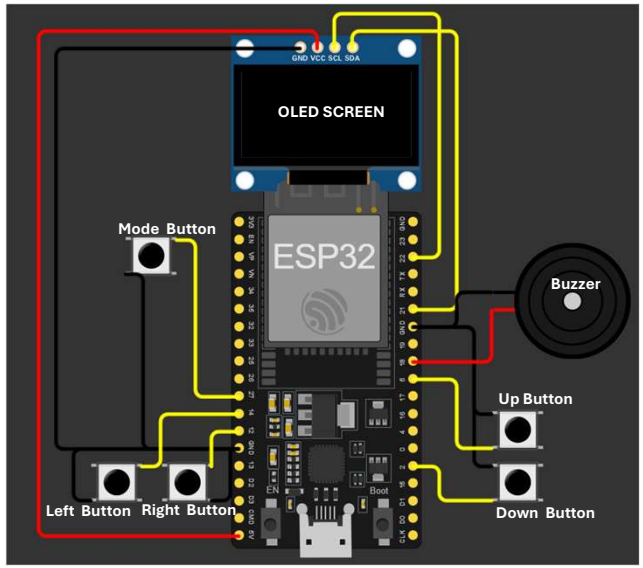
Mini RC Car with obstacle avoidance

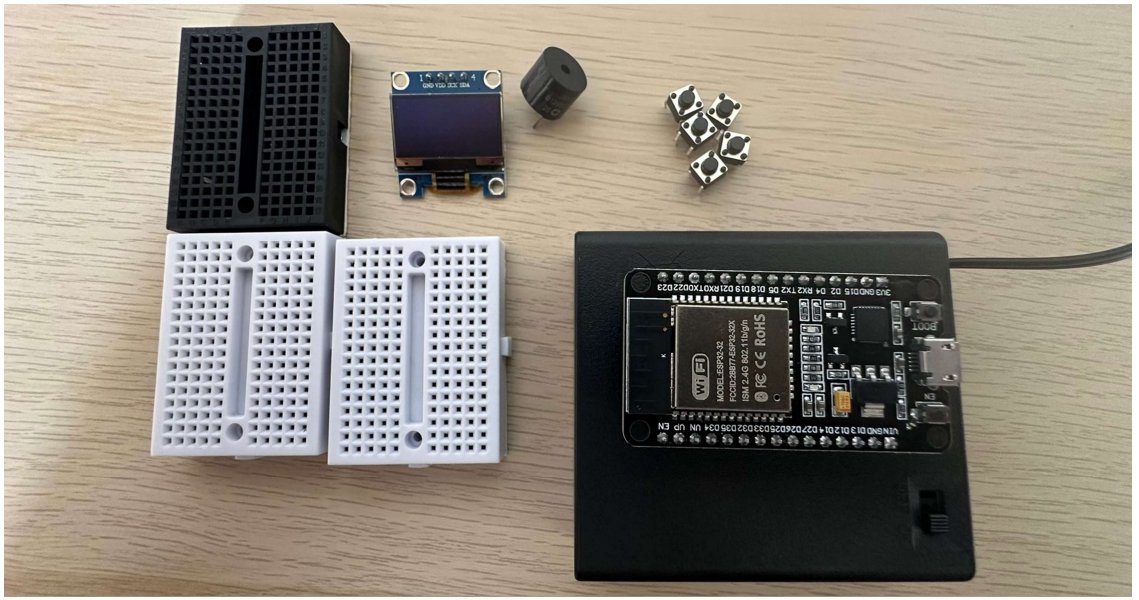
How will it work

It will be comprised of 2 ESP32s, one for the remote control, and another for the actual RC car, and they will communicate with each other using the ESP-NOW Protocol

The remote

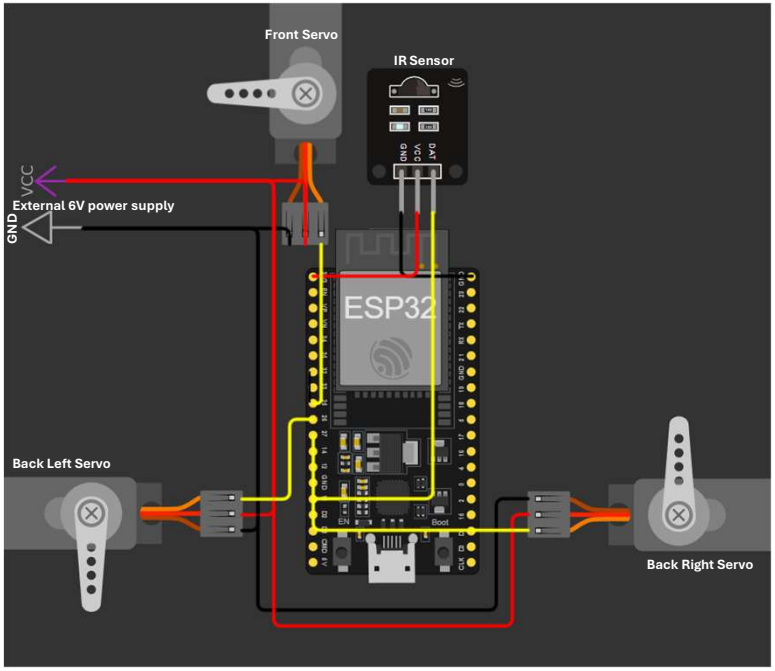
- The remote will be comprised of 3xmini breadboards, 1xESP32, 1x0.96 inch OLED screen, a buzzer, and 5xmini buttons (4 directional ones and 1 to change the mode of the vehicle), a battery pack to power it, and jumper wires to connect everything

