Group 10

Arduino Mars Rover

李昀達 張守豐 陳冠晰
劉祐瑋 王彥智 陳柏翰

June 2024

Team Members

李昀達

陳冠晰

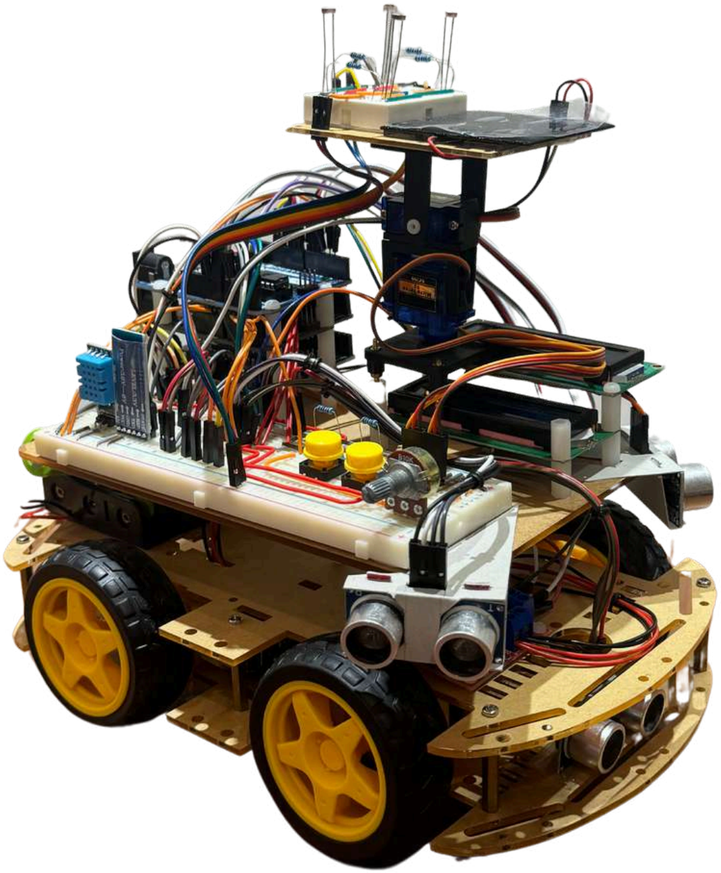
劉祐瑋

張守豐

陳柏翰

王彥智

OVERVIEW



https://github.com/vic9112/MarsRover/

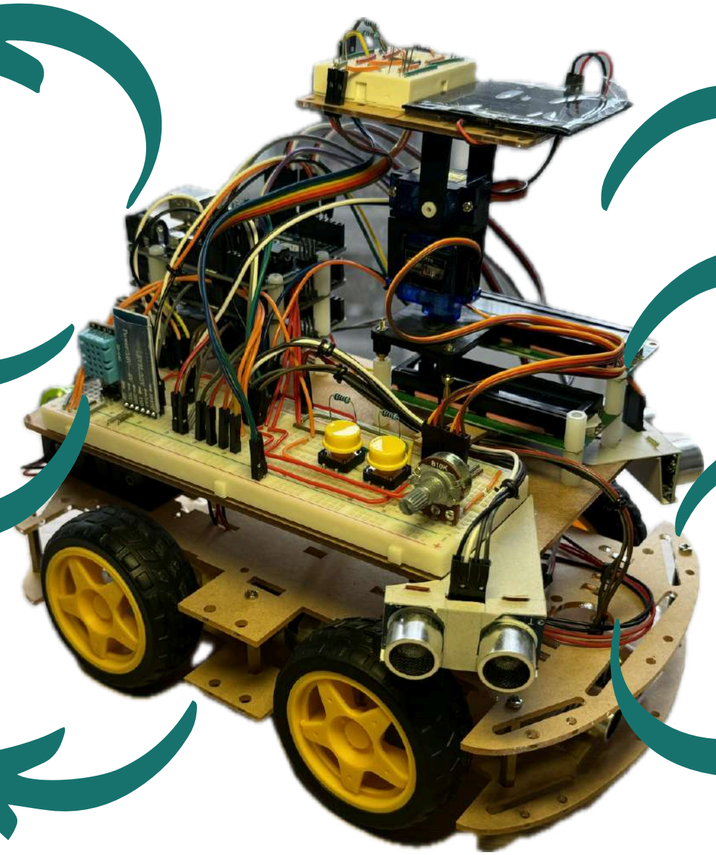
Project Overview

Bluetooth Module
HM10-BLE

RGB Intensity Sensing Module
APDS-9960

Temperature&Humidity Sensor
DHT11

Four-wheel drive



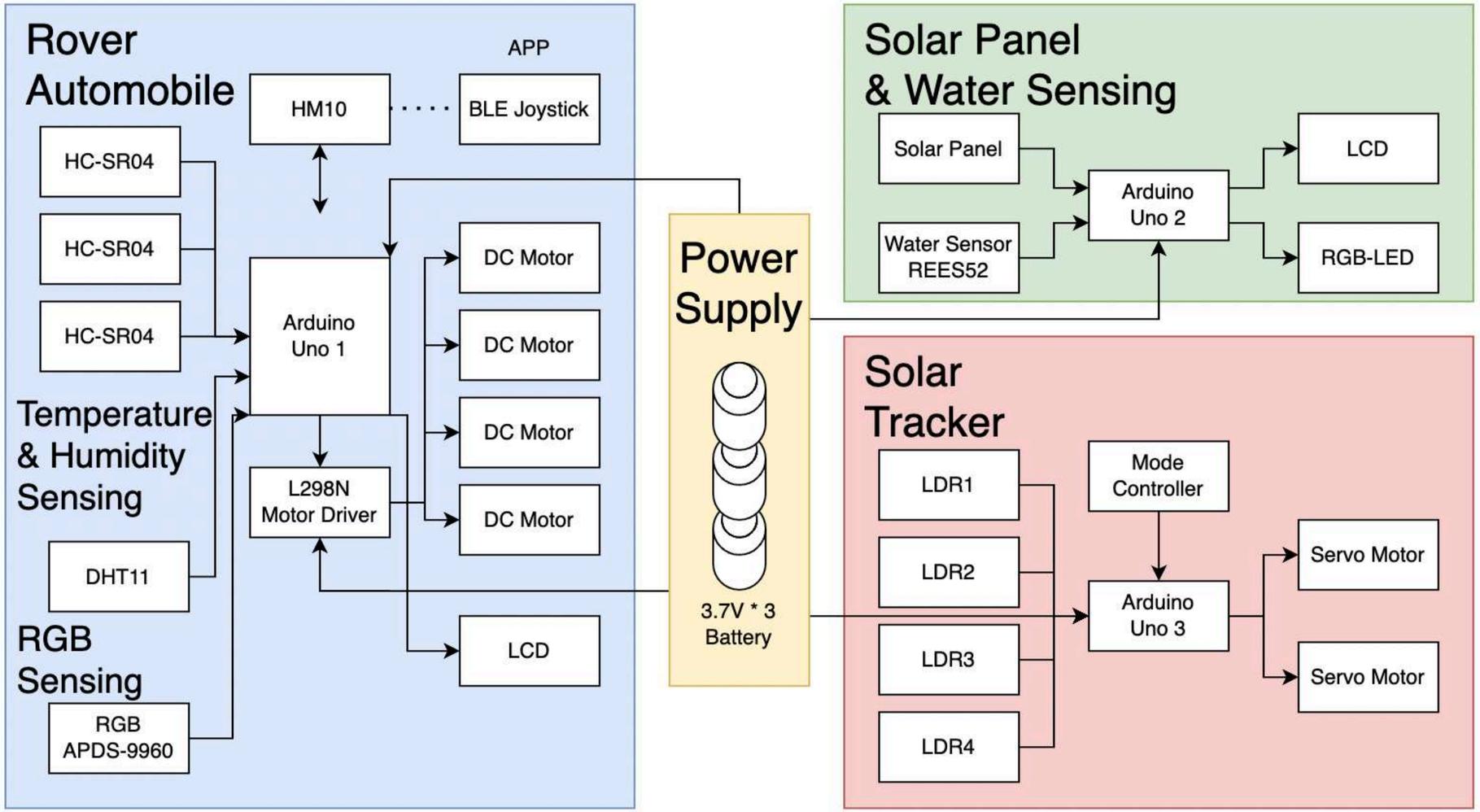
Dual-Axis Solar Tracker

Two LCD Displays

Water Sensing Module
REES52
with RGB LED

Ultrasonic Sensor
HC-SR04

Project Overview



TIMELINE

Dual Axis Solar Tracker

1 week
LDR with two motors
Auto and Manual mode

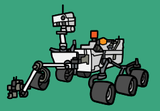
Temperature and humidity sensing

1 week
Sensing with
LCD Display

Solar Panel Installation

1 week
structure setup

Rover Construction



2 weeks
Construction
Joystick-controlled
Bluetooth-controlled
Self-driving

Routing & Installation

1 week
Integration
Installation modules
on the rover
Testing & Refining

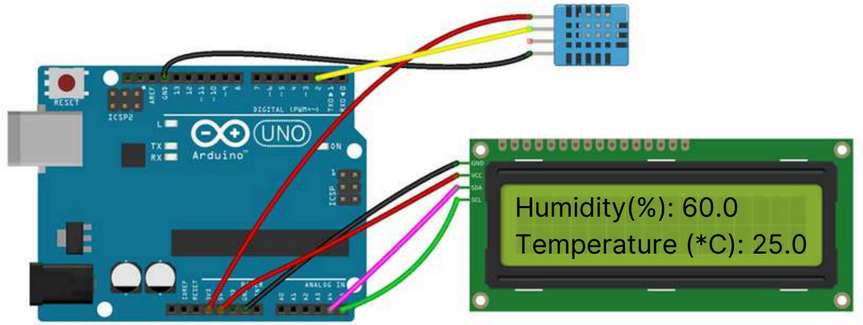
Midterm Presentation

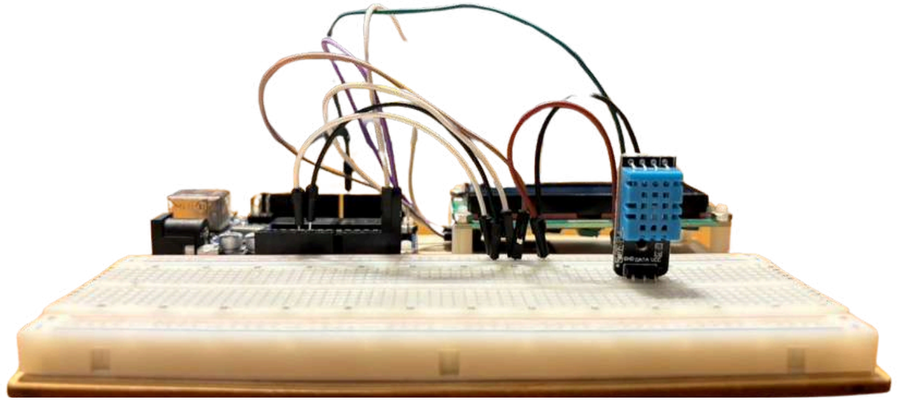
Temperature and Humidity Sensing

Implementation

