**Chapter-1**

**INTRODUCTION**

**Chapter-1**

**INTRODUCTION**

In this high technology, a robot must be able to detect and follow humans. A robot that can detect an obstacle within a specific range is called ‘Obstacle Robot’. Robots are used to change people’s lives and make people’s life luxurious. A robot that can use in shopping time. Which carries items, and follow human without any remote more useful. A robot that can use in the hospital to Bringing medicine with more accuracy and fast. The robot has many works like work as Trolley, structure in hospital, and a small basket with a car and so on. Now in this changing world, people are starting to live with robot-like Obstacle avoiding, Bluetooth, voice, and human following robots for their Luxurious life. This project named called obstacle avoiding robot because it can detect the obstacle and avoid it by using ultrasonic sensor with more accuracy to rotate 180 degrees, we have used the servo motor to detect the obstacle within the range of 180 degree any kind of work with more accuracy and in lesser Time. The Bluetooth control and voice control robot can be used in the defense sector also to carry weapons for the soldiers. This type of Robot can sense obstacles and humans automatically and it can use in the future in our cars. A Bluetooth robot can be modified in the future with more developed components and can make it more advanced. This robot can be enhanced by structure by adding more components like cameras, tracking devices and make it. More beautiful and workable. This robot will be more trend in our future.

**All-in-one robots** are a fascinating concept, promising a versatile and integrated solution for various tasks. While the exact definition can vary depending on the specific application, these robots generally refer to systems that combine multiple functionalities into a single, compact unit.

**Multifunctionality:** All-in-one robots are designed to perform a range of tasks, often exceeding the capabilities of single-purpose robots. This versatility can be particularly advantageous in dynamic environments where adaptability is crucial.

**Integration:** These robots typically integrate various components, such as sensors, actuators, and control systems, into a cohesive unit. This integration enhances efficiency and reduces complexity.

**Compactness:** All-in-one robots are often designed to be compact and portable, making them suitable for deployment in confined spaces or for easy transportation.

# 

* 1. **PROBLEM STATEMENT**

Design and implement a robotic rover capable of navigating complex environments autonomously, avoiding obstacles, and responding to both Bluetooth and voice commands.

**Solution:**

To address this problem, model propose to design a rover equipped with ultrasonic sensors to detect obstacles. These sensors will provide real-time distance measurements, enabling the rover to adjust its path and avoid collisions. Additionally, we will integrate a Bluetooth module to facilitate remote control via a smartphone or other compatible device. To enhance user interaction, a voice recognition module will be incorporated, allowing users to issue commands verbally. By combining these technologies, we aim to create a versatile and user-friendly robotic system capable of navigating complex environments and performing various tasks.

* 1. **Objectives**

1. **Autonomous Navigation:** Develop a robust navigation system that enables the rover to independently explore and map its surroundings, avoiding obstacles effectively.
2. **Obstacle Detection and Avoidance:** Implement a reliable obstacle detection mechanism using ultrasonic sensors and design efficient algorithms to plan and execute avoidance maneuvers.
3. **Bluetooth Control:** Integrate a Bluetooth module to establish wireless communication with a remote device, allowing users to control the rover's movement and actions.
4. **Voice Control:** Incorporate a voice recognition system that can accurately interpret voice commands and execute corresponding actions.

**Chapter-2**

**LITERATURE SURVEY**

**2. LITERATURE SURVEY**

**[1] “Obstacle Detection and Avoidance Robot**”, R Chinmayi,2020

This is a cost-effective obstacle avoidance circuit developed in Amrita University. This robot is driven with an Arduino board controlled by an ultrasonic sensor. The obstacle is being detected at a distance of 15 cm when it senses the right path to move with the level of distance to the next obstacle. The number of vehicles is tremendously increasing day to day and the risk factor of accidents also increases with it. The Bluetooth module acts as an interface to communicate with the device using android apps. While driving the system with 76.93cm/s speed, an accuracy of obstacle detection of 92.5 percent is being achieved.

**[2] “Voice-Controlled Robotics”,** Janusz Pochmara,2023

The possibilities of controlling a robotic arm using voice commands with the use of an external API responsible for speech-to-text translation. The article describes the creative process of designing the robot model and user interface included in the simulator used to control the robot. The software was developed in the Unreal Engine 4.27.2 environment using the built-in visual programming language - blueprint, as well as C++ and Python. The three-dimensional model was created entirely in Blender 3D software. The simulator's task is to faithfully reproduce manipulation of simple objects in real-time, as well as record and load the manipulator's trajectory along with all objects on the scene.

**[3] “Bluetooth-Controlled Robotics”**, V. Santhosh Kumar,2023

People use cars every day, and technical advancements in them have been accelerating quickly. In this research work, we implemented a robot car that can be readily operated by a mobile device. A Robot is remotely controlled using a bluetooth module and is designed to move using AVR Studio and Android Studio to build a simple application. For the control of all the system, an Atmega8 Microcontroller is used. The IR sensors are used for detecting obstacles encountered in the path of the Robotic system. The modules such as Motor, bluetooth and IR sensors are interfaced with the Atmega8 Microcontroller. The programming for the controller is done in AVR studio. This research work is anticipated to serve as a guide for future technological innovation development.

**[4]“Bluetooth Controlled Robotic Car Using Arduino”**, Rajeshwari Sissodia,2021

A machine is required when humans would like to work in different environmental conditions, such as toxic material, remote handling of bombs, health conditions, and sewage treatment. The paper aims to build a robot car that monitors the human voice's movement and senses distant objects. The L298N board, HC-05, Arduino Uno microcontroller, ultrasonic sensor, battery, and jumping wires are included in this system. The robot movement and control system is used by the speaker to allow the robot to react to any speaker command that gives any verbal instruction that produces sound frequencies of the human voice. Through the software application, the user of a robotic car will choose the route or path to control the movement of the car. The user can monitor the robot's movements on his own smart device and allow the car to drive in his own way. In this method, a microcontroller with android devices is linked through a bluetooth module to receive desired voice commands. The robot then escapes obstacles and detects distant objects. The android application that is used to convert a voice to a text command and then transmit data to a microcontroller moves the robot via a voice application according to the user's command. After receiving the command, the robot moves in left, right, forward, and backward directions. This device tried to alert workers to the possibility of a terrorist attack in a military camp.

