# GESTURE CONTROLLED ROBOT

B.ANUSHA (22261A0409)
HEEBA FIRDOUSE (22261A0424)
N.INDUPRIYA (22261A0441)


Department of Electronics and Communication Engineering

**MAHATMA GANDHI INSTITUTE OF TECHNOLOGY (AUTONOMOUS)**

Chaitanya Bharathi P.O., Gandipet, Hyderabad – 500 075

# GESTURE CONTROLLED ROBOT

MINI PROJECT REPORT

SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENTS FOR THE DEGREE OF

BACHELOR OF TECHNOLOGY

IN

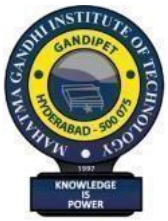
ELECTRONICS AND COMMUNICATION ENGINEERING

BY


B.ANUSHA (22261A0409)

HEEBA FIRDOUSE (22261A0424)

N.INDUPRIYA (22261A0441)



Department of Electronics and Communication Engineering

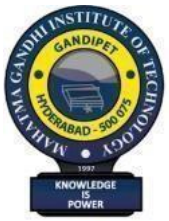
**MAHATMA GANDHI INSTITUTE OF TECHNOLOGY (AUTONOMOUS)**

Chaitanya Bharathi P.O., Gandipet, Hyderabad – 500 075


2025

# MAHATMA GANDHI INSTITUTE OF TECHNOLOGY
# (AUTONOMOUS)

Chaitanya Bharathi P.O., Gandipet, Hyderabad-500 075




## Department of Electronics and Communication Engineering

## CERTIFICATE

Date: 09 May 2025

This is to certify that the Mini project work entitled **"GESTURE CONTROLLED ROBOT"** is a bonafide work carried out by

| | |
|---|---|
| **B.ANUSHA** | **(22261A0409)** |
| **HEEBA FIRDOUSE** | **(22261A0424)** |
| **N.INDUPRIYA** | **(22261A0441)** |

in partial fulfillment of the requirements for the degree of **BACHELOR OF TECHNOLOGY** in **ELECTRONICS & COMMUNICATION ENGINEERING** by the Jawaharlal Nehru Technological University, Hyderabad during the academic year 2025.

(Signature)

---

**Dr. Madhavi**
**Associate Professor**

(Signature)

---

**Dr. S. P. Singh**
**Professor & Head**

# ACKNOWLEDGEMENT

We express our deep sense of gratitude to our Faculty Advisor, Dr. MADHAVI, Associate Professor, Department of Electronics and Communication Engineering, MGIT, Hyderabad, for his guidance, support and encouragement which have been invaluable throughout this project. His insights and advice have greatly contributed to its successful completion.

We wish to express our sincere thanks to Dr. S.P.Singh, Head of the Department of Electronics and Communication Engineering, M.G.I.T., for permitting us to pursue our Project, and encouraging us throughout the Project.

Finally, we thank all the people who have directly or indirectly helped us through the course of our Project.

B.ANUSHA
HEEBA FIRDOUSE
N.INDUPRIYA

# ABSTRACT

This project presents the design and implementation of a **Gesture Controlled Robot** operated through a Bluetooth-enabled RC controller app and powered by an **Arduino Uno** microcontroller. The robot is developed to respond to directional commands forward, backward, left, and right sent from a smartphone via Bluetooth communication. The system uses a **Bluetooth RC controller app** in the user's hand to detect hand gestures and convert them into corresponding motion commands. These gestures are interpreted and transmitted wirelessly to the robot using a **Bluetooth module (HC05)**.

The Arduino Uno processes the received commands and drives the motors accordingly through a **motor driver module (L298N)** connected to the robot's wheels. The chassis consists of four wheels ensuring smooth movement in multiple directions. This project highlights the integration of embedded systems, wireless communication, and sensor technology to create an intuitive and interactive robotic control system. The robot serves as a prototype for hands-free control in applications such as surveillance, remote assistance, or human-computer interaction. With its low cost, portability, and ease of use, the gesture-controlled robot stands as an excellent learning platform for students and hobbyists in robotics and automation.

Table of contents

# LIST OF FIGURES



(

# CHAPTER 1: OVERVIEW

## 1.1 INTRODUCTION

Robotics and automation have become an integral part of modern technology, with applications ranging from industrial automation to personal assistance and smart surveillance systems. As the demand for more intuitive and human-friendly interfaces increases, gesture-based control systems have gained significant attention. The use of hand gestures offers a natural and effective way of interacting with machines without the need for traditional input devices such as keyboards, joysticks, or remotes. This project focuses on the development of a **Gesture Controlled Robot** that is operated via a **Bluetooth-enabled RC controller app** and managed using an **Arduino Uno** microcontroller.

Gesture-controlled systems are highly advantageous in scenarios where physical contact with the device is not feasible or practical, such as hazardous environments, defense operations, or in aiding people with disabilities. The primary objective of this project is to design a simple and cost-effective robot that can interpret user hand gestures and respond with accurate movement, using common components such as an accelerometer, a Bluetooth module (HC-05), an Arduino Uno board, and a motor driver (L298N). The system is intended to be compact, affordable, and accessible for students and hobbyists interested in embedded systems and robotics.

The basic working principle of this robot is that a user uses mobile phone which has **Bluetooth controller app,** which detects the movements as phone has inbuilt **Gyroscope**. This sensor captures the motion and orientation of the hand and sends corresponding data to the Arduino. The Arduino processes this data and sends control signals via Bluetooth to another Arduino mounted on the robot chassis. Upon receiving the data through the HC-05 Bluetooth module, the second Arduino interprets the instructions and activates the motor driver to move the robot in the desired direction.

In addition to gesture control, the robot can also be operated using a **Bluetooth RC controller app**, which offers an alternative method of manual control for voice command.

This dual-mode system enhances the usability of the robot and allows users to switch between gesture-based control and traditional smartphone control depending on the situation.

