensor, a 16x2 *Liquid Crystal Display* (LCD) will display a welcome message and instructions to the user to press the button to print a parking ticket. This system records the vehicle's entry time and generates a ticket with a unique *Identification* (ID) code and arrival time. All vehicle data is stored in a *Comma-Separated Values* (CSV) file to facilitate monitoring and calculation of parking fees when the vehicle exits. This system can calculate the total parking cost based on the duration of parking at the predetermined rate and display parking user data on the GUI. Based on testing, the system can detect vehicles at an effective distance of 5–15 cm using ultrasonic sensors, open and close the barrier using a servo motor in approximately 1–2 seconds, and print tickets via a *thermal* printer in less than 3 seconds. These results indicate that the developed system can operate automatically and efficiently, so that it has the potential to be a solution to reduce problems in manual parking management.

Keywords— Automatic parking system, Arduino Uno, ultrasonic sensor, Python GUI, parking ticket, 16x2 LCD, servo motor.

I. INTRODUCTION

As the number of motor vehicles increases, the need for an efficient and organized parking system becomes increasingly important, especially in urban areas. Conventional parking systems that are still operated manually often cause various problems, such as long queues, errors in recording parking times, and potential ticket abuse. Therefore, the development of an automated parking system is one relevant solution to address these challenges.

One approach in developing this system is to utilize microcontrollers such as Arduino Uno connected to various sensors and actuators, such as ultrasonic sensors to detect the presence of vehicles and servo motors to automatically adjust parking barriers. In addition, the use of a Python-based *Graphical User Interface* (GUI) allows the system to digitally record vehicle data and calculate parking fees based on the duration of the vehicle's stay in the parking area.

Previous research shows that the HC SR04 ultrasonic sensor controlled by Arduino Uno is capable of detecting vehicles with an accuracy of nearly 99%, after undergoing a characterization process using fuzzification and defuzzification methods [1]. Meanwhile, research by Priyulida et al. developed an Arduino Uno-based automatic parking barrier system, which utilizes HC SR04 ultrasonic sensors and servo motors to automatically open and close the barrier when a vehicle is detected [2].

Based on these findings, this study aims to design and build an automatic parking system using Arduino Uno, HC SR04 ultrasonic sensors, push buttons, servo motors, 16x2 *Liquid Crystal Display* (LCD), and a Python GUI connected to a *thermal* printer to automatically print tickets. This system focuses only on the process of vehicles entering and exiting with digital tickets, not on monitoring parking slots, and is expected to be an efficient and practical solution for parking areas in public places such as shops, offices, restaurants, and tourist areas.

II. MATERIALS AND METHODS

This research uses an (*engineering design*) method with an experimental approach to develop an automatic parking system that can detect vehicles, print entry tickets, and automatically control parking barriers. The prototype was developed with an Arduino Uno microcontroller as the hardware control center and a Python-based desktop interface for ticket data management and parking ticket printing.

A. Tools and Materials

This study uses several main components and supporting tools in designing an Arduino Uno-based automatic parking system, including:

1) Arduino

Figure 1. Arduino Uno

The Arduino Uno is an ATmega328P-based microcontroller that serves as the control center for all system components. The Arduino Uno has 14 digital input/output pins and 6 analog input pins, and supports serial communication used to communicate with computers via the *Universal Serial Bus* (USB) port [3]. Arduino Uno is used as the central microcontroller of the parking system, tasked with reading sensors, controlling servos, and communicating with Computer.

2) Ultrasonic Sensor HC-SR04 (2 pcs)

Figure 2. Ultrasonic Sensor HC-SR04

This sensor works by emitting ultrasonic waves and measuring the reflection time to calculate distance, with high accuracy in the range of 2 cm to 4 m. The sensor has four pins: VCC (*Voltage Common Collector*), Trigger (*trigger for sending ultrasonic signals*), Echo (*receiver for signal reflections*), and GND (*Ground*), and is commonly used in automatic parking applications [4]. This sensor is used to detect vehicles entering and exiting the lane.

