VISVESVARAYA TECHNOLOGICAL UNIVERSITY
Jnana Sangama, Belagavi - 590 018

MINI PROJECT REPORT ON

AUTONOMOUS ROVER

Thesis submitted in partial fulfillment for the Award of Degree of
Bachelor of Engineering
in
Electronics and Communication Engineering

Submitted by

SHARANYA N	1RN21EC128
SHREYA G PRASAD	1RN21EC135
SUSHRUTHA S ATHREYA	1RN21EC147
VIDYASAGAR JAIN	1RN21EC166

Under the Guidance of
Chethana J
Assistant Professor

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
(Accredited by NBA for the Academic years 2022-25)

RNS INSTITUTE OF TECHNOLOGY
Autonomous Institution Affiliated to VTU Recognized by GOK,
Approved by AICTE (NAAC 'A+' Accredited, NBA Accredited (UG - CSE, ECE, ISE, EIE and EEE)
Channasandra, Dr.Vishnuvardhan Road, Bengaluru-560098
2023-2024

VISVESVARAYA TECHNOLOGICAL UNIVERSITY
Jnana Sangama, Belagavi - 590 018

MINI PROJECT REPORT ON

AUTONOMOUS ROVER

Thesis submitted in partial fulfillment for the Award of Degree of
Bachelor of Engineering
in
Electronics and Communication Engineering

Submitted by

SHARANYA N	1RN21EC128
SHREYA G PRASAD	1RN21EC135
SUSHRUTHA S ATHREYA	1RN21EC147
VIDYASAGAR JAIN	1RN21EC166

Under the Guidance of
Chethana J
Assistant Professor

ESTD 2001

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
(Accredited by NBA for the Academic years 2022-25)

RNS INSTITUTE OF TECHNOLOGY
Autonomous Institution Affiliated to VTU Recognized by GOK,
Approved by AICTE (NAAC 'A+' Accredited, NBA Accredited (UG - CSE, ECE, ISE, EIE and EEE)
Channasandra, Dr.Vishnuvardhan Road, Bengaluru-560098
2023-2024

RNS INSTITUTE OF TECHNOLOGY

Autonomous Institution Affiliated to VTU Recognized by GOK,
Approved by AICTE (NAAC 'A+' Accredited, NBA Accredited (UG - CSE, ECE, ISE, EIE and EEE)
Channasandra, Dr.Vishnuvardhan Road, Bengaluru-560098
2023-2024

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
(Accredited by NBA for the Academic years 2022-25)

CERTIFICATE

Certified that the thesis work entitled **"AUTONOMOUS ROVER"** is carried out by **SHARANYA N(1RN21EC128), SHREYA G PRASAD(1RN21EC135), SUSHRUTHA S ATHREYA(1RN21EC147), VIDYASAGAR JAIN (1RN21-EC166)** in partial fulfillment for the award of degree of Bachelor of Engineering in **Electronics and Communication Engineering** of Visvesvaraya Technological University, Belgavi, during the year 2022-2023. It is certified that all corrections/suggestions indicated during internal assessment have been incorporated in the report. The project report has been approved as it satisfies the academic requirements in aspect of the project work prescribed for the award of degree of **Bachelor of Engineering**.

....	....	....
Ms.Chethana J	Dr. Vipula Singh	Dr.Ramesh Babu H S
Assistant Professor	Head of the Department	Principal

RNS INSTITUTE OF TECHNOLOGY

Autonomous Institution Affiliated to VTU Recognized by GOK, Approved by AICTE (NAAC 'A+' Accredited, NBA Accredited (UG - CSE, ECE, ISE, EIE and EEE)

Channasandra, Dr.Vishnuvardhan Road, Bengaluru-560098

2023-2024

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

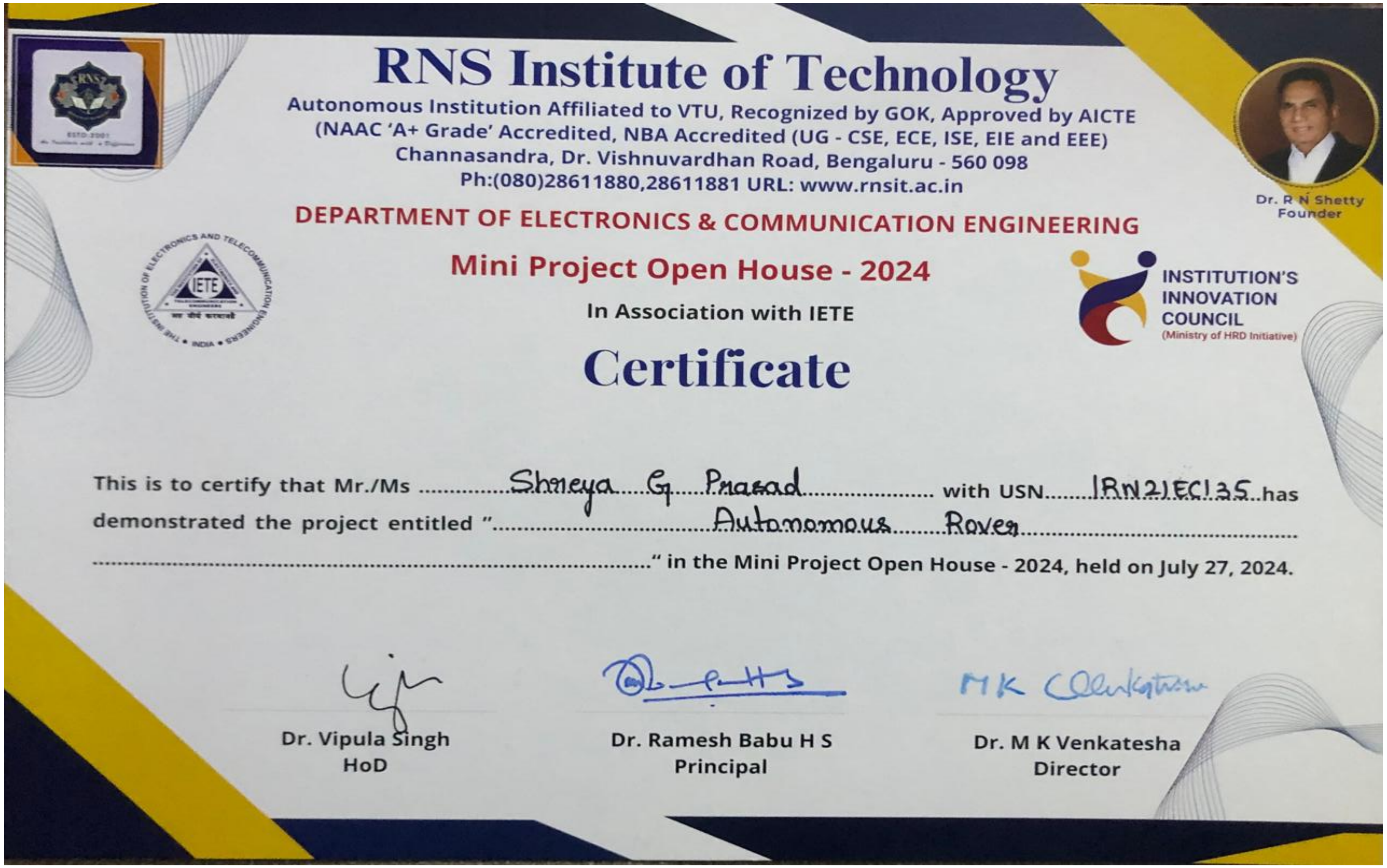
(Accredited by NBA for the Academic years 2022-25)

ESTD 2001

DECLARATION

We here by declare that the entire work emobodied in this thesis titled, **" AUTONOMOUS ROVER"** submitted to **Visvesvaraya Technological University**, Belagavi, is carried out at the department of **Electronics and Communication Engineering, RNS Institute of Technology, Bengaluru** under the guidance of **Ms. Chethana J**, Assistant Professor . This report has not been submitted for the award of any Diploma or Degree of this or any other University.

Name	USN	Signature
1.SHARANYA N	1RN21EC128	....
2.SHREYA G PRASAD	1RN21EC135	....
3.SUSHRUTHA S ATHREYA	1RN21EC147	....
4.VIDYASAGAR JAIN	1RN21EC166	....



RNS Institute of Technology

Autonomous Institution Affiliated to VTU, Recognized by GOK, Approved by AICTE
(NAAC 'A+ Grade' Accredited, NBA Accredited (UG - CSE, ECE, ISE, EIE and EEE)
Channasandra, Dr. Vishnuvardhan Road, Bengaluru - 560 098
Ph:(080)28611880,28611881 URL: www.rnsit.ac.in

Dr. R N Shetty
Founder

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

INSTITUTION'S INNOVATION COUNCIL
(Ministry of HRD Initiative)