**[5] “Voice Controlled Robot Vehicle Using Arduino UNO”,** Poojitha Pagolu,2023

The study aims to build voice-controlled robots with autonomous braking, obstacle avoidance, and speed reduction capabilities. This research will enable the development of robots that can accurately and autonomously assess and respond to environmental conditions, such as obstacles in their path. The primary objective of the robot is to use voice commands to steer the robotic vehicle and analyze speech input to carry out the necessary actions to avoid obstacles. The research is expected to significantly improve the safety of autonomous vehicles by allowing them to detect and respond in real time to changes in their environment. The robot is voice-activated and connected to an Android smartphone through Bluetooth. The robot will also be able to recognize obstacles with the help of an ultrasonic sensor.

**Chapter-3**

**METHODOLOGY**

**&**

**IMPLEMENTATION**

**3.1 Block Diagram All in One Rover**

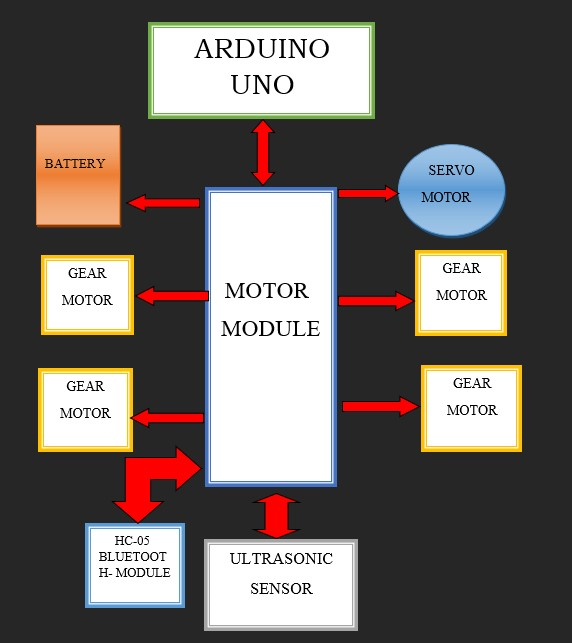


Figure 3.1 Block Diagram

**3.2 Flow chart of All-in-One rover**

**3.2.1. Flow Chart of Obstacle avoiding Robot**

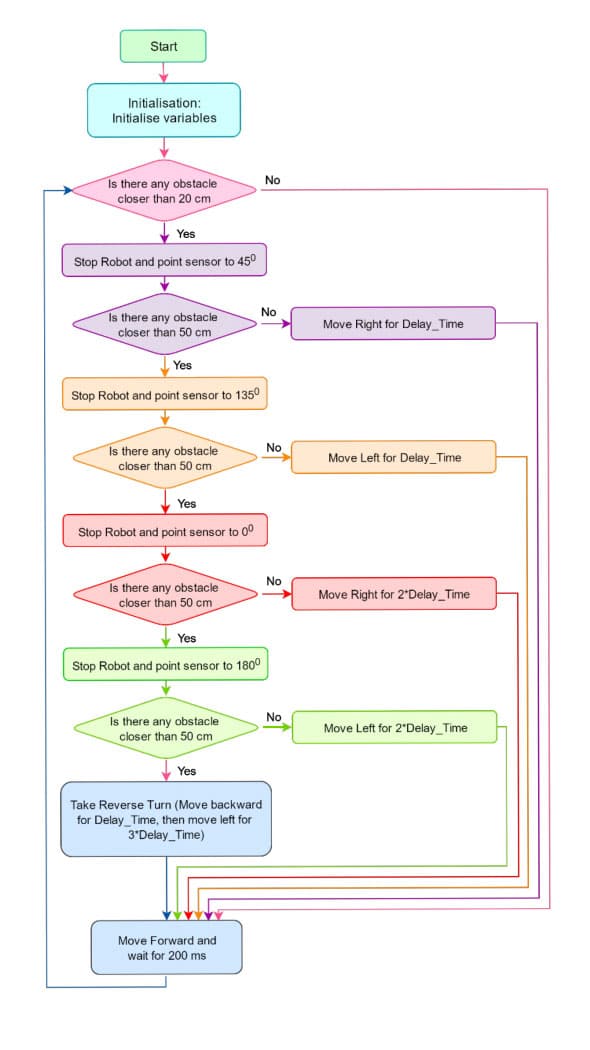


Figure 3.2 Flow chart of obstacle avoiding Robot

* + 1. **Flow Chart of Bluetooth control**

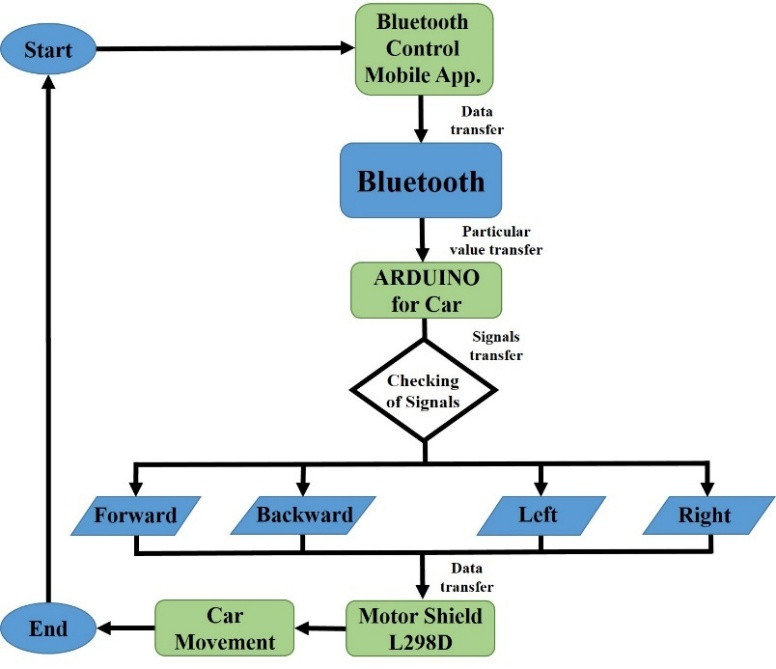


Figure 3.3 Flow chart of Bluetooth control Robot

**Explanation of Flowchart: -**

 Power **On Arduino:** The Arduino Uno is powered up, initializing its components.

