Design and Development of Rover Using ESP32 And Mission Planner

D. Pavana Kumari[1], Dr. R. Prasad Rao[2], S. Divya[3], V.Sai Suvarna[4], B. Dileep[5], K.Rama Mohan[6]
[1] *Assistant Professor, Department of ECE, Avanthi Institute of Engineering & Technology*
[2] *Professor, Department of ECE, Avanthi Institute of Engineering & Technology*
[3, 4, 5,6] *Student, Department of ECE, Avanthi Institute of Engineering & Technology*

***Abstract*—Autonomous robots are becoming an important part of modern technology due to their ability to perform tasks without direct human intervention. These robots can operate in dangerous, remote, or difficult environments where human presence may not be possible or safe. One of the most common applications of autonomous robots is the rover, which can move across terrain while collecting data or performing missions. This project focuses on developing a rover that can navigate autonomously using GPS coordinates. The rover uses the ESP32 microcontroller as its main control unit. ESP32 is a powerful microcontroller with built-in WiFi and Bluetooth capabilities, making it suitable for IoT-based robotics applications. [1][2] [3]**

The navigation of the rover is controlled using Mission Planner software. Mission Planner allows users to define waypoints and mission paths that the rover must follow. These waypoints represent GPS coordinates that guide the rover along a predefined route. A NEO-6M GPS module is used to determine the rover's real-time geographic position. The GPS module communicates with the ESP32 and continuously provides latitude and longitude data. The rover compares its current location with the target waypoint and calculates the distance between them. [4]

***Index Terms*—ESP32, Autonomous Rover, GPS Navigation, Mission Planner, IoT, Wi-Fi Communication, NEO-6M GPS, Cloud-Based Monitoring.To enable remote communication and monitoring, the system uses WiFi connectivity. Waypoints are uploaded to a cloud database using a Python script and Google Sheets integration. The rover downloads the waypoints and begins navigation automatically. Additionally, the rover includes a WiFi-based manual control system. When the rover reaches the final waypoint, it switches to WiFi control mode, allowing users to manually control the rover through a mobile browser.This project demonstrates the integration of robotics, GPS navigation, IoT communication, and cloud-based mission management to create an intelligent autonomous rover system.**

I. INTRODUCTION

Autonomous robots are becoming an important part of modern technology due to their ability to perform tasks without direct human intervention. These robots can operate in dangerous, remote, or difficult environments where human presence may not be possible or safe. One of the most common applications of autonomous robots is the rover, which can move across terrain while collecting data or performing missions. [1][2]

This project focuses on developing a rover that can navigate autonomously using GPS coordinates. The rover uses the ESP32 microcontroller as its main control unit. ESP32 is a powerful microcontroller with built-in Wi-Fi and Bluetooth capabilities, making it suitable for IoT-based robotics applications. [3]

The navigation of the rover is controlled using Mission Planner software. Mission Planner allows users to define waypoints and mission paths that the rover must follow. These waypoints represent GPS coordinates that guide the rover along a predefined route. [4]

A NEO-6M GPS module is used to determine the rover's real-time geographic position. The GPS module communicates with the ESP32 and continuously provides latitude and longitude data. The rover compares its current location with the target waypoint and calculates the distance between them.

To enable remote communication and monitoring, the system uses Wi-Fi connectivity. Waypoints are

uploaded to a cloud database using a Python script and Google Sheets integration. The rover downloads the waypoints and begins navigation automatically.
Additionally, the rover includes a Wi-Fi based manual control system. When the rover reaches the final waypoint, it switches to Wi-Fi control mode, allowing users to manually control the rover through a mobile browser.
This project demonstrates the integration of robotics, GPS navigation, IoT communication, and cloud-based mission management to create an intelligent autonomous rover system.

II. LITERATURE SURVEY

2.1 Traditional Methods

Traditional robotic rovers were mainly controlled manually using RF transmitters or wired communication. These systems required continuous human supervision. Basic microcontrollers like Arduino UNO or PIC were used for simple tasks such as motor control and sensor handling.

Navigation methods included line-following using infrared sensors and obstacle detection using ultrasonic sensors. While effective in controlled environments, these methods failed in outdoor conditions. Manual joystick control provided flexibility but lacked autonomy.
Communication was limited to short-range technologies like RF and Bluetooth, restricting operational range. Due to these limitations, traditional systems were not suitable for autonomous and large-scale applications.