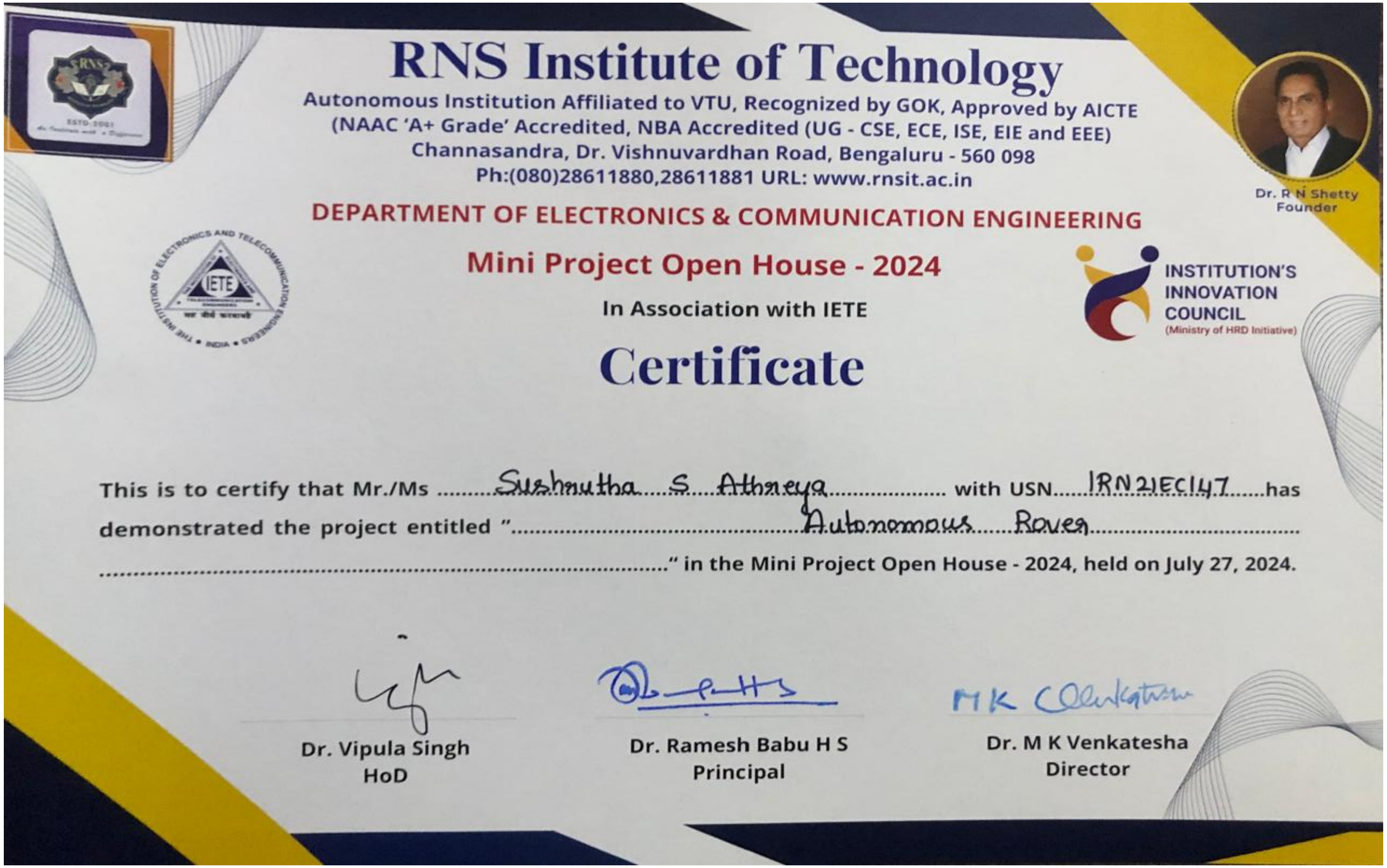
Mini Project Open House - 2024

In Association with IETE

Certificate

This is to certify that Mr./Ms ....Shreya G Prasad.......... with USN..........1RN21EC135..has demonstrated the project entitled "..........Autonomous Rover..........
.........." in the Mini Project Open House - 2024, held on July 27, 2024.

[illegible]	[illegible]	[illegible]
Dr. Vipula Singh	Dr. Ramesh Babu H S	Dr. M K Venkatesha
HoD	Principal	Director



RNS Institute of Technology

Autonomous Institution Affiliated to VTU, Recognized by GOK, Approved by AICTE
(NAAC 'A+ Grade' Accredited, NBA Accredited (UG - CSE, ECE, ISE, EIE and EEE)
Channasandra, Dr. Vishnuvardhan Road, Bengaluru - 560 098
Ph:(080)28611880,28611881 URL: www.rnsit.ac.in

Dr. R N Shetty
Founder

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

Mini Project Open House - 2024

In Association with IETE

INSTITUTION'S INNOVATION COUNCIL
(Ministry of HRD Initiative)

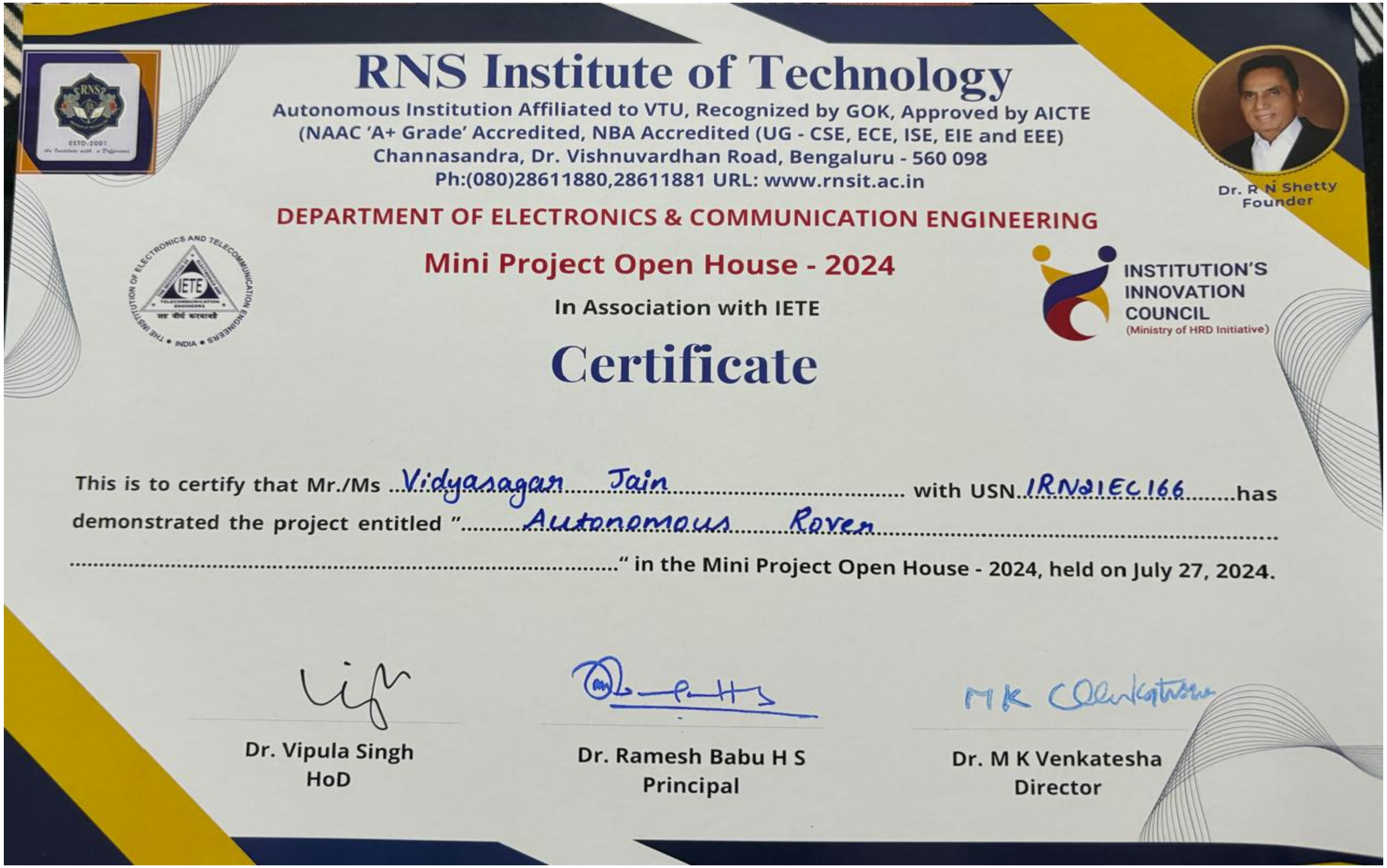
Certificate

This is to certify that Mr./Ms ......Sushrutha S Athreya.................. with USN......1RN21EC147......has demonstrated the project entitled "..........Autonomous Rover.." in the Mini Project Open House - 2024, held on July 27, 2024.

[illegible]
Dr. Vipula Singh
HoD

[illegible]
Dr. Ramesh Babu H S
Principal

MK [illegible]
Dr. M K Venkatesha
Director



RNS Institute of Technology

Autonomous Institution Affiliated to VTU, Recognized by GOK, Approved by AICTE
(NAAC 'A+ Grade' Accredited, NBA Accredited (UG - CSE, ECE, ISE, EIE and EEE)
Channasandra, Dr. Vishnuvardhan Road, Bengaluru - 560 098
Ph:(080)28611880,28611881 URL: www.rnsit.ac.in

Dr. R N Shetty
Founder

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

INSTITUTION'S INNOVATION COUNCIL (Ministry of HRD Initiative)