This project serves not only as a demonstration of wireless gesture control but also as a hands-on introduction to fundamental concepts in electronics, sensor integration, wireless communication, and robotic movement. It provides a foundation upon which more advanced autonomous and semi-autonomous systems can be developed. The Gesture Controlled Robot is an excellent learning tool and proof of concept for human machine interaction, making it ideal for academic, research, and hobbyist applications.

## 1.2 AIM OF THE PROJECT

- To design and build a robot that can be controlled using hand gestures detected by an accelerometer.
- To implement wireless communication between the gesture sensor and the robot using a Bluetooth module (HC-05).
- To program the Arduino Uno for interpreting gesture data and controlling motor movements accordingly.
- To provide an alternative control option through a Bluetooth RC controller mobile app.
- To integrate basic electronics components like sensors, motor driver (L298N), and DC motors for robot movement.
- To create a low-cost, beginner-friendly platform for learning robotics and embedded systems.
- To explore real-time applications of gesture control in robotics, such as surveillance and remote control.
- To enhance user-machine interaction by using intuitive, touch-free control methods.

## 1.3 METHODOLOGY

This project involves the development of a **Gesture Controlled Robot** that uses **Arduino Uno**,

**HC-05 wireless modules**, and an **L298N motor driver** to enable wireless, gesture-based control. The methodology is structured into four key parts: hardware and software components, system development, system testing, and maintenance. [1]

**Hardware Components**

- **Arduino Uno (x2):** One for the transmitter (gesture control unit) and one for the receiver (robot).
- **L298N Motor Driver Module:** Controls the robot's motors based on commands from the Arduino.
- **DC Motors (x4):** Provide movement to the 4-wheel robot chassis.
- **Robot Chassis:** Mechanical base that holds the motors, wheels, and circuitry.
- **Battery Pack:** Supplies power to both the transmitter and the robot.

**Software Components**

- **Arduino IDE:** Used to write and upload the code for both transmitter and receiver units.

**System Development**

- **Transmitter Part:** The gesture controller is connected to an Arduino Uno. When gestures or button inputs are made, the Arduino sends commands via bluetooth module. In this project we are using the mobile phone as transmitter.
- **Receiver Circuit Design:** The robot's Arduino receives these commands using the HC-05 module and controls motor actions via the L298N motor driver.
- **Mechanical Assembly:** The 4-wheel chassis is assembled, and motors, battery, and L298N driver are securely mounted.
- **Programming:** Code is written to interpret transmitter inputs (mobile phone) and control the motors accordingly on the receiver side.

**System Testing**

- **Module Testing:** Each component (L298N, motors, HC-05) is tested individually.
- **Communication Testing:** Ensuring reliable wireless communication between transmitter and receiver within a defined range.

- **Functional Testing:** Validating the robot's response to all gesture/button inputs under different conditions.
- **Stress Testing:** Testing for packet loss, signal interference, and battery performance over time.

**System Maintenance**

- Sensor Calibration – Regular calibration of sensors to ensure accurate measurements.
- Firmware Updates – Periodic updates to improve performance and add features.
- App Modifications – Updating Blynk interface as needed for improved user experience.
- Hardware Checks – Inspecting and replacing worn-out components like jumper wires or sensors to maintain system reliability.

**FLOW CHART**

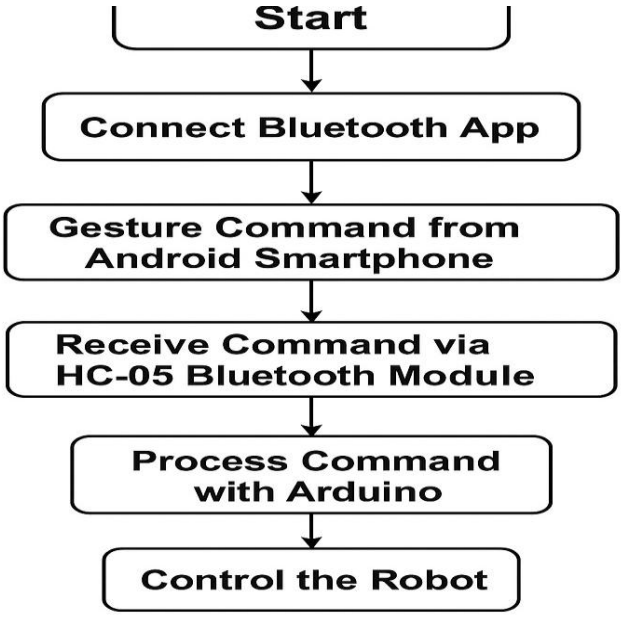



Fig:1.1 *flowchart of Gesture-Controlled Robot*

## 1.4 SIGNIFICANCE AND APPLICATIONS

**Significance**

- **Hands-free Control**: Enables wireless robot operation using hand gestures with mobile phone, which is intuitive and reduces dependency on physical remotes.
- **Wireless Communication**: The use of **HC-05 module** ensures short-range, reliable 2.4GHz communication between transmitter and receiver units.
- **Cost-Effective Design**: Utilizes affordable components like Arduino Uno and HC-05, making it ideal for students and hobbyists.
- **Modular and Expandable**: Easily upgradeable to include more functions such as sensors for obstacle avoidance or automation.
- **Learning Opportunity**: Enhances understanding of embedded systems, wireless communication, and motor control.
- **Real-Time Response**: Offers almost instant response to gestures or movement, suitable for real-time control applications.
- **Safe Remote Access**: Useful in hazardous environments where human entry is unsafe, like disaster zones or chemical plants.