2.2 IoT in Robotics

IoT enhances robotic systems by enabling communication between devices, sensors, and cloud platforms. Controllers like ESP32 provide built-in Wi-Fi for internet connectivity. [6]

In this project, waypoint data from Mission Planner is processed using Python and stored in Google Sheets. The ESP32 retrieves this data for navigation, allowing dynamic updates without reprogramming.
IoT also enables real-time monitoring using Telegram, improving flexibility, scalability, and remote control capabilities.

2.3 Learning Modules

Learning modules improve navigation and decision-making in robotic systems. GPS-based algorithms help the rover move toward waypoints accurately.
The system continuously processes GPS data to calculate distances and adjust movement. Cloud-based data storage allows performance improvements through analysis.
In this project, integration of Mission Planner, Python, Google Sheets, and ESP32 enables autonomous navigation based on waypoint learning.

2.4 Research Gaps

Key challenges in existing systems include:

- High cost due to advanced hardware like LiDAR
- Complex algorithms requiring high computational power
- Lack of simple cloud-based mission management
- Limited real-time communication and monitoring
- Absence of GPS navigation in low-cost systems

These gaps highlight the need for a low-cost, IoT-enabled autonomous rover.

2.5 GPS-Based Navigation

GPS enables accurate outdoor navigation using satellite signals. Modules like NEO-6M provide latitude and longitude data. [5]
Distance to waypoints is calculated using formulas such as the Haversine formula. GPS allows operation without predefined paths but may face signal issues in obstructed areas.In this project, GPS data guides the rover toward target locations in real time.

2.6 Cloud-Based Mission Planning

Cloud-based systems allow remote mission updates and centralized control. Mission Planner generates waypoint files, which are processed and uploaded to Google Sheets. [8]
The ESP32 retrieves this data via the internet, eliminating the need for firmware updates. This improves scalability, flexibility, and system coordination.

2.7 Wireless Communication and Monitoring

Wireless communication enables remote monitoring and control. ESP32 uses Wi-Fi for data transfer and control functions. [7]

The rover sends real-time updates via Telegram. After reaching the destination, it creates a Wi-Fi access point for manual control through a web interface.
This improves usability and supports applications like autonomous vehicles and remote inspection systems.

III. PROPOSED SYSTEM

The proposed system introduces a low-cost autonomous rover that integrates ESP32, GPS navigation, cloud connectivity, and Mission Planner software. The system is designed to provide waypoint-based navigation with remote monitoring and manual control capabilities.
In this system, the ESP32 microcontroller serves as the main controller responsible for navigation and communication. The NEO-6M GPS module continuously provides real-time location information, which is used by the ESP32 to calculate the rover's position relative to predefined waypoints.
Mission Planner software is used to define and generate waypoint missions. These waypoints are stored in a waypoint file, which is monitored by a Python application. The Python script automatically extracts GPS coordinates and uploads them to Google Sheets.The ESP32 connects to the internet via Wi-Fi and downloads the waypoint data from the cloud server. The rover then navigates toward the target waypoint by calculating the distance between the current GPS position and the waypoint coordinates.
The L298N motor driver controls the rover's motors, allowing it to move forward, backward, and turn left or right. When the rover reaches the waypoint, it stops and switches to Wi-Fi control mode.In Wi-Fi control mode, the ESP32 creates a wireless access point and hosts a web interface that allows users to control the rover using a smartphone or computer.The system also sends real-time status messages and GPS coordinates to a Telegram bot, allowing users to monitor the rover remotely.

Advantages of Proposed System

- Low-cost hardware implementation
- IoT-based waypoint management
- GPS-based autonomous navigation
- Real-time monitoring using Telegram
- Wi-Fi-based manual control system
- Easy integration with Mission Planner
- Scalable for future robotics applications

3.1 BLOCK DIAGRAM

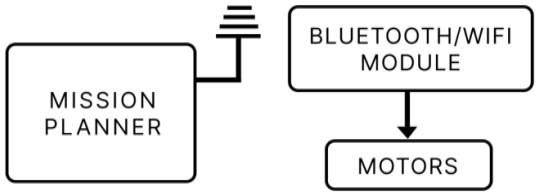


Figure 1.Block Diagram

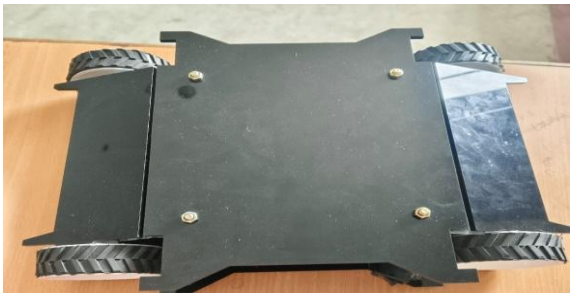
Block Diagram Explanation
The block diagram represents the overall architecture of the autonomous rover system and shows how different modules interact with each other.