3) Servo Motor SG90 (2 pcs)

Figure 3. Servo Motor SG90

SG90 servo motor is used as a barrier actuator in automatic parking systems, with a maximum rotation angle of 180°, low power consumption, and a torque of approximately 1,8 kg·cm. These characteristics make the SG90 suitable for small-scale mechanical applications. A study by Khamidah et al. shows the application of the SG90 in an Arduino Uno-based kWh token activator system with good motion stability [5].

4) LCD 16 x 2 I2C

Figure 4. LCD 16 x 2 I2C

The 16×2 I2C LCD is used to display messages to users, such as "*Welcome*" and "*Please Press the Button*." This module only requires two pins, namely the *Serial Data Line* (SDA) and *Serial Clock Line* (SCL), making it more practical and efficient than a regular parallel LCD. The use of I2C LCDs is considered to simplify system design by reducing the complexity of wiring and component installation, as explained by Rizakir and Sukarno in their research on Arduino-based automated systems [6].

5) Thermal Printer ESC/POS

Figure 5. Thermal Printer ESC/POS

The *Epson Standard Code for Point of Sale* (ESC/POS) *thermal* printer is used to automatically print parking tickets via a serial connection to a laptop. With the help of the *python-escpos* library, the printer can be controlled directly from a Python application using print commands such as *text()* and *cut()*. The official documentation states that it is sufficient to specify the port (e.g., COM4) and *baudrate* (9600) to send print commands via serial [7].

6) Supporting Tools (Breadboard, *jumper* cable, adaptor, input button (*push button*))

figure 6. Supporting tools used: breadboard, jumper cable, adaptor 5 V 2 A, and input button (push button)

It is a tool used in the assembly and testing process. Breadboards and *jumper* cables are used to assemble prototype circuits without soldering, adapters are used as a 5 V power source for components, and buttons are used as manual user input when printing tickets.

B. System Series (Wiring Diagram)

This automatic parking system is designed to integrate various electronic components that coordinate with each other through an Arduino Uno microcontroller. Each component has a specific function, such as ultrasonic sensors that detect the presence of vehicles, servo motors that move the parking barriers, and I2C LCDs that provide an interactive display for users.

The input button (*Push Button*) is used as a trigger for ticket printing, while the *thermal* printer will print ticket information such as *Identification* (ID) or unique code, entry

time, and parking rates. All of these components are powered by a 5 V adapter and connected using a breadboard and *jumper* cables to facilitate prototype assembly without soldering.

Cable arrangement and connections between components are carried out carefully to ensure system stability and prevent interference during the automation process.

The following image shows the overall wiring configuration of the system :

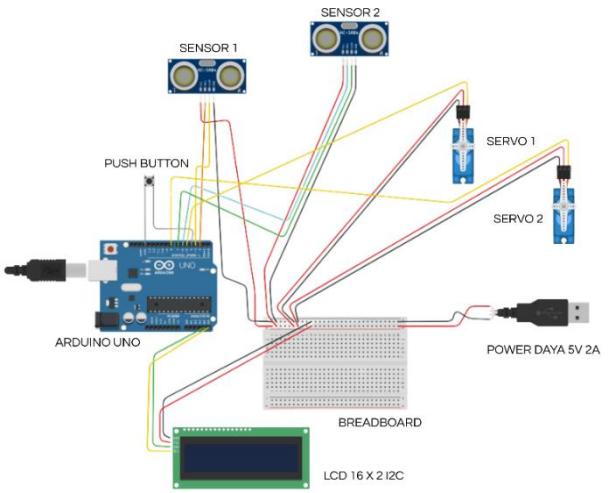

Figure 7. Wiring diagram for an Arduino Uno-based automatic parking system

TABLE I. COMPONENT PIN CONNECTION

No.	Component	Component Pin	Connected to Arduino Pin
1.	Ultrasonic Sensor 1 (In)	Trig	D2
		Echo	D3
2.	Ultrasonic Sensor 2 (Out)	Trig	D7
		Echo	D8
3.	Input Button (*Push Button*)	1 foot	D4
		The other foot	GND
4.	Servo 1 (Entry Gate)	Signal (yellow/orange)	D5
5.	Servo 2 (Exit Gate)	Signal (yellow/orange)	D6
6.	LCD 16x2 I2C	SDA	A4
		SCL	A5
7.	Adaptor 5 V 2 A	VCC	Breadboard +5 V
		GND	Breadboard GND

Description:

All VCC and GND connections from components such as ultrasonic sensors, servo motors, and LCDs are not connected directly to the power pins on the Arduino. Instead, these components are powered from a breadboard connected to an external 5 V 2 A adapter to ensure current and voltage stability, especially when the system load increases.