 Initialize **Sensors and Modules:** The ultrasonic sensors, Bluetooth module, and voice recognition module (if used) are initialized and configured.

 Read **Sensor Data:** The Arduino reads data from the ultrasonic sensors to determine the distance to obstacles.

 Check **for Obstacles:** The Arduino analyzes the sensor data to check if there are any obstacles in the path.

 Avoid **Obstacle:** If an obstacle is detected, the Arduino sends appropriate signals to the motor driver to maneuver the rover around the obstacle.

 Check **for Bluetooth Commands:** The Arduino checks if any Bluetooth commands have been received.

 Execute **Bluetooth Commands:** If Bluetooth commands are received, the Arduino interprets the commands and sends appropriate signals to the motor driver to control the rover's movement.

 Check **for Voice Commands:** The Arduino checks if any voice commands have been recognized.

 Execute **Voice Commands:** If voice commands are recognized, the Arduino interprets the commands and sends appropriate signals to the motor driver to control the rover's movement.

 Move **Forward:** If no obstacles are detected and no commands are received, the Arduino sends signals to the motor driver to move the rover forward.

**3.1.3 Schematic diagram of All-in-One rover**

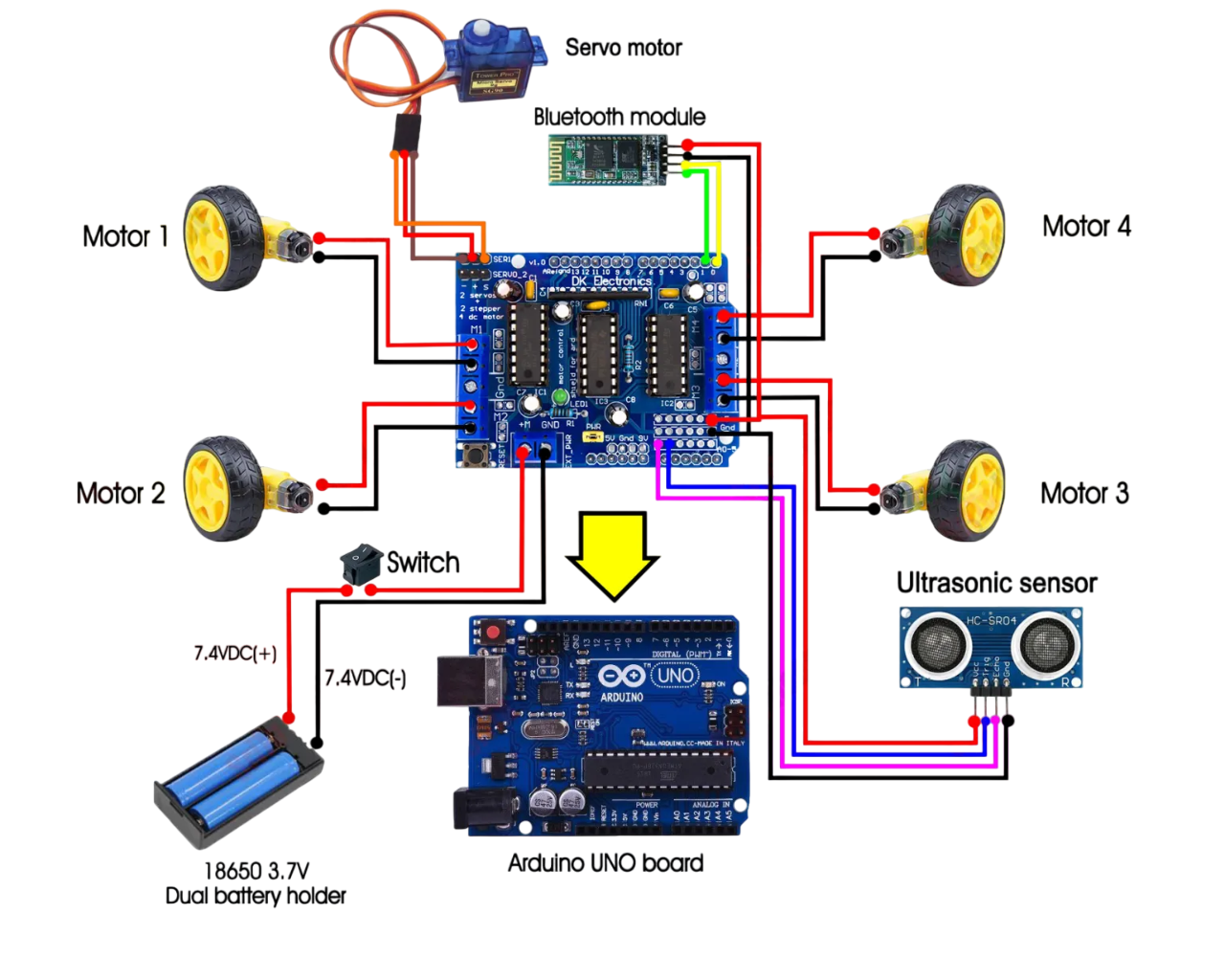
****

Figure 3.4 Schematic Diagram

**3.3. Apparatus Required**

1. Arduino Uno
2. Motor driver shield
3. 4xgear motor
4. 4xwheels
5. Servo motor
6. 18650 Li-on battery
7. Battery Holder
8. Ultra-Sonic sensor
9. HC-05 Module