LCD_DHT §

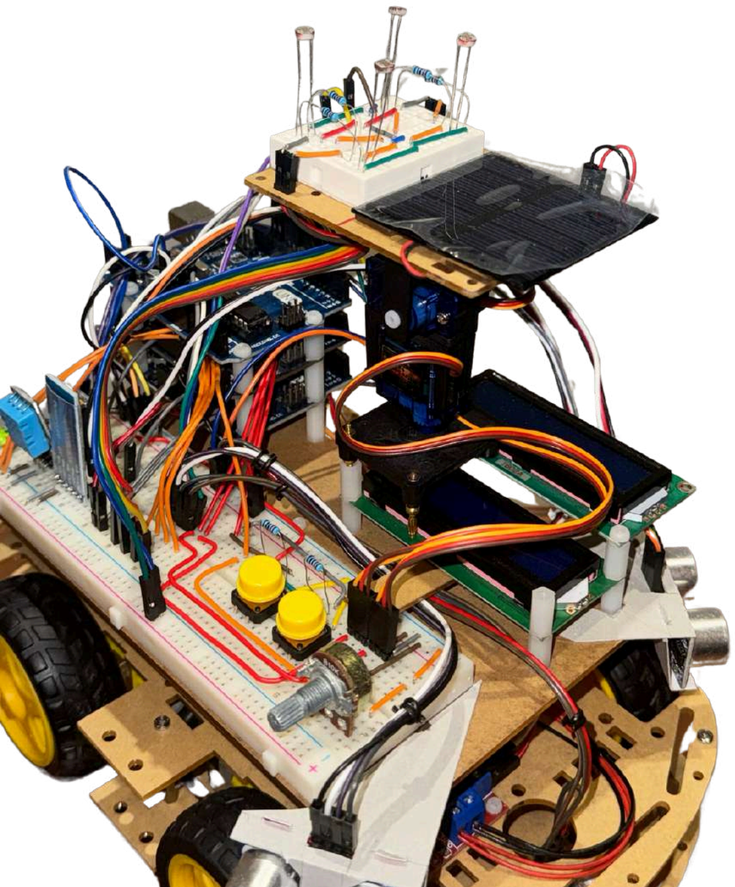
```
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd (0x27, 16, 2);
#include <DHT.h>
DHT dht(7, DHT11);
void setup(){
  Serial.begin(9600);
  dht.begin();
  lcd.init() ;
  lcd.backlight();      //Serial 不需要此指令
}
void loop(){
  float H = dht.readHumidity();
  float T = dht.readTemperature();
  lcd.setCursor(0, 0);
  lcd.print("Humidity(%):");
  lcd.setCursor(12, 0);
  lcd.print(H);
  lcd.setCursor(0, 1);
  lcd.print("Temp.(*C):");
  lcd.setCursor(10, 1); lcd.print(T);
  delay(2000);
}
```





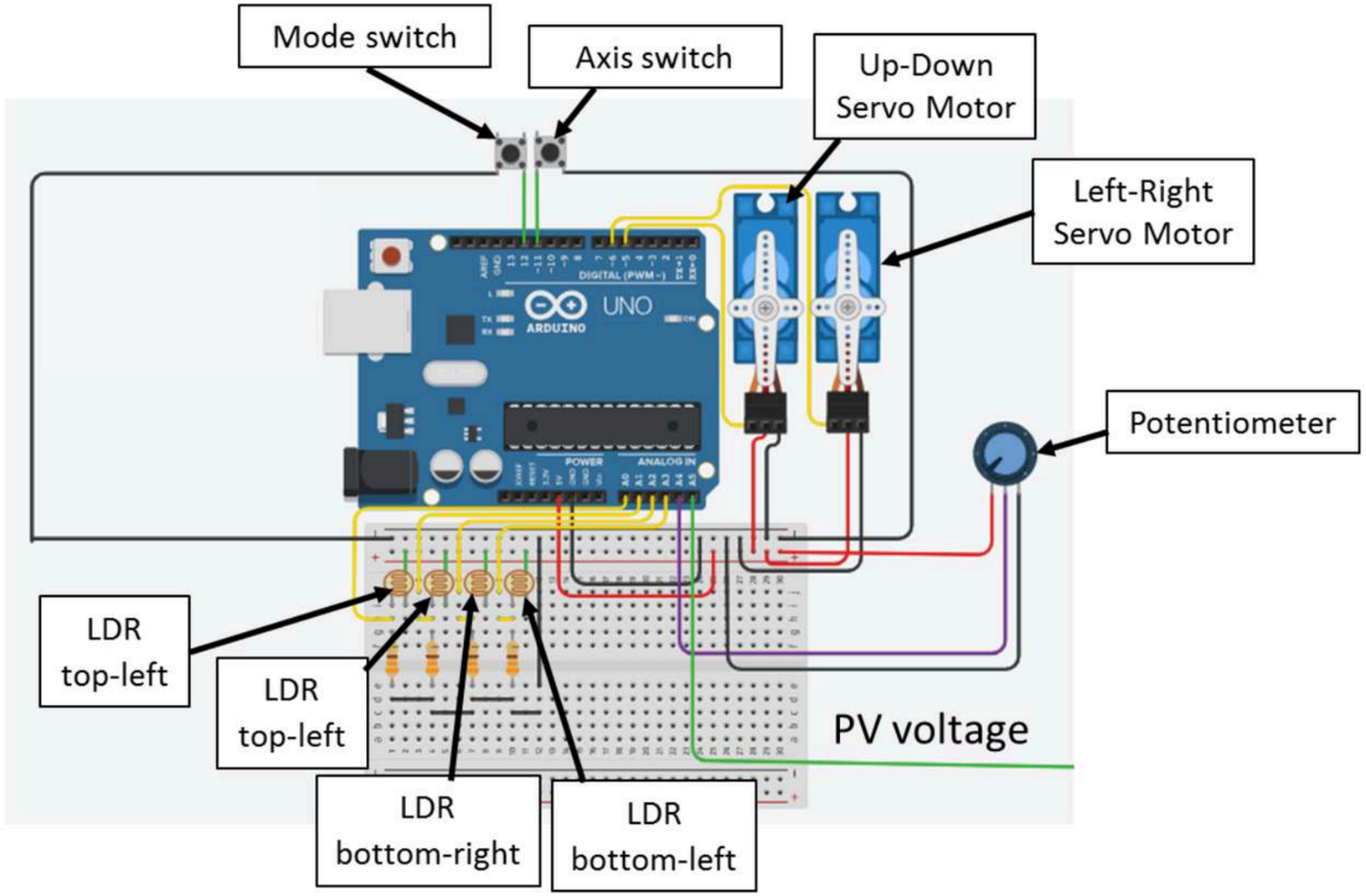
https://github.com/vic9112/MarsRover/tree/main/src

Dual Axis Solar tracker

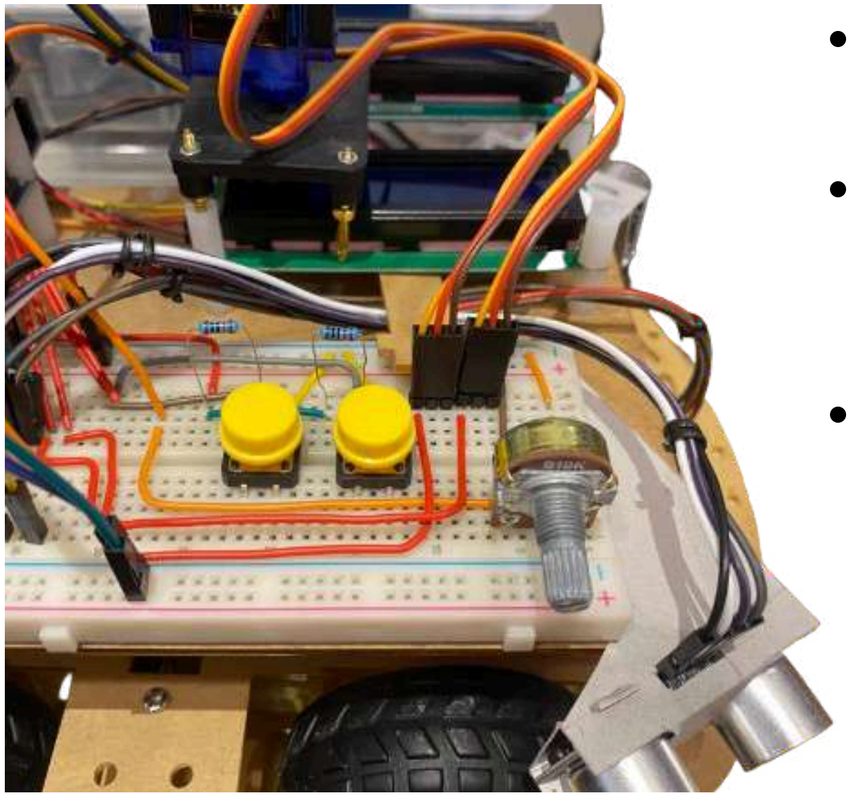


- Utilizing four Light Dependent Resistors (LDR) sensors to detect light intensity.
- Switch between the two modes (automatic and manual) with push-button1.
- Push-button2 is used to link either the SM1 (up-down servomotor) or SM2 (left-right servomotor) to the potentiometer to control their movement.
- Build the structure for two axis to function properly using cardboard
- Installation of the solar panel
- Display the Voltage on LCD