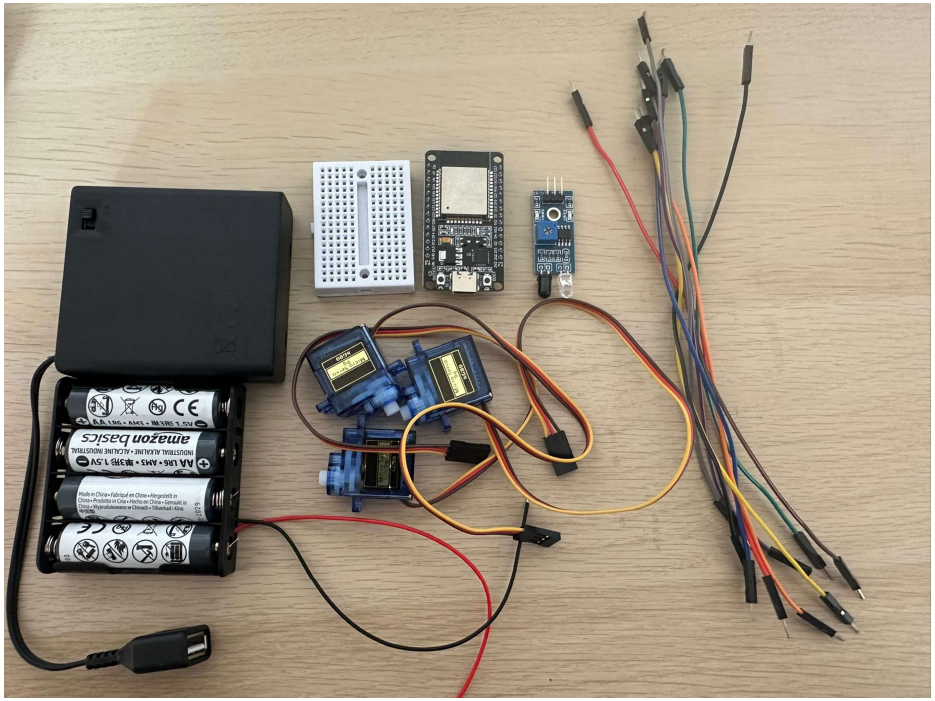




The vehicle

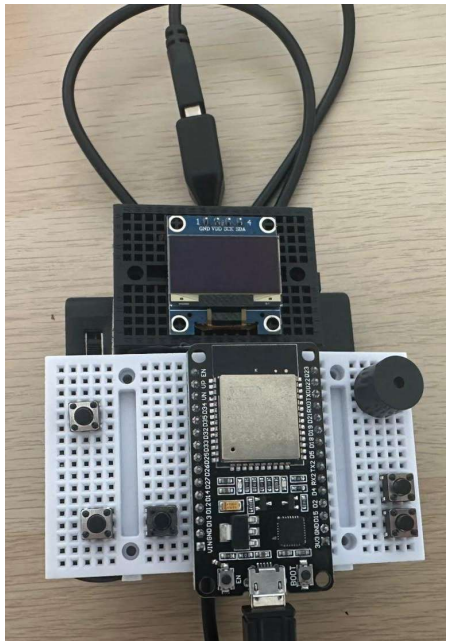
- The vehicle will have 1xESP32, 3xSG90 360degree servo motors, 1xmini breadboard, 3x3D printed wheels, 2xbattery packs (one to power the ESP32, and another to power the servos), some jumper wires to connect everything together, an infrared distance sensor, and a 3D printed Chassis to contain everything

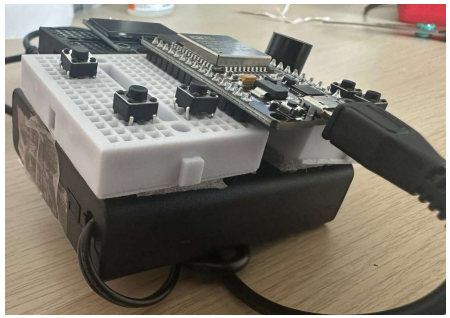




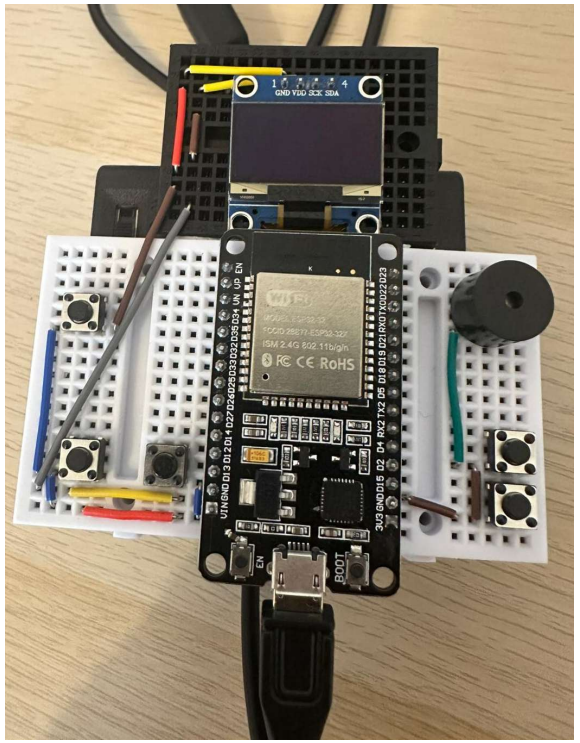
Making the remote

1 – First, I placed everything where it needed to go on the breadboards and glued the breadboards together and attached the breadboards to the battery pack to make it portable

