SMART VACUUM CLEANER



**PRESENTED BY**

**NIMMALA SUJITH MUTYALA PURNA RAMA SATYA SAI**

**Branch : E.C.E Branch : E.C.E**

**Reg NO: 20B91A04H1 Reg No : 20B91A04G0**

**MALLAVARAPU MAANVITA**

**Branch : E.C.E**

**Reg NO: 20B91A04D7**

# **1.ABSTRACT :**

“Robotics” - the word has a deep meaning. Robotics in turn takes the scale of development by employing various branches, tools, and mechanisms and performs a wide variety of functions for the benefit of mankind. From this whole ocean of robotics, we have intended to build a basic Remote controlled vacuum cleaner robot -“SHUBHROT‟.

Being the students of 3th sem Electronics & Communication branch, the concept that emerged in our mind was to develop a product that would be useful to most of the people, so we decided to design and build a robot capable of vacuuming the floor of a room or area without any human interaction other than just starting and sometime operating the unit thus saving valuable human time.

In the robot a robotic arm, a vacuum cleaner and obstacle detector sensors, object tracking sensors, line tracking sensors and manual controlling ( Bluetooth ) has been integrated and controlled by a smart remote, which is useful in cleaning our home.

The project is comprised of basically three modules, which handles all the basic functionalities of the robot. The modules which we have integrated in this project are:

1. The controller part,

2. The Robot and

3. The Vacuum Cleaner.

# **2.INTRODUCTION :**

## 

## PRESENT SCENARIO :

# The tedious process of floor cleaning has become labor demanding and time consuming task.

# Also as the activity is manual so scheduling, regularity and quality cannot be maintained.

# The elderly who lives by themselves do not have the strength or ability to clean by themselves, and the cost of hiring a maid is expensive.

## 

## PROPOSED SYSTEM:

To overcome this issues we are proposing a machine which will timely clean the floor and maintain cleanliness throughout the day, which will reduce human efforts.

It is also able to identify obstacles, perform timely operation, optimize uses of cleaning detergents and water, and reduce noise generated during cleaning.

# 

# **3.COMPONENTS :**

1.ESP 32

2.Arduino Nano

3.Motor Driver

4.Ultrasonic sensor

5.Bluetooth Module

6.MPU6050

7.OLED Screen

8.Proximitty Sensor Circuit

LM358\*4 units

IR ( Transmitter )\*8

Photo Diode\*8 units

10K Variable Resistor\*8 units

( 220, 360, 10K ) ohm’s

9.LED Flasher circuit

NE555

100UF capacitor

220K ohm resistor

100k ohm Variable resistor

10.Switch

TIP122\*4

1k ohm resistor

Relay

BC547 Transistor

1N4007 Diode

1k resistor

11.Buzzer

12.Vacuum Motor with Fan