Dual Axis Solar tracker



Manual mode

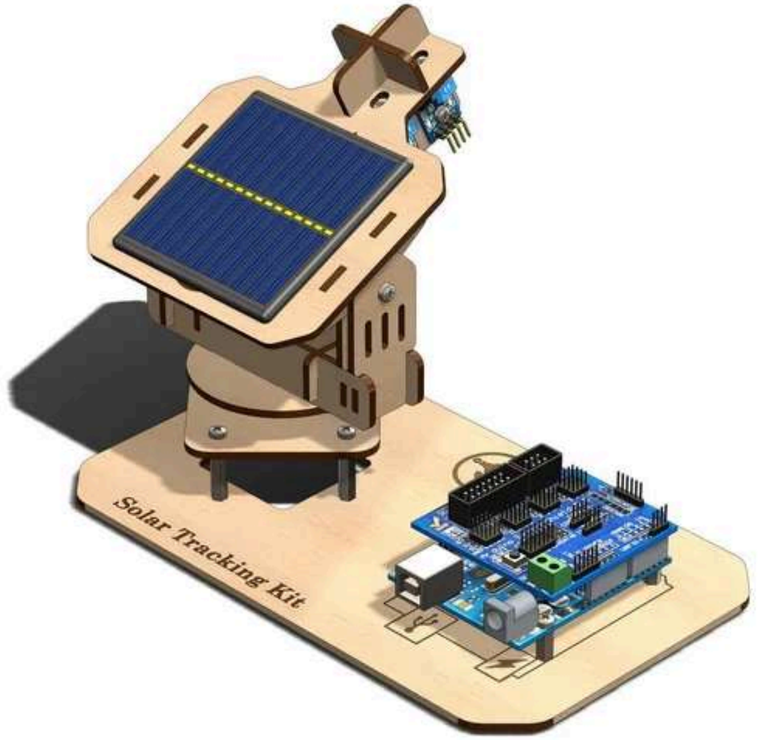


- Switch between the two modes (automatic and manual) with push-button1.
- Push-button2 is used to link either the SM1 (up-down servomotor) or SM2 (left-right servomotor) to the potentiometer to control their movement.
- Potentiometer maps resistive values from 0-1023 to angles on the two servo motors 0-180 degrees.

Dual Axis Solar tracker

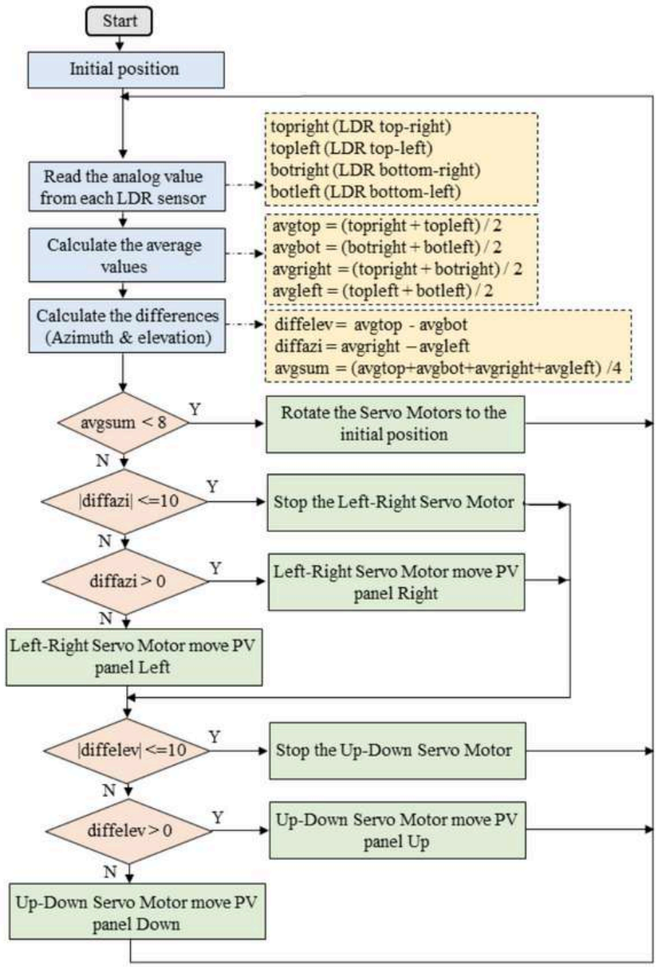
Implementation-Manual

```
void manualsolartracker(){
  buttonState2  = digitalRead(11);
  if (buttonState2 != prevButtonState2) {
    if (buttonState2  == HIGH) {
      //Change mode and ligh up the correct indicator
      if  (axe == 1) {
        axe = 0;
      } else {
        axe = 1;
      }
    }
  }
  prevButtonState2 = buttonState2;
  delay(50); // Wait for 50  millisecond(s)
  if (axe == 0) {     //control right-left movement
    servo_rightleft.write(map(analogRead(A4),  0, 1023, 0, 180)); // strange
  } else { // //control up-down movement
    servo_updown.write(map(analogRead(A4),  0, 1023, 0, 180));
  }
}
```



https://github.com/vic9112/MarsRover/tree/main/src

Automatic mode



- The average values from two right LDRs and two left LDRs (two top LDRs and two bottom LDRs) are compared.
- If one side receives more light, the solar tracker will move in that direction through the servomotors.
- It will continue to rotate until the difference result is in the range [–15, 15] making sure the solar tracker is perpendicular to the light source.

Dual Axis Solar tracker

Implementation-Auto

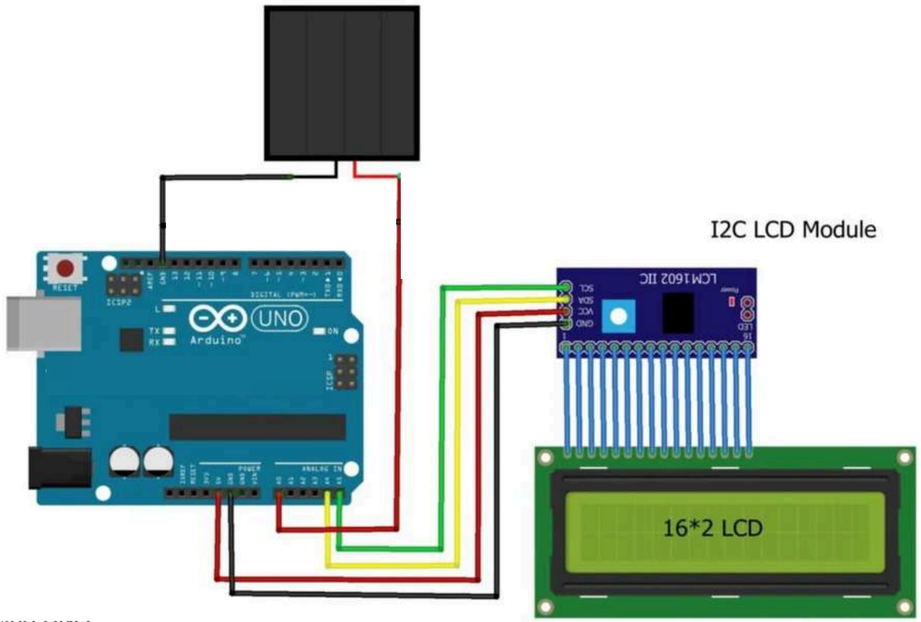
```
void  automaticsolartracker(){
//capturing analog values of each LDR
     topr= analogRead(ldrtopr);          //capturing analog value of top right LDR
     topl= analogRead(ldrtopl);          //capturing analog value of top left LDR
     botr= analogRead(ldrbotr);          //capturing analog value of bot right LDR
     botl= analogRead(ldrbotl);          //capturing analog value of bot left LDR
// calculating average
     int avgtop = (topr + topl) / 2;     //average  of top LDRs
     int avgbot = (botr + botl) / 2;     //average of bottom LDRs
     int avgleft = (topl + botl) / 2;    //average of left LDRs
     int avgright  = (topr + botr) / 2;   //average of right LDRs
//Get the different
     int diffelev = avgtop - avgbot;      //Get the different average betwen  LDRs top and LDRs bot
     int diffazi = avgright - avgleft;    //Get the different  average betwen LDRs right and LDRs left
```

```
//left-right movement of  solar tracker
     if (abs(diffazi) >= threshold_value){       //Change  position only if light difference is bigger then the threshold_value
       if  (diffazi > 0) {
         if (servo_rightleft.read() < 180) {
           servo_rightleft.write((servo_updown.read()  + 5));
         }
       }
       if (diffazi <  0) {
         if (servo_rightleft.read()  > 0) {
           servo_rightleft.write((servo_updown.read() - 5));
         }
       }
     }
//up-down movement of solar tracker
     if (abs(diffelev) >= threshold_value){    //Change position only if light  difference is bigger then thethreshold_value
       if (diffelev > 0) {
         if  (servo_updown.read() < 180) {
           servo_updown.write((servo_rightleft.read()  - 5));
         }
       }
       if (diffelev <  0) {
         if (servo_updown.read()  > 0) {
           servo_updown.write((servo_rightleft.read() + 5));
         }
       }
     }
 }
```

https://github.com/vic9112/MarsRover/tree/main/src

Dual Axis Solar tracker

Solar panel voltage readout

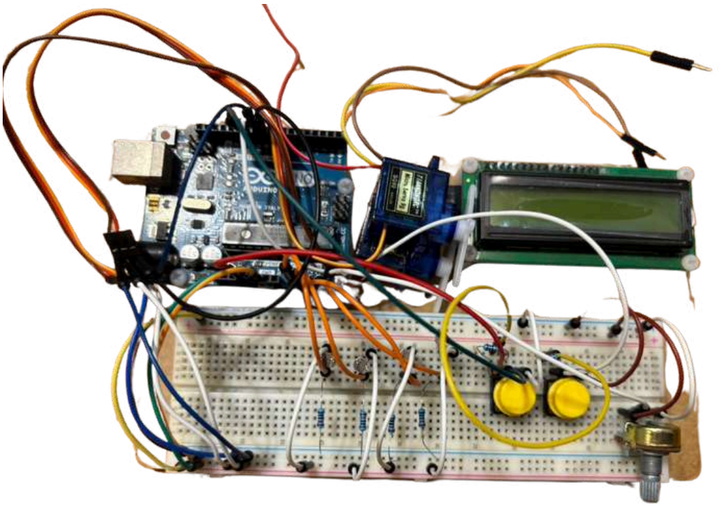



```
void loop()
{
  solar_cell = (analogRead(A0) * (5.001 / 1023.001));
  Serial.println(solar_cell);
  lcd.setCursor(0, 0);
  lcd.print("Solar Cell Volt");
  lcd.setCursor(0, 1);
  lcd.print("Value = ");
  lcd.setCursor(8, 1);
  lcd.print(solar_cell);
  delay(10); // Delay a little bit to improve simulation performance
}
```