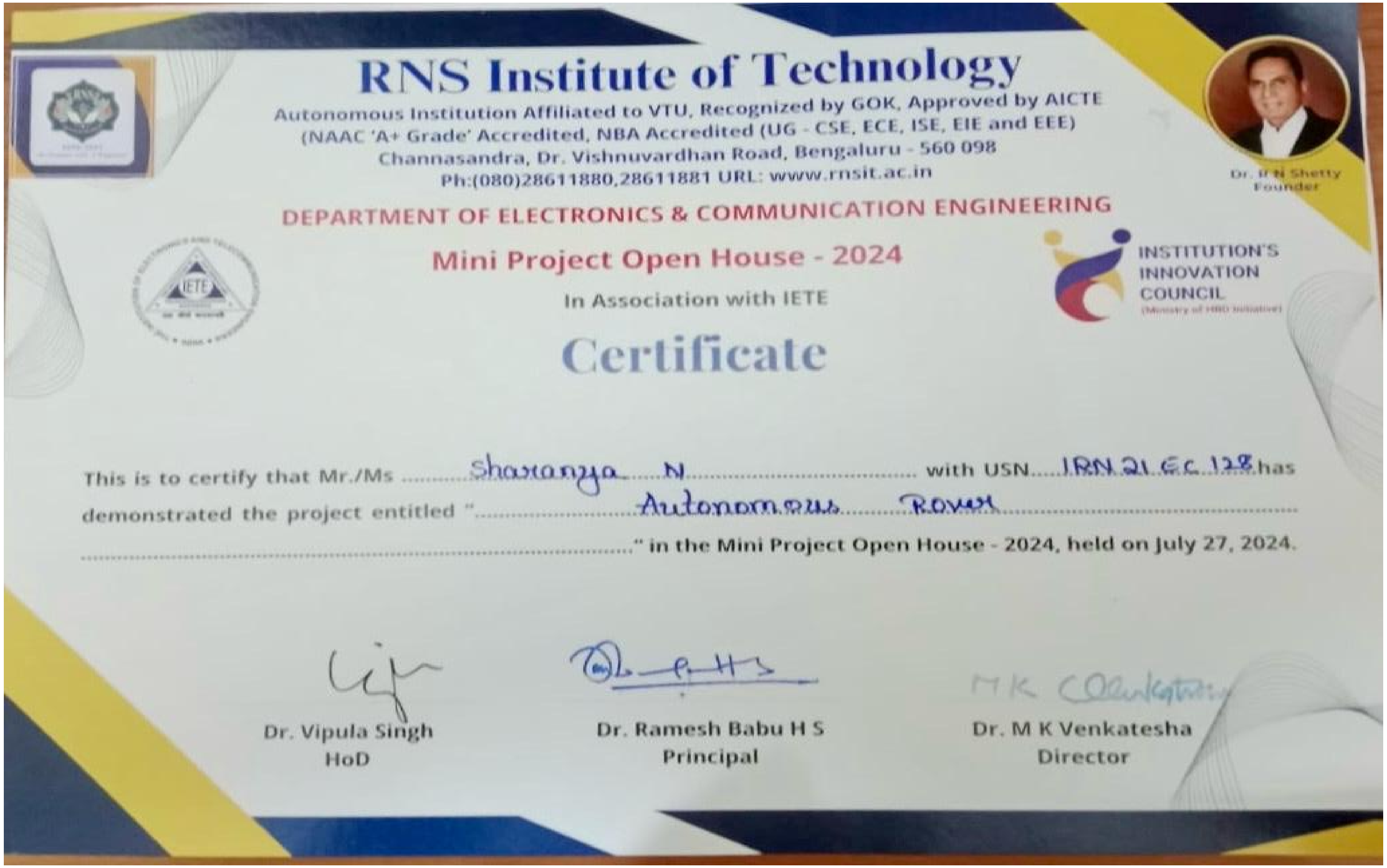
Mini Project Open House - 2024

In Association with IETE

Certificate

This is to certify that Mr./Ms ...Vidyasagar Jain............ with USN.1RN21EC166........has demonstrated the project entitled "........Autonomous Rover.." in the Mini Project Open House - 2024, held on July 27, 2024.

[illegible]	[illegible]	[illegible]
Dr. Vipula Singh HoD	Dr. Ramesh Babu H S Principal	Dr. M K Venkatesha Director



RNS Institute of Technology

Autonomous Institution Affiliated to VTU, Recognized by GOK, Approved by AICTE
(NAAC 'A+ Grade' Accredited, NBA Accredited (UG - CSE, ECE, ISE, EIE and EEE)
Channasandra, Dr. Vishnuvardhan Road, Bengaluru - 560 098
Ph:(080)28611880,28611881 URL: www.rnsit.ac.in

Dr. R N Shetty
Founder

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

INSTITUTION'S INNOVATION COUNCIL
(Ministry of HRD Initiative)

Mini Project Open House - 2024

In Association with IETE

Certificate

This is to certify that Mr./Ms ....Sharanya N........ with USN....1RN21EC128 has demonstrated the project entitled "........Autonomous Rover........" in the Mini Project Open House - 2024, held on July 27, 2024.

[illegible] [illegible] MK Venkatesha

Dr. Vipula Singh	Dr. Ramesh Babu H S	Dr. M K Venkatesha
HoD	Principal	Director

Acknowledgement

The joy and satisfaction that accompany the successful completion of any task would be incomplete without thanking those who made it possible. We consider ourselves proud to be a part of RNS Institute of Technology, the institution which moulded us in all our endeavors.

We are grateful to our Founder Chairman, RNS Group of Institutions, **Late. Dr. R N Shetty**, for providing state of art facilities.We would like to express our gratitude to our honourable **Mr.Satish R Shetty**, Managing Director, RNS Groups of Institutions, for providing state of art facilities.

We would like to express our sincere thanks to **Dr. M K Venkatesha**, Director, **Dr. Ramesh Babu H S**, Principal and **Dr. Vipula Singh**, Professor and HOD, Department of ECE, for their valuable guidance and encouragement throughout our program.

We express our profound gratitude to the coordinators who have given valuable suggestions and guidance throughout the project. We would like to express our sincere gratitude to our guide **Ms. Chethana J**, Associate Professor , for her guidance, continuous support and motivation in completing the project successfully.

Finally, we take this opportunity to extend our earnest gratitude and respect to our parents, teaching and non-teaching staff of the department, the library staff and all our friends who have directly or indirectly supported us.

SHARANYA N

SHREYA G PRASAD

SUSHRUTHA S ATHREYA

VIDYASAGAR JAIN

Abstract

This project develops an autonomous rover designed for efficient navigation in constrained environments. The rover integrates three key functions: localization, navigation, and locomotion. Localization is achieved using the Neo 7 GPS module, accurately determining the rover's position. For navigation, a customized path planning algorithm calculates the optimal route to a target location, while ultrasonic sensors facilitate real-time obstacle detection and avoidance, allowing the rover to autonomously navigate around obstacles.

Additionally, the rover incorporates basic decision-making capabilities, enabling it to execute simple decisions based on algorithms for route planning and obstacle avoidance. Locomotion is powered by motor drivers and motors, ensuring reliable movement along the planned path.

The primary expected outcome is a fully functional rover capable of perceiving its environment and navigating autonomously, without human intervention. This integrated approach not only enhances the rover's operational efficiency but also contributes to advancements in robotic navigation technologies, making it suitable for various applications in dynamic terrains. The successful implementation of these features will represent a significant step forward in the development of sophisticated autonomous systems.

Table of Contents

List of Figures

Acronyms

AGV : AUTOMATED GUIDED VEHICLE

IoT : INTERNET OF THINGS

IDE : INTEGRATED DEVELOPMENT ENVIRONMENT L

GPS : GLOBAL POSITIONING SYSTEM

VLSI : VERY LARGE SCALE INTEGRATION

SLAM : SIMULTANEOUS LOCALALIZATION AND MAPPING

IMU : INTERTIAL MEASUTEMENT UNIT

4WD : FOUR WHEEL DRIVE

Chapter 1

Introduction

The Fourth Industrial Revolution has ushered in significant advancements in information and telecommunication technology, driving research in core areas such as big data, artificial intelligence, and the Internet of Things (IoT). Within this context, Automated Guided Vehicles (AGVs) have gained prominence for their potential to revolutionize industrial operations. This project presents the design and prototype of a Mini Autonomous Rover utilizing a microcontroller to navigate autonomously within a given environment.
In traditional industrial settings, wired guidance methods are predominantly employed for AGV navigation due to their cost-effectiveness and reliable tracking capabilities. However, these methods suffer from significant drawbacks, including the inflexibility and high maintenance costs associated with changing or maintaining the physical guidance lines embedded in the workplace floors.
To address these challenges, this project explores a wireless guidance method, where a virtual route is pre-defined, and the AGV uses positioning sensors to determine its current location and navigate accordingly. This approach allows for autonomous travel, making route changes and maintenance more straightforward. Through this project, we aim to develop a mini AGV prototype that leverages wireless guidance for enhanced flexibility and efficiency in navigation, addressing some of the limitations of current wired guidance systems. By integrating advanced sensor technologies and robust algorithms, our solution strives to provide a more adaptable and maintainable AGV system suitable for dynamic industrial environments.

1.1 Internship Objectives

The primary objective of this project is to develop an advanced autonomous system capable of operating with minimal human intervention. By emphasizing real-time sensing and adaptation, we aim to equip our rover with the ability to navigate diverse environments and make informed decisions autonomously. The key objectives of this project are:

- Path Planning and Control:

Efficient Navigation: To develop robust path planning algorithms that enable the rover to navigate efficiently from its current location to designated target points. Dynamic Adaptation: To integrate control systems that allow the rover to dynamically adapt to environmental changes, ensuring smooth and precise movement. Algorithm Optimization: To employ advanced optimization techniques to enhance the speed and reliability of the path planning process.

- Obstacle Avoidance:

  Real-Time Detection: To implement sophisticated obstacle avoidance algorithms that enable the rover to detect obstacles in real-time using a variety of sensors. Autonomous Navigation: To ensure the rover can autonomously navigate around detected obstacles by recalculating its path in real-time, thereby avoiding collisions and ensuring continuous movement. Sensor Fusion: To utilize sensor fusion techniques to improve the accuracy of obstacle detection and environmental awareness, combining data from multiple sensor sources for comprehensive situational understanding.

1.2 Internship

As per the VTU academic scheme of 2021 we were introduced to the internship program in October.Here we were exposed to some of the major fields of electronics and communication It was effective at implying electronic methods and concepts in a variety of fields and domains, including computer engineering, solid-state physics,robotics, signal processing, telecommunication, and radio engineering. Some of the main domains we learned were:

- signals and system
- Photonics
- Communication
- Embedded system
- VLSI
- IoT

1.2.1 Photonics

The study of light waves is called photonics. It discusses the science involved in the production, detection, and control of light. The wave-particle duality of light is a dual nature. In other words, light possesses properties of both a continuous electromagnetic wave and a particle (photon). Depending on the type of interaction being watched, a certain type of light may be used.here we learned about

- FBG sensing application.
- modeling and simulation of an optical passive device.
- modeling and simulation of an optical wave guid.