**Applications**

1. **Military and Surveillance**: For scouting or inspecting dangerous areas without risking human life.
2. **Search and Rescue Operations**: Can be used to navigate disaster-struck regions to locate survivors.
3. **Warehouse Automation**: Controlling robots to carry loads within an industrial space.
4. **Medical Assistance**: Assisting elderly or disabled individuals in moving objects without physical strain.
5. **Agriculture**: Controlling small inspection or spraying robots remotely across fields.
6. **Education and Training**: Excellent tool for teaching robotics, IoT, and embedded systems in academic institutions.
7. **Home Automation**: Controlling basic domestic robots for cleaning or monitoring.
8. **Entertainment and DIY Projects**: Building gesture-controlled toy cars or home robots as a fun hobby.

## 1.5 ORGANIZATION OF WORK

**Phase 1: Research and Requirement Analysis**

- Studied existing gesture-controlled and wireless robot systems.
- Identified the scope, limitations, and feasibility of using Arduino Uno, HC-05, and L298N motor driver.
- Finalized the list of required hardware and software tools.
- Defined clear project goals and constraints.

**Phase 2: Component Procurement and Circuit Design**

- Procured all electronic components: Arduino Uno boards, HC-05 module, L298N motor driver, DC motors, chassis, batteries, etc.
- Designed the receiver circuit on paper and software (e.g., Fritzing/Proteus).
- Tested individual components like Arduin UNO, HC-05 and L298N.

**Phase 3: System Integration and Programming**

- Assembled the transmitter and robot hardware circuits.
- Wrote and uploaded separate Arduino codes for transmitter and receiver.
- Established bluetooth communication protocol between the two Arduinos.

**Phase 4: Testing and Debugging**

- Conducted unit testing on receiver and motor movement individually.

# CHAPTER 2: LITERATURE REVIEW

## 2.1 INTRODUCTION

With the growing interest in human-machine interaction (HMI) and gesture-based control systems, gesture controlled robotics has become an active area of research and innovation in recent years. Traditional control systems for RC cars often rely on joysticks, push buttons, or smartphone apps. However, these methods can lack the natural, intuitive interaction that gestures provide.

To overcome these limitations, recent projects have explored the use of wireless communication modules like Bluetooth HC-05 to enable gesture-based control of mobile robots and RC cars.

Several studies and DIY projects demonstrate how simple hand movements, when processed by microcontrollers such as Arduino UNO, can be used to control the direction and speed of robotic vehicles. These systems typically rely on the tilt of the hand, captured by bluetooth RC controller app in mobile phones, and then transmitted wirelessly to the vehicle, which responds in real time.

The control of remote-controlled vehicles has evolved significantly over the past few decades. Early RC cars and robots relied on simple wired connections, where physical cables linked a handheld controller to the motors. These wired systems were limited by mobility constraints and susceptibility to physical damage but offered straightforward and reliable control.

Gesture-based control systems offer several advantages:

- Intuitive and natural control
- Wireless operation
- Low cost and easy to implement
- Educational value for learning embedded systems and robotics

This chapter presents a review of existing literature, systems, and technologies related to gesture-controlled robotics and RC vehicles. The goal is to understand current advancements, identify challenges, and highlight how this project contributes to this emerging area.

## 2.2 Problem Statement

In today's world, robots are increasingly being used in various domains for remote operations, surveillance, and assistance, particularly in inaccessible or hazardous environments. A major challenge in robotics, especially for beginners and educational environments, is the ease of controlling and interacting with the robot without relying on complex hardware interfaces.

Traditional remote-controlled robots often require dedicated remotes or complex coding interfaces, limiting their usability for everyday users or educational purposes. Moreover, most robots cannot switch control methods dynamically or adapt based on the user's convenience.

To address these issues, this project introduces a **dual-controlled robot** that can be operated both via **hand gestures** (using sensors or switches) and **Bluetooth-based mobile control**. This flexible control mechanism enables the user to maneuver the robot either using intuitive hand gestures or a smartphone app when gesture control is inconvenient.

This solution is ideal for all these. They are robotics education and demonstrations , home automation experiments and remote-controlled assistance in areas unsafe for humans

By integrating gesture and Bluetooth modules, the robot ensures portability, ease of use, and adaptability to different control modes, promoting hands-on learning and innovation in embedded systems and robotics.

## 2.3 Existing Methodologies

Over time, various techniques have been developed to control RC cars and mobile robots, evolving from basic wired remotes to advanced wireless systems such as RF, Bluetooth, Wi-Fi, and IoT-based solutions. Initially, traditional wired remote-controlled systems were used, where push buttons or joysticks were connected directly to the motors through wires. These systems were simple and free from wireless interference but had significant limitations in mobility due to the physical tethering, making them impractical for real-world use.

To improve freedom of movement, wireless joystick-based control methods were introduced. These systems typically use RF modules (e.g., 433 MHz) to transmit joystick signals wirelessly to the robot. The microcontroller on the robot interprets the received signals to control motor movements. While this method improves user experience by enabling remote operation, it still relies on dedicated hardware and does not provide a very natural interface for human-robot interaction.

In recent years, mobile app-based control through Bluetooth or Wi-Fi has become increasingly popular. These systems utilize modules like HC-05 (Bluetooth) or ESP8266/ESP32 (Wi-Fi) to receive commands from a smartphone application. The app interface includes on-screen buttons or a virtual joystick for controlling the robot's movement. This approach removes the need for separate remotes and is highly customizable. However, it still requires the user to interact with the screen manually, which can be less intuitive and engaging.

The focus of this project is on gesture-controlled robots with additional support for Bluetooth-based mobile control. In the gesture control mode, the user can control the robot’s movement through hand gestures using a handheld module or switch-based mechanism. The detected gestures are translated into directional commands and sent wirelessly to the robot using a Bluetooth module like HC-05. The robot’s microcontroller then processes these commands and drives the motors through an L298N motor driver. This gesture-based method offers a more natural and interactive experience compared to traditional methods.