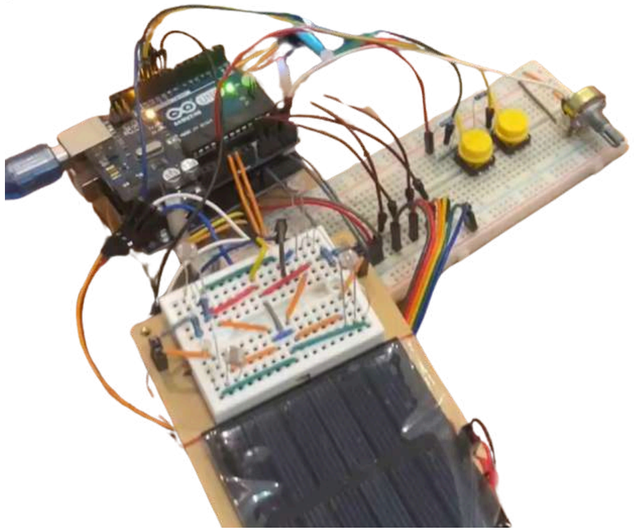
https://github.com/vic9112/MarsRover/tree/main/src

Dual Axis Solar tracker

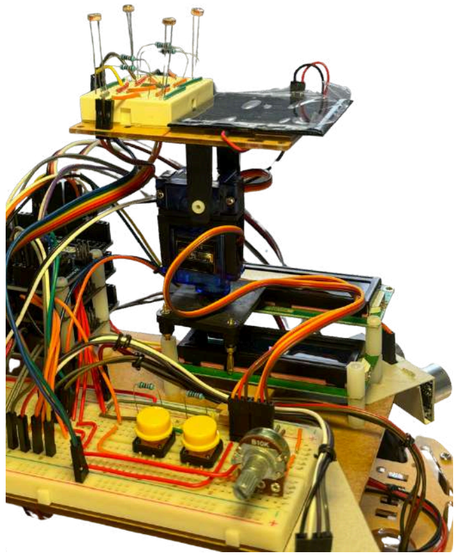
Design Process



Plain circuit



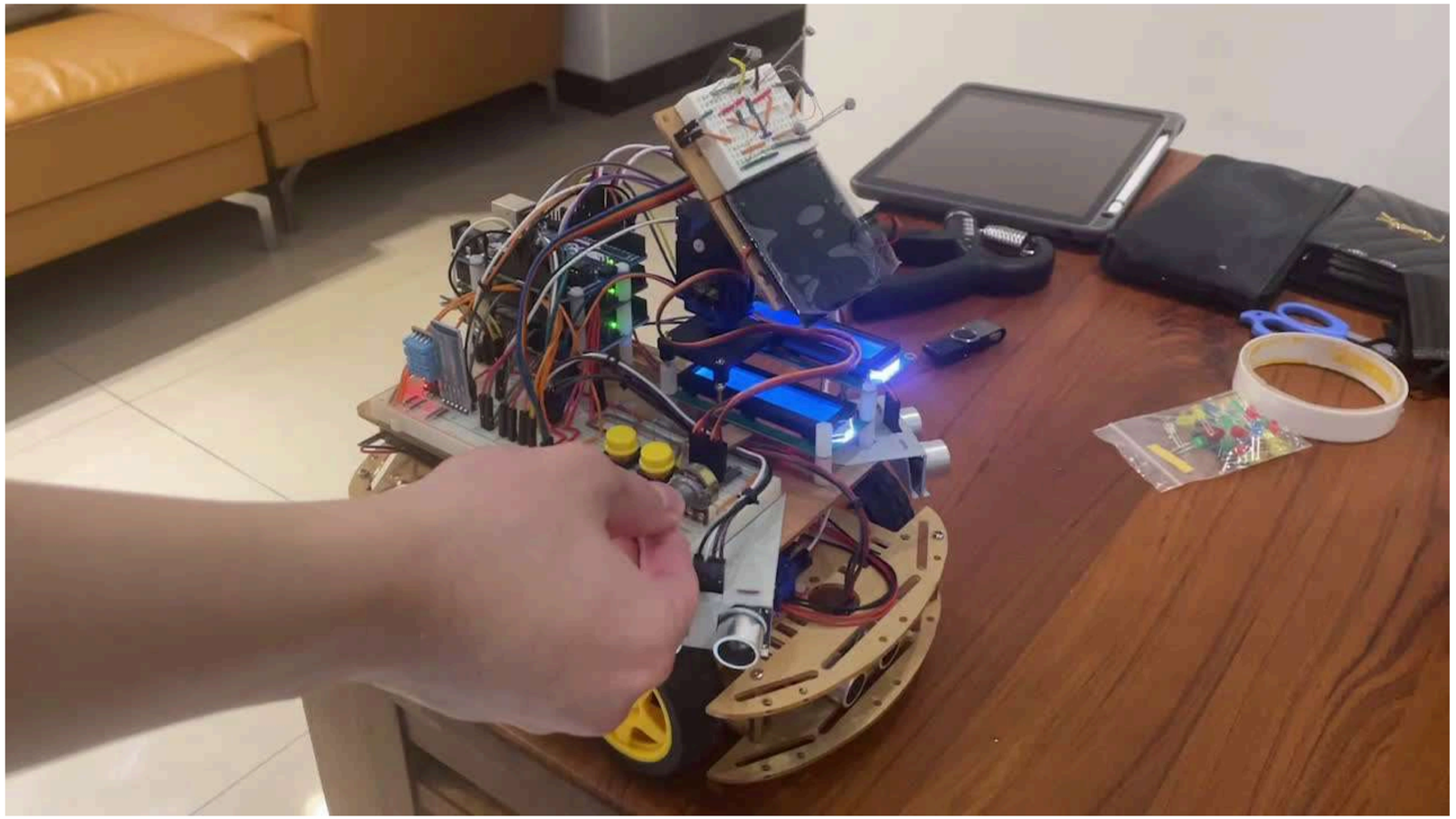
Dual-axis construction with solar panel installation