Arduino only acts as a logic signal controller, namely through digital pins and I2C pins (SDA/SCL) to control the overall operation of the system, without burdening the internal resources of the Arduino board itself.

C. System Flow and Flowchart

To facilitate understanding of the automatic parking system's workings, a *flowchart* is used to illustrate the logic of interaction between sensors, buttons, LCDs, servos, and the Python GUI interface and *thermal* printer. This *flowchart* represents the data flow and control logic from the beginning to the end of the process.

The system is divided into two main parts, namely vehicle entry and exit processes. Each has a structured, automated, and integrated workflow so that the system can work efficiently and respond to real conditions in the field.

1) Vehicle entrance lane

The process begins when the ultrasonic sensor detects the presence of a vehicle near the entrance gate. After that, the system will display a "*Welcome*" message on the LCD as an initial greeting. The driver is then asked to press a button to print a ticket. The system will automatically record the time of entry, print the ticket through a *thermal* printer, and send a signal to the servo motor to open the entrance barrier. Once the vehicle has successfully entered and is no longer detected by the sensor, the barrier will automatically close again. This process is fast and requires minimal manual intervention, improving the efficiency of incoming traffic.

2) Vehicle exit lane

When the vehicle is about to exit, the user must enter the ticket ID into the Python GUI via the computer interface. The system will read the ticket data, then calculate the parking duration based on the difference between the entry time and the current time. After that, the system will display the parking fee to be paid, which can be adjusted according to specific tariff policies. If the validation process is successful, the system will send a command to Arduino to automatically open the exit barrier. Just like when entering, after the vehicle has completely exited and is no longer detected by the sensor, the barrier will automatically close again, ensuring the safety and orderliness of the exit lane.

.

Figure 8. Flowchart vehicle entry process

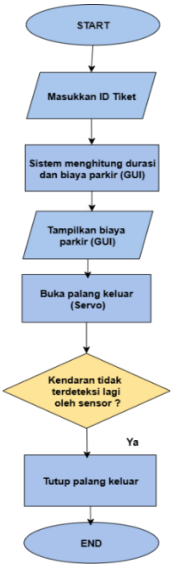

Figure 9. Flowchart vehicle exit pricess

D. Software Implementation

The software in this automatic parking system is designed using the Python programming language on the computer side and the C/C++ language on the Arduino microcontroller side. Communication between devices is carried out bidirectionally via the *Universal Asynchronous Receiver Transmitter* (UART) serial port with an easy-to-integrate string-based protocol.

The software implementation is arranged according to the *flowchart* in Figure 8 (vehicle entry process) and Figure 9 (vehicle exit process). Each block in the *flowchart* is represented in the form of Arduino and Python code snippets as follows.

1). Vehicle Entry Detection (Arduino)

```
// Arduino
long jarakMasuk = bacaJarak(TRIG_MASUK, ECHO_MASUK);
if (jarakMasuk <= 10) {
  Serial.println("MOBIL_TERDETEKSI");
  lcd.setCursor(0, 0);
  lcd.print("Selamat Datang!");
  lcd.setCursor(0, 1);
  lcd.print("Tekan Tombol");
}
```

The code above represents the "*Vehicle detected by sensor*?" block in the vehicle entry process *flowchart*. If the distance between the car and the sensor is less than 10 cm, the system will display a "*Welcome*" message on the LCD and send a signal to the computer via the serial port.

2). Process of saving entry times and printing tickets (Arduino & Python)

```
// Arduino
if (digitalRead(BUTTON_PIN) == LOW) {
  Serial.println("TOMBOL_DITEKAN");
  lcd.clear();
  lcd.print("Cetak Tiket...");
  delay(2000);
  lcd.setCursor(0, 1);
  lcd.print("Silakan Masuk");
}

# Python
times = datetime.now()
def proses_tiket_masuk():
    tid = generate_id() # create a unique ticket ID
    masuk = times.strftime('%Y-%m-%d %H:%M:%S')
    simpan_tiket(tid, masuk) # save to CSV database
    cetak_tiket(tid, masuk) # print tickets
    kirim_perintah("BUKA_PALANG_MASUK") # open the gate
```

The code in the snippet above represents the "*Button pressed* ?" block in the *flowchart*. When the physical button is pressed by the user, Arduino will display the status "*Printing Ticket...*" on the LCD, then Python continues the process by generating a unique ticket ID, recording the entry time in the database, printing the parking ticket via a *thermal* printer, and sending a command to the microcontroller to open the entrance barrier so that vehicles can pass through.