At the center of the system is the ESP32 microcontroller, which acts as the main processing unit. It receives GPS data from the NEO-6M GPS module through serial communication. The GPS module continuously provides real-time latitude and longitude information.

Mission Planner software is used to generate waypoint files containing navigation coordinates. These files are monitored by a Python script running on a computer. The script extracts the waypoint coordinates and uploads them to a Google Sheets database.The ESP32 connects to the cloud server through WiFi and downloads waypoint data using HTTP requests. Once the waypoint coordinates are received, the ESP32 calculates the distance between the rover's current location and the target waypoint.

The motor driver module (L298N) receives control signals from the ESP32 and drives the DC motors accordingly. These motors rotate the rover wheels, allowing the rover to move toward the destination. When the rover reaches the waypoint, the ESP32 stops the motors and activates WiFi control mode. A web interface is generated to allow manual control of the rover. Additionally, the ESP32 sends real-time updates about the rover's location and navigation status to a Telegram bot, enabling remote monitoring.

IV. OUTPUTS



Rover Top View

Rover Side View

Figure 2: Rover top view and side view

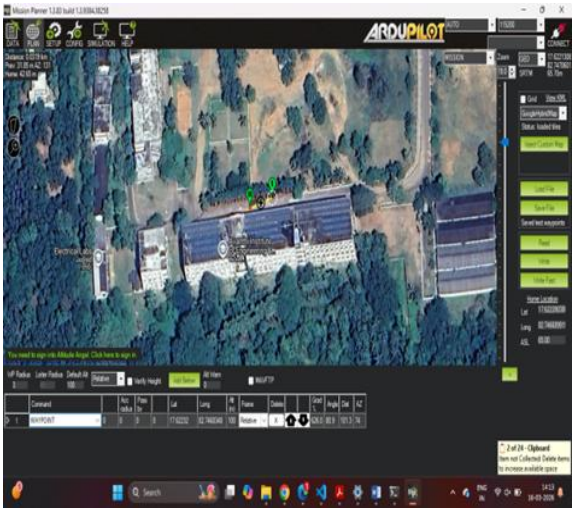


Figure 3: Google Map

V. CONCLUSION

The project titled “Design and Development of Rover using ESP32 and Mission Planner” successfully demonstrates the implementation of an autonomous navigation system using IoT technologies and GPS-based waypoint tracking. The system integrates hardware components such as ESP32 microcontroller, GPS module, motor driver, and DC motors with software tools including Mission Planner, Python scripts, and cloud-based databases. By combining these technologies, the rover is capable of navigating autonomously toward predefined waypoints while continuously monitoring its location.

The developed system also provides additional features such as WiFi-based manual control and real-time monitoring through Telegram notifications. This enhances the flexibility and usability of the rover for various real-world applications. The project demonstrates a low-cost and scalable solution for autonomous robotic navigation and can be further enhanced by integrating obstacle detection sensors, cameras, and artificial intelligence algorithms for advanced navigation and decision-making.

REFERENCES

[1] S. Thrun et al., “Stanley: The robot that won the DARPA Grand Challenge,” Journal of Field Robotics, vol. 23, no. 9, pp. 661–692, 2006.

[2] R. Siegwart, I. Nourbakhsh, and D. Scaramuzza, Autonomous Mobile Robots, MIT Press, 2011.

[3] Espressif Systems, “ESP32 Series Datasheet,” 2023.

[4] M. Oborne, “Mission Planner Ground Control Station,” ArduPilot Documentation, 2022.

[5] U-blox, “NEO-6 GPS Modules Data Sheet,” 2021.

[6] A. Al-Fuqaha et al., “Internet of Things: A Survey on Enabling Technologies,” IEEE Communications Surveys & Tutorials, vol. 17, no. 4, pp. 2347–2376, 2015.

[7] K. Ashton, “That ‘Internet of Things’ Thing,” RFID Journal, 2009.

[8] H. Choset et al., Principles of Robot Motion: Theory, Algorithms, and Implementations, MIT Press, 2005.