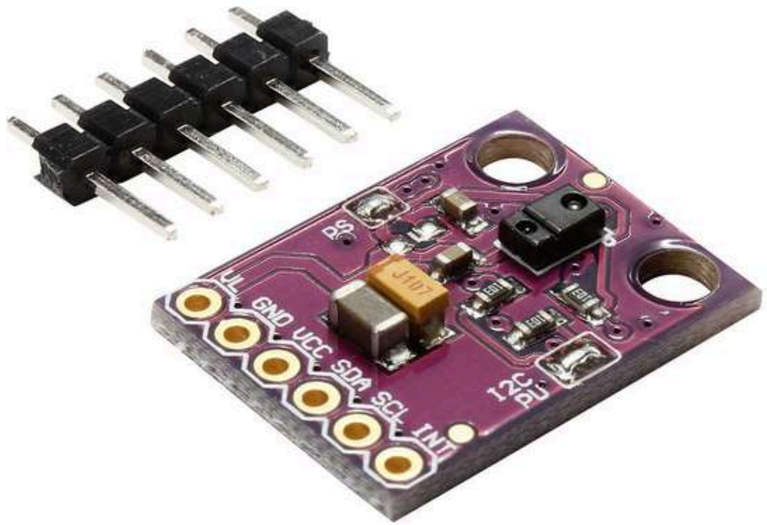
Final integration on rover

Dual Axis Solar tracker

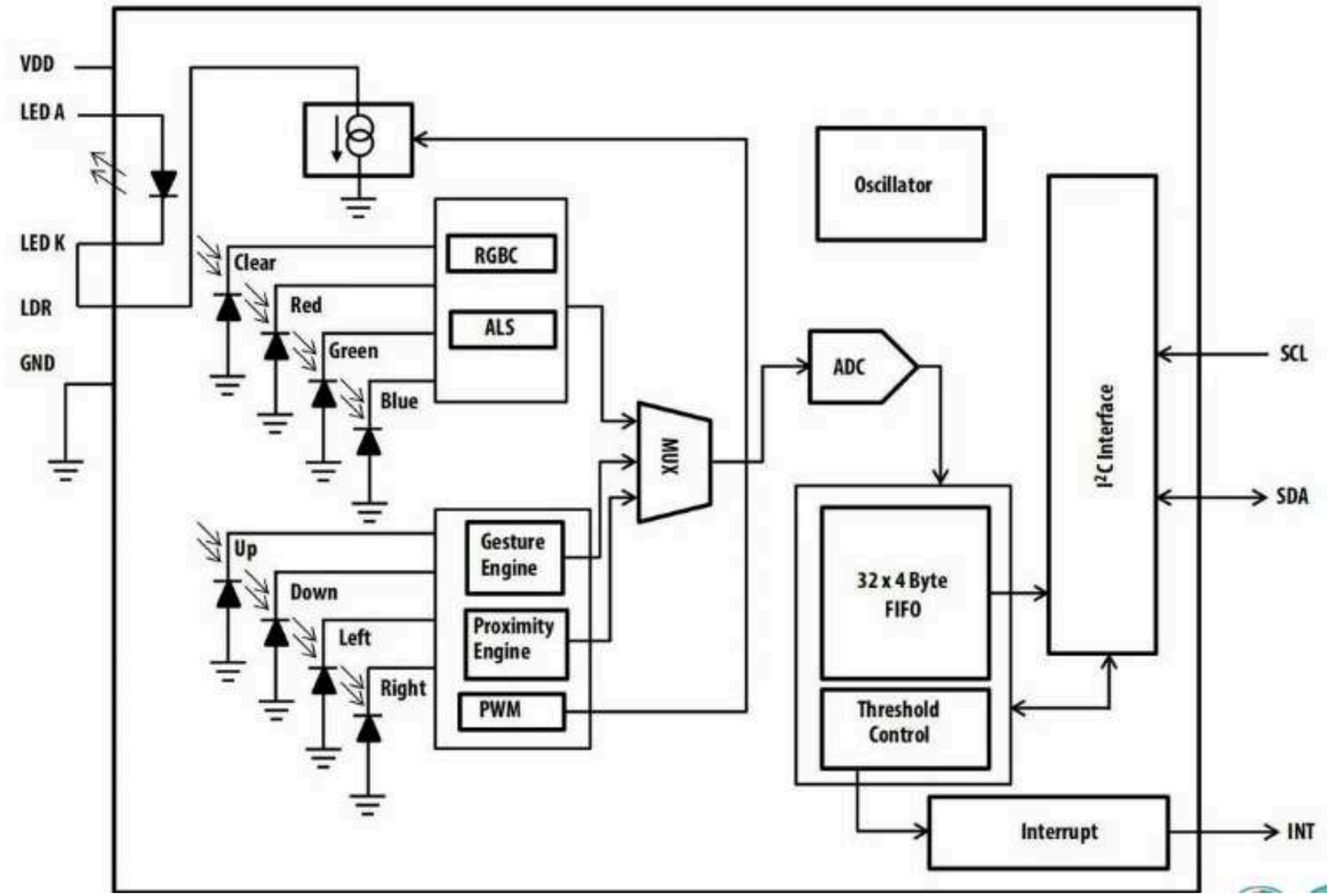
Demo



RGB intensity Sensing



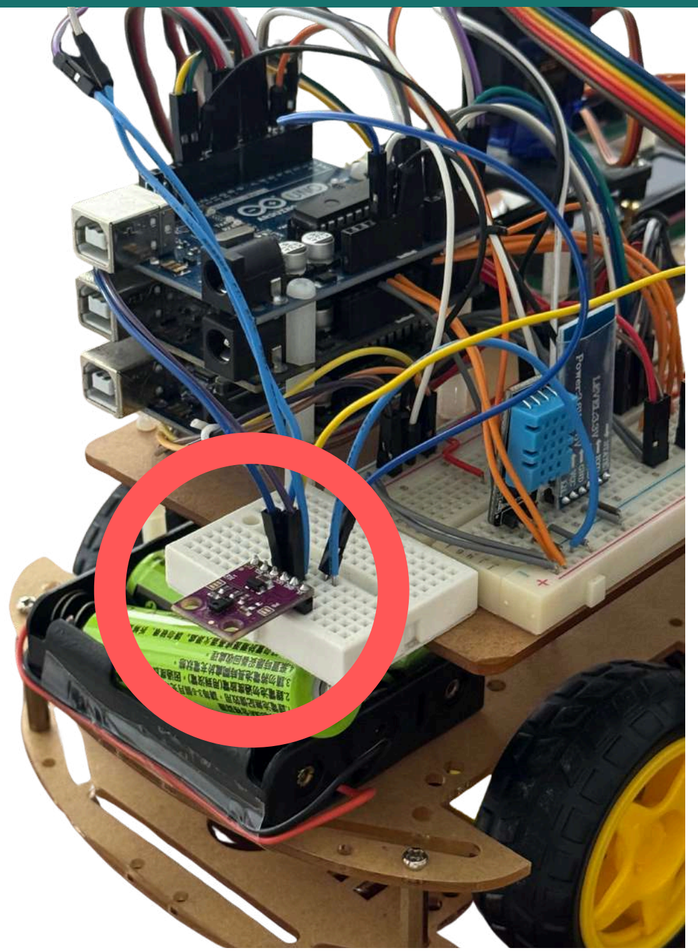
RGB Sensing
APDS-9960



APDS-9960

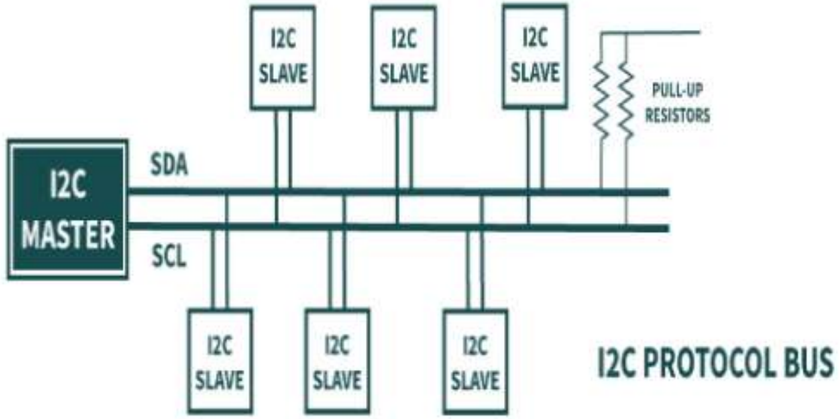
Implementation

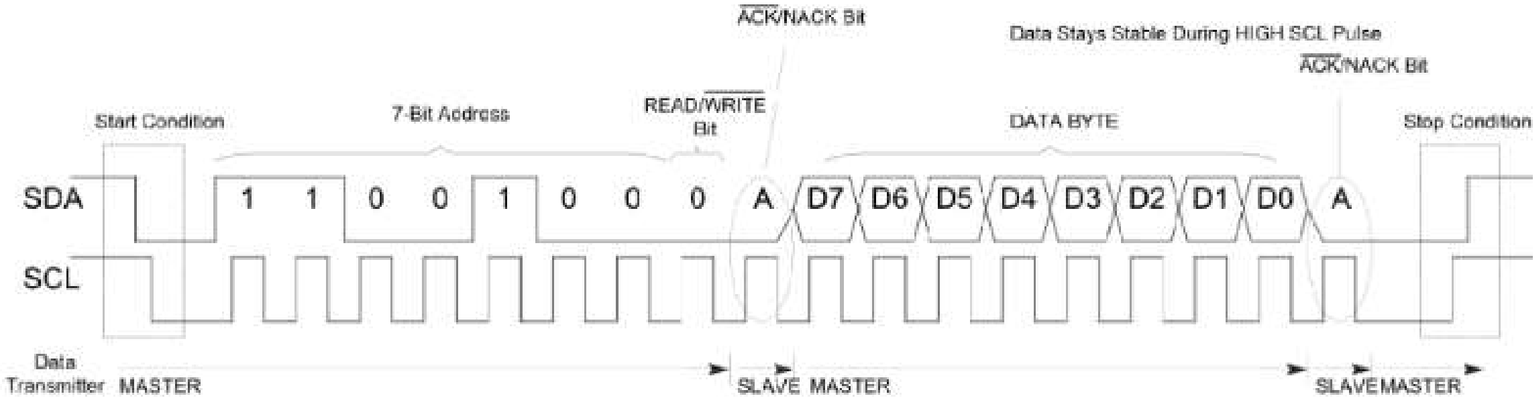
```
else if(cnt==40){lcd.clear();
      cnt++;
    if (  !apds.readAmbientLight(ambient_light) ||
      !apds.readRedLight(red_light) ||
      !apds.readGreenLight(green_light) ||
      !apds.readBlueLight(blue_light) ) {
  Serial.println("Error reading light values");
} else {
  Serial.print("Ambient: ");
  Serial.print(ambient_light);
  Serial.print(" Red: ");
  Serial.print(red_light);
  Serial.print(" Green: ");
  Serial.print(green_light);
  Serial.print(" Blue: ");
  Serial.println(blue_light);
}
}
else if(cnt<80){
    lcd.setCursor(0, 0);
    lcd.print("A:");
    lcd.print(ambient_light);
    lcd.setCursor(8, 0);
    lcd.print("R:");
    lcd.print(red_light);
    lcd.setCursor(0, 1);
    lcd.print("G:");
    lcd.print(green_light);
    lcd.setCursor(8, 1);
    lcd.print("B:");
    lcd.print(blue_light);
    cnt++;
  }
```



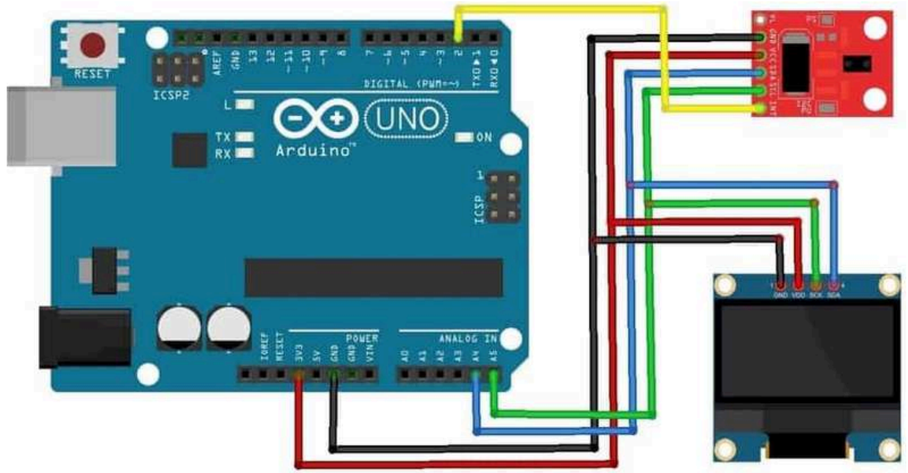
https://github.com/vic9112/MarsRover/tree/main/src

I2C protocol



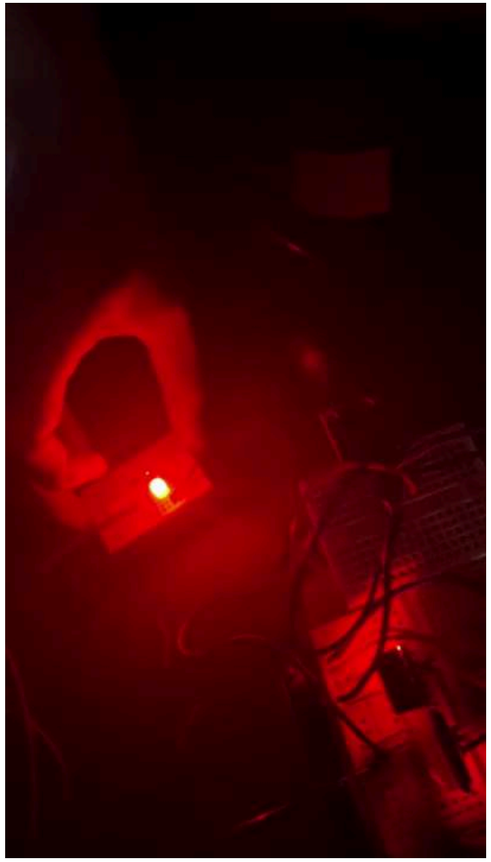


Circuit sturcture



RGB intensity Sensing

Demo



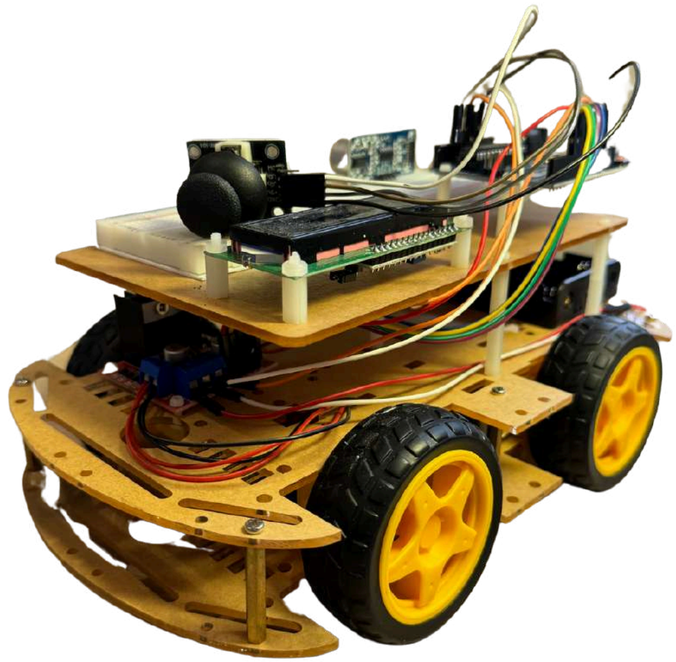
Bluetooth Controlled Rover with Self-Driving

Version 0

Joystick-controlled

- Initial implementation is to control the rover using wire-connected joystick controller.