To enhance usability, the project also includes a mobile app control mode. This allows users to switch to a smartphone-based interface whenever gesture control is not convenient. By combining both gesture and mobile control, the robot becomes more versatile and user-friendly. It is especially suitable for students, hobbyists, and beginners looking to explore embedded systems, wireless communication, and robotics in an engaging and educational way.

## 2.4 Proposed Solution

The proposed system is a low-cost, Bluetooth-controlled robot car designed using fundamental electronics and microcontroller-based components. It aims to provide wireless and user-friendly control of a robotic vehicle using a smartphone application, making it highly suitable for students, hobbyists, and beginners in robotics. The system operates on a simple yet effective principle — sending directional commands from a phone to the robot using Bluetooth communication, which the robot interprets to perform specific movements. At the heart of the system is the **Arduino UNO**, a widely used microcontroller board that processes incoming commands and controls the movement of the robot accordingly. The **HC-05 Bluetooth module** is paired with the smartphone and acts as a wireless bridge, receiving data from the mobile app. Once the data is received, the Arduino decodes the command (such as forward, backward, left, right, or stop) and triggers the corresponding motor action using the **L298N motor driver module**. This module allows bi-directional control of two motors, making it perfect for controlling the four-wheel drive system used in the robot.

The **robot chassis**, equipped with **four DC motors and wheels**, forms the mechanical base. The L298N motor driver is connected to each pair of motors on the left and right sides of the chassis, enabling smooth directional control. The power supply for the entire system comes from **Li-ion batteries**, making the robot fully mobile and independent of wired power sources. These batteries are lightweight, rechargeable, and provide sufficient current to run both the control system and the motors efficiently.

One of the key advantages of this project is its **simplicity and portability**. Unlike systems that rely on accelerometers or complex sensor-based gesture controls, this robot simplifies the interaction by using a smartphone application. This makes it highly accessible for those without advanced electronics knowledge while still offering meaningful insights into microcontroller programming, wireless communication, and motor control. It also avoids the need for calibrating sensors or handling complex gesture recognition algorithms.

Furthermore, the robot offers a great deal of **educational value**. By building and testing the system, students can learn about how Bluetooth modules interface with microcontrollers, how motor drivers function, and how user input can be translated into physical movement. This understanding lays the groundwork for more advanced robotic applications in the future, including automation, remote surveillance, and IoT-enabled systems.

In addition to being an excellent learning tool, the robot's design is **modular and scalable**. It can be enhanced by integrating additional features such as ultrasonic sensors for obstacle avoidance, speed control using PWM (Pulse Width Modulation), or even a camera for live streaming via Wi-Fi modules like ESP32. This flexibility ensures that the project is not limited to a fixed scope and can be continuously improved as the builder’s skills grow.

The use of Bluetooth control via a mobile app also ensures **user convenience and adaptability**. The mobile interface can be designed with buttons or voice control, depending on the chosen application. This wireless mode of operation gives the user freedom to maneuver the robot from a distance without being physically close to the system, making it suitable for various real-world scenarios like search-and-rescue simulations, warehouse movement training, or just educational demonstrations.

## 2.5 Hardware Components

### 1.Arduino UNO (Receiver)

This board is chosen for its **simplicity, affordability, and wide community support**. Programming the Arduino UNO is straightforward using the **Arduino IDE**, and a large number of libraries are available for controlling motors, modules, and sensors. Its compatibility with various components and ease of prototyping make it an ideal choice for beginners and intermediate-level robotics projects. In this Bluetooth-controlled robot,

the Arduino UNO acts as the brain, making real-time decisions based on wireless inputs to control the  movement of the robot car effectively. [3]

Fig. 2.1 : *Arduino UNO*

**2. HC-05 Bluetooth Module :**

The **HC-05 Bluetooth module** plays a key role in enabling wireless communication between the user's smartphone and the robot car. It is responsible for receiving control commands sent from a mobile app and forwarding them to the Arduino UNO for further processing. This module operates on the Bluetooth 2.0 standard and communicates with the Arduino using serial communication (UART). The HC-05 is configured in slave mode, allowing it to pair with any standard smartphone Bluetooth connection. Once connected, it receives characters or strings corresponding to commands like 'F' for forward, 'B' for backward, 'L' for left, 'R' for right, and 'S' for stop. These commands are interpreted by the Arduino to control the direction of the robot. [4]

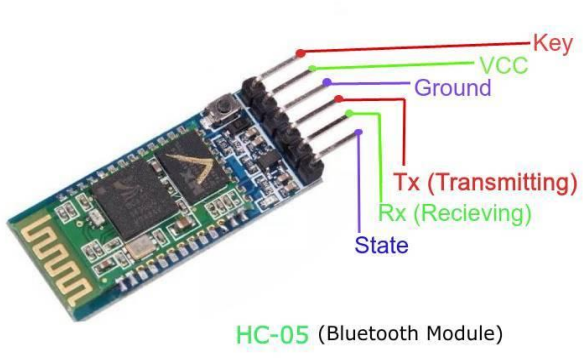



*Fig. 2.2 HC-05*

**3. L298N Motor Driver Module**

The **L298N motor driver module** is used to control the movement of the DC motors in the robot car. It acts as a bridge between the low-power control signals from the Arduino and the higher-power requirements of the motors. The module can control two motors independently and allows for forward, reverse, and braking functionality. It uses an H-bridge configuration, enabling bidirectional control of each motor. The Arduino sends digital control signals (IN1, IN2, IN3, IN4) to the L298N module, which then activates the motors in the desired direction.