1.2.2 Signals and System

A signal is a function that transmits information about a phenomenon in signal processing. Any quantity that is capable of changing across time or place can be employed as a signal to communicate between observers. As examples of signals, signal processing covers audio, video, voice, picture, sonar, and radar. Even if it doesn't carry information, a signal can be described as any observable change in a quantity over time or place (a time series).

- Speech emotion recognition
- Audio signal compression
- Image compression
- Analysis and detection of brain disease using MRI brain images
- ECG denoising using wavelet transform
- Speech background noise suppression with deep learning

1.2.3 Communication

A communication system is a collection of individual telecommunications networks, transmission systems, relay stations, tributary stations, and terminal equipment usually capable of interconnection and interoperation to form an integrated whole. some of the main domain we learnt in communication are

- Standard amplitude modulator and demodulator
- DC motor speed control
- Parity encoder/Decoder
- Design of pulse code modulation/demodulation
- CDMA mux/demux

1.2.4 Embedded system

An embedded system is a computer system—a combination of a computer processor, computer memory, and input/output peripheral devices—that has a dedicated function within a larger mechanical or electronic system. It is embedded as part of a complete device often including electrical or electronic hardware and mechanical parts. Because an embedded system typically controls physical operations of the machine that it is embedded within, it often has real-time computing constraints.some of the main domain we learnt in embedded system are

- Water level indicator circuit.
- Automatic room light controller.
- Controlling speed of Fan based on temperature.
- DC motor Speed control system.

1.2.5 VLSI

Very large-scale integration (VLSI) is the process of creating an integrated circuit (IC) by combining millions or billions of MOS transistors onto a single chip. VLSI began in the 1970s when MOS integrated circuit (Metal Oxide Semiconductor) chips were developed and then widely adopted, enabling complex semiconductor and telecommunication technologies. The microprocessor and memory chips are VLSI devices.
Before the introduction of VLSI technology, most ICs had a limited set of functions they could perform. An electronic circuit might consist of a CPU, ROM, RAM and other glue logic. VLSI enables IC designers to add all of these into one chip.some of the main domain we learnt in VLSI are

- Ring Counter -Johnson Counter
- 2:4 DECODER using NAND
- Detection of Even and Odd numbers
- Design of 4:1 MUX using basic gates / universal gates

1.2.6 Internet of Things (IoT)

The study of the Internet of Things (IoT) focuses on the interconnectivity of everyday devices through the internet. IoT explores the technology and processes involved in the production, detection, and control of interconnected systems. The core of IoT lies in the ability of devices to communicate and share data with each other and centralized systems. This interconnected network facilitates automation, remote monitoring, and intelligent decision-making. Depending on the application and the type of data being processed, specific IoT technologies and protocols are utilized to achieve efficient and secure communication. Here, we delve into the fundamentals and advancements in IoT.

1.3 Mobile robotics

Mobile robotics is a branch of robotics concerned with the development and application of robots that can move and operate in various environments. Unlike stationary robots, mobile robots possess the capability to navigate through their surroundings, enabling them to perform tasks in dynamic and unstructured settings. These robots integrate various technologies including mechanics, electronics, computer science, and artificial intelligence to achieve autonomy and adaptability. Core Components of Mobile robotics are:

1.3.1 Locomotion:

- Types of Locomotion:

  Mobile robots can use wheels, legs, tracks, or a combination of these to move. Wheeled robots are common due to their simplicity and efficiency on flat surfaces, while legged robots are designed to handle rough terrain and obstacles. Kinematics and Dynamics: The study of motion (kinematics) and the forces causing motion (dynamics) are crucial for designing efficient locomotion systems. This includes understanding the robot's degrees of freedom and ensuring stability during movement.

1.3.2 Perception:

Sensors: Mobile robots are equipped with various sensors to perceive their environment. Common sensors include cameras, LIDAR, ultrasonic sensors, and IMUs (Inertial Measurement Units). These sensors provide data that the robot processes to understand its surroundings. Sensor Fusion: Combining data from multiple sensors to create a comprehensive understanding of the environment. Techniques like Kalman filtering and particle filtering are used to improve accuracy and reliability.

1.3.3 Control Systems:

Control Theory: Essential for the precise movement and operation of mobile robots. It includes techniques like PID (Proportional-Integral-Derivative) control, model predictive control, and adaptive control. Motion Control: Ensuring that the robot follows the desired path smoothly and accurately. This involves trajectory planning and real-time adjustments based on sensor feedback.

1.3.4 Application areas

Mobile robotics can be used for a wide range of applications across various industries, offering flexibility, efficiency, and the ability to perform tasks in environments that may be hazardous or difficult for humans to access. Here are some possible uses and specific applications of mobile robotics:

1. Healthcare Hospital Logistics: Autonomous mobile robots (AMRs) can transport medical supplies, medications, and linens within hospitals, reducing the workload on healthcare staff. Surgical Assistance: Robotic systems like the Da Vinci Surgical System aid in minimally invasive surgeries, providing precision and reducing recovery times for patients. Disinfection: Robots equipped with UV-C light or chemical sprayers can autonomously disinfect hospital rooms, reducing the spread of infections.

2. Agriculture Precision Farming: Robots can plant seeds, monitor crop health, and apply fertilizers or pesticides precisely where needed, improving yield and reducing resource use. Harvesting: Autonomous harvesters can pick fruits and vegetables, operating continuously and efficiently, even in adverse weather conditions. Weeding: Robots equipped with vision systems can identify and remove weeds, reducing the need for herbicides.

3. Manufacturing and Warehousing Material Handling: Automated Guided Vehicles (AGVs) and Autonomous Mobile Robots (AMRs) can transport materials and products within factories and warehouses, optimizing workflow and reducing labor costs. Inventory Management: Robots can scan and manage inventory, ensuring accurate stock levels and reducing the likelihood of stockouts or overstocking. Assembly Line Tasks: Robotic arms and mobile platforms can work on assembly lines, performing repetitive tasks with high precision and consistency.

4. Exploration and Environmental Monitoring Space Exploration: Rovers like NASA's Mars rovers explore planetary surfaces, collecting data and samples

from environments that are not accessible to humans. Marine Research: Autonomous underwater vehicles (AUVs) can explore and map the ocean floor, study marine life, and monitor environmental conditions. Wildlife Monitoring: Drones and ground-based robots can track wildlife movements and behaviors, providing data for conservation efforts.

5. Security and Surveillance Patrolling: Security robots can autonomously patrol premises, detecting intrusions and providing real-time video feeds to security personnel. Emergency Response: Robots can be deployed in hazardous situations such as bomb disposal or search and rescue operations, reducing risk to human responders. Border Security: Autonomous systems can monitor and patrol borders, detecting unauthorized crossings and alerting authorities.

6. Domestic and Service Robotics Cleaning: Robotic vacuum cleaners and lawn mowers can autonomously clean floors and maintain lawns, providing convenience to homeowners. Personal Assistance: Service robots can assist elderly or disabled individuals with daily tasks, enhancing their quality of life and independence. Delivery: Autonomous delivery robots can transport packages, food, and other goods to customers, especially in urban environments.

7. Education and Research STEM Education: Educational robots can be used to teach students programming, robotics, and engineering concepts in an interactive and hands-on manner. Research Platforms: Mobile robots serve as research platforms for developing and testing new algorithms in fields such as artificial intelligence, machine learning, and control systems.

8. Construction and Infrastructure Surveying and Mapping: Robots can autonomously survey construction sites, creating detailed maps and models that aid in planning and monitoring progress. Maintenance and Inspection: Mobile robots can inspect infrastructure such as bridges, pipelines, and power lines, detecting faults and performing maintenance tasks without the need for human intervention. These applications illustrate the versatility and impact of mobile robotics across various sectors, highlighting their potential to improve efficiency, safety, and convenience in numerous tasks and environments.

Chapter 2

Autonomous Control in Mobile Robotics

Autonomous control in mobile robotics refers to the capability of a robot to perform tasks and navigate its environment without human intervention. This involves a combination of sensing, processing, decision-making, and action to achieve specified goals in dynamic and often unpredictable environments.[2]
Key Components of Autonomous Control:

2.1 Perception:

Sensors: Mobile robots are equipped with various sensors, such as cameras, LIDAR, ultrasonic sensors, and IMUs (Inertial Measurement Units). These sensors collect data about the environment, such as distances to objects, light levels, and motion. Sensor Fusion: Combining data from multiple sensors to create a comprehensive understanding of the robot's surroundings. Techniques such as Kalman filtering and particle filtering are commonly used to enhance data accuracy and reliability.

2.2 Localization and Mapping:

Localization: The process of determining the robot's position within its environment. This can be achieved through GPS for outdoor environments or through techniques like visual odometry and LIDAR for indoor settings. Simultaneous Localization and Mapping (SLAM): A method used by robots to build a map of an unknown environment while simultaneously keeping track of their location within that map. SLAM integrates sensor data to create real-time maps, essential for navigation in unfamiliar areas.

2.3 Path Planning:

Global Path Planning: Determining an optimal route from the robot's current location to its destination, considering the layout of the environment and any static obstacles. Algorithms such as A*, Dijkstra's, and Rapidly-exploring Random Trees

(RRT) are often used. Local Path Planning: Involves navigating the immediate vicinity, avoiding dynamic obstacles in real-time. Techniques like the Dynamic Window Approach (DWA) and potential fields guide the robot around obstacles while moving toward the goal.