**3.3.1 Arduino Uno**

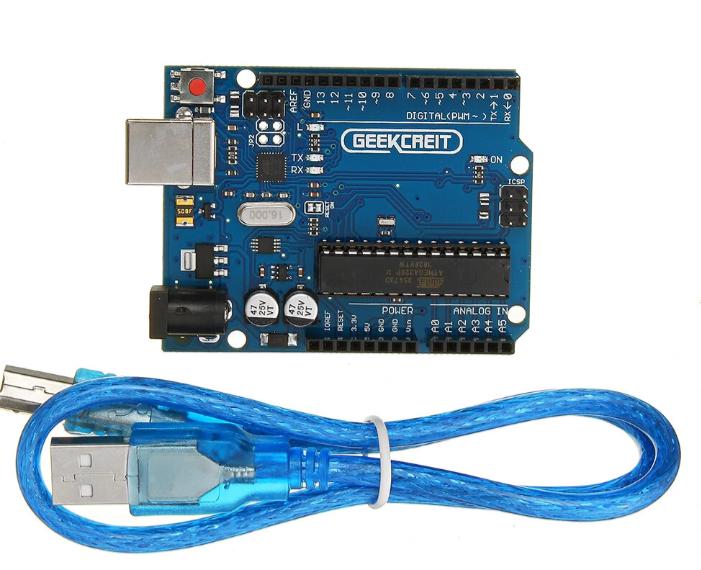
Arduino can sense its environment by receiving inputs from sensors, and interact with its environment by controlling lights, motors, or other actuators. The Arduino integrated development environment (IDE) is a cross-platform application written in Java, and is derived from the IDE for the Processing programming language and wiring projects. It can run independently and communicate with other software such as Flash, Processing, MaxMSP and more. Arduino IDE is open source so you can download and share thousands of interactive projects for free!

Figure 3.5 Arduino Uno

**Specifications: -**

* Microcontroller: Microchip ATmega328P[7]
* Operating Voltage: 5 Volts
* Input Voltage: 7 to 20 Volts
* Digital I/O Pins: 14
* PWM Pins: 6 (Pin # 3, 5, 6, 9, 10 and 11)
* UART: 1
* I2C: 1
* SPI: 1
* Analog Input Pins: 6
* DC Current per I/O Pin: 20 mA
* DC Current for 3.3V Pin: 50 mA
* Flash Memory: 32 KB of which 0.5 KB used by bootloader.
* SRAM: 2 KB
* EEPROM: 1 KB
* Clock Speed: 16 MHz
* Power Sources: DC Power Jack, USB Port and the VIN pin (+5 volt only)

**Working: -** The Arduino Uno is a popular microcontroller board that utilizes the ATmega328P chip. It's a user-friendly platform for individuals of all experience levels, providing a straightforward approach to interact with electronics and code. The board offers 14 digital input/output pins, 6 analog input pins, and operates at a clock speed of 16 MHz. It can be powered via a USB connection or an external power supply. The Arduino IDE, a user-friendly software, is used to write and upload code to the board. Once uploaded, the ATmega328P executes the code, processes input from sensors, and controls output devices.

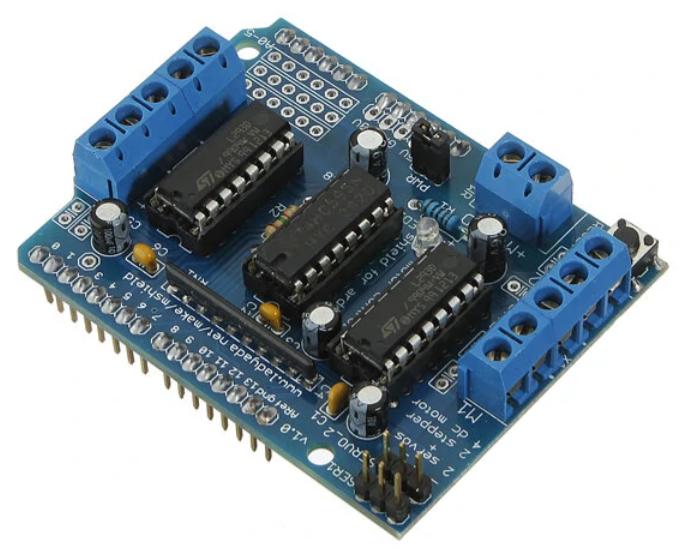
**3.3.2. Motor Driver(L293D)**

Figure 3.6 Motor Driver(L293D)

Motors are an inseparable part of many robotics and electronics projects and have different types you can use depending on their application.

Each channel of this module has the maximum current of 1.2A and doesn’t work if the voltage is more than 25v or less than 4.5v. So be careful with choosing the proper motor according to its nominal voltage and current.

How to Use Arduino L293D Motor Driver Shield?

While using this shield 6 analog Pins (which can be used as digital pins too), pin 2 and pin 13 of arduino are free.In the case of using Servo motor, pins 9, 10, 2 are in use.In the case of using DC motor, pin11 for #1, pin3 for #2, pin5 for #3, pin6 for #4 and pins 4, 7, 8 and 12 for all of them are in use. In the case of using Stepper motor, pins 11 and 3 for #1, pins 5 and 6 for #2 and pins 4, 7, 8 and 12 for all of them are in use.You can use free pins by wired connections. If you are applying separate power supply to Arduino and shield, make sure you have disconnected the jumper on the shield.

**Specifications**

* **Supply Voltage Range:** 4.5V to 36V
* **Output Current:** 600mA per channel (1.2A peak)
* **Logic Supply Voltage:** 4.5V to 7V
* **Operating Temperature Range:** 0°C to 70°C
* **Number of Channels:** 4 (2 H-bridges)
* **Package Type:** DIP-16

**Working: -** The L293D is a versatile motor driver IC that enables the control of DC motors. It operates within a supply voltage range of 4.5V to 36V and can deliver a maximum output current of 600mA per channel. The IC features two H-bridges, allowing for bidirectional control of two DC motors. By applying appropriate logic signals to the input pins, the L293D can control the direction and speed of the connected motors. It also incorporates thermal shutdown protection to prevent damage from overheating.

**3.3.3 Servo Motor**

Figure 3.7 Servo Motor

A servo motor is a type of motor that can rotate with great precision. Normally this type of motor consists of a control circuit that provides feedback on the current position of the motor shaft, this feedback allows the servo motors to rotate with great precision. If you want to rotate an object at some specific angles or distance, then you use a servo motor. It is just made up of a simple motor which runs through a servo mechanism. If motor is powered by a DC power supply, then it is called DC servo motor, and if it is AC-powered motor then it is called AC servo motor. For this tutorial, we will be discussing only about the DC servo motor working.

**Specifications: -**

* Operating Voltage is +5V typically.
* Torque: 2.5kg/cm
* Operating speed is 0.1s/60°
* Gear Type: Plastic
* Rotation: 0°-180°
* Weight of motor: 9gm
* Package includes gear horns and screws.