3). The barrier opens and closes after the vehicle passes through (Arduino)

```
// Arduino
void bukaPalangMasuk() {
  servoMasuk.attach(SERVO_MASUK_PIN);
  gerakServoHalus(servoMasuk, 90, 0);
  Serial.println("PALANG_MASUK_DIBUKA");
  servoMasuk.detach();
}
tungguKendaraanKeluar(TRIG_MASUK, ECHO_MASUK, servoMasuk, true);

if (perintah == "BUKA_PALANG_KELUAR") {
  Serial.println("PALANG_KELUAR_DIBUKA");
  bukaPalangKeluar();
  tungguKendaraanKeluar(TRIG_KELUAR, ECHO_KELUAR, servoKeluar, true);
  Serial.println("PALANG_KELUAR_DITUTUP");
}
```

The code snippet above represents the "*Open entrance barrier*", "*Vehicle passes sensor*", and "*Close barrier*" blocks on the automatic parking system *flowchart*. When a vehicle enters, the servo slowly moves from the 90° position to 0° so that the barrier opens. The *tungguKendaraanKeluar* function ensures that the vehicle has completely passed the sensor before the barrier is closed again. A similar process applies when exiting: when the "*BUKA_PALANG_KELUAR*" command is received from the computer, the system opens the exit barrier, waits for the vehicle to pass the sensor, and then automatically closes the barrier again.

According to Pradana et al., the application of serial communication between Arduino Uno and desktop applications has proven effective in smart parking systems, enabling the direct exchange of sensor status and barrier commands without the need for additional networks [8].

1) Arduino and Python Communication

The Arduino Uno reads data from ultrasonic sensors and input buttons (*push buttons*), and controls actuators such as servo motors and LCDs. When a vehicle is detected by the sensor and the button is pressed, the Arduino sends a specific signal via USB serial to the Python interface.

Conversely, Python can also send commands back to Arduino to move the servo, open or close the gate, and set the system flow sequence. The communication protocol is carried out using regular text messaging.

Example signal:

- "KENDARAAN_TERDETEKSI": sent from Arduino when the sensor detects a vehicle
- "TOMBOL_DITEKAN": sent when the button is pressed
- "BUKA_PALANG_MASUK" and "BUKA_PALANG_KELUAR": sent from Python to Arduino to control the servo.

2) User Interface (GUI)

The user interface was created with the Tkinter library, Python's built-in GUI module. This GUI has two main functions:

1. Entry Ticket Recording When the button signal is received, the system records the ticket ID and entry time in a CSV file (data_tiket.csv) and immediately prints the ticket.
2. Exit and Fee Calculation The user enters the ticket ID. The system calculates the duration based on the entry time and the current time, then displays and prints the parking fee.

Additional GUI features:

- Display a list of parked vehicles
- Save all data to a local file
- Directly connected to a *thermal* printer (ESC/POS).

3) Automatic Ticket Printing

The parking ticket printing system uses a *thermal* printer that supports the ESC/POS protocol and is connected via a serial port (e.g., COM4). The Python interface manages printing using the *python-escpos* library, which allows text to be sent directly to the printer. The information printed includes the ticket ID, entry time, and hourly rate. Once the ticket is printed, the GUI immediately sends a signal to the Arduino to open the entry gate.

III. Results and Discussion

After going through the design and testing process, the result was a prototype of an automatic parking system built using an Arduino Uno microcontroller and a Python-based GUI interface. This prototype was realized in the form of a miniature parking booth that represented the main features of an automatic parking system, such as vehicle detection, ticket printing, and control of entry and exit barriers. The physical visualization of the implemented system is shown in Figure 10.

Figure 10. Miniature prototype of an automated parking system.