2.4 Motion Control:

Kinematics and Dynamics: Understanding and modeling the robot's movement capabilities and constraints. This involves the robot's degrees of freedom and ensuring stable motion. Control Algorithms: Implementing strategies to manage the robot's motion, such as PID (Proportional-Integral-Derivative) control, model predictive control, and adaptive control. These algorithms process sensor feedback to adjust the robot's actions, ensuring smooth and accurate movement.

2.5 Decision-Making:

Artificial Intelligence (AI): Using AI to enhance the robot's ability to make decisions autonomously. Machine learning algorithms can be employed for tasks such as object recognition, path optimization, and adaptive behavior. Behavior-Based Control: Designing the robot's actions based on predefined rules and conditions. This approach allows the robot to react to environmental changes and make decisions in real-time.

2.6 Obstacle Avoidance:

Real-Time Sensing: Continuously monitoring the environment using sensors to detect obstacles. The robot must process this data rapidly to adjust its path and avoid collisions. Reactive Control: Implementing algorithms like Vector Field Histograms (VFH) and Artificial Potential Fields to dynamically adjust the robot's trajectory, ensuring safe navigation.

Achieveing autonomy in any vehicle, three major functions must be seamlessly integrated: localization, locomotion, and navigation. Each of these functions plays a crucial role in ensuring the vehicle can operate independently and effectively in various environments.

Chapter 3

Methodology and Implementation - Hardware Integration

The successful implementation of our project relies on the seamless integration of various hardware and software components.The GPS module and compass provide accurate localization, the 4WD motor driver kit and DC motors enable effective locomotion, and the ultrasonic sensors and microcontrollers facilitate advanced navigation. An explanation of the integrated hardware and software elements used are as below. Achieving autonomous control of the rover involves the seamless integration of various hardware components, each contributing essential functions to ensure effective and reliable operation. The integration of these components allows the rover to navigate its environment, make real-time decisions, and execute complex tasks autonomously. The primary hardware components include sensors, actuators, control systems, power sources, and communication devices.
The hardware components used are as explained below.

3.1 Sensors

3.1.1 GPS NEO-7M Module

The NEO-7M GPS module,as referred in fig 3.1 with its inbuilt compass, is a versatile and reliable choice for various GPS applications. Its high sensitivity, low power consumption, and multiple interfacing options make it ideal for integration into projects requiring precise location and orientation data.
Specifications :

- Receiver Type: 56-channel u-blox 7 GNSS engine
- Time Accuracy: 30 ns
- Supply Voltage: 2.7 V to 3.6 V
- Communication Interfaces: UART, USB, DDC (I2C compliant), SPI

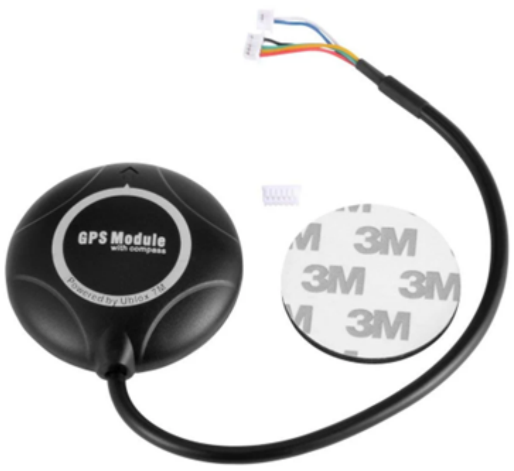


Figure 3.1: GPS 7M WITH COMPASS

By leveraging the NEO-7M's capabilities , we have successfully extracted the latitude and longitude of the pre-defined waypoints and the current location of the rover.We have used the inbuilt compass of the NEO-7M GPS module to calculate the heading of our rover, this involves interpreting magnetometer data to determine the rover's orientation relative to magnetic west. This integration is crucial for accurate navigation and path following, allowing the rover to adjust its direction effectively. The GPS module is connected to the ESP32, which extracted the coordinates and heading information. This data is then transmitted to the Arduino Uno via serial communication, where it is utilized in the code to control the rover's navigation and movement.

3.1.2 HC-SR04 Ultrasonic Sensor Module

The HC-SR04 is a widely used ultrasonic sensor module that provides precise distance measurements.as referred in fig 3.2 .The HC-SR04 operates based on the principle of echo reflection, which involves emitting ultrasonic waves and detecting the time taken for these waves to return after reflecting off an object. This time-of-flight measurement is then used to calculate the distance to the object.

Specifications:

- Operating Voltage: 5V DC
- Operating Current: 15mA
- Ultrasonic Frequency: 40 kHz

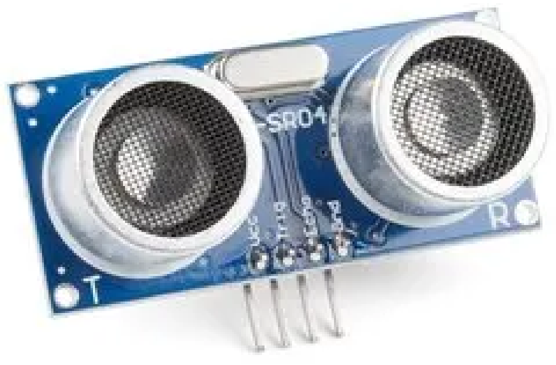


Figure 3.2: ULTRASONIC SENSOR

- Max Range: 4 meters
- Min Range: 2 cm
- Accuracy: ±3 mm

In our project, the HC-SR04 ultrasonic sensor plays a crucial role in calculating the rover's distance from obstacles and implementing obstacle avoidance. The sensor is connected to ESP32, which processes the distance measurements and then transmits the data to an Arduino Uno via serial communication for further action.

3.2 Actuators

3.2.1 L293D Motor Driver

The L293D is a popular motor driver integrated circuit (IC) used for controlling the direction and speed of DC motors and stepper motors. as referred in fig 3.3 .It acts as a bridge between microcontrollers and the motors, allowing microcontrollers to command high-current motors that they cannot directly drive. The L293D is a dual H-bridge motor driver, meaning it can control two motors independently.
Key Features and Specifications :

- Dual H-Bridge:

  Function: Allows control of two DC motors or one stepper motor.
  Bridge Configuration: Each H-bridge consists of four switches (transistors) that

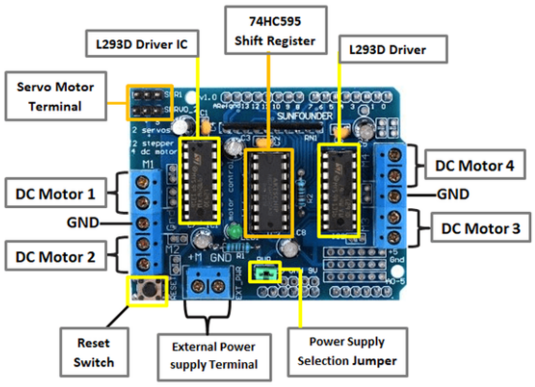


Figure 3.3: MOTOR DRIVER (as mentioned in figure 3.1.2.1)

can direct current in either direction through the motor, enabling both forward and reverse operation.

- Operating Voltage:

  Motor Supply Voltage (Vcc2): 4.5V to 36V.
  Logic Supply Voltage (Vcc1): 4.5V to 5.5V.

- Current Handling:

  Output Current (per channel): Up to 600 mA continuous.
  Peak Output Current (per channel): 1.2 A for short duration.

- Control Logic:

  Inputs: Four input pins (two per H-bridge) to control the direction of the motors.
  Enable Pins: Two enable pins (one per H-bridge) to enable or disable the operation of the motor drivers.

In our project, the L293D motor driver is utilized to drive the four DC motors that control the locomotion and direction of your rover. This allows for precise control of the rover's movement, enabling it to navigate its environment effectively.

3.2.2 Four Wheel Drive Kit

The 4WD configuration provides stability and power,as referred in fig 3.4 allowing the robot to navigate various terrains.
Components:

- Chassis: It is made of durable acrylic and provides the structural framework to mount motors, wheels, sensors, and electronics.

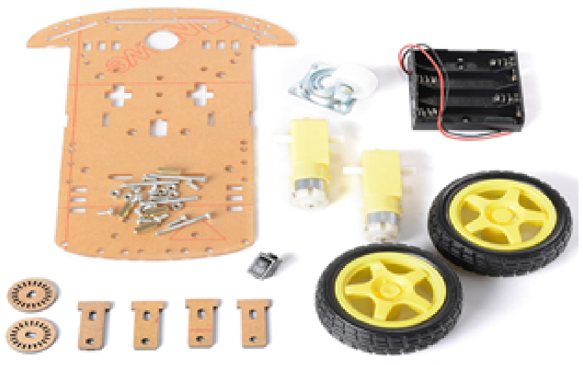


Figure 3.4: FOUR WHEEL DRIVE KIT

- DC Motors: Four independent DC motors are used, one for each wheel. They are operated via motor drivers to control speed and direction.Each motor is independently controlled, enabling differential steering (i.e., varying the speed of wheels on either side to turn).
  DC motors are an essential component in our project, providing the necessary mechanical power for locomotion and direction control. By leveraging the capabilities of DC motors, our rover can navigate and move effectively.
- Wheels: Four rubber wheels to provide good traction on various surfaces. They are connected directly to the motors.

In our project this kit is particularly beneficial for navigating various terrains and facilitating the locomotion of the rover.