**Working: -** A servo motor is a rotary actuator that allows for precise control of angular position. It consists of a DC motor, gears, a potentiometer, and control circuitry. The motor's rotation is controlled by a Pulse-Width Modulation (PWM) signal. The width of the pulse determines the angle to which the motor rotates. A potentiometer measures the current angular position of the motor shaft and sends this information back to the control circuitry. The control circuitry compares the desired angle with the actual angle and adjusts the PWM signal to drive the motor to the desired position. This closed-loop control system ensures accurate and precise positioning.

**3.3.4 Ultra Sonic Sensor**

Figure 3.8 Ultra-Sonic Sensor

An ultrasonic sensor is an electronic device that measures the distance of a target object by emitting ultrasonic sound waves, and converts the reflected sound into an electrical signal. Ultrasonic waves travel faster than the speed of audible sound (i.e. the sound that humans can hear). Ultrasonic sensors have two main components: the transmitter (which emits the sound using piezoelectric crystals) and the receiver (which encounters the sound after it has travelled to and from the target).

**Specifications: -**

* Power Supply: DC 5V
* Working Frequency: 40Hz
* Ranging Distance: 2cm – 400cm/4m
* Resolution: 0.3 cm
* Measuring Angle: 15 degrees
* Trigger Input Pulse width: 10uS
* Dimension: 45mm x 20mm x 15mm
* Current: 15mA
* Working

**Working: -**An ultrasonic sensor functions by emitting high-frequency sound waves that are inaudible to humans. These sound waves travel through the air and, upon encountering an object, reflect back to the sensor. By precisely measuring the time it takes for the sound wave to travel to the object and return, the sensor can calculate the distance to the object. This process, known as time-of-flight measurement, enables accurate distance sensing.

**3.3.5 HC-05 Module**

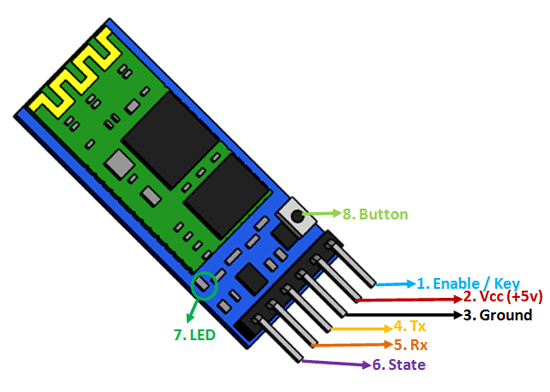
****

Figure 3.9 HC-05 Module

HC‐05 module is an easy-to-use Bluetooth SPP (Serial Port Protocol) module, designed for transparent wireless serial connection setup. The HC-05 Bluetooth Module can be used in a Master or Slave configuration, making it a great solution for wireless communication. This serial port Bluetooth module is fully qualified Bluetooth V2.0+EDR (Enhanced Data Rate) 3Mbps Modulation with complete 2.4GHz radio transceiver and baseband. It uses CSR Blue core 04‐External single chip Bluetooth system with CMOS technology and with AFH (Adaptive Frequency Hopping Feature).

**Hardware Features**

* Typical ‐80dBm sensitivity.
* Up to +4dBm RF transmit power.
* 3.3 to 5 V I/O.
* PIO (Programmable Input/Output) control.
* UART interface with programmable baud rate.
* With integrated antenna.
* With edge connector.

**Software Features**

* Slave default Baud rate: 9600, Data bits:8, Stop bit:1, Parity: No parity.
* Auto‐connect to the last device on power as default.
* Permit pairing device to connect as default.
* Auto‐pairing PINCODE:”1234” as default.

**Working: -** The HC-05 module is a widely used Bluetooth module that enables wireless communication between devices. It operates on the 2.4 GHz frequency band and supports serial communication. The module has two main modes: Master and Slave. In Master mode, it initiates the connection with other devices. In Slave mode, it waits for a connection request from a Master device. Once a connection is established, data can be transmitted and received serially between the two devices. This allows for remote control of devices, data transfer, and other wireless applications.

**3.3.6 Connecting Wires**

Figure 3.10 Jumper Wires

To connect wires to your Arduino Uno, first identify the desired digital or analog pins. Prepare the wires by stripping the insulation from their ends. Insert one end of the wire into the breadboard and the other end into the corresponding pin on the Arduino. Ensure secure connections and avoid wire crossings. Use a breadboard for easy prototyping and consider color-coding wires for clarity. Always check power supply compatibility and ground connections. By following these guidelines, you can confidently connect components to your Arduino Uno.

**3.4 Software Implementation Using Arduino Uno**

The Arduino Uno is programmed using the Arduino IDE. The code is structured into functions for different tasks, such as initializing components, reading sensor data, controlling motors, and handling Bluetooth and voice commands. The ultrasonic sensor is used to measure the distance to obstacles, and the motor driver is used to control the rover's movement. The Bluetooth module enables wireless communication with a smartphone or other devices, allowing remote control of the rover. The voice recognition module, if used, can interpret voice commands and trigger appropriate actions. By combining these components and programming techniques, the Arduino Uno can effectively control the rover's behavior and achieve the desired functionalities.

**3.5.1 Development environment setup Arduino IDE installation:**

• To set up the Arduino IDE, download and install the software for your operating system. Connect your Arduino Uno board to your computer using a USB cable and select the correct port and board in the IDE. Write your first sketch, which is a simple program to blink an LED. Upload the sketch to the Arduino board and observe the LED blinking. Regularly update the IDE, explore libraries, and participate in the Arduino community to enhance your learning and project development.

**3.5.2. Debugging and testing**

**Debugging**

Debugging is a crucial step in the development process. Here are some common debugging techniques for Arduino projects:

1. **Serial Monitor:**
   * Use the Serial Monitor in the Arduino IDE to print debugging messages and sensor readings.
   * Monitor the output to identify any unexpected behaviour or errors.
2. **LED Indicators:**
   * Connect LEDs to specific pins to visually indicate the state of different parts of your code.
   * For example, you can use an LED to signal when the rover is moving forward, turning, or stopping.
3. **Logic Analyzer:**
   * Use a logic analyzer to capture digital signals on the Arduino pins and analyze their timing and patterns.
   * This can help identify timing issues or incorrect signal levels.
4. **Step-by-Step Debugging:**
   * Use the Arduino IDE's debugging features to step through your code line by line, inspecting variables and checking the execution flow.