In this project, the L298N controls four motors by grouping them in pairs — left and right. Each side receives control from one channel of the driver, enabling coordinated movement for turning and straight motion. The module also includes onboard terminals for power input and an optional 5V regulator, simplifying the connection process. Its ability to handle high current loads (up to 2A per channel) makes it well-suited for robotic applications using multiple DC motors. The L298N ensures precise control of the robot's motion while protecting both the Arduino and the motors from electrical stress. [5]

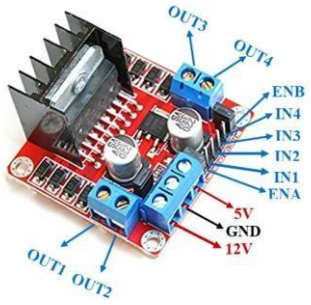



*Fig : 2.3: L298N*

**4. Li-ion Battery (Power Supply):**

The **Li-ion battery pack** serves as the primary power source for the robot, providing energy to both the control electronics (Arduino and HC-05) and the motors. These batteries are chosen for their high energy density, compact size, and reusability. In this project, multiple 3.7V Li-ion cells are connected in series or parallel (depending on required voltage and current) to provide sufficient power to the system. The power from the battery is connected to the L298N motor driver's power input, which then distributes it to the motors, while the Arduino is powered either through its VIN pin or directly from a regulated 5V line.

Li-ion batteries are ideal for portable robotics due to their ability to deliver continuous current for motor operation without significantly increasing the weight of the robot. However, they require careful handling, proper charging circuits like TP4056, and protective circuitry to avoid overcharging or deep discharge. In this robot, the use of Li-ion batteries ensures untethered operation and enhances the overall mobility of the system, allowing it to function effectively in wireless, remote-controlled scenarios.


*Fig. 2.4: Li-ion Battery*

**5. DC Motors + Wheels**

The **DC motors with wheels** are essential components that provide the actual movement of the robot car. These motors are mounted on a chassis and connected to wheels, allowing the robot to move forward, backward, and make turns based on the commands received. Controlled through the L298N motor driver and powered by a Li-ion battery pack, these motors convert electrical energy into mechanical motion, enabling the car to navigate its environment effectively.

Typically, the project uses **6V to 12V brushed DC motors**, which are compact yet capable of delivering **high torque**, making them suitable for small robotic applications. These motors are ideal for battery-powered systems due to their efficiency, low cost, and ease of control using simple H-bridge circuits. When paired with standard rubber or plastic wheels, they provide reliable traction and movement on most indoor surfaces.

*Fig: 2.5 : DC Motors + Wheels*

**6. Chassis Frame**

The **chassis** forms the structural foundation of the robot car, providing support and housing for all essential components including the Arduino UNO, L298N motor driver, DC motors, wheels, HC-05 Bluetooth module, and the Li-ion battery pack. Typically made of lightweight yet durable materials like acrylic, plastic, or aluminum, the chassis ensures the robot remains stable during movement and can handle minor impacts or vibrations without compromising performance.


*Fig. 2.6: Chassis Frame*

**7.Jumper Wires :**

In this project, jumper wires are used to connect the Arduino's digital pins to the L298N control inputs (IN1, IN2, IN3, IN4), link the HC-05 module's TX/RX pins with the Arduino's RX/TX respectively, and distribute power (5V and GND) across all modules. Their plug-and-play nature makes them ideal for quick prototyping and testing, eliminating the need for soldering and allowing easy rewiring if the setup needs changes or troubleshooting.

*Fig. 2.7: Jumper wires*

## 2.6 Software Tools

The main software tool used in this project is the **Arduino IDE**, which plays a critical role in writing, compiling, and uploading code to the Arduino UNO boards used in the system. As the central development environment, the Arduino IDE supports C/C++ programming and provides an easy-to-use interface, making it highly suitable for beginners and hobbyists. It includes essential features such as a built-in **Serial Monitor**, which is valuable for debugging and observing real-time data or command execution. This cross-platform tool is free, open-source, and works seamlessly on Windows, macOS, and Linux systems, making it widely accessible for development. [9]

To facilitate Bluetooth communication, the project uses the **SoftwareSerial library**, which is especially useful when working with the **Arduino UNO**. Since the UNO has only one hardware serial port that is usually used for USB communication with the computer, the SoftwareSerial library allows the creation of additional serial ports on other digital pins. This is necessary to establish serial communication with the **HC-05 Bluetooth module** without interfering with USB communication. With this setup, the Arduino UNO can simultaneously receive control commands via Bluetooth while maintaining serial output to the Serial Monitor for testing and debugging.

The **Serial Monitor**, a built-in tool within the Arduino IDE, is another vital part of the software toolkit. It allows the developer to observe what commands (such as 'F' for forward, 'B' for backward, 'L' for left, 'R' for right, and 'S' for stop) are being sent and received. This is especially helpful for ensuring that the Bluetooth module is properly receiving the correct data and that the Arduino is interpreting and executing the commands as expected. It significantly improves troubleshooting during development and testing phases.

Since this project does not use any sensors like MPU6050, there is no need for additional sensor libraries. The focus remains on code logic, Bluetooth communication, and motor control — all handled efficiently using basic Arduino functions, the SoftwareSerial library, and the Serial Monitor for verification.