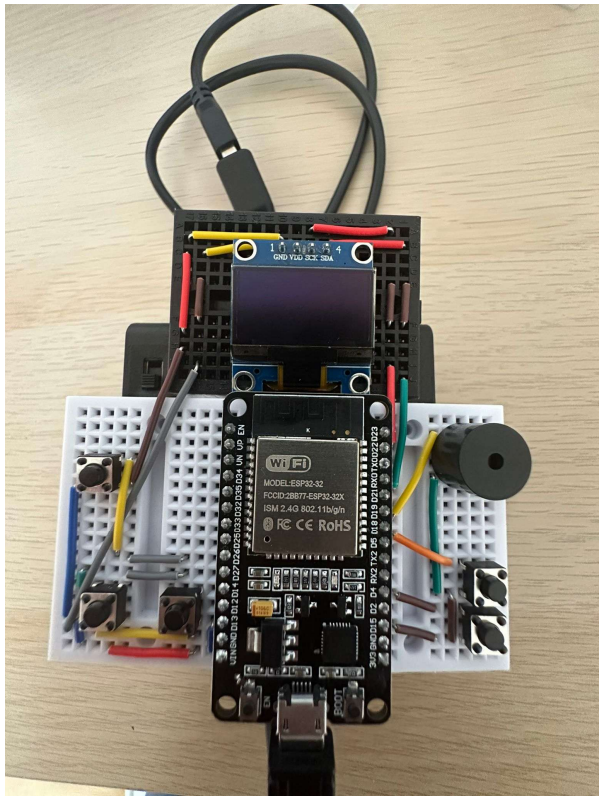




2 – Then I connected everything to ground using jumper wires and I also wired the OLED Screen power supply to the 5V pin of the ESP32



3 – Then I connected the rest of the wires which connect the different components to the main ESP32



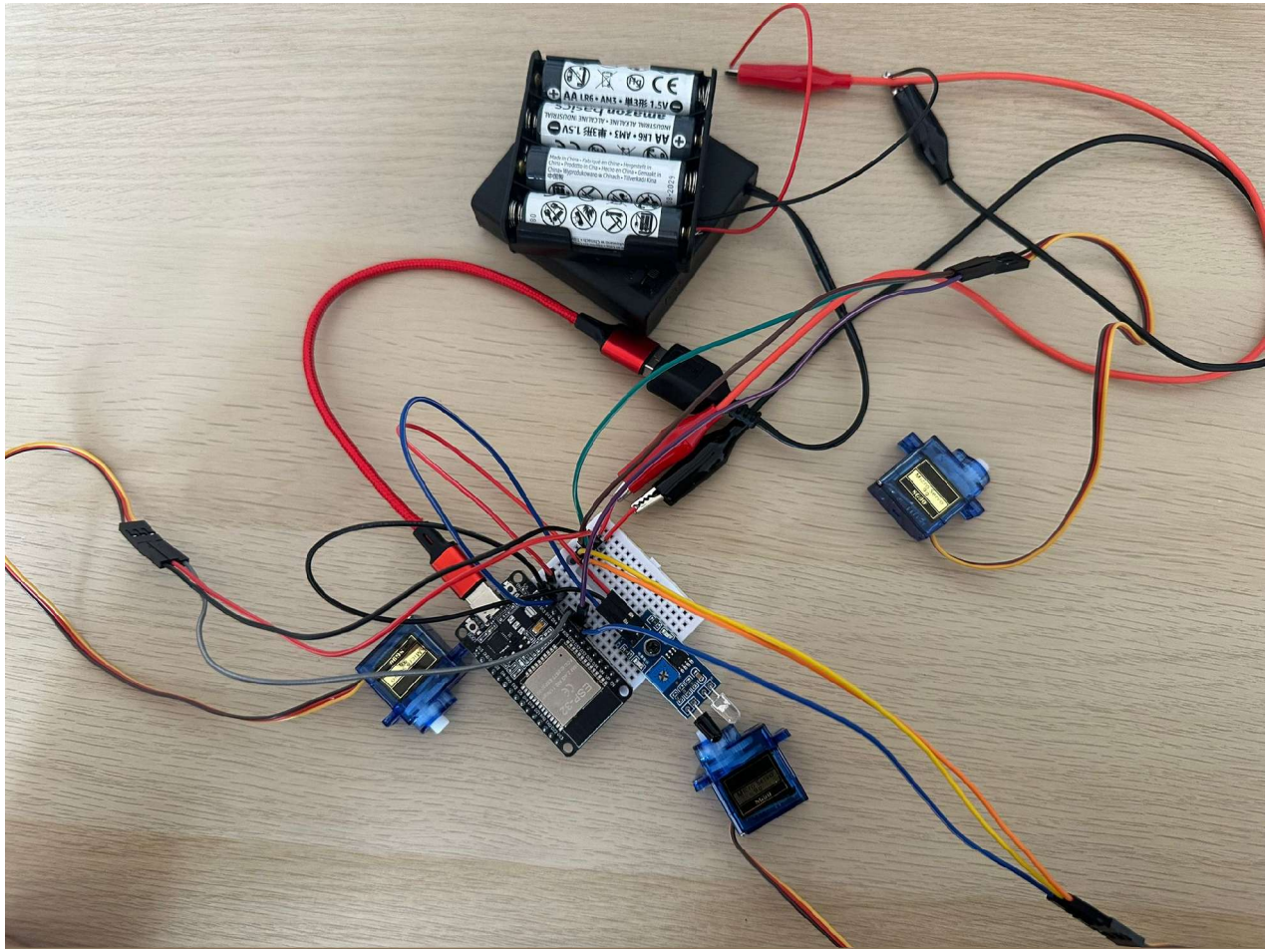
Making the vehicle

1 - I wired everything together first.