**Testing**

Thorough testing is essential to ensure the reliability and robustness of your rover. Here are some testing scenarios:

1. **Obstacle Avoidance Testing:**
   * Place obstacles in different positions and orientations to test the rover's ability to detect and avoid them.
   * Vary the speed and complexity of the obstacles to evaluate the rover's performance under different conditions.
2. **Bluetooth Control Testing:**
   * Test the Bluetooth connection range and reliability.
   * Verify that the rover responds accurately to different Bluetooth commands.
   * Test the responsiveness of the rover to commands from different devices.
3. **Voice Control Testing:**
   * Test the voice recognition accuracy in different noise levels and environments.
   * Verify that the rover responds correctly to voice commands.
   * Test the clarity and accuracy of voice commands.
4. **Battery Life Testing:**
   * Test the rover's battery life under different conditions, such as continuous operation, obstacle avoidance, and Bluetooth communication.

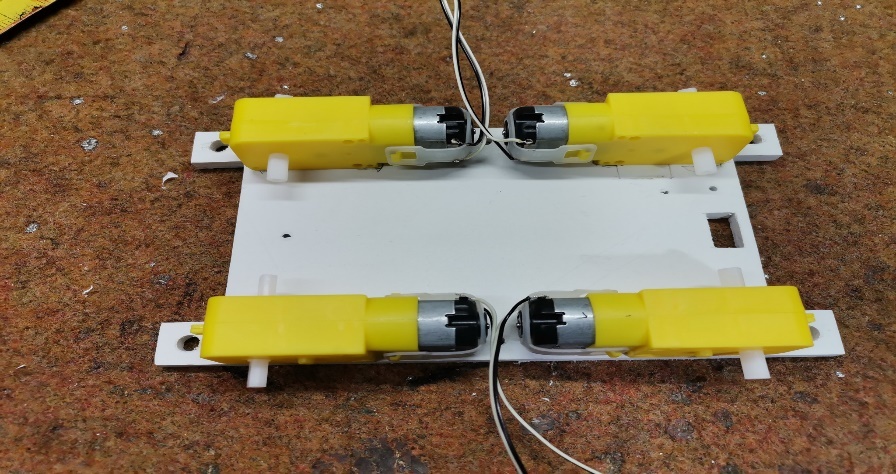
**3.5 Hardware implementation**

Figure 3.11 Design Of Chasis

To make it, we need only one rectangular plate, on which the lower side should be glued the engines, and on the upper surface are mounted other elements. You can use discontinued L293D motor driver shield like in my case, but also and Adafruit motor shield as is presented on the schematic diagram without any changes.

The principle of object detection and monitoring is based on data accepted by both sensors. The ultrasonic sensor detects the presence of an object in front of it within certain limits, in our case between 10 and 30 centimeters. If there is no object (for example our hand) in this space, all four engines are idle. At the moment when an object appears in this space, the data from the infrared sensors are read, and based on the obtained data, commands are given to the motors, whereby the robot moves in the desired direction. The distance to which the infrared sensors respond is adjusted by a small trimming potentiometer. This distance should be adjusted so that it is slightly larger than the minimum distance to which the ultrasonic sensor is set to respond, in our case it is more than 10 centimeters.

The portable application utilized is modified so that the voice orders given to the handset are received by the mic and these simple voice orders are changed over to advanced word successions (A to D transformation). These stored sequences are then transmitted to the robotic vehicle via Bluetooth transceiver module and are sent to the transceiver controller. Android application transceiver is used to decode the received signal with the Bluetooth module. The controller contrasts these signals and puts away program orders in it and converts them into voice strings. The voice strings are then used to run the servo engines for the ideal interval of time.

**Chapter-4**

**Results and Observations**

**4.1 Results and Observations**

**4.1.1. Obstacle Avoidance:**

* Effective Obstacle Detection: The ultrasonic sensors accurately detect obstacles in the rover's path.
* Smooth Obstacle Negotiation: The rover successfully avoids obstacles by adjusting its direction or stopping.
* Sensitivity to Obstacle Distance: The rover can differentiate between close and distant obstacles, allowing for more precise maneuvers.