- With Joystick controller give us two axis analog signals, we convert the two signals into motor rotation speed.

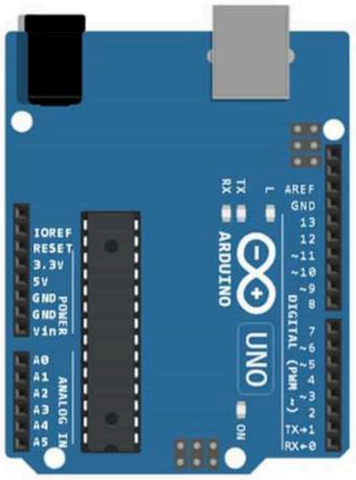


Bluetooth Controlled Rover with Self-Driving

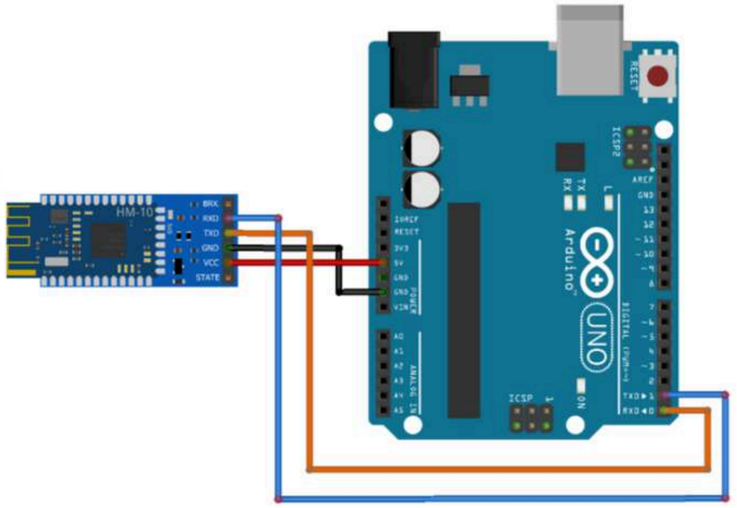
Version 1

Bluetooth Control mode

We replaced joystick controller to bluetooth controller.
As for bluetooth module, we used HM-10, for IOS



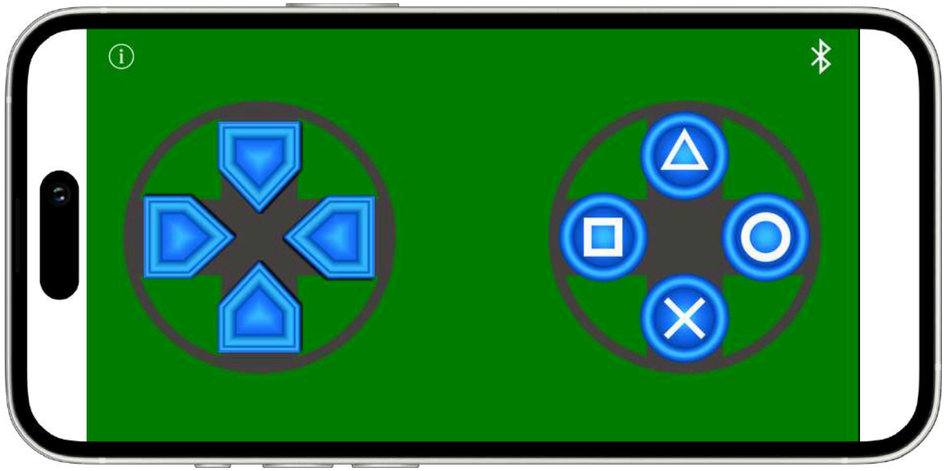
Bluetooth Module



Bluetooth Controlled Rover with Self-Driving

Implementation

- The left panel used to control the direction of the rover in manual mode.
- The right panel, we used △ to switch to self-driving and ⊗ back to manual mode.



When the press started ,sending the character '**A**'
When pressed long, sending the character '**a**'.

When the press started ,sending the character '**D**'
When pressed long, sending the character '**d**'.

When the press started ,sending the character '**B**'
When pressed long, sending the character '**b**'.

When the press started ,sending the character '**C**'
When pressed long, sending the character '**c**'.

DONE

Every time you release the button, sending data 0.

When the press started ,sending the character '**E**'
When pressed long, sending the character '**e**'.

When the press started ,sending the character '**H**'
When pressed long, sending the character '**h**'.

When the press started ,sending the character '**F**'
When pressed long, sending the character '**f**'.

When the press started ,sending the character '**G**'
When pressed long, sending the character '**g**'.

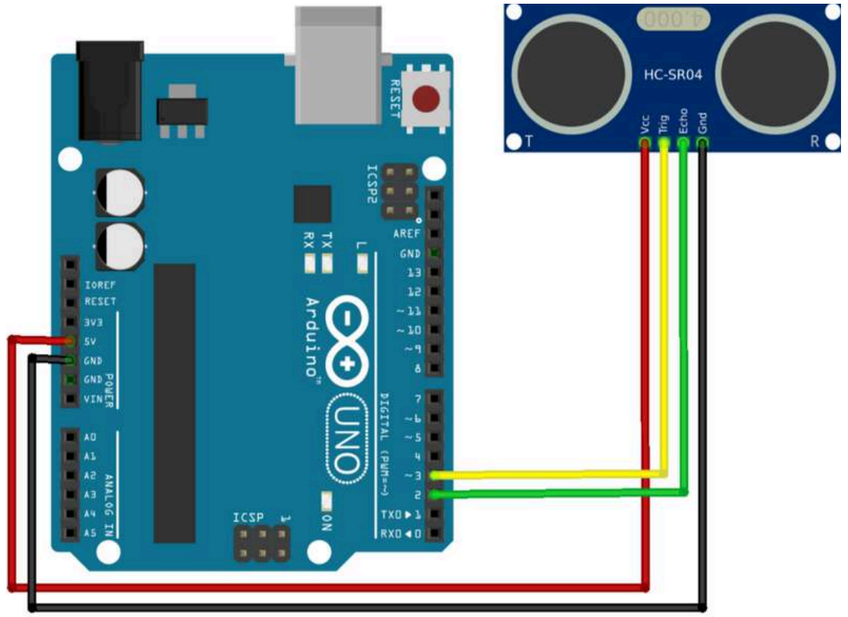
Bluetooth Controlled Rover with Self-Driving

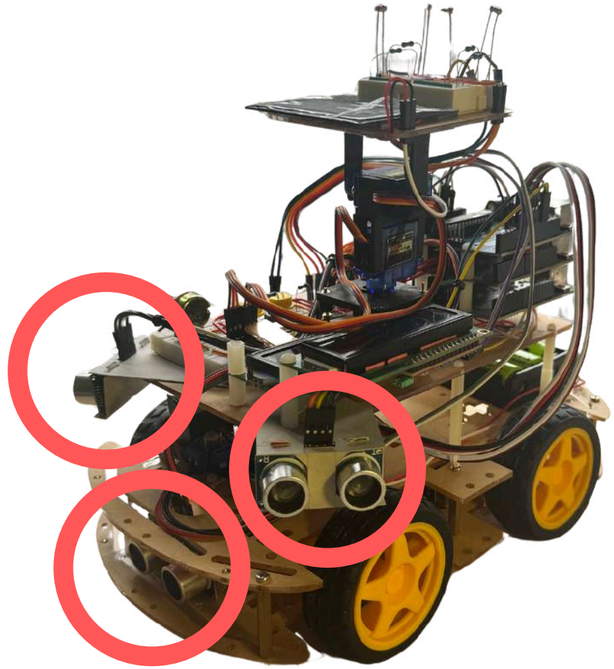
Version 2

Self-Driving Mode

- Arduino ultrasonic sensor (HC-SR04) to get the distance of the surrounding obstacles then turn the rover.
- We installed three sensors to get more adaptive self-driving capability.