3.3 Control Systems

3.3.1 Arduino Uno

It is one of the most popular microcontroller boards in the Arduino family.As referred in fig 3.5 It is widely used in education, prototyping, and hobby projects due to its simplicity, versatility, and ease of use. The Arduino Uno is based on the ATmega328P microcontroller and provides a platform for building a variety of interactive and embedded applications.
Key Features and Specifications :

- Microcontroller (ATmega328P) : The core of the Arduino Uno is the ATmega328P microcontroller, which is an 8-bit AVR microcontroller by Atmel (now part of

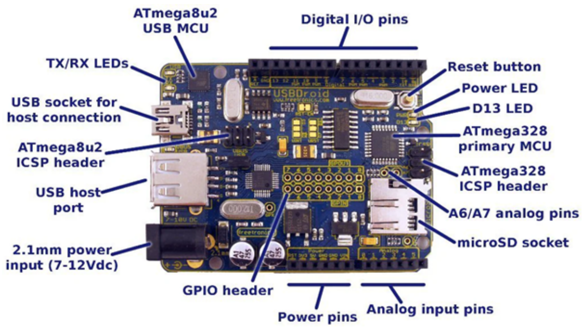


Figure 3.5: ARDUINO UNO

Microchip Technology). It runs the main code that controls the project's behavior, handling logic, decision-making, and task execution.

- Memory:

  Flash Memory: 32 KB, used for storing the program code.
  SRAM: 2 KB, used for runtime data storage.
  EEPROM: 1 KB, used for storing non-volatile data.

- Input/Output (I/O) Pins: Manages input from sensors and controls outputs to actuators, enabling interaction with the physical world.

  Digital I/O Pins: 14 digital pins (0-13), of which 6 can be used as PWM (Pulse Width Modulation) outputs.
  Analog Input Pins: 6 analog pins (A0-A5), which can read analog signals and convert them to digital values using a 10-bit ADC (Analog-to-Digital Converter).

- Communication: Data Handling: Facilitates communication between different system components, ensuring that data is processed and relayed as needed.
  Serial Communication between ESP32 and Arduino (Arduino as Receiver)

- Power: Power Supply: Can be powered via USB connection (5V) or an external power supply (7-12V recommended).

In our project, the Arduino Uno functions as the primary microcontroller, executing and managing the central portion of our code. It integrates various system components, processes data, and coordinates tasks to ensure smooth operation.

3.3.2 ESP32

The ESP32 is a highly capable microcontroller with a range of features that make it suitable for various applications,as referred in fig3.6 including IoT devices, wireless

communication, and sensor integration. Its powerful processing capabilities, extensive connectivity options, and low power consumption make it a popular choice for modern embedded systems.
Key Features and Specifications :

- Microcontroller and Processing: Dual-core or Single-core Tensilica LX6 Processor: The ESP32 can have either a dual-core or a single-core processor, which operates at up to 240 MHz. The dual-core configuration allows for parallel task execution, enhancing performance for complex applications. The high clock speed ensures the microcontroller can handle intensive computational tasks efficiently.

- Memory:

  Flash Memory (4 MB) : Flash memory is used for storing the firmware and application code. 4 MB is sufficient for most applications, allowing for the storage of large programs and data.

  SRAM (520 KB) : SRAM provides temporary storage for variables and data during program execution. 520 KB is ample for running complex programs with multiple variables and data structures.

- Connectivity:

  Wi-Fi: The ESP32 supports IEEE 802.11 b/g/n standards and operates on the 2.4 GHz band. It can function as a Wi-Fi station, access point, or both simultaneously. This flexibility makes it ideal for IoT applications requiring wireless connectivity.

  Bluetooth: Supporting both Classic Bluetooth and BLE (Bluetooth Low Energy), the ESP32 can communicate with a wide range of Bluetooth devices, enabling applications like wireless sensors, health monitors, and smart home devices.

- Input/Output (I/O):

  Digital I/O Pins (34 GPIO Pins) : General Purpose Input/Output (GPIO) pins can be programmed to perform various functions, such as reading sensor data or controlling actuators. Having 34 GPIO pins provides ample options for interfacing with other components.

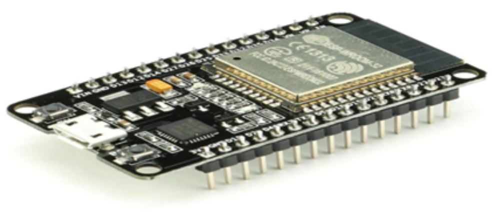


Figure 3.6: ESP 32

Analog Inputs (18 Channels , 12-bit ADC): The analog-to-digital converter (ADC) allows the ESP32 to read analog signals from sensors. With 18 channels and 12-bit resolution, it can measure a wide range of analog inputs with good precision.

- Communication Interfaces (I2C, SPI, UART, CAN) : These communication protocols enable the ESP32 to interface with various peripheral devices. I2C and SPI are commonly used for sensors and displays, UART for serial communication, and CAN for automotive and industrial applications.

In our project, the ESP32 plays a crucial role in sensor fusion by reading data from an ultrasonic sensor and a GPS module. This data is then transmitted to Arduino Uno. The communication between the ESP32 and the Arduino Uno is established using serial communication, with the ESP32 acting as the transmitter (TX) and the Arduino Uno as the receiver (RX).

Chapter 4

Methodology and Implementation - Software

The microcontroller code serves as the foundation for the rover's autonomous capabilities, managing overall control and functionality. Navigation is a key aspect of this code, involving path planning, obstacle avoidance, and waypoint navigation. By integrating sensor data, executing control algorithms, and facilitating communication between components, the software ensures that the rover can operate independently and efficiently.

4.1 Way-point Navigation

Waypoint navigation is a fundamental technique in autonomous vehicle systems, allowing the vehicle to follow a predefined path by traveling through a series of specified locations, known as waypoints. This method involves several key components and processes to ensure accurate and efficient navigation.

- Waypoints Definition: Waypoints are specific geographic coordinates or locations that the vehicle is programmed to reach sequentially. Each waypoint represents a target position that guides the vehicle along its intended route.

- Path Planning and Navigation Algorithms : Before the vehicle begins its journey, a path planning algorithm determines the optimal route from the starting point to the destination. This route is divided into segments, each ending at a waypoint.

- Positioning Data: The vehicle continuously updates its position using sensors such as GPS modules. By comparing its current location with the coordinates of the next waypoint, the system determines how to adjust its movement.

- Obstacle Avoidance: During waypoint navigation, the vehicle must also detect and avoid obstacles that may obstruct its path. Ultrasonic sensors, LiDAR, or other obstacle detection systems provide data to adjust the vehicle's route and prevent collisions.

- Completion and Transition: Once the vehicle reaches a waypoint, the navigation system verifies the arrival and prepares for the transition to the next waypoint. The process repeats until the final destination is reached.

The implementation of waypoint navigation in the Arduino Uno code involves defining waypoints, tracking position, calculating direction, and generating movement commands. The code manages real-time adjustments, error handling, and obstacle avoidance to ensure that the rover follows its planned route accurately.

4.2 Embedded C programming

Embedded C programming is a specialized extension of the C programming language tailored for developing software for embedded systems. These systems are typically characterized by constraints such as limited memory, processing power, and energy consumption, and they are designed for specific tasks or functions. This guide delves into the essentials of Embedded C programming, its significance, typical use cases, and key concepts that are foundational to embedded systems development.

- Understanding Embedded Systems :

  Before diving into Embedded C, it is essential to understand what embedded systems are. An embedded system is a computer system that is dedicated to performing one or a few specific functions, often within a larger system. It typically consists of:

- Microcontroller or Microprocessor:The brain of the embedded system, responsible for executing instructions.

- Memory:Includes both ROM (Read-Only Memory) for storing firmware and RAM (Random Access Memory) for runtime data.

- Input/Output Interfaces:allow the embedded system to interact with the external world, including sensors, actuators, communication ports, and user interfaces.

  Embedded systems are found in a wide range of applications, from household appliances like microwaves and washing machines to complex systems like automotive controls and medical devices.

4.2.1 The Role of C in Embedded Systems

C is the dominant language used in embedded systems development for several reasons:

- Efficiency:C allows direct manipulation of hardware resources through pointers and bitwise operations, which is crucial for systems with limited resources.
- Portability :C code can be written to run on different hardware platforms with minimal modifications, which is beneficial in embedded systems that often need to support multiple hardware configurations.
- Control :provides low-level access to memory and hardware, allowing precise control over timing and resource usage, which is essential in real-time applications.
- Widely Supported :C compilers and tools are available for almost every microcontroller and processor architecture.
- . Key Concepts in Embedded C Programming :
- Memory Management :

  Embedded systems often have limited memory, making efficient memory management crucial. There are several types of memory in an embedded system:

  ROM (Read-Only Memory) :Stores the firmware, which is the software permanently programmed into the system. RAM (Random Access Memory) :Used for temporary data storage during program execution. EEPROM/Flash Memory :Non-volatile memory used for storing configuration data or firmware updates.

  In Embedded C, memory management involves careful allocation and deallocation of memory, often using static allocation instead of dynamic allocation (e.g., ‘malloc()‘ and ‘free()‘) due to the unpredictable nature of memory requirements in dynamic allocation, which can lead to fragmentation and potential system crashes.

4.2.2 I/O Programming

Input/output programming in embedded systems involves direct interaction with hardware components. This is typically done through registers, which are special memory locations within the microcontroller used to control hardware peripherals.

GPIO (General Purpose Input/Output) :Used to interact with external components like LEDs, buttons, and sensors. Timers and Counters :Used for timing operations, generating delays, and event counting. Serial Communication :Involves protocols like UART, SPI, and I2C for communication with other devices or microcontrollers.

Embedded C provides constructs for manipulating these registers directly, usually through pointer operations, allowing precise control over hardware.

- Interrupt Handling :

  Interrupts are signals that indicate an event requiring immediate attention, such as a timer overflow or an external signal from a sensor. In Embedded C, interrupt handling is a critical concept, enabling the system to respond promptly to real-time events.