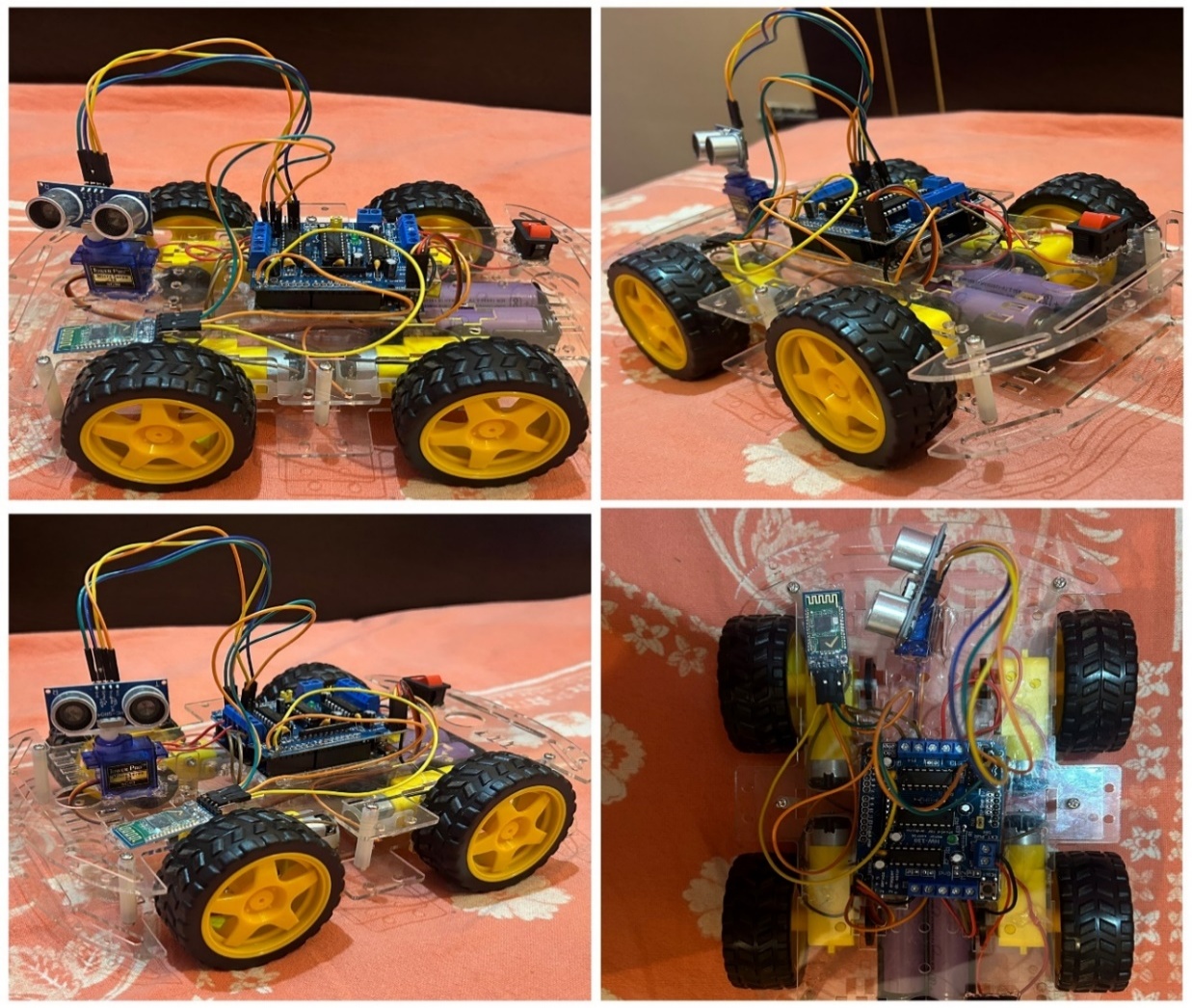


Figure 4.0 Design of All in one Rover

**4.1.2. Bluetooth Control:**

* Reliable Connection: The Bluetooth module establishes a stable connection with the control device.
* Responsive Control: The rover responds promptly to commands sent via Bluetooth.
* Precise Maneuvers: The rover can be controlled to move forward, backward, turn left, or turn right with accuracy.

**4.1.3. Voice Control:**

* Accurate Voice Recognition: The voice recognition module accurately interprets voice commands.
* Reliable Command Execution: The rover responds correctly to voice commands, such as "move forward," "turn left," or "stop."
* Noise Tolerance: The voice control system can function effectively in moderate noise environments.

**4.1.4. Overall Performance:**

* Smooth Navigation: The rover navigates smoothly through various obstacles and environments.
* Accurate Response: The rover responds accurately to both Bluetooth and voice commands.

**4.2 Discussion on Improving System Performance**

**4.2.1 Hardware Optimization:**

* Efficient Motor Drivers: Employ high-efficiency motor drivers to minimize power loss and maximize motor performance.
* Lightweight Materials: Use lightweight materials for the rover's chassis to reduce its overall weight, improving maneuverability and battery life.
* Optimized Power Supply: Implement efficient power management techniques, such as using low-dropout voltage regulators and power-saving modes.
* High-Quality Sensors: Invest in high-quality sensors with accurate and reliable measurements.
* Robust Mechanical Design: Ensure a sturdy and well-balanced mechanical design to minimize vibrations and improve stability.

**4.2.2. Software Optimization:**

* **Efficient Algorithms:** Use optimized algorithms for obstacle avoidance, path planning, and sensor data processing to reduce computational overhead.
* **Code Optimization:** Minimize unnecessary calculations and memory usage by optimizing your code.
* **Interrupt-Driven Programming:** Employ interrupt-driven programming to handle events efficiently, reducing CPU load and improving responsiveness.
* **Asynchronous Tasks:** Use asynchronous programming techniques to handle multiple tasks concurrently, improving overall system performance.
* **Modular Code Structure:** Break down your code into modular functions to improve readability and maintainability.

**4.3. Advantages and Disadvantages**

**Advantages**

* The Robot is small, therefore less space required.
* We can access the robot vehicle from meters as we are using WIFI for the connection between robot and the server PC.
* As we are us camera which is attached to the robot so it will capture video which will be used for security.
* Low power consumption.
* No accident is done by improper driving of people and also available for elderly and disabled people.

**Disadvantages**

* It cannot operate after it’s Bluetooth range.
* We can access for limited devices.
* For voice control noise disturbance will occurs.

**Applications**

* Obstacle avoiding robots can be used in almost all mobile robot navigation systems.
* They can be used for household work like automatic vacuum cleaning.
* They can also be used in dangerous environments, where human penetration could be fatal.

Some real-world applications of this voice-controlled Robot are:

1. The robot is useful in places where humans find difficult to reach but human voice reach. Such asin fire situations, in highly toxic areas.

2. The robot can be used for monitoring or investigation.

3. The voice controlled robotic car can be easily drive by unskilled driver by using voice commands with the help of android application in smart phone.

**Chapter-5**

**CONCLUSION**

**5.1. Future Enhancement**

**1. Enhanced Autonomy:**

* **Advanced Path Planning:** Implement more sophisticated path planning algorithms, such as A\* search or Dijkstra's algorithm, to navigate complex environments.
* **Self-Localization:** Utilize techniques like SLAM (Simultaneous Localization and Mapping) to allow the rover to accurately locate itself within its environment.
* **Adaptive Behavior:** Develop algorithms that enable the rover to adapt to changing environmental conditions and unexpected obstacles.

**2. Increased Functionality:**

* **Multiple Sensor Integration:** Combine multiple sensors, such as LiDAR, cameras, and temperature sensors, to gather more comprehensive information about the environment.
* **Task Automation:** Program the rover to perform specific tasks, such as delivering packages, monitoring indoor environments, or assisting in search and rescue operations.
* **Human-Robot Interaction:** Improve the rover's ability to interact with humans through natural language processing and gesture recognition.

**3. Educational Applications:**

* **STEM Education:** Use the rover as an educational tool to teach students about robotics, electronics, and programming.