HC-SR04

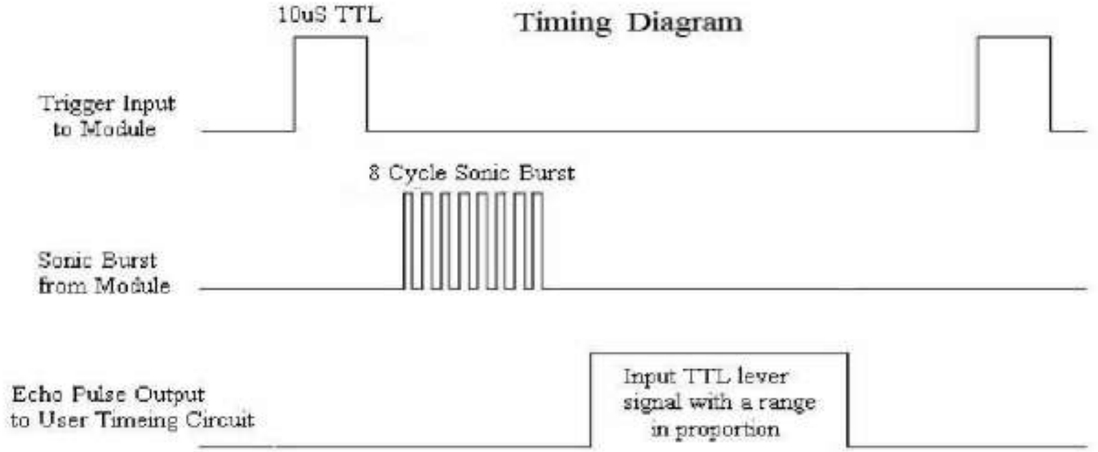


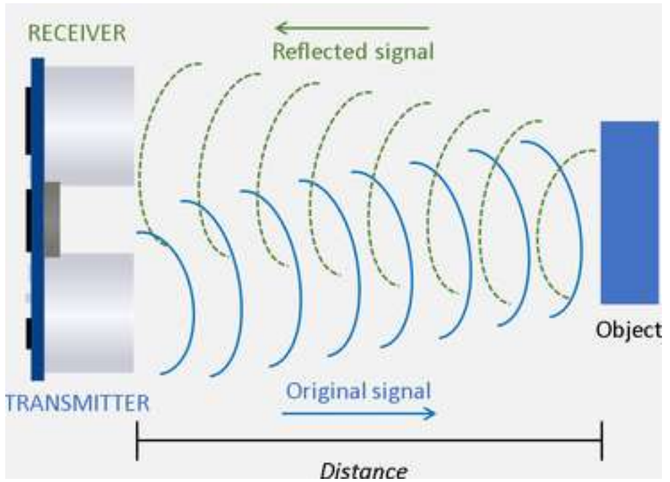


HC-SR04

Implementation

```
digitalWrite(trigPinR, LOW);
delayMicroseconds(5);
digitalWrite(trigPinR, HIGH);
delayMicroseconds(10);
digitalWrite(trigPinR, LOW);
pinMode(echoPinR, INPUT);
durationR = pulseIn(echoPinR, HIGH);
cmR = (durationR/2) / 29.1;
```





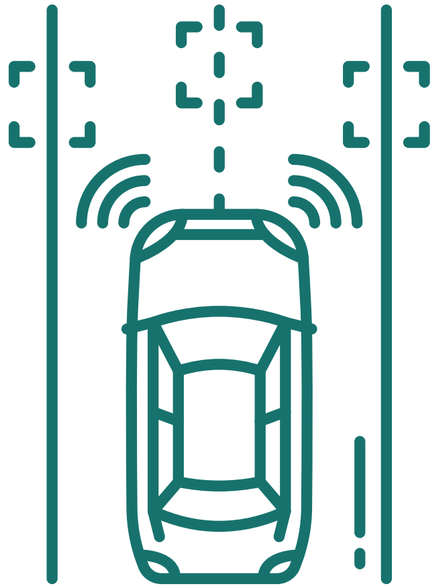
Bluetooth Controlled Rover with Self-Driving

Implementation

For self-driving mode, we use three ultra-sonic sensors to check the distance between the rover the the front environment.

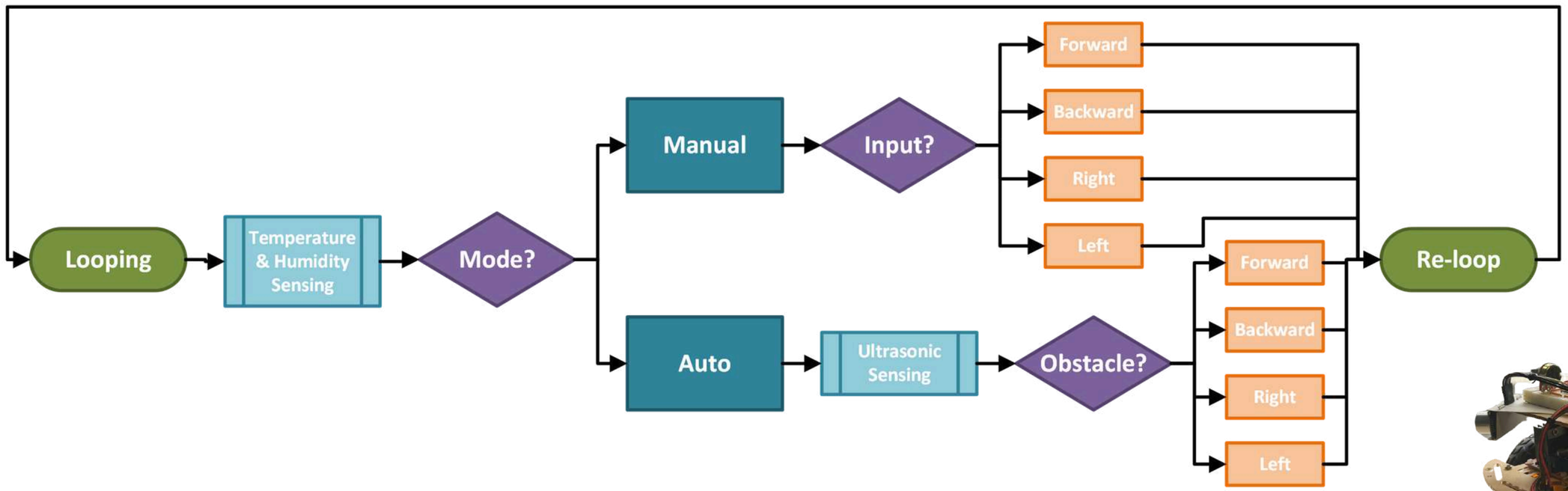
4 scenarios

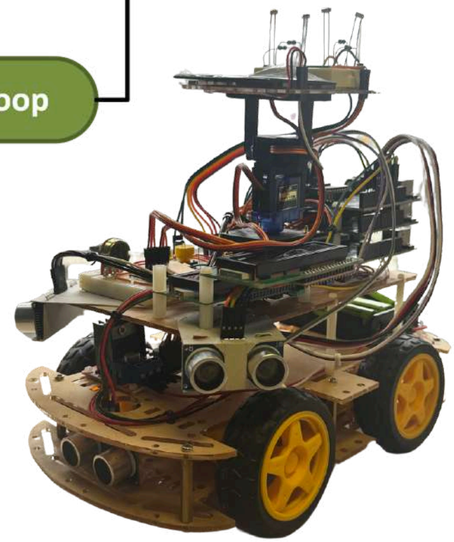
- Right-front distance too short: TURN LEFT
- Left-front distance too short: TURN RIGHT
- Front distance too short: BACK
- Otherwise : FORWARD



Bluetooth Controlled Rover with Self-Driving

Implementation



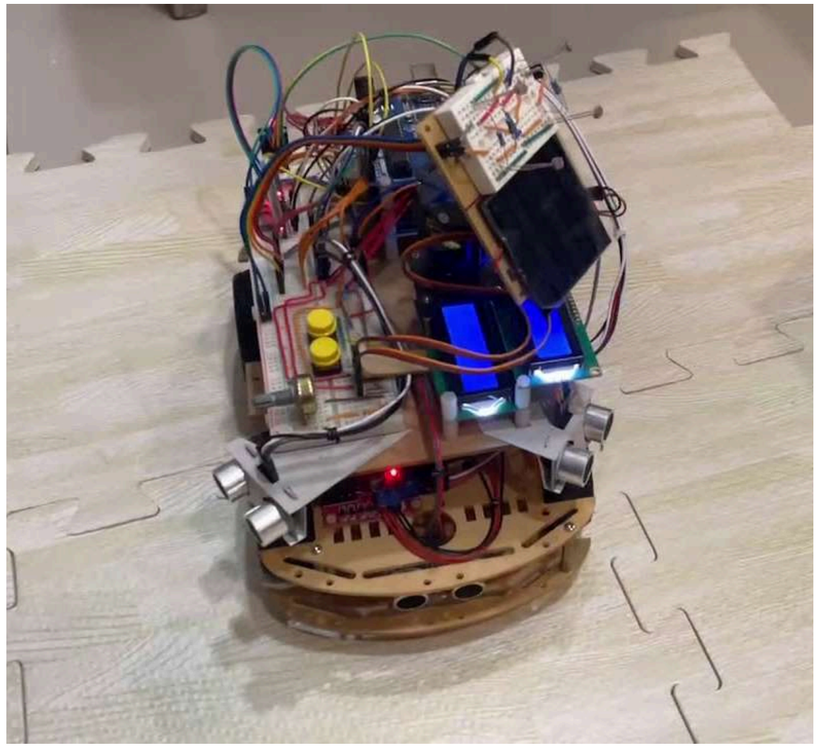


Implement with temperature & humidity sensing and Ultrasonic sensing

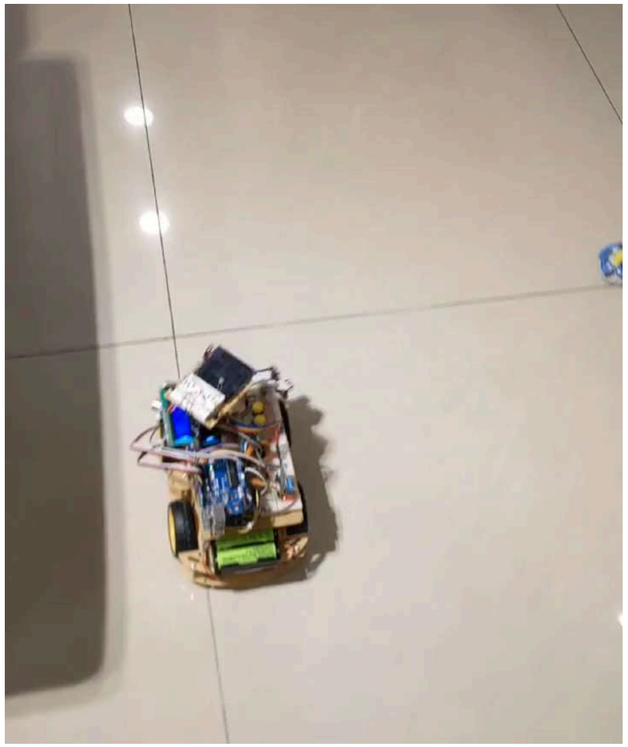
https://github.com/vic9112/MarsRover/tree/main/src