2 - I then calibrated each servo to make sure I get the value for which it is at its max forward speed, max backward speed, and completely stationary as these values do vary when they are produced as they are mass produced and it depends on their placement. The values are:

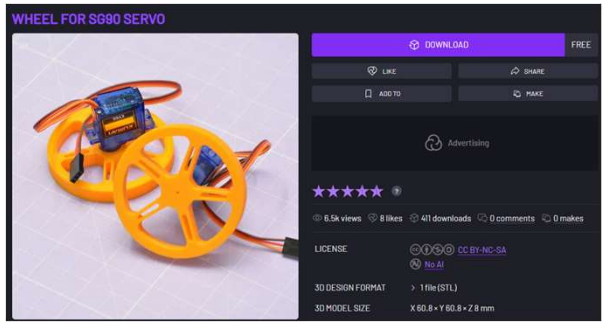
Servo	Placement	Forward	Reverse	Stationary
Rear Left		180	0	92
Rear Right		180	0	94
Front		0	180	93

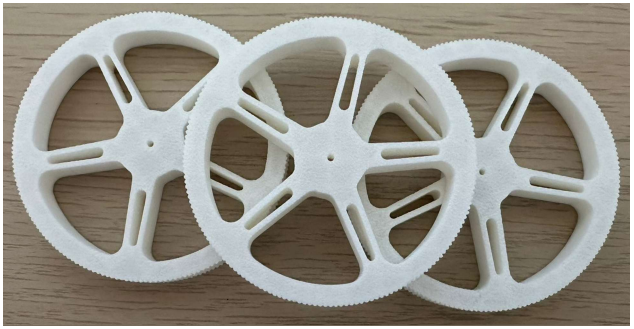
These values will also probably need to be changed as once the car is built as they need to be calibrated even more.



Making the vehicle – 3D printing the chassis:

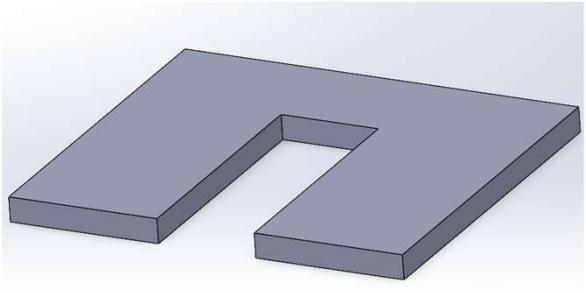
First I 3D printed some wheels which I found online and attached them to the servo motors:



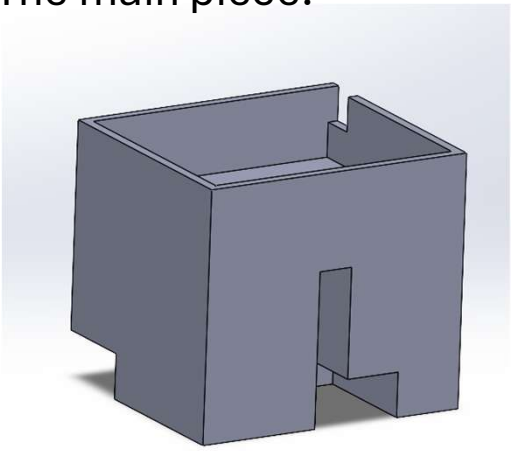


I then designed the main chassis by splitting it into 2 parts which I will glue together (The main piece and the base):

The base:



The main piece:

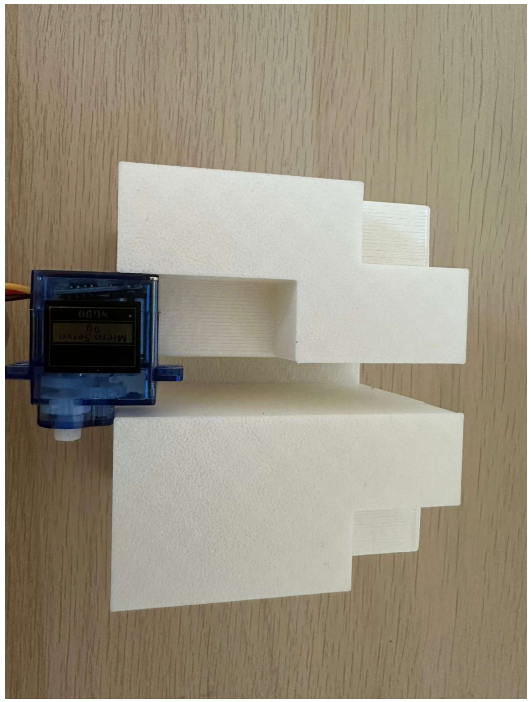


Making the vehicle - continued

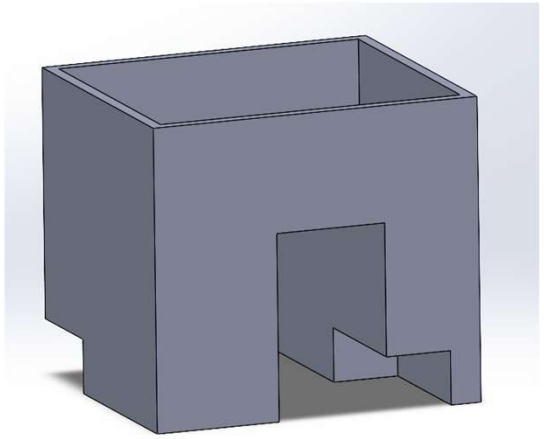
I then 3D printed the main piece and realised that I had made some errors:

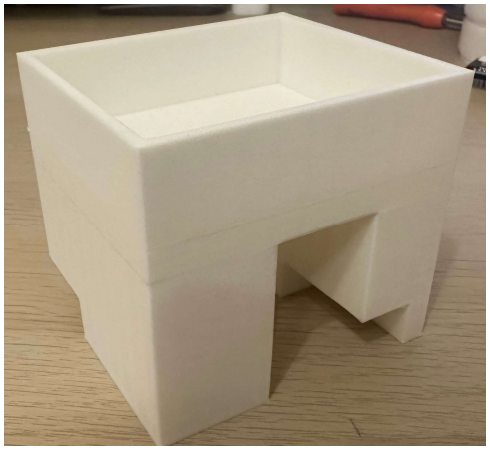
Firstly, I didn't really need the hole for the battery pack wire as it can bend upwards which gives the battery pack a more secure fit:

Secondly, I forgot to account for the gearbox of the front servo motor as it faces into the chassis not out of it like the rear servo motors so I need to adjust the hole for the wheel and the hole for the motor:



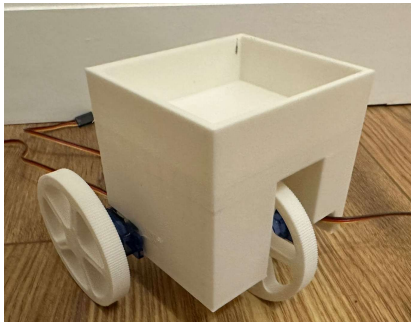
I then redesigned it and 3D printed it again:

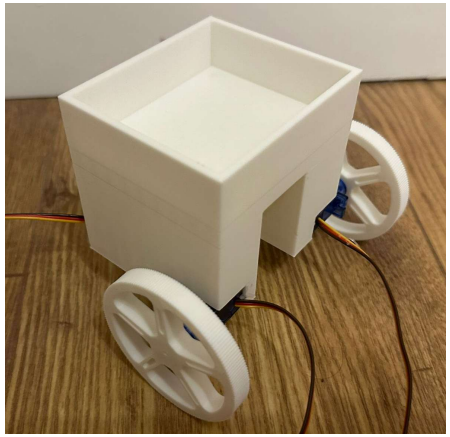




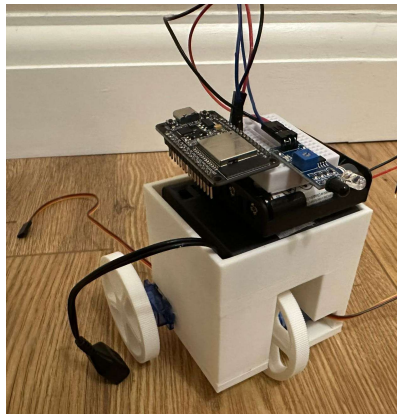
Making the vehicle – continued:

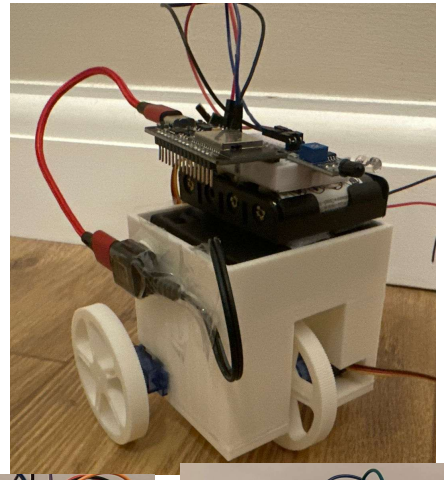
First I unplugged the servo motors and glued them in whilst also gluing the wheels to the servos:

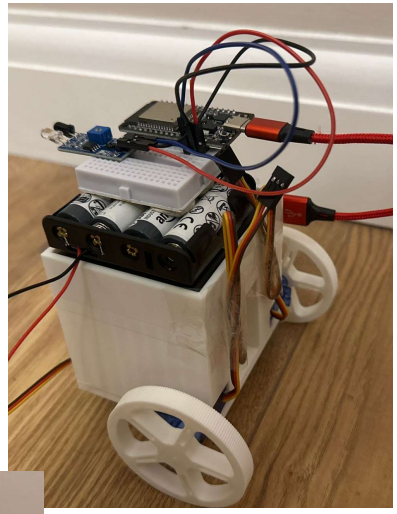


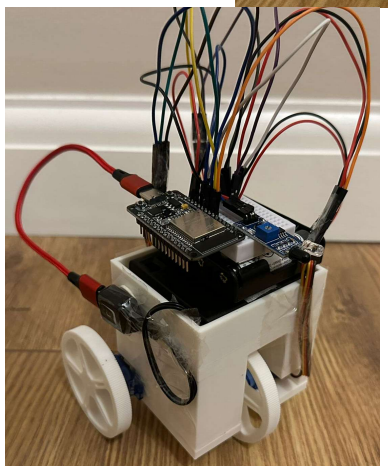


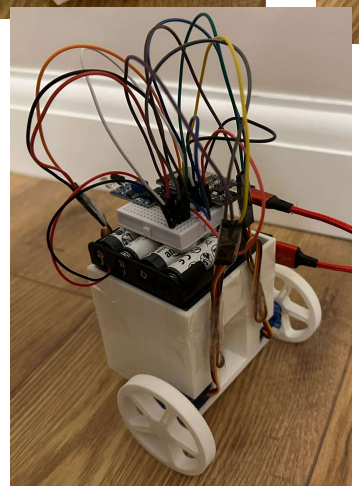
Then I attached the motor battery pack, ESP32, and breadboards together using sticky tape and placed it into the chassis and wired everything back up:





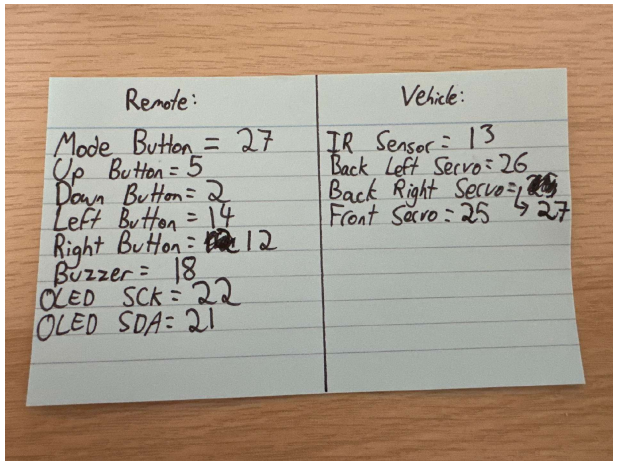






Coding it:

To make Coding easier I first noted which pins all of the components are connected to:



I then looked through my pre-existing code to find the code for button inputs, OLED display outputs, buzzer outputs, and servo motor outputs. I then went online to find out code on how to allow the 2 ESP32s to communicate via Wi-Fi and how to code the Infrared distance sensor as an input. Then, I coded both the remote and vehicle with some assistance from AI. I did run into some errors throughout the process, but these were mostly syntax errors (eg: missing off a semicolon at the end of a line) as I am not used to coding in Arduino Language (a simplified version of C/C++). The following code will also be published in the GitHub branch,

First I coded the remote:

ObstacleAvoidaceCar_RemoteCode | Arduino IDE 2.3.6

File Edit Sketch Tools Help

ESP32 Dev Module

ObstacleAvoidaceCar_RemoteCode.ino

```ino
#include <esp_now.h>
#include <WiFi.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
#define OLED_RESET    -1
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);

// Button pins
#define UP_BUTTON     5
#define DOWN_BUTTON   2
#define LEFT_BUTTON   14
#define RIGHT_BUTTON  12
#define MODE_BUTTON   27

// Buzzer pin
#define BUZZER_PIN    18

// OLED pins
#define OLED_SDA 21
#define OLED_SCL 22

// Struct to send data
typedef struct struct_message {
  int command;
} struct_message;

struct_message myData;

// Receiver MAC address (replace with your robot's ESP32 MAC)
uint8_t broadcastAddress[] = {0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF};

esp_now_peer_info_t peerInfo;

bool manualMode = true;
```

```ino
void setup() {
  // Init Serial Monitor
  Serial.begin(115200);

  // Init buttons
  pinMode(UP_BUTTON, INPUT_PULLUP);
  pinMode(DOWN_BUTTON, INPUT_PULLUP);
  pinMode(LEFT_BUTTON, INPUT_PULLUP);
  pinMode(RIGHT_BUTTON, INPUT_PULLUP);
  pinMode(MODE_BUTTON, INPUT_PULLUP);

  // Init buzzer
  pinMode(BUZZER_PIN, OUTPUT);
  digitalWrite(BUZZER_PIN, LOW);

  // Init OLED
  Wire.begin(OLED_SDA, OLED_SCL);
  if (!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
    Serial.println(F("SSD1306 allocation failed"));
    for (;;);
  }
  display.clearDisplay();
  display.setTextSize(1);
  display.setTextColor(SSD1306_WHITE);
  display.setCursor(0, 0);
  display.println("Remote Starting...");
  display.display();

  // Set device as a Wi-Fi Station
  WiFi.mode(WIFI_STA);

  // Init ESP-NOW
  if (esp_now_init() != ESP_OK) {
    Serial.println("Error initializing ESP-NOW");
    return;
  }

  // Register peer
  memcpy(peerInfo.peer_addr, broadcastAddress, 6);
  peerInfo.channel = 0;
  peerInfo.encrypt = false;

  if (esp_now_add_peer(&peerInfo) != ESP_OK) {
    Serial.println("Failed to add peer");
    return;
  }
}

void loop() {
  // Mode toggle
  if (digitalRead(MODE_BUTTON) == LOW) {
    manualMode = !manualMode;
    delay(300); // debounce
    display.clearDisplay();
    display.setCursor(0, 0);
    display.println(manualMode ? "Mode: Manual" : "Mode: Auto");
    display.display();
  }
```

```ino
void loop() {
  // Mode toggle
  if (digitalRead(MODE_BUTTON) == LOW) {
    manualMode = !manualMode;
    delay(300); // debounce
    display.clearDisplay();
    display.setCursor(0, 0);
    display.println(manualMode ? "Mode: Manual" : "Mode: Auto");
    display.display();
  }

  if (manualMode) {
    // Manual mode commands
    if (digitalRead(UP_BUTTON) == LOW) {
      myData.command = 1; // Forward
      sendData();
      digitalWrite(BUZZER_PIN, HIGH);
    }
    else if (digitalRead(DOWN_BUTTON) == LOW) {
      myData.command = 2; // Backward
      sendData();
      digitalWrite(BUZZER_PIN, HIGH);
    }
    else if (digitalRead(LEFT_BUTTON) == LOW) {
      myData.command = 3; // Left
      sendData();
      digitalWrite(BUZZER_PIN, HIGH);
    }
    else if (digitalRead(RIGHT_BUTTON) == LOW) {
      myData.command = 4; // Right
      sendData();
      digitalWrite(BUZZER_PIN, HIGH);
    }
    else {
      myData.command = 0; // Stop
      sendData();
      digitalWrite(BUZZER_PIN, LOW);
    }
  }
  else {
    // Auto mode
    myData.command = 5; // Auto
    sendData();
    digitalWrite(BUZZER_PIN, LOW);
  }
}

void sendData() {
  esp_now_send(broadcastAddress, (uint8_t *) &myData, sizeof(myData));
}
```

Then I coded the vehicle:

ObstacleAvoidaceCar_CarCode | Arduino IDE 2.3.6

File Edit Sketch Tools Help

ESP32 Dev Module

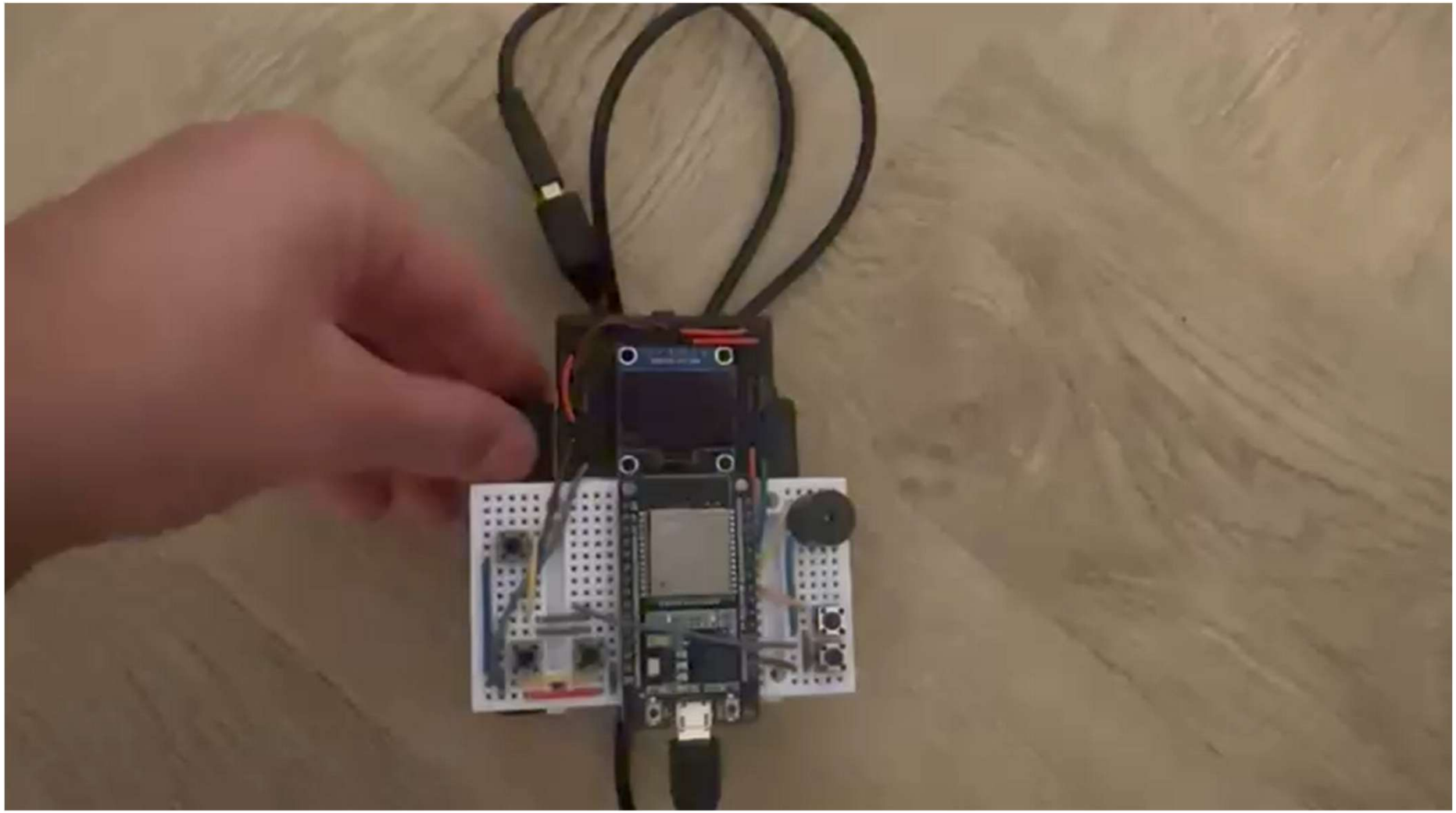
ObstacleAvoidaceCar_CarCode.ino

```ino
#include <ESP32Servo.h>
#include <WiFi.h>
#include <esp_now.h>

#define IR 13

Servo f;
Servo br;
Servo bl;

int command = 0;
int x = 0;
int y = 0;
unsigned long irLowStart = 0;
int i = 100;

void forward() {
  br.write(5);
  bl.write(180);
  f.write(0);
}
void reverse() {
  f.write(180);
  br.write(180);
  bl.write(24);
}
void left() {
  br.write(0);
  bl.write(0);
  f.write(180);
}
void right() {
  br.write(180);
  bl.write(180);
  f.write(180);
}
void stationary() {
  br.write(94);
  bl.write(92);
  f.write(93);
}
void onReceive(const esp_now_recv_info_t *info, const uint8_t *incomingData, int len) {
  memcpy(&command, incomingData, sizeof(command));
  Serial.print("Received command: ");
  Serial.println(command);
}
```

```ino
void setup() {
  pinMode(IR, INPUT);

  br.attach(27);
  bl.attach(26);
  f.attach(25);

  Serial.begin(115200);

  WiFi.mode(WIFI_STA);
  WiFi.disconnect();

  if (esp_now_init() != ESP_OK) {
    Serial.println("ESP-NOW init failed");
    return;
  }

  esp_now_register_recv_cb(onReceive);

}
```

```ino
void loop() {
  int IRvalue = digitalRead(IR);
  if (irLowStart == 0) irLowStart = millis();
  else if (millis() - irLowStart >= i) {
    reverse();
    delay(1000);
    stationary();
    delay(250);
    if (y == 1) {
      int number = random(1,3);
      if (number == 1) {
        right();
        delay(random(1000,2000));
        stationary();
        delay(250);
      }
      else {
        left();
        delay(random(1000,2000));
        stationary();
        delay(250);
      }
    }
    irLowStart = millis();
  }
  else if (IRvalue == LOW) {
    if (irLowStart == 0) irLowStart = millis();
  }
  else {
    if (command == 1) {
      stationary();
      y = 0;
      i = 100;
    }
    else if (command == 2) {
      forward();
      i = 10;
      y = 1;
    }
    else if (command == 3) {
      forward();
    }
    else if (command == 4) {
      reverse();
    }
    else if (command == 5) {
      left();
    }
    else if (command == 6) {
      right();
    }
    else if (command == 7) {
      stationary();
    }
    irLowStart = 0;
  }
}
```