- (Interrupt Service Routine) :A special function in Embedded C that is executed when an interrupt occurs. ISRs must be concise and efficient to minimize latency. Interrupt Priorities :In systems with multiple interrupts, priority levels are assigned to ensure that the most critical tasks are addressed first.

  Proper management of interrupts is essential to ensure that the system remains responsive and that high-priority tasks are handled appropriately.

- Real-Time Operating Systems (RTOS)

  In more complex embedded systems, an RTOS is often used to manage multiple tasks and ensure that real-time constraints are met. Embedded C can be used to write code that runs on an RTOS, which typically involves:

  Task Scheduling :CPU time to various tasks based on their priority and timing requirements. Inter-Task Communication :like semaphores, queues, and message buffers are used to coordinate between tasks. Memory Management :Ensuring that tasks have sufficient resources without conflicts or deadlocks.

  An RTOS helps in structuring the code into distinct tasks, making the system more manageable, scalable, and responsive.

4.2.3 Embedded C Development Process

Developing software for embedded systems involves several stages, from writing and testing code to deploying it on the target hardware. The following outlines the typical development process:

- Writing Code :

  The code is usually written in a text editor or an Integrated Development Environment (IDE) tailored for embedded systems, such as Keil, MPLAB, or Atmel Studio. The code is structured into modules, typically following a layered architecture to separate hardware-dependent code from the application logic.

- Compilation and Linking :

  Once the code is written, it is compiled using a cross-compiler, which generates machine code that can run on the target microcontroller. The linking process combines various object files and libraries into a single executable or firmware image.

- Debugging :

  Debugging embedded systems can be challenging due to the limited resources and the real-time nature of the application. Developers use tools like in-circuit debuggers, emulators, and logic analyzers to test and debug the code. In many cases, debugging involves stepping through the code, monitoring variable values, and setting breakpoints to isolate issues.

- Testing :

  Testing is done both in simulation (using software tools that emulate the hardware) and on actual hardware to ensure that the code behaves as expected. This stage is crucial for validating that the system meets its performance and reliability requirements.

- Deployment :

  Once the code is tested and validated, it is deployed to the target hardware, often using a programmer or through over-the-air updates in the case of wireless devices. Deployment also involves configuring the system's peripherals, setting up initial parameters, and ensuring the

4.3 The Arduino IDE

The Arduino IDE (Integrated Development Environment) is a robust and user-friendly platform designed for developing and uploading code to Arduino-compatible microcontrollers. It offers a simplified coding environment with a clear and intuitive interface, making it accessible to both beginners and advanced users. The IDE supports C/C++ programming languages and provides a comprehensive
Key Components of Arduino IDE

- Sketch: A sketch is the name given to a program written using the Arduino IDE. Sketches are typically written in C or C++ with some additional syntax and functions specific to Arduino. Each sketch consists of two main functions: setup() and loop(). The setup() function runs once when the program starts, and the loop() function runs continuously after the setup has been executed.

- Libraries: Libraries are collections of code that make it easy to connect to sensors, displays, motors, and other hardware components. The Arduino IDE comes with several built-in libraries, and additional libraries can be downloaded and included to extend the functionality of sketches.
- Serial Monitor: The Serial Monitor is a tool within the IDE that allows for communication between the Arduino board and the computer. It is used to send and receive serial data, which is helpful for debugging and monitoring the status of the Arduino board.
- Board and Port Selection:The IDE allows users to select the specific type of Arduino board they are using and the communication port to which it is connected. This ensures that the compiled code is compatible with the hardware and can be successfully uploaded.

4.4 Libraries we have used

- AFMotor: Used for controlling motors with the Adafruit Motor Shield.
- SoftwareSerial: Allows serial communication on other digital pins.
- Arduino: Core library that provides the basic functionality for Arduino sketches
- ArduinoQueue: Allows the use of queue data structures in Arduino sketches.
- Math: Provides mathematical functions that go beyond the basic arithmetic functions included in the core Arduino library.

4.5 Functions used in our Arduino Code

The Arduino-based code is designed for a robot (or rover) that navigates through a set of waypoints using an ESP32 module for communication, sensors for distance measurement, and DC motors for movement. The code employs several functions to manage navigation, obstacle avoidance, and communication between the ESP32 and the Arduino. Below is an explanation of the key functions used in this code.

- setup() :
  The ‘setup()‘ function is executed once when the Arduino starts. It initializes the serial communication between the Arduino and ESP32, as well as the Arduino’s serial monitor. It also calls ‘findPathFromStartToDestination()‘ to determine the path from the starting waypoint to the destination.

- loop() :
  The ‘loop()‘ function is the main execution loop of the Arduino program, running repeatedly. It checks if data from the ESP32 has been read, then follows these steps:
  - Checks if the destination waypoint has been reached using ‘isDestinationNodeReached()‘.
  - If outside any waypoint, it navigates to the nearest waypoint using ‘navigateToNearestWayPoint()‘.
  - Determines the next waypoint to move towards using ‘getNextWaypoint()‘ and navigates to it.
  - Continuously reads data from the ESP32 if it hasn’t been read before.
- isDestinationNodeReached() :
  This function checks if the robot has reached the destination waypoint by comparing the current latitude and longitude with the destination’s coordinates. It returns ‘true‘ if the destination is reached, otherwise ‘false‘.
- isOutsideAWayPoint() :
  This function checks if the robot is outside any defined waypoint by comparing its current coordinates with the waypoints’ coordinates. If the robot’s current position doesn’t match any waypoint, it returns ‘true‘, indicating that the robot is outside a waypoint.
- findIndexOfNearestWayPoint() :
  This function calculates the distance between the robot’s current position and each waypoint, using the Haversine formula (‘calculateDistanceInCm()‘), and returns the index of the nearest waypoint.
- navigateToNearestWayPoint() :
  This function calls ‘findIndexOfNearestWayPoint()‘ to determine the nearest waypoint, then sets this as the starting waypoint and navigates the robot towards it using ‘navigateToAWayPoint()‘.
- findPathFromStartToDestination() :
  This function uses Breadth-First Search (BFS) to find the shortest path from the starting waypoint to the destination waypoint, considering the connectivity between waypoints. It populates the ‘correctPath‘ queue with the sequence of waypoints the robot should follow.
- getNextWaypoint() :
  This function dequeues the next waypoint from the ‘correctPath‘ queue and

returns it. If the queue is empty, it returns ‘-1‘, indicating no further waypoints are available.

- readData()‘ and ‘readData(long timeOut) :
  These functions manage the reading of data from the ESP32. The ‘readData()‘ function attempts to read data within a specified timeout period. If data is received, it calls ‘readAMessageFromESP32()‘ and ‘extractInformationFromMessage()‘ to process the data.
- readAMessageFromESP32() :
  This function reads a message from the ESP32 using the ‘SoftwareSerial‘ interface. It reads the message into the ‘dataFromESP32‘ buffer and ensures the message is null-terminated.
- extractInformationFromMessage(char data[]) :
  This function processes the data received from the ESP32 by parsing the message to extract distance to an obstacle, latitude, longitude, and heading. It uses string manipulation to extract and convert these values from the received data.
- navigateToAWayPoint(int index) :
  This is a critical function that handles the navigation of the robot to a specific waypoint. It calculates the bearing to the waypoint using ‘calculateBearing(int index)‘, rotates the robot to face the waypoint if necessary, and then moves it forward until the waypoint is reached, avoiding obstacles if any are detected.
- moveForwardUntilWayPointIsReached(int index) :
  This function controls the forward movement of the robot towards a waypoint. It calculates the distance to the waypoint and moves the robot in increments, checking for obstacles along the way. If an obstacle is detected, it calls ‘avoidObstucle()‘ to navigate around it.
- avoidObstucle() :
  This function handles obstacle avoidance. It performs a sequence of rotations and movements to steer the robot around an obstacle and then realigns the robot to its original heading.
- calculateDurationToMoveForwardInMs(double distance) :
  This function calculates the time required for the robot to move forward by a specified distance, based on the wheel diameter and the rotation speed (RPM) of the motors.

- degreesToRadians(double degrees) :
  A helper function that converts an angle from degrees to radians, which is useful for trigonometric calculations.

- calculateDistanceInCm(float iLat1, float iLon1, float iLat2, float iLon2) :
  This function calculates the distance between two geographic coordinates (latitude and longitude) using the Haversine formula. It returns the distance in centimeters.

- rotateUntilWeFaceDestinationWaypoint(int bearing) :
  This function rotates the robot until its heading aligns with the bearing of the destination waypoint. It calculates the duration for which the motors should be engaged to achieve the required rotation.

- calculateBearing(int index) :
  This function calculates the bearing from the robot's current position to the specified waypoint, using trigonometric functions. The bearing is normalized to be within 0-360 degrees.

- calculateDurationToRotate(int bearing) :
  This function calculates the time required to rotate the robot by a specified angle, based on the robot's rotational speed.

- rotateRoverClockWise(double duration) :
  This function rotates the robot clockwise for a specified duration by controlling the direction and speed of the motors.

- moveRoverForward(double duration) :
  This function moves the robot forward for a specified duration by setting all motors to move in the forward direction.

- stopMotors() :
  This function stops all the motors, halting the robot's movement. It sets the motor speeds to zero and releases the motor control signals.

Chapter 5

Results and Outcomes

The major outcome of this project is the realization of a functional autonomous rover. This rover is designed to autonomously navigate through a predefined set of waypoints while dynamically avoiding obstacles. The integration of various hardware components and sophisticated software algorithms ensures the rover's capability to operate independently in diverse environments.

5.1 Technical Outcomes

- Navigation : The rover employs a NEO-7M GPS module to determine its current location with high accuracy.Using GPS data, the rover autonomously navigates to predefined waypoints, adjusting its path as necessary to reach each target destination.The navigation algorithm, embedded within the microcontroller code, processes real-time location data to make precise movement decisions.