## 2.7 Circuit Diagram:

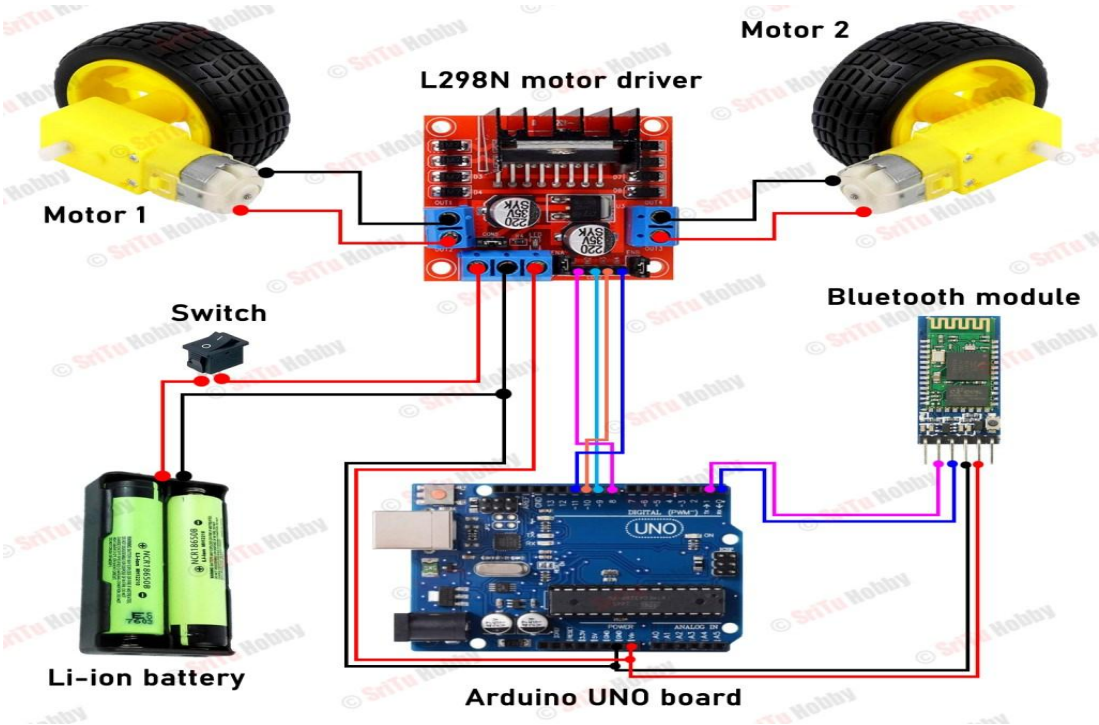



*Fig. 2.8: CIRCUIT DIAGRAM*

# CHAPTER 3: SYSTEM ARCHITCTURE AND IMPLEMENTATION

## 3.1 INTRODUCTION

The **Bluetooth-Controlled RC Car** is a wireless, Arduino-based project that allows an RC car to be operated using a mobile device via Bluetooth communication. Instead of using traditional wired remotes, joysticks, or gesture-based systems, this project uses a smartphone connected to the car through an **HC-05 Bluetooth module**, enabling directional control via button commands sent from a mobile app. These commands are received by an **Arduino UNO** on the car, which processes them and sends appropriate signals to the **L298N motor driver**. The motor driver then controls the **DC motors** attached to the **four-wheel chassis**, allowing the car to move forward, backward, left, or right in real time.

Powered by **rechargeable Li-ion batteries**, the system is fully portable and untethered, making it suitable for practical demonstrations and learning. The project stands out as a **low-cost and beginner-friendly solution** to introduce students and hobbyists to key concepts in embedded systems, wireless communication, and mobile robotics. It also showcases how Bluetooth technology can be leveraged for remote control applications in robotics.

## 3.2 SYSTEM ARCHITECTURE

The system consists of two main sections: the Bluetooth-controlled mobile app (Transmitter Unit) and the RC Car (Receiver Unit). The user controls the RC car by sending directional commands from a smartphone app connected via Bluetooth. When the user presses a control button on the app (such as Forward, Backward, Left, Right, or Stop), the command is transmitted to the HC-05 Bluetooth module. This Bluetooth module is connected to an Arduino UNO, which acts as the receiver on the car.

Upon receiving the command, the Arduino UNO processes the instruction and sends the appropriate signals to the L298N motor driver module, which then controls the rotation and direction of the DC motors attached to the car's chassis and wheels. The car moves accordingly in real time, based on the received command. This setup provides a simple yet effective wireless control system for RC cars using only readily available components **and** a smartphone, making it ideal for educational and hobby robotics projects.

## 3.3 BLOCK DIAGRAM

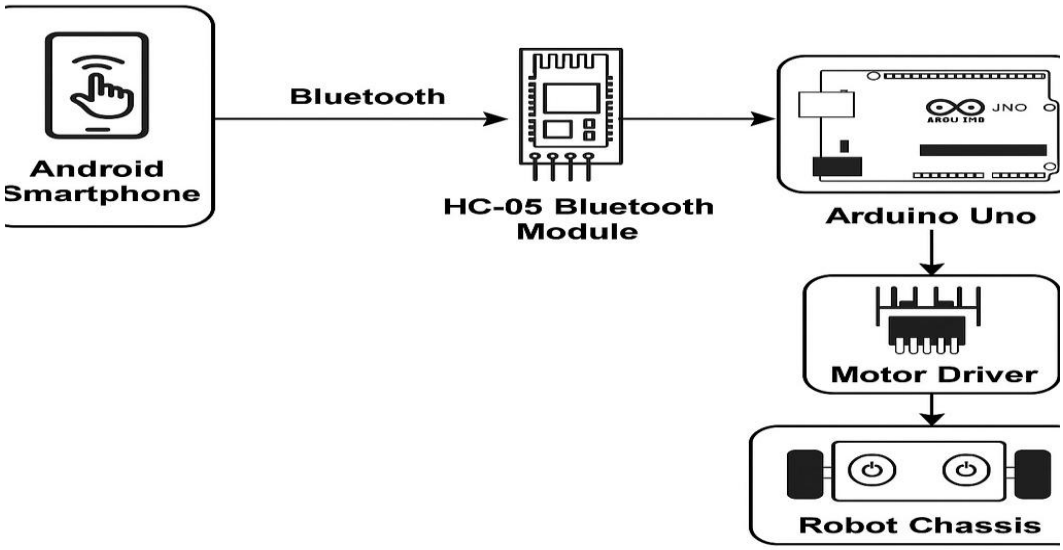



*Fig. 3.1: BLOCK DIAGRAM*

## 3.4 WORKING PRINCIPLE

The **Bluetooth-Controlled RC Car** operates through wireless communication between a smartphone and the RC car, using the **HC-05 Bluetooth module**. The process begins on the transmitter side, where a **mobile phone app** is used to send specific control commands such as 'F' for forward, 'B' for backward, 'L' for left, 'R' for right, and 'S' for stop. When a button is pressed on the app interface, the command is transmitted via Bluetooth using the **Serial Port Profile (SPP)** communication protocol at a baud rate of 9600. [1]

This command is wirelessly received by the **HC-05 Bluetooth module** mounted on the RC car. The received signal is then sent to the **Arduino UNO**, which acts as the main microcontroller on the receiver side. Based on the character received, the Arduino processes the command and sends the appropriate signals to the **L298N motor driver module**. The motor driver activates specific combinations of digital output pins, thereby controlling the direction and rotation of the **DC motors** attached to the **car chassis** and wheels.