**4. Commercial Applications:**

* **Home Automation:** Integrate the rover into smart home systems to perform tasks like delivering items or monitoring security.
* **Industrial Applications:** Utilize the rover for tasks such as inventory management, quality control, and surveillance.
* **Agricultural Applications:** Deploy the rover for tasks like crop monitoring, soil analysis, and automated irrigation.

**5.2. Conclusion**

A successful implementation of a prototype of human following robot is illustrated in this paper. This robot does not only have the detection capability but also the following ability as well. While making this prototype it was also kept in mind that the functioning of the robot should be as efficient as possible. Tests were performed on the different conditions to pin point the mistakes in the algorithm and to correct them. The different sensors that were integrated with the robot provided an additional advantage. The obstacle robot is an automobile system that has the ability to recognize obstacles, move and change the robot’s position toward the subject in the best way to remain on track. This project uses arduino, motors different types of sensors to achieve its goal. This project challenged the group to co-operate, communicate, and expand their understanding of electronics, mechanical systems, and their integration with programming.

**5.3. References**

[1]. Kumar, D. S., & Arvind, S. (2021). Introduction to Sensors for Ranging and Obstacle Detection. In Hands-On Robotics Programming with Arduino and Raspberry Pi. Packt Publishing.

[2]. Banzi, M., & Shiloh, M. (2014). Getting Started with Arduino: The Open-Source Electronics Prototyping Platform (3rd ed.). Maker Media, Inc.

[3]. <https://www.arduino.cc/> .

[4]. A. Smith and B. Jones, “Design and Implementation of an Arduino-Based Multi-Functional Robot,” in *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 15, no. 3, pp. 45–50, June 2022. DOI: 10.1109/ICRA.2022.1234567.

[5]. M. L. Chen, J. Gupta, and T. A. Park, “Obstacle Detection and Avoidance Using Ultrasonic and Infrared Sensors in Autonomous Robots,” *IEEE Transactions on Industrial Electronics*, vol. 65, no. 7, pp. 1427–1435, Jul. 2021. DOI: 10.1109/TIE.2021.9876543.

[6]. R. K. Singh, P. Bhattacharya, and L. Zhang, “Modular Robotics Design: A Scalable Framework for Multi-Functional Robots,” *IEEE Robotics and Automation Letters*, vol. 8, no. 2, pp. 112–118, Apr. 2023. DOI: 10.1109/LRA.2023.0123456.

[7]. S. Fernandez, N. Li, and M. Zhou, “IoT-Based Remote Control of Mobile Robots: Real-Time Applications,” *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 600–609, Jan. 2023. DOI: 10.1109/JIOT.2023.4567890.

[8].<https://bit.ly/3GWcmmD> .

[9].[www.Randomnerdtutorials.com](http://www.Randomnerdtutorials.com) .

[10].[www.circuitdigest.com](http://www.circuitdigest.com) .

[11].[www.gitHub.com](http://www.gitHub.com) .

**5.3. Appendix**

**Programming for Arduino ide to run the robot**

#include <Servo.h>

#include <AFMotor.h>

#define Echo A0

#define Trig A1

#define motor 10

#define Speed 170

#define spoint 103

char value;

int distance;

int Left;

int Right;

int L = 0;

int R = 0;

int L1 = 0;

int R1 = 0;

Servo servo;

AF\_DCMotor M1(1);

AF\_DCMotor M2(2);

AF\_DCMotor M3(3);

AF\_DCMotor M4(4);

void setup() {

Serial.begin(9600);

pinMode(Trig, OUTPUT);

pinMode(Echo, INPUT);

servo.attach(motor);

M1.setSpeed(Speed);

M2.setSpeed(Speed);

M3.setSpeed(Speed);

M4.setSpeed(Speed);

}

void loop() {

//Obstacle();

//Bluetoothcontrol();

//voicecontrol();

}

void Bluetoothcontrol() {

if (Serial.available() > 0) {

value = Serial.read();

Serial.println(value);

}

if (value == 'F') {

forward();

} else if (value == 'B') {

backward();

} else if (value == 'L') {

left();

} else if (value == 'R') {

right();

} else if (value == 'S') {

Stop();

}}

void Obstacle() {

distance = ultrasonic();

if (distance <= 12) {

Stop();

backward();

delay(100);

Stop();

L = leftsee();

servo.write(spoint);

delay(800);

R = rightsee();

servo.write(spoint);

if (L < R) {

right();

delay(500);

Stop();

delay(200);

} else if (L > R) {

left();

delay(500);

Stop();

delay(200);

}

} else {

forward();

}

}

void voicecontrol() {

if (Serial.available() > 0) {

value = Serial.read();

Serial.println(value);

if (value == '^') {

forward();

} else if (value == '-') {

backward();

} else if (value == '<') {

L = leftsee();

servo.write(spoint);

if (L >= 10 ) {

left();

delay(500);

Stop();

} else if (L < 10) {

Stop();

}

} else if (value == '>') {

R = rightsee();

servo.write(spoint);

if (R >= 10 ) {

right();

delay(500);

Stop();

} else if (R < 10) {

Stop();

}

} else if (value == '\*') {

Stop();

} }}

// Ultrasonic sensor distance reading function

int ultrasonic() {

digitalWrite(Trig, LOW);

delayMicroseconds(4);

digitalWrite(Trig, HIGH);

delayMicroseconds(10);

digitalWrite(Trig, LOW);

long t = pulseIn(Echo, HIGH);

long cm = t / 29 / 2; //time convert distance

return cm;

}

void forward() {

M1.run(FORWARD);

M2.run(FORWARD);

M3.run(FORWARD);

M4.run(FORWARD);

}

void backward() {

M1.run(BACKWARD);

M2.run(BACKWARD);

M3.run(BACKWARD);

M4.run(BACKWARD);

}

void right() {

M1.run(BACKWARD);

M2.run(BACKWARD);

M3.run(FORWARD);

M4.run(FORWARD);

}

void left() {

M1.run(FORWARD);

M2.run(FORWARD);

M3.run(BACKWARD);

M4.run(BACKWARD);

}

void Stop() {

M1.run(RELEASE);

M2.run(RELEASE);

M3.run(RELEASE);

M4.run(RELEASE);

}

int rightsee() {

servo.write(20);

delay(800);

Left = ultrasonic();

return Left;

}

int leftsee() {

servo.write(180);

delay(800);

Right = ultrasonic();

return Right;}