A. *System Implementation Results*

This automatic parking system consists of several main components, namely Arduino Uno as the control center, HC SR04 ultrasonic sensor, servo motor, input button (push button), 16×2 LCD, and *thermal* printer. These components have been proven effective in various similar studies. Research by Auliani et al. shows that the Arduino Uno and HC SR04 can be used accurately and responsively in microcontroller-based automatic parking systems through simulations and prototypes that have been tested for functionality [9].

The entire system is assembled in the form of a miniature parking booth. When a vehicle is detected by the entry sensor, the LCD displays a message, then a button is pressed to print a ticket. The entry data is automatically stored, the barrier opens, and closes again after the vehicle passes the sensor.

Figure 11. The vehicle is detected by the sensor upon entry and the system automatically prints a ticket.

For the exit process, the sensor will detect that a vehicle wants to exit. The driver gives the ticket to the operator, who simply enters the ticket ID on the GUI, then the system calculates the parking duration and displays the total cost.

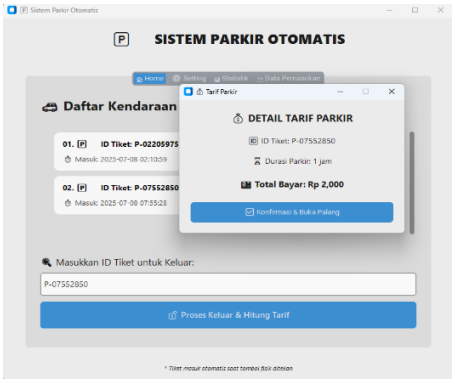

Figure 12. Python GUI Interface for entering vehicle exit ticket IDs

After the ticket ID is entered by the operator into the GUI, the system will automatically process the data and send a command to the Arduino to open the exit barrier. When a vehicle is detected exiting through the ultrasonic sensor, the system will wait until there are no more objects in the sensor area, then the barrier will automatically close again. This process ensures that the vehicle has completely left the parking area before the barrier closes again, thus avoiding errors upon exit.

According to Tanjung et al., the application of ultrasonic sensors on automatic parking barriers ensures that the barrier will only close after the vehicle has completely passed through, thereby improving accuracy and preventing operational disruptions [10].

Figure 13. Simulasi Simulation of vehicles exiting through automatic barriers
.

B. *Component Function Testing*

Testing was conducted on all components used in the automatic parking system to ensure that they functioned as designed. The following are the test results for each component:

- HC-SR04 Ultrasonic Sensor: This sensor can detect vehicles at a distance of less than 15 cm. When a vehicle is detected at the entrance gate, the sensor sends a signal to the Arduino to display the message "*Welcome*" on the LCD and activate the ticket input flow. The sensor is also used at the exit gate to detect vehicle movement before and after the barrier opens. The effectiveness of the HC-SR04 in detecting close-range objects has been proven in research by Brilliantoro & Fitriani, where this sensor was able to provide accurate responses in an Arduino-based system [11]. After the vehicle is detected by the sensor. The test results show that Arduino receives the input well and immediately executes the ticket printing process and opens the barrier. No delays or interruptions were found in the input retrieval.
- Two servo motors are used to move the barrier in and out. The servo movement is smooth and responsive. The servo opens after the ticket is printed, then closes again after the vehicle is no longer detected by the sensor. The servo function has also been tested with a simulation of a vehicle exiting, and shows stable performance. This is in line with research results showing that the use of two servo motors in an automatic parking barrier system is capable of producing automatic, responsive, and stable opening and closing movements after a vehicle is detected by an ultrasonic sensor [12].
- 16x2 I2C LCD, used as an information display during the parking process. Testing shows that the LCD can display dynamic messages such as "*Welcome*", "*Press Button*", and "*Parking System*".
- *Thermal* Printer, Printer connected via COM4 serial port. Testing was conducted by printing tickets when the button was pressed. The printer can print ticket information quickly and clearly, as long as serial communication is not interrupted.
- Arduino–GUI (Python) Serial Communication, Two-way communication via the COM3 port has been tested and is running stably. Arduino successfully sends commands such as KENDARAAN_TERDETEKSI and TOMBOL_DITEKAN to the Python GUI. Conversely, the BUKA_PALANG_KELUAR command from the GUI can be received by Arduino and executed according to the system logic.