This process enables real-time control of the car's movements wirelessly and efficiently, with a typical range of up to 10 meters. The setup eliminates the need for complex gesture detection or sensor-based interpretation, making it a simple, cost-effective, and beginner-friendly project ideal for learning about embedded systems and wireless control.

## 3.5 IMPLEMENTATION

The Bluetooth-Controlled RC Car project is implemented using basic and easily accessible components, making it ideal for students and hobbyists. The central controller of the system is the Arduino UNO, which is used both at the transmitter end (smartphone) and receiver end (RC car). The car does not use any gesture sensors like the MPU6050. Instead, the user controls the car using a smartphone with a Bluetooth terminal or custom app. The commands are transmitted wirelessly via the HC-05 Bluetooth module, which is connected to the RX and TX pins of the Arduino UNO.

On the receiver side, the Arduino UNO receives Bluetooth commands sent from the smartphone and processes them to drive the motors through the L298N motor driver module. The L298N is connected to the DC motors mounted on the car chassis. It receives direction and enable signals from the Arduino to control the motion — forward, backward, left, right, and stop. The entire setup is powered using Li-ion batteries, with separate power lines for the Arduino and the motor driver to ensure smooth and stable operation.

Programming is done using the Arduino IDE. The code listens for specific characters such as 'F' for forward, 'B' for backward, 'L' for left, 'R' for right, and 'S' for stop. [2]

# CHAPTER 4:RESULT AND OBSERVATION

## 4.1 RESULT

The **Gesture Controlled RC Car** project has successfully demonstrated real-time control of a car using Bluetooth-based wireless communication. The system is designed to operate by transmitting control signals from a mobile device to the RC car, eliminating the need for physical remotes or traditional joystick-based interfaces. Instead of using an accelerometer or sensor glove, this model simplifies the design by allowing gesture-based or button-tap control from a Bluetooth-connected mobile app, making the car respond promptly to directional commands such as forward, backward, left, right, and stop.

One of the key achievements of the system is the **wireless maneuverability** enabled by the **HC-05 Bluetooth module**, which reliably transmits commands to the car within a range of approximately 10 meters. The system architecture includes two **Arduino UNO boards**—one acting as the Bluetooth command receiver and the other optionally used in transmitter-side testing setups. The **L298N motor driver** is used to control the direction and motion of the **DC motors** mounted on a **four-wheeled chassis**, powered by **Li-ion batteries** for portability and uninterrupted movement.

The **mobile app interface**, acting as the controller, is simple and intuitive. With clearly labeled buttons for motion commands, it allows the user to easily operate the RC car. This removes the learning curve and makes the project suitable for beginners, hobbyists, and students exploring embedded systems and wireless robotics.

The design is **lightweight, compact, and portable**, making it easy to demonstrate in classrooms or competitions. Since it doesn't rely on external sensors like accelerometers or gyroscopes, it reduces complexity and cost, while still offering a fun and educational experience. The RC car’s modular structure also allows future expansion, such as adding speed control, obstacle detection using ultrasonic sensors, or even integrating a camera module for live video streaming.

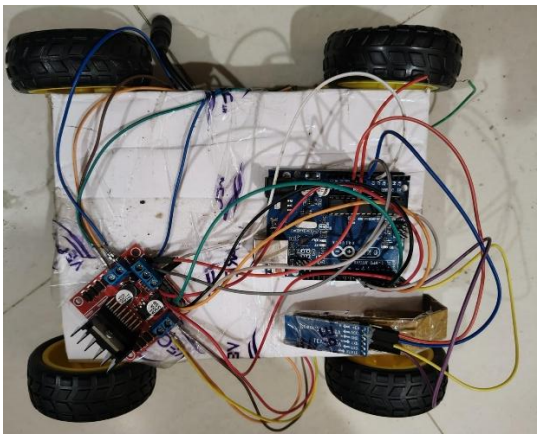


*Fig: 4.1 Gesture Controlled Robot*

## 4.2 OBSERVATION

The **Bluetooth communication** between the **Transmitter Arduino** and **Receiver Arduino** remained stable within a range of **approximately 8–10 meters**.

Overall, the prototype demonstrates **intuitive and real-time gesture control** of an RC car, fulfilling the intended objectives of this project.


*Fig : 4.2 Working of Gesture Controlled Robot*

# CHAPTER 5: CONCLUSION AND FUTURE WORK

## 5.1 CONCLUSION

The development of this **Bluetooth-Controlled RC Car** demonstrates a practical and efficient implementation of **Human-Machine Interaction (HMI)** using low-cost embedded hardware. By utilizing **Arduino UNO**, **HC-05 Bluetooth modules**, **DC motors**, and a **motor driver (L298N)**, the system achieves reliable and real-time wireless control of an RC car directly from a smartphone application.

The car responds accurately to directional commands such as **forward, backward, left, right, and stop**, allowing users to control its movement with ease. The entire system is **wireless, battery-powered, compact**, and highly suitable for educational and hobbyist purposes. It highlights how basic components and simple Bluetooth communication can be combined to create an effective robotic system.