BLE Car DEMO

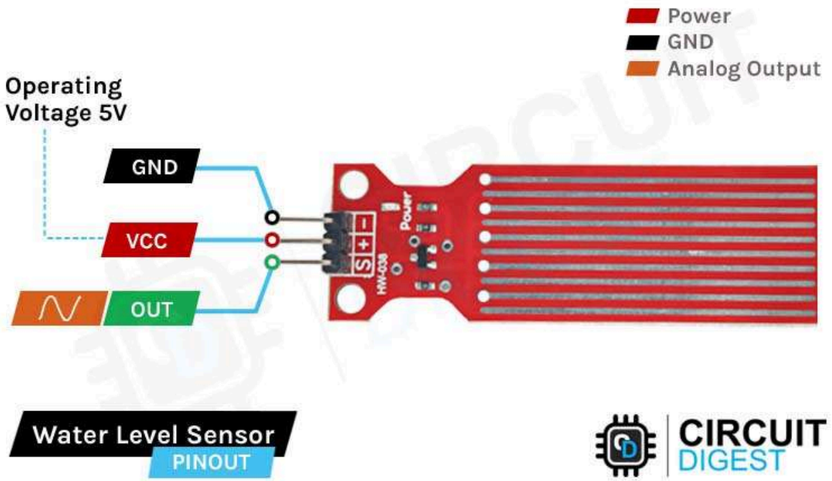
Manual



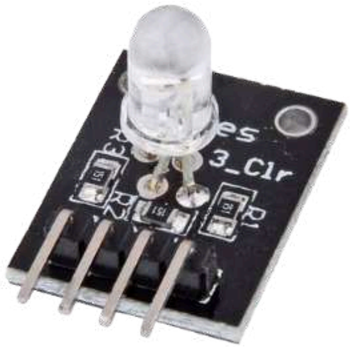
Self-driving



Water Sensing Module with RGB Indicator



REES52



RGB_LED

- No water
- Discover water
- Discover large amounts of water

Water Sensing Module with RGB Indicator

Implementation

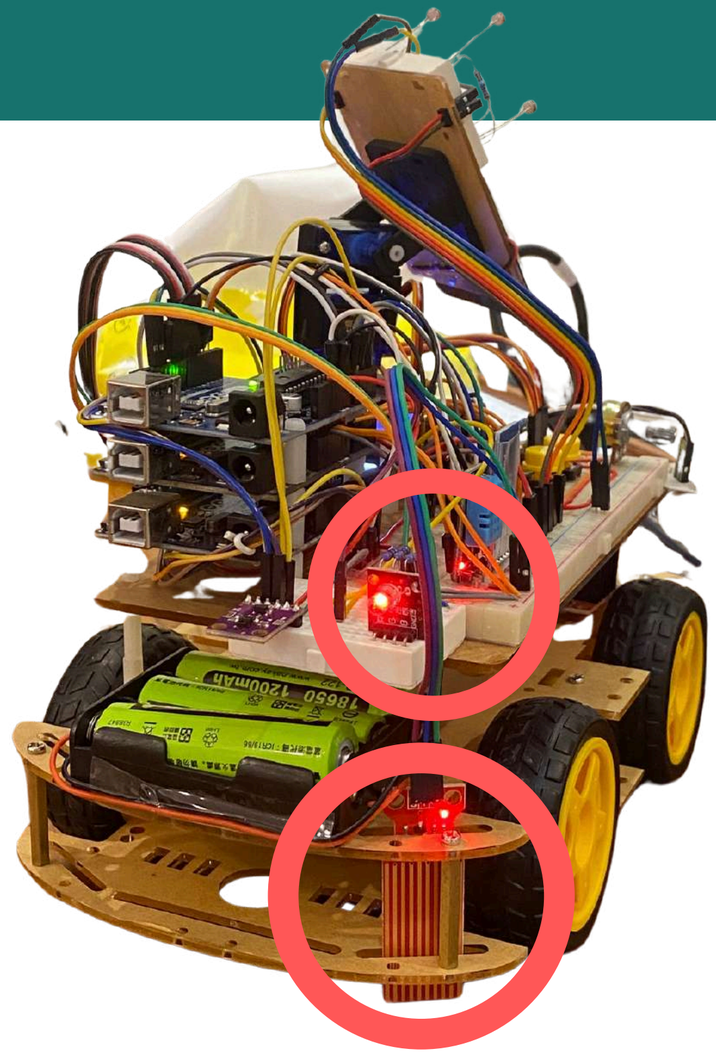
```
#define R 8
#define G 9
#define B 10

void setup(){
Serial.begin(9600);     // Communication starte

  pinMode(R,OUTPUT);
  pinMode(G,OUTPUT);
  pinMode(B,OUTPUT);

}
void loop(){
int sensor=analogRead(A0); // Incoming analog
Serial.println(sensor);   //Wrote serial port

if (sensor <= 100) {
    digitalWrite(R,HIGH);
    digitalWrite(G,LOW);
    digitalWrite(B,LOW);
} else if (100 < sensor and sensor < 400) {
    digitalWrite(R,LOW);
    digitalWrite(G,HIGH);
    digitalWrite(B,LOW);
} else {
    digitalWrite(R,LOW);
    digitalWrite(G,LOW);
    digitalWrite(B,HIGH);
```



https://github.com/vic9112/MarsRover/tree/main/src

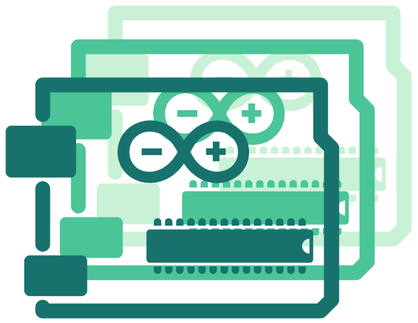
Water Sensing Module with RGB Indicator

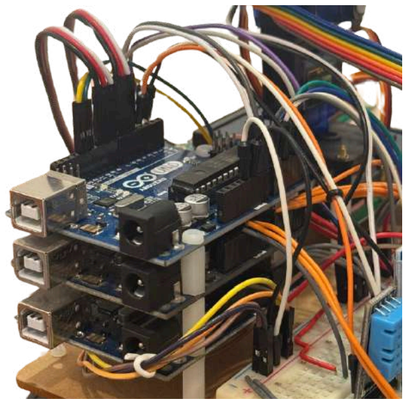
Demo

https://github.com/vic9112/MarsRover/tree/main/src

Final Integration and Routing

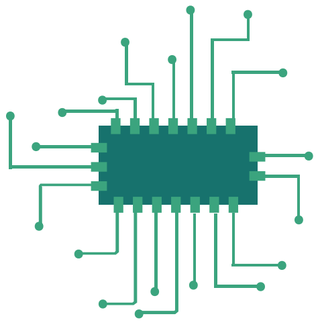
Arduino UNO Stack

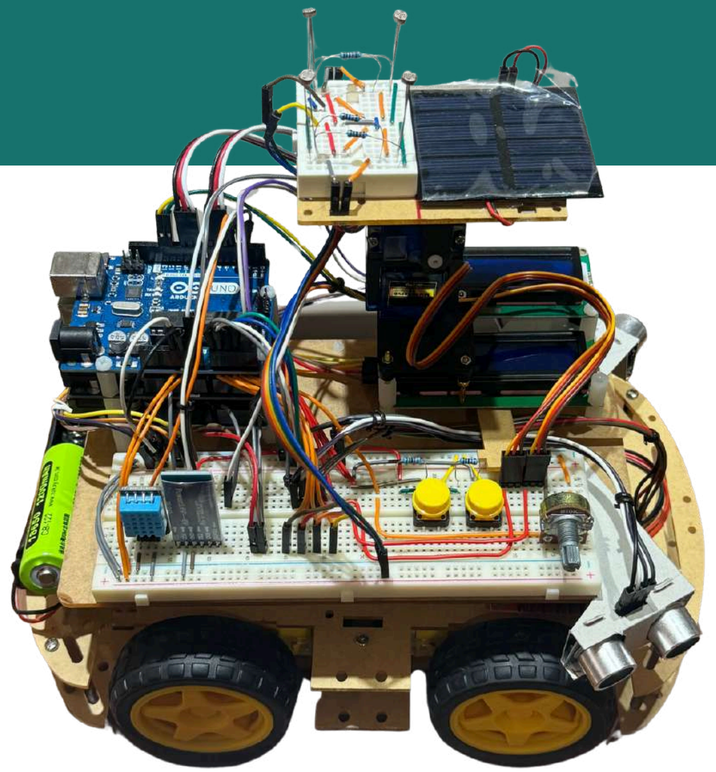




Due to the limitation of pin number, we have to use three Arduino UNOs to make all the module run concurrently. Therefore, in order to save installation space, we stack up three boards

Routing



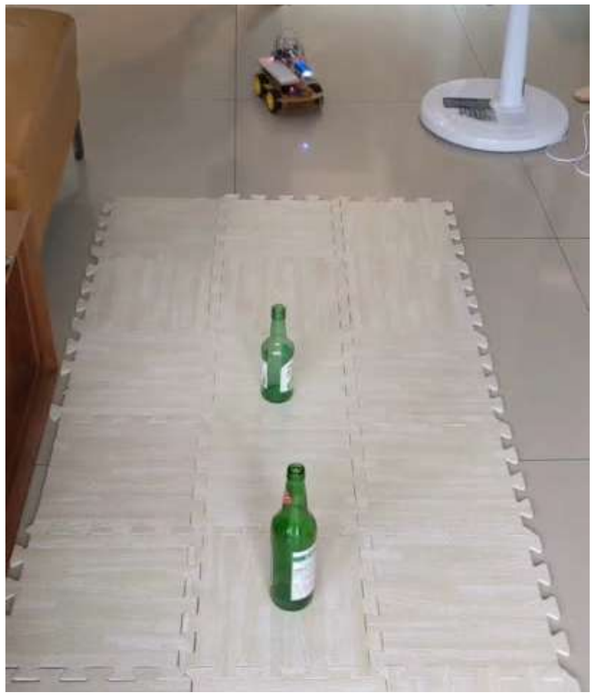


Organized circuit routing will make the design more clean and easier to debug. We spent a lot of time to place the circuit as clean as possible.

Behind the Scene

Beyblade Mode

No Wheel Mode



Work Distribution

- **李昀達**: Manual dual-axis control, automatic dual-axis control, solar panel, routing optimization, bracket fabrication, LCD control, report preparation
- **陳冠晰**: Bluetooth module, RGB capacitive module, manual car control, automatic car control, ultrasonic sensing, mode switch button, report preparation
- **劉祐瑋**: Automatic car control, ultrasonic sensing, RGB sensing module
- **張守豐**: Light sensor module, temperature and humidity sensing, mode switch button
- **陳柏翰**: Joystick car control, motor drive module, car body setup, report preparation
- **王彥智**: Water source detection module, RGB capacitive module, report preparation

Thank you !