Overall, all components were successfully tested and demonstrated good functionality. Inter-component interactions also ran synchronously, enabling the system to execute the automatic parking flow completely and efficiently.

TABLE II. SUMMARY OF PARKING SYSTEM COMPONENT TEST RESULTS

No.	Component	Function	Brief Test Results
1.	HC-SR04 (Sensor)	Detection of vehicles entering and exiting	Stable response at a distance ≤ 15 cm
2.	Input Button (*Push Button*)	Ticket printing trigger	No delay, stable input
3.	Servo Motor	Automatic crossbar movement in & out	Smooth, accurate, open-close function
4.	LCD 16x2 I2C	Information Display	Messages appear directly without error
5.	*Thermal* Printer	Print parking tickets	Fast, clear, via COM4 serial
6.	Arduino – GUI Serial	System communication	Bidirectional stable (COM3 & COM4 active)

Table II summarizes the main test results for each component that makes up this automated parking system. All components operate synchronously according to a pre-designed logic flow.

C. *System Advantages and Limitations*

This automated parking system has a number of advantages that support the efficiency and effectiveness of the parking process, especially on a prototype scale and in a limited environment. One of the main advantages is the system's ability to run the entire process automatically, from vehicle detection and ticket printing to controlling the entry and exit barriers. The integration between the hardware (Arduino) and software (Python GUI) runs smoothly and stably. The system is also designed to be *user-friendly* with

a simple interface that is easy for parking attendants to operate.

In addition, the use of ultrasonic sensors as vehicle detectors has proven effective in identifying the presence of vehicles at short distances. The use of *thermal* printers that can be used immediately without requiring complex configuration or driver installation is an added value because it enables instant and efficient ticket printing. The addition of an LCD as an information medium further strengthens the interactive aspect of the system. This is in line with the findings of Savitri and Paramytha, who stated that *thermal* printers in parking systems can print information directly without the need for further configuration, thereby speeding up the process and minimizing technical errors [13].

However, this system still has several limitations. One of them is the limited detection range of ultrasonic sensors, which are only optimal at a certain distance range (5–15 cm), so the system is not yet fully ready to be applied on a large scale in the field without modification. This is in line with research by Nugroho et al., which states that ultrasonic sensors have limited accuracy under certain conditions, such as uneven surfaces and inconsistent object reflections [14]. In addition, dependence on a cable connection between the Arduino and laptop can limit the flexibility and mobility of the system. The system is also not equipped with an online database or data backup feature, so data is only stored locally in CSV files. On the other hand, ticket ID input still requires the involvement of an operator, so it is not yet completely free of manual interaction. As stated by Bahri & Hutagalung, smart parking systems that do not use centralized storage and still rely on manual input will face efficiency issues and the risk of data loss [15].

By understanding these advantages and limitations, this system can be further developed to become a more robust automated parking solution that is ready to be implemented on a wider scale.

IV. CONCLUSION

This research successfully designed and implemented a prototype of an Arduino Uno microcontroller-based automatic parking system integrated with a Python-based GUI software interface. This system is capable of performing the parking process automatically, from vehicle detection, ticket printing, entry time data storage, to fare calculation and exit gate opening.

The tests conducted showed that all system components, including ultrasonic sensors, servo motors, input buttons (*push buttons*), LCDs, and *thermal* printers, worked well and were fully integrated. The system can respond to real-world conditions such as the presence of vehicles in front of the barrier, and is capable of printing tickets and calculating fares automatically based on parking duration.

The advantages of this system lie in its automated parking flow, *hardware* and *software* integration, and simple user interface. However, the system still has several limitations, such as limited sensor range, dependence on data cables, and the lack of a *cloud-based* storage system.

Overall, this prototype shows potential as an initial solution in the development of small-scale automated parking systems, and can be further improved to support implementation in real-world environments with the addition of features such as *Quick Response* (QR) Code reading, *Internet of Things* (IoT)-based monitoring, and wireless connectivity.

REFERENCES

[1] M. Monica Mardhalena and N. Dian Nathasia, "Parking Sensor System Untuk Mendeteksi Jarak Aman Kendaraan Menggunakan Sensor Ultrasonic Berbasis Arduino Uno Atmega328," Jakarta, 2022.