Overall, the project meets its goals of building a **low-cost, intuitive, and reliable Bluetooth-controlled RC car**, forming a solid base for future enhancements such as voice control, obstacle detection, or camera integration—opening doors to more advanced robotics and smart vehicle systems.

## 5.2 FUTURE ENHANCEMENTS

Several advancements can be incorporated to improve the functionality and user experience of the Gesture Controlled RC Car. One promising direction is the integration of Machine Learning (ML) techniques to enable recognition of more complex and diverse gestures. Instead of relying solely on simple tilt-based controls, gesture patterns could be analyzed, allowing for richer and more intuitive interactions.

Another enhancement is extending the communication range. The existing Bluetooth connection limits control to about 10 meters, but upgrading to Wi-Fi modules like the

ESP32 or using advanced RF modules can significantly increase the operational range. Additionally, implementing mesh networking could allow simultaneous control of multiple cars, opening up new possibilities for group coordination and swarm robotics. Obstacle avoidance is another key area for improvement. By adding ultrasonic or infrared sensors, the car could detect and automatically avoid obstacles, enabling semi-autonomous navigation. This would not only enhance safety but also introduce a higher level of autonomy.

Speed control can be made more sophisticated by incorporating PWM (Pulse Width Modulation) techniques for variable speed operation. Furthermore, speed adjustments could be tied to gesture input, such as varying the tilt angle to control how fast the car moves, providing a more dynamic driving experience.

Integrating an FPV (First Person View) camera would add a new dimension to the project. By streaming live video to the user's smartphone, the car could be remotely driven based on the camera feed, offering a dronelike exploration experience. This could greatly enhance usability in environments where direct line of sight is limited.

Lastly, combining voice commands with gesture control would create a hybrid control system, enhancing the human-machine interface (HMI). For example, a voice command like "Start" could activate the gesture control mode, making the system more interactive and accessible.

The technology behind gesture-controlled RC cars has broad applications. It can be used effectively in educational robotics settings, such as schools, colleges, and workshops, to teach students about embedded systems and wireless communication. In assistive robotics, this approach can help differently-abled users operate devices more intuitively. Looking forward, the methodology can be extended to gesture-controlled drones, providing new ways to pilot aerial devices. Additionally, the same gesture-based architecture can be adapted for smart home devices, creating seamless, touchless control systems for everyday environments.

## REFERENCES


[1] M. Banzi and M. Shiloh, *Getting Started with Arduino*, 3rd ed. Maker Media, Inc., 2014.

[2] A. Kumar, R. Singh, and P. Sharma, “Bluetooth Controlled Car using Arduino,” *International Journal of Engineering Research & Technology (IJERT)*, vol. 7, no. 5, pp. 10–14, May 2018.

[3] Arduino, "Arduino UNO Board," [Online]. Available: https://www.arduino.cc/en/Main/ArduinoBoardUno/ [Accessed: Jun. 12, 2025].

[4] Components101, "HC-05 Bluetooth Module Datasheet," [Online]. Available: https://components101.com/wireless/hc-05-bluetooth-module [Accessed: Jun. 12, 2025].

[5] ElectronicWings, "L298N Motor Driver Module with NodeMCU," [Online]. Available: https://www.electronicwings.com/nodemcu/l298n-motor-driver-module [Accessed: Jun. 12, 2025].

[6] Instructables, "Bluetooth Controlled Car Using Arduino," [Online]. Available: https://www.instructables.com/Bluetooth-Controlled-Car-Using-Arduino/ [Accessed: Jun. 12, 2025].

[7] Blynk, "Blynk IoT Platform," [Online]. Available: https://blynk.io/. [Accessed: Jun. 12, 2025].

[8] TutorialsPoint, "Arduino Tutorial," [Online]. Available: https://www.tutorialspoint.com/arduino/index.htm [Accessed: Jun. 12, 2025].

[9] Random Nerd Tutorials, "Guide to HC-05 Bluetooth Module with Arduino," [Online]. Available: https://randomnerdtutorials.com/. [Accessed: Jun. 12, 2025].

# APPENDIX

## Appendix

```
 // Define motor pins
const int IN1 = 2;
const int IN2 = 3;
const int IN3 = 4;
const int IN4 = 5;
const int en1 = 6;
const int en2 = 7;
char data = 0;            //Variable
for storing received data
void  setup()
{
    Serial.begin(9600);   //Sets the
baud for serial data transmission
   pinMode(13, OUTPUT);  //Sets digital
pin  13 as output pin
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  pinMode(IN3, OUTPUT);
  pinMode(IN4, OUTPUT);
}
void loop()
{
   if(Serial.available() > 0)
   {
      data = Serial.read();
      Serial.println(data);
         if(data == 'F') {
         front();
         Serial.println("going front");
         }
          else  if(data == 'B') {
         back();
          Serial.println("going back");
          }

         else  if(data == 'L')  {
         left();
         Serial.println("going left");
         }
         else  if(data == 'R')  {
         right();
         Serial.println("going right");
         }
         else if(data == 's'){
```

```
          stop();
          Serial.println("no
movement");
         }

   }
}

void front(){
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);
  digitalWrite(IN3, HIGH);
  digitalWrite(IN4, LOW);
  analogWrite(en1,150);
  analogWrite(en2,150);
 }

void back(){
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, HIGH);
  digitalWrite(IN3, LOW);
  digitalWrite(IN4, HIGH);
  analogWrite(en1,150);
  analogWrite(en2,150);
 }

void left(){
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);
  digitalWrite(IN3, LOW);
  digitalWrite(IN4, HIGH);
  analogWrite(en1,150);
  analogWrite(en2,150);

 }

void right(){
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, HIGH);
  digitalWrite(IN3, HIGH);
  digitalWrite(IN4, LOW);
    analogWrite(en1,150);
  analogWrite(en2,150);
 }

void stop(){
  analogWrite(en1,0);
  analogWrite(en2,0);   }
```