**RC CAMERA ROVER**

**DIY PROJECT REPORT**

**SECTION-14**

**GROUP ID-11**

TEAM MEMBERS

UJJWAL SHARMA

23AG10045

PARAG MAHADEO CHIMANKAR

23CS10049

BHUKYA SHASHI KUMAR

23EE10014

AYUSH GUPTA

23MI10012

**PROBLEM STATEMENT**

There are some places where the human head can’t go or risky. Just think of your ceiling with many electrical connections. What can we do to check such places for faults?

**MOTIVE**

* To explore inaccessible or hard-to-reach areas such as ceilings or high shelves.
* Remote surveillance and exploration of these areas without risking human safety or requiring physical access.
* Easy to operate and control

Our project aims to achieve the following objectives:

* Develop a robust RC car platform capable of traversing ceilings and other surfaces.
* Integrate a high-quality camera module for real-time video streaming.
* Implement wireless communication using ESP32 for remote control and data transmission.
* Ensure user-friendly operation through intuitive control interfaces.
* Explore potential applications in surveillance, inspection, and education.

**INTRODUCTION**

* RC rover is a 4-wheeled car whose movement is powered by gear motors.
* The rover is connected to Wifi by an ESP32 Wifi module and is controlled by the Wifi network-connected web page.
* The Pan Tilt Assembly is made for the movement of the camera.
* A Lithium-ion battery powers the car.

**WORK METHODOLOGY**

The ESP starts and requests a connection to the given WiFi network.

After the connection, a web page using the same Wi-Fi network is created.

Users access the network to send commands through the HTML webpage

The ESP receives these Signals and performs the allotted functions as specified.

**COMPONENTS USED**

1. L298N 2A Dual H Bridge Motor Driver Module

* Logical voltage: 5V
* Drive voltage: 5V-35V
* Logical current: 0-36mA
* Drive current: 2A (max. single bridge)
* Max power: 25W
* Dimensions: 44 x 44 x 28 (LxWxH)mm
* Weight: 26 gm

2. 2 Axis Servo Pan Tilt Assembly for SG90 MG90

* Two axial platform

3. Mini Servo Motor SG90 180 Rotation

* Weight: 9 g
* Dimension: 22.2 x 11.8 x 31 mm approx.
* Stall torque: 1.8 kgf·cm
* Operating speed: 0.1 s/60 degree
* Operating voltage: 4.8 V (~5V)
* Dead band width: 10 μs
* Temperature range: 0 oC – 55 oC

4. ESP32 WROOM Devkit V1

* Microcontroller: Tensilica 32-bit Single-/Dual-core CPU Xtensa LX6
* Operating Voltage: 3.3V
* Input Voltage: 7-12V
* Digital I/O Pins (DIO): 25
* Analog Input Pins (ADC): 6
* Analog Outputs Pins (DAC): 2
* UARTs: 3
* SPIs: 2
* I2Cs: 3
* Flash Memory: 4 MB
* SRAM: 520 KB
* Clock Speed: 240 Mhz

5. 4-wheel Robot Car Chassis Kit

6. Jumper Wire and Connecting wire

7. 9V HW Battery,Li ion Battery and Battery Holder

8. AMS1117 5V Step Down Power Supply Module

* Input: DC 6.5V – 12V (input voltage must be more than output voltage by least 1V)
* Output: 5.0V (+-0.05v error), 800mA (load current cannot exceed 800ma)
* Onboard power indicator to indicate power is turned on: Red LED.
* Small Compact Size, 4Pin Type, Breadboard Friendly.

Other components to be

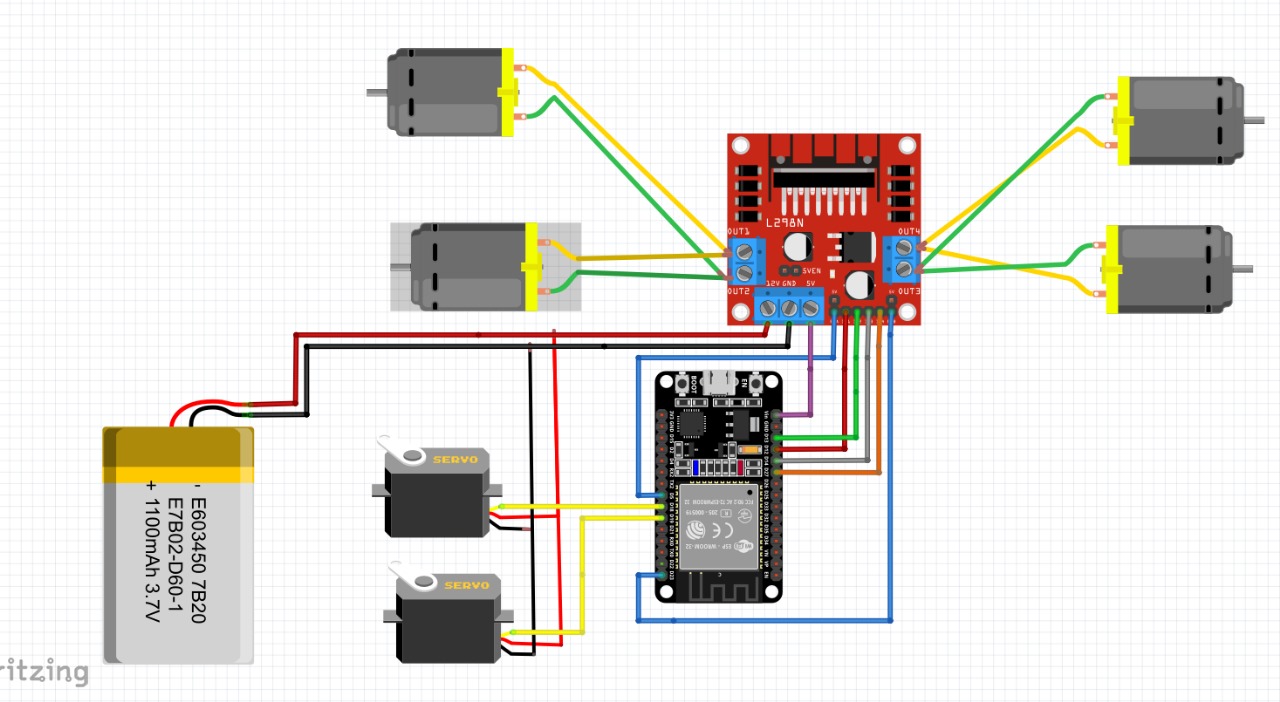
9. ESP32 Cam Module

* SPI Flash: default 32Mbit
* RAM: built-in 520 KB+external 4MPSRAM
* IO port: 9
* Image Output Format: JPEG( OV2640 support only ), BMP, GRAYSCALE
* Spectrum Range: 2412 ~2484MH
* Power consumption: Turn off the flash: 180mA@5V
  + Turn on the flash and adjust the brightness to the maximum: 310mA@5V
  + Deep-sleep: the lowest power consumption can reach 6mA@5V
  + Moderm-sleep: up to 20mA@5V
  + Light-sleep: up to 6.7mA@5V
* Transmit Power: 802.11b: 17±2 dBm (@11Mbps);
  + 802.11g: 14±2 dBm (@54Mbps);
  + 802.11n: 13±2 dBm (@MCS7)
* Receiving Sensitivity: CCK, 1 Mbps : -90dBm;
  + CCK, 11 Mbps: -85dBm;
  + 6 Mbps (1/2 BPSK): -88dBm;
  + 54 Mbps (3/4 64-QAM): -70dBm;
  + MCS7 (65 Mbps, 72.2 Mbps): -67dBm

10.OV7670 Cam

11. Arduino UNO

**CIRCUIT DIAGRAM**



**CODE**

**#include <WiFi.h>**

**#include <AsyncTCP.h>**

**#include <ESPAsyncWebServer.h>**

**#include <iostream>**

**#include <sstream>**

**#include <ESP32Servo.h>**

**const char\* ssid = "ParagC";**

**const char\* password = "12345678";**

**// Motor driver pins**

**const int enA = 5; // Speed control for motor A**

**const int enB = 23; // Speed control for motor B**

**const int in1 = 12;**

**const int in2 = 13;**

**const int in3 = 14;**

**const int in4 = 27;**

**// Servo pins**

**const int servoPin1 = 19;**

**const int servoPin2 = 18;**

**Servo servo1;**

**Servo servo2;**

**// Initialize AsyncWebServer object on port 80**

**AsyncWebServer server(80);**

**void setup() {**

**Serial.begin(115200);**

**// Connect to Wi-Fi**

**WiFi.begin(ssid, password);**

**while (WiFi.status() != WL\_CONNECTED) {**

**delay(1000);**

**Serial.println("Connecting to WiFi...");**

**}**

**Serial.println("Connected to WiFi");**

**// Print IP address to Serial Monitor**

**Serial.print("IP Address: ");**

**Serial.println(WiFi.localIP());**

**// Configure motor driver pins as outputs**

**pinMode(enA, OUTPUT);**

**pinMode(in1, OUTPUT);**

**pinMode(in2, OUTPUT);**

**pinMode(enB, OUTPUT);**

**pinMode(in3, OUTPUT);**

**pinMode(in4, OUTPUT);**

**// Attach servos to pins**

**servo1.attach(servoPin1);**

**servo2.attach(servoPin2);**

**// Print generated web page to Serial Monitor for debugging**

**// Serial.println(generateWebPage());**

**// Start the web server**

**server.on("/", HTTP\_GET, [](AsyncWebServerRequest \*request){**

**request->send(200, "text/html", generateWebPage());**

**});**

**server.on("/motorAForward", HTTP\_GET, [](AsyncWebServerRequest \*request){**

**motorAForward();**

**request->send(200);**

**});**

**server.on("/motorBBackward", HTTP\_GET, [](AsyncWebServerRequest \*request){**

**motorABackward();**

**request->send(200);**

**});**

**server.on("/motorBForward", HTTP\_GET, [](AsyncWebServerRequest \*request){**

**motorBForward();**

**request->send(200);**

**});**

**server.on("/motorABackward", HTTP\_GET, [](AsyncWebServerRequest \*request){**

**motorBBackward();**

**request->send(200);**

**});**

**server.on("/stopMotors", HTTP\_GET, [](AsyncWebServerRequest \*request){**

**stopMotors();**

**request->send(200);**

**});**

**server.on("/turnServo1Positive10Degrees", HTTP\_GET, [](AsyncWebServerRequest \*request){**

**turnServo1Positive10Degrees();**

**request->send(200);**

**});**

**server.on("/turnServo1Negative10Degrees", HTTP\_GET, [](AsyncWebServerRequest \*request){**

**turnServo1Negative10Degrees();**

**request->send(200);**

**});**

**server.on("/turnServo2Positive10Degrees", HTTP\_GET, [](AsyncWebServerRequest \*request){**

**turnServo2Positive10Degrees();**

**request->send(200);**

**});**

**server.on("/turnServo2Negative10Degrees", HTTP\_GET, [](AsyncWebServerRequest \*request){**

**turnServo2Negative10Degrees();**

**request->send(200);**

**});**

**server.begin();**

**}**

**void loop() {**

**}**

**String generateWebPage() {**

**String page = "<!DOCTYPE html><html><head><title>ESP32 Motor Control</title></head><body>";**

**page += "<h1>ESP32 Motor Control</h1>";**

**page += "<h2>Motors:</h2>";**

**page += "<button ontouchstart='sendCommand(\"/motorAForward\")'>Forward</button><br>";**

**page += "<button ontouchstart='sendCommand(\"/motorABackward\")'>Right</button><br>";**

**page += "<button ontouchstart='sendCommand(\"/motorBForward\")'>Left</button><br>";**

**page += "<button ontouchstart='sendCommand(\"/motorBBackward\")'>Backward</button><br>";**

**page += "<button ontouchstart='sendCommand(\"/stopMotors\")'>Stop Motors</button><br>";**

**page += "<h2>Servo 1:</h2>";**

**page += "<button ontouchstart='sendCommand(\"/turnServo1Positive10Degrees\")'>Turn +20°</button><br>";**

**page += "<button ontouchstart='sendCommand(\"/turnServo1Negative10Degrees\")'>Turn -20°</button><br>";**

**page += "<h2>Servo 2:</h2>";**

**page += "<button ontouchstart='sendCommand(\"/turnServo2Positive10Degrees\")'>Turn +20°</button><br>";**

**page += "<button ontouchstart='sendCommand(\"/turnServo2Negative10Degrees\")'>Turn -20°</button><br>";**

**page += "<script>";**

**page += "function sendCommand(command) {";**

**page += " fetch(command);"; // Send a GET request to the specified command**

**page += "}";**

**page += "</script>";**

**page += "</body></html>";**

**return page;**

**}**

**void motorAForward() {**

**// A forward**

**digitalWrite(in1, HIGH);**

**digitalWrite(in2, LOW);**

**analogWrite(enA, 255);**

**// B forward**

**digitalWrite(in3, HIGH);**

**digitalWrite(in4, LOW);**

**analogWrite(enB, 255);**

**}**

**void motorABackward() {**

**// A Backward**

**digitalWrite(in3, LOW);**

**digitalWrite(in4, HIGH);**

**analogWrite(enB, 255);**

**// B Backward**

**digitalWrite(in1, LOW);**

**digitalWrite(in2, HIGH);**

**analogWrite(enA, 255);**

**}**

**void motorBForward() {**

**// making left B Forward and A LOW**

**digitalWrite(in3, HIGH);**

**digitalWrite(in4, LOW);**

**digitalWrite(in1, LOW);**

**digitalWrite(in2, LOW);**

**analogWrite(enA, 255);**

**analogWrite(enB, 255);**

**}**

**void motorBBackward() {**

**// making right A Forward and B LOW**

**digitalWrite(in1, HIGH);**

**digitalWrite(in2, LOW);**

**digitalWrite(in3, LOW);**

**digitalWrite(in4, LOW);**

**analogWrite(enA, 255);**

**analogWrite(enB, 255);**

**}**

**void stopMotors() {**

**digitalWrite(in1, LOW);**

**digitalWrite(in2, LOW);**

**digitalWrite(in3, LOW);**

**digitalWrite(in4, LOW);**

**analogWrite(enA, 0);**

**analogWrite(enB, 0);**

**}**

**void turnServo1Positive10Degrees() {**

**int currentPosition = servo1.read(); // Get the current position of servo 1**

**int targetPosition = currentPosition + 20; // Calculate the target position by adding 20 degrees**

**// Make sure the target position stays within the valid range of 0 to 180 degrees**

**if (targetPosition > 180) {**

**targetPosition = 180;**

**}**

**// Set servo 1 to the target position**

**servo1.write(targetPosition);**

**}**

**void turnServo1Negative10Degrees() {**

**int currentPosition = servo1.read(); // Get the current position of servo 1**

**int targetPosition = currentPosition - 20; // Calculate the target position by subtracting 20 degrees**

**// Make sure the target position stays within the valid range of 0 to 180 degrees**

**if (targetPosition < 0) {**

**targetPosition = 0;**

**}**

**// Set servo 1 to the target position**

**servo1.write(targetPosition);**

**}**

**void turnServo2Positive10Degrees() {**

**int currentPosition = servo2.read(); // Get the current position of servo 2**

**int targetPosition = currentPosition + 20; // Calculate the target position by adding 20 degrees**

**// Make sure the target position stays within the valid range of 0 to 180 degrees**

**if (targetPosition > 180) {**

**targetPosition = 180;**

**}**

**// Set servo 2 to the target position**

**Serial.print("Turning servo 2 to position: ");**

**// Debugging statement**

**servo2.write(targetPosition);**

**}**

**void turnServo2Negative10Degrees() {**

**int currentPosition = servo2.read(); // Get the current position of servo 2**

**int targetPosition = currentPosition - 20; // Calculate the target position by subtracting 20 degrees**

**// Make sure the target position stays within the valid range of 0 to 180 degrees**

**if (targetPosition < 0) {**

**targetPosition = 0;**

**}**

**// Set servo 2 to the target position**

**Serial.print("Turning servo 2 to position: ");**

**// Debugging statement**

**servo2.write(targetPosition);**

**}**

**SAMPLE PHOTO OF THE WEBPAGE GENERATED**



**CHALLENGES FACED**

1. We planned to work with ESP32 CAM module, but it didn’t the module malfunction.

2. Working with Arduino-supported OV7670 CAM is not feasible for the project.

3. We have to start the code from scratch and understand the connections of components and pins due to the unavailability of the reference code.