- Obstacle Avoidance: The HC-SR04 ultrasonic sensor continuously measures the distance to potential obstacles in the rover's path. The ESP32 microcontroller receives distance measurements and transmits this data to the Arduino Uno, which processes the information to dynamically alter the rover's course. The obstacle avoidance system ensures the rover can navigate safely by detecting and circumventing obstacles, preventing collisions and ensuring smooth operation.

- Hardware Integration: The L293D motor driver effectively controls the rover's four DC motors, providing precise locomotion capabilities. The GPS module and ultrasonic sensor are seamlessly integrated with the ESP32 and Arduino Uno microcontrollers, ensuring reliable data transmission and processing.

- System Performance: The rover's ability to autonomously navigate to waypoints and avoid obstacles demonstrates the successful integration of hardware and software components. The system's robustness and reliability are validated through extensive testing in various environments, confirming the rover's capability to operate independently and efficiently.

Image 4.1 below shows the functional rover we have built.Image 4.2 below shows the serial monitor output.

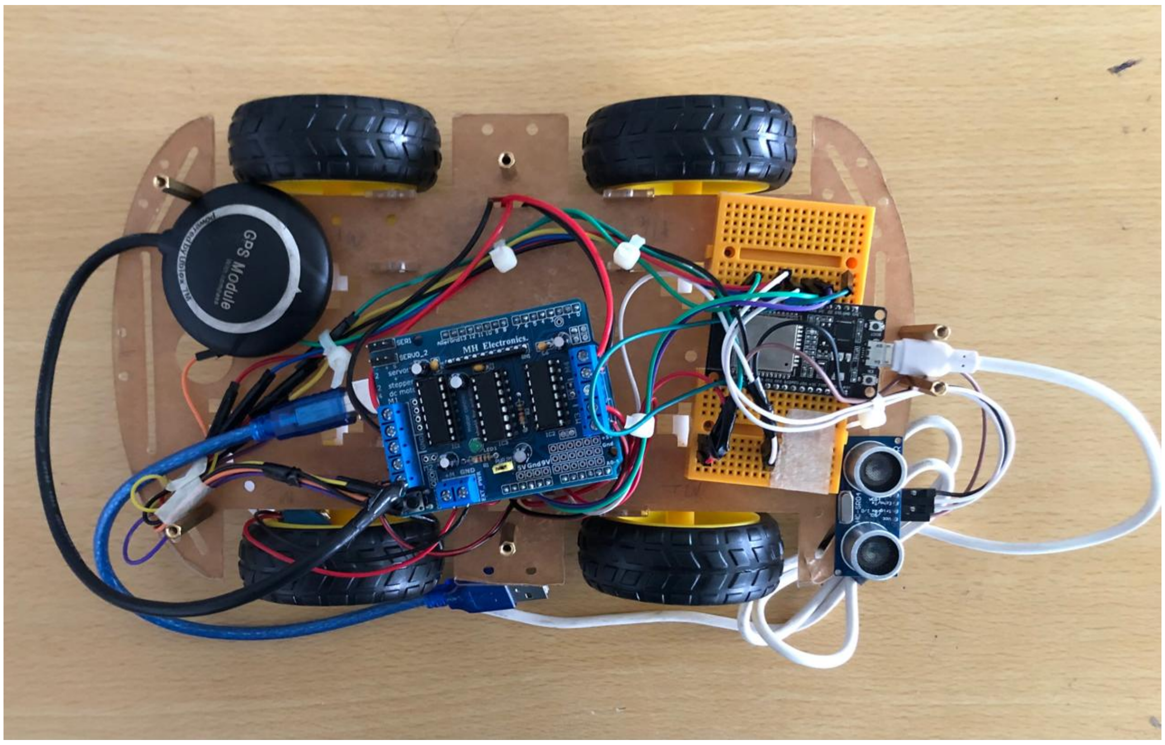
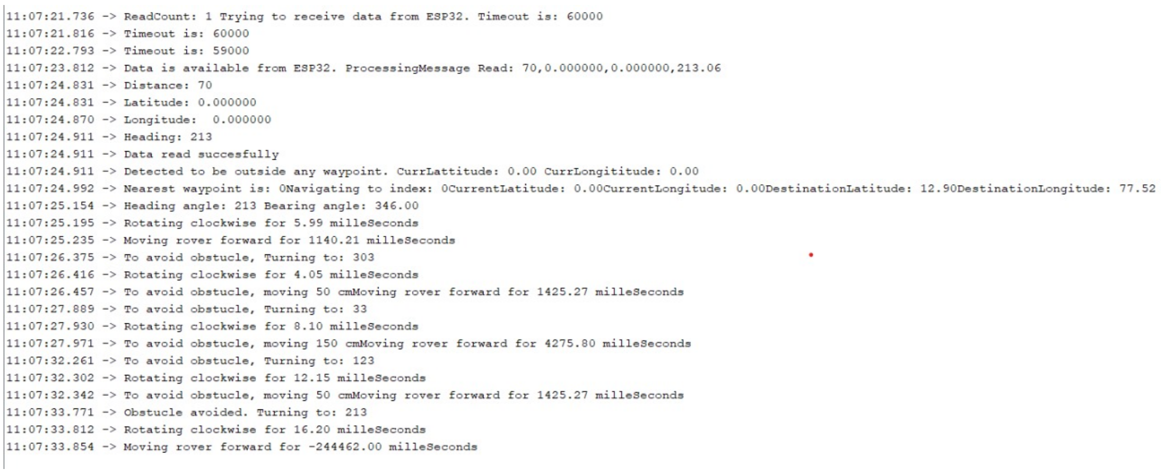


Figure 5.1: AUTONOMOUS ROVER



```
11:07:21.736 -> ReadCount: 1 Trying to receive data from ESP32. Timeout is: 60000
11:07:21.816 -> Timeout is: 60000
11:07:22.793 -> Timeout is: 59000
11:07:23.812 -> Data is available from ESP32. ProcessingMessage Read: 70,0.000000,0.000000,213.06
11:07:24.831 -> Distance: 70
11:07:24.831 -> Latitude: 0.000000
11:07:24.870 -> Longitude:  0.000000
11:07:24.911 -> Heading: 213
11:07:24.911 -> Data read succesfully
11:07:24.911 -> Detected to be outside any waypoint. CurrLattitude: 0.00 CurrLongititude: 0.00
11:07:24.992 -> Nearest waypoint is: 0Navigating to index: 0CurrentLatitude: 0.00CurrentLongitude: 0.00DestinationLatitude: 12.90DestinationLongitude: 77.52
11:07:25.154 -> Heading angle: 213 Bearing angle: 346.00
11:07:25.195 -> Rotating clockwise for 5.99 milleSeconds
11:07:25.235 -> Moving rover forward for 1140.21 milleSeconds
11:07:26.375 -> To avoid obstucle, Turning to: 303
11:07:26.416 -> Rotating clockwise for 4.05 milleSeconds
11:07:26.457 -> To avoid obstucle, moving 50 cmMoving rover forward for 1425.27 milleSeconds
11:07:27.889 -> To avoid obstucle, Turning to: 33
11:07:27.930 -> Rotating clockwise for 8.10 milleSeconds
11:07:27.971 -> To avoid obstucle, moving 150 cmMoving rover forward for 4275.80 milleSeconds
11:07:32.261 -> To avoid obstucle, Turning to: 123
11:07:32.302 -> Rotating clockwise for 12.15 milleSeconds
11:07:32.342 -> To avoid obstucle, moving 50 cmMoving rover forward for 1425.27 milleSeconds
11:07:33.771 -> Obstucle avoided. Turning to: 213
11:07:33.812 -> Rotating clockwise for 16.20 milleSeconds
11:07:33.854 -> Moving rover forward for -244462.00 milleSeconds
```

Figure 5.2: SERIAL MONITOR OUTPUT

5.2 Non-Technical Outcomes

- Enhanced Team Collaboration : The project required coordinated efforts from various team members with different skill sets, such as coding, hardware assembly, and testing. This collaboration improved team dynamics and communication skills.

- Problem-Solving Skills: Encountering and overcoming various challenges during the development process enhanced problem-solving abilities. Team members learned to think critically and devise creative solutions to technical problems.

- Knowledge Sharing and Learning: The project served as a platform for learning and sharing knowledge among team members. Those with expertise in certain areas, such as programming or electronics, had the opportunity to mentor others, fostering a collaborative learning environment. Increased Motivation and Confidence:

  Successfully completing a complex project boosted the confidence of team members in their technical and non-technical abilities. This success serves as motivation for tackling future projects and challenges.

- Improved Communication Skills: Explaining technical concepts and project progress to non-technical stakeholders, such as supervisors or potential investors, helped improve communication skills. This is crucial for effective teamwork and stakeholder engagement.

- Inspiration and Outreach: The project has the potential to inspire others, especially students and young enthusiasts, to explore robotics and engineering. Sharing the project's journey through presentations or publications can contribute to educational outreach and community engagement.

References

[1] Shambhavi Lalsinge1, Yash Mali2, Akshit Mahale3, Mrunmayee Kulkarni4, Omkar Kurade5, Professor. Pramod Patil , Autonomous Car using Arduino.

[2] Zhao, J.; Liang, B.; Chen, Q. The key technology toward the self-driving car. Int. J. Intell. Unmanned Syst. 2018, 6, 2–20.

[3] K.Karur, N.Sharma, C.Dharmatti, JESiegel. A Survey of Path Planning Algorithms for Mobile Robots. Vehicles.2021 Sep;3(3):448–68

[4] AMegha M1, Namratha shetty T G2, Pavan Kumar E3 WORKING OF AUTONOMOUS VEHICLES International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056 Volume: 06 Issue: 12 — Dec 2019