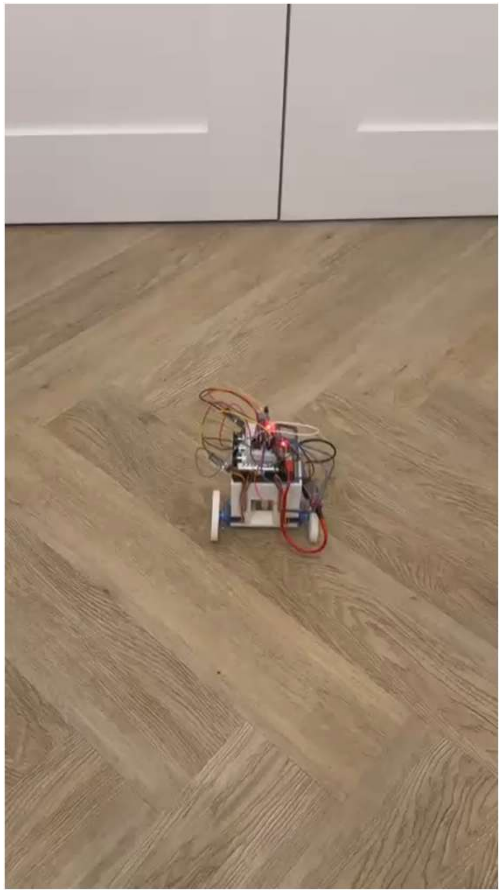
Checking that the remote is fully functioning:



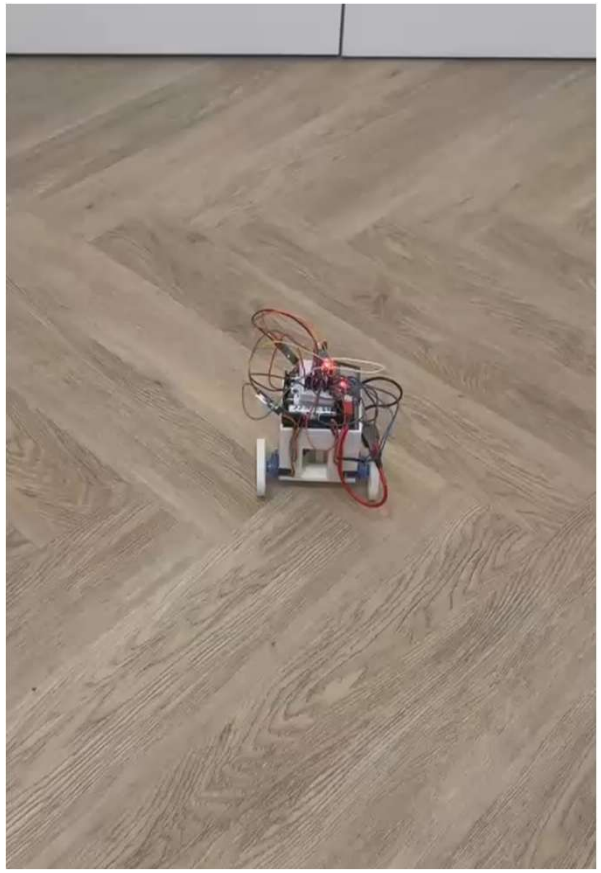
Testing the remote:

Through testing the remote I realised that sometimes the IR sensor was misfiring and the car wasn't travelling straight when it was meant to so after some calibration and tinkering with the code, I managed to get the car to function as it is intended to, as seen in the videos:

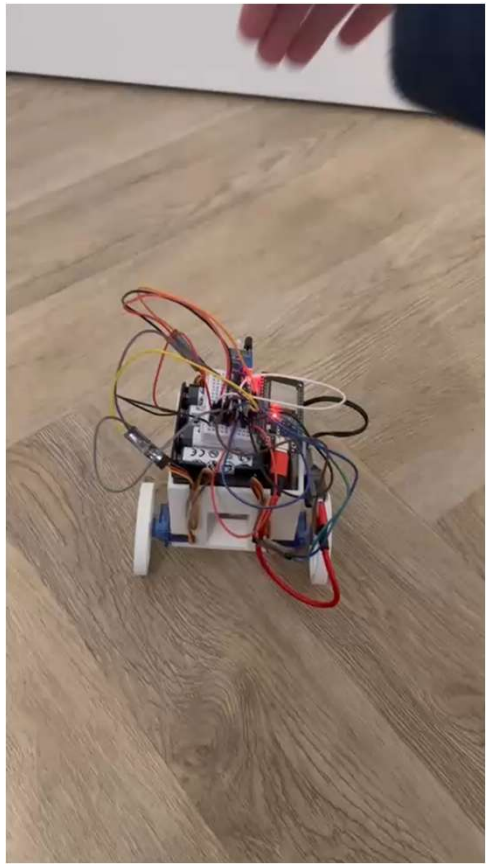
Automatic Mode:



Manual Mode:



Testing Obstacle Avoidance:



Evaluation:

It just about works and functions how I would have liked it too. However, as you can see in the testing videos, it struggled quite a bit trying to move itself. Therefore, if I were to make this again, I would probably use 4 servo motors and wheels, like a car does, and use stronger servos and make the wheels out of a material with more grip like TPU, and this would hopefully ensure that it moves more stably.