4. Lack of knowledge of Web development.

5. Power Supply Mismanagement to ESP and Servos

**LEARNINGS**

1. Understanding Pins of ESP32 and L298N Motor driver.

2. Programming ESP32

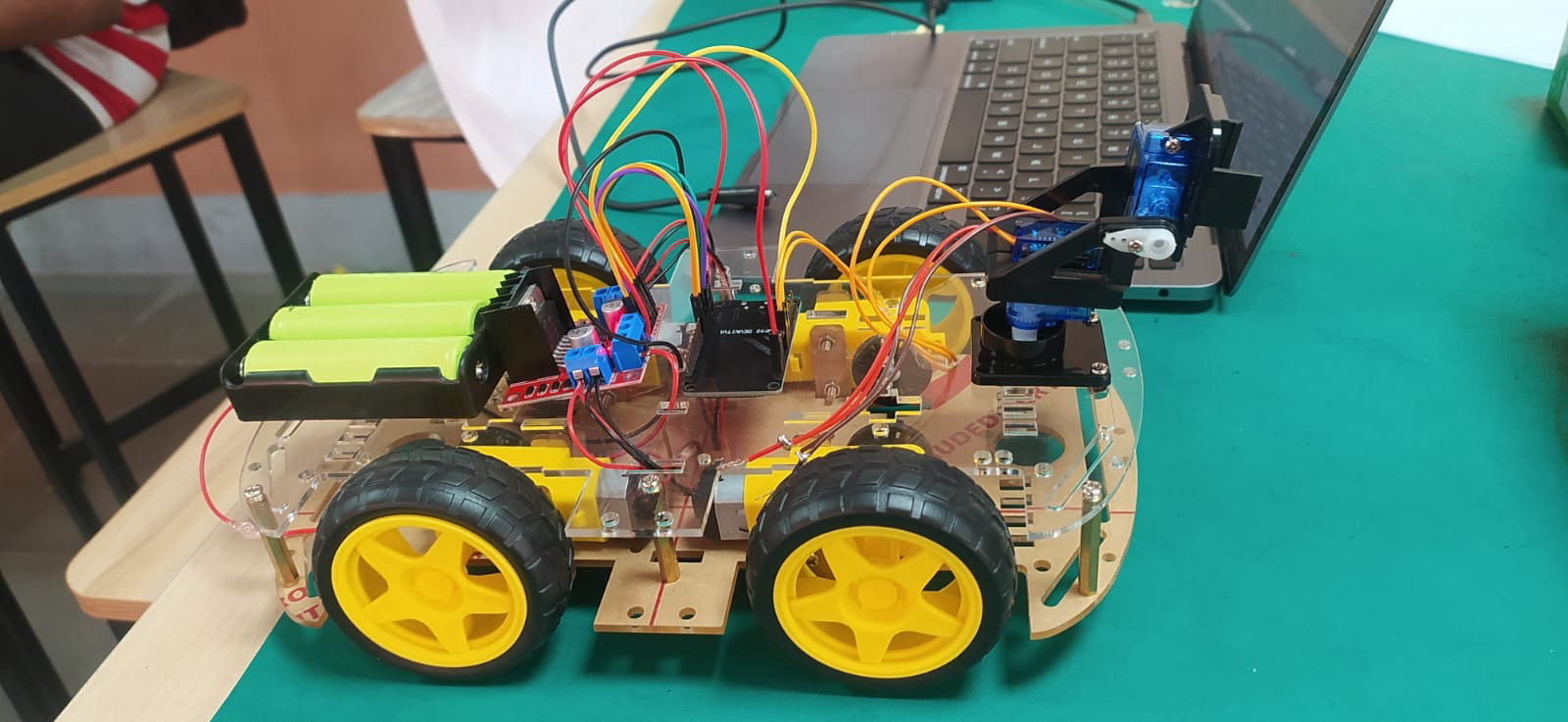
3. Learning Wifi connectivity with Web Server.

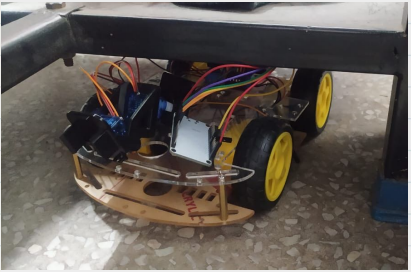
4. Understanding the workings of Servo Motors and Gear Motors.

5. Data(Video) transfer via Wifi through CAM Modules.

**FUTURE DIRECTION**

1. Making the RC car more compact and sturdy.
2. Controlling the gear motor speed.
3. Pan Tilt is to be controlled by a scrollbar.
4. Increased range of wifi connectivity.
5. The camera is to be intact with the car and connected to the Wi-Fi module.





**CONCLUSION**

The Model solves the daily life problem of survelling through compact dark places. Scaling down the size will increase its scale of use.

**RESOURCES**

1. Youtube Videos by hashincludeelectronics,and others
2. AI tools( Chat GPT, Gemini)
3. W3school website for webpage related queries.
4. Projects by arduino website