[2] F. Priyulida, R. A. Putra, and H. Situmorang, "Palang Pintu Parkir Otomatis Berbasis Arduino Uno," *Go Infotech: Jurnal Ilmiah STMIK AUB*, vol. 30, no. 1, pp. 87–95, Jun. 2024, doi: 10.36309/goi.v30i1.263.

[3] S. Tammimah, "Sistem Ketersediaan Slot Parkir Mobil Menggunakan Mikrokontroler Atmega328p (Studi Kasus : Stt Wastukancana)."

[4] T. N. Arifin, G. Febriyani Pratiwi, and A. Janrafsasih, "Sensor Ultrasonik Sebagai Sensor Jarak," vol. 2, pp. 55–62, Jul. 2022, [Online]. Available: http://jurnal.undira.ac.id/index.php/jurnaltera/

[5] M. Khamidah, W. Suciyati, H. R. Ayu, G. A. Pauzi, and A. Royhan, "Activator Token KWh Meter Using Servo Motor SG90 Based on Arduino Uno Microcontroller," Lampung, Apr. 2023. [Online]. Available: https://jemit.fmipa.unila.ac.id/

[6] F. Rizakir and S. A. Sukarno, "Sistem Kunci Otomatis Pada Casing Rokok Berbasis Arduino Nano Dengan Lcd I2c," *Jurnal Informatika dan Teknik Elektro Terapan*, vol. 13, no. 1, Jan. 2025, doi: 10.23960/jitet.v13i1.5661.

[7] python-escpos developers, "python-escpos Documentation – Serial Printer," ReadTheDocs / GitHub.

[8] R. G. Pradana, "E-Jurnal Prodi Teknik Elektronika Edisi Proyek Akhir D3," Yogyakarta, 2016.

[9] Qonita Auliani, Riyan Aditya, Muhammad Dicky Saputra, Muhammad Aldi Firdaus, Bryant Reza Pahlevi, and Didik Aribowo, "Simulasi Sensor Parkir Berbasis Mikrokontroler Arduino Uno dengan Sensor HC-SR04 Menggunakan Website Wokwi," *Jurnal Teknik Mesin, Industri, Elektro dan Informatika*, vol. 3, no. 4, pp. 259–273, Nov. 2024, doi: 10.55606/jtmei.v3i4.4557.

[10] R. A. Tanjung, T. Rijanto, F. Baskoro, and R. Firmansyah, "Pengembangan Sistem Palang Pintu Otomatis Di Tempat Parkir FT UNESA Menggunakan Sensor RFID dan Sensor Ultrasonik Berbasis Bot Telegram," Surabaya, 2025.

[11] B. Brilliantoro, "Rancang Bangun Alat Pendeteksi Jarak Aman Mobil Menggunakan Sensor Ultrasonik Hc-Sr04 Dan Arduino Uno," Bandung.

[12] M. H. Susanta, "Prototype Penggunaan Empat Sensor Ultrasonik Pada Palang Parkir Otomatis Berbasis Arduino Uno," *Jurnal Ilmiah Sains dan Teknologi*, Jul. 2024.

[13] C. E. Savitri and N. Paramytha, "Prototipe Sistem Monitoring Parkir Mobil Berbasis Esp32 Di Universitas Bina Darma Palembang," *Jurnal Ampere*, 2023, doi: 10.31851/ampere.

[14] E. Surya Aby Nugroho, N. Diana Resty, I. Hudati, and P. Studi Sarjana Terapan Teknologi Rekayasa Instrumentasi dan Kontrol Fakultas Sekolah Vokasi, "Implementasi Filter Kalman Pada Sensor Jarak Berbasis Ultrasonik," *Jurnal Listrik, Instrumentasi dan Elektronika Terapan*, vol. 2, no. 2, Oct. 2021.

[15] S. Bahri and D. Durbin Hutagalung, "Sistem Parkir Cerdas Berbasis Internet Of Things," *OKTAL : Jurnal Ilmu Komputer dan Science Sistem Parkir Cerdas Berbasis Internet Of Things*, vol. 2, Nov. 2023, Available: https://journal.mediapublikasi.id/index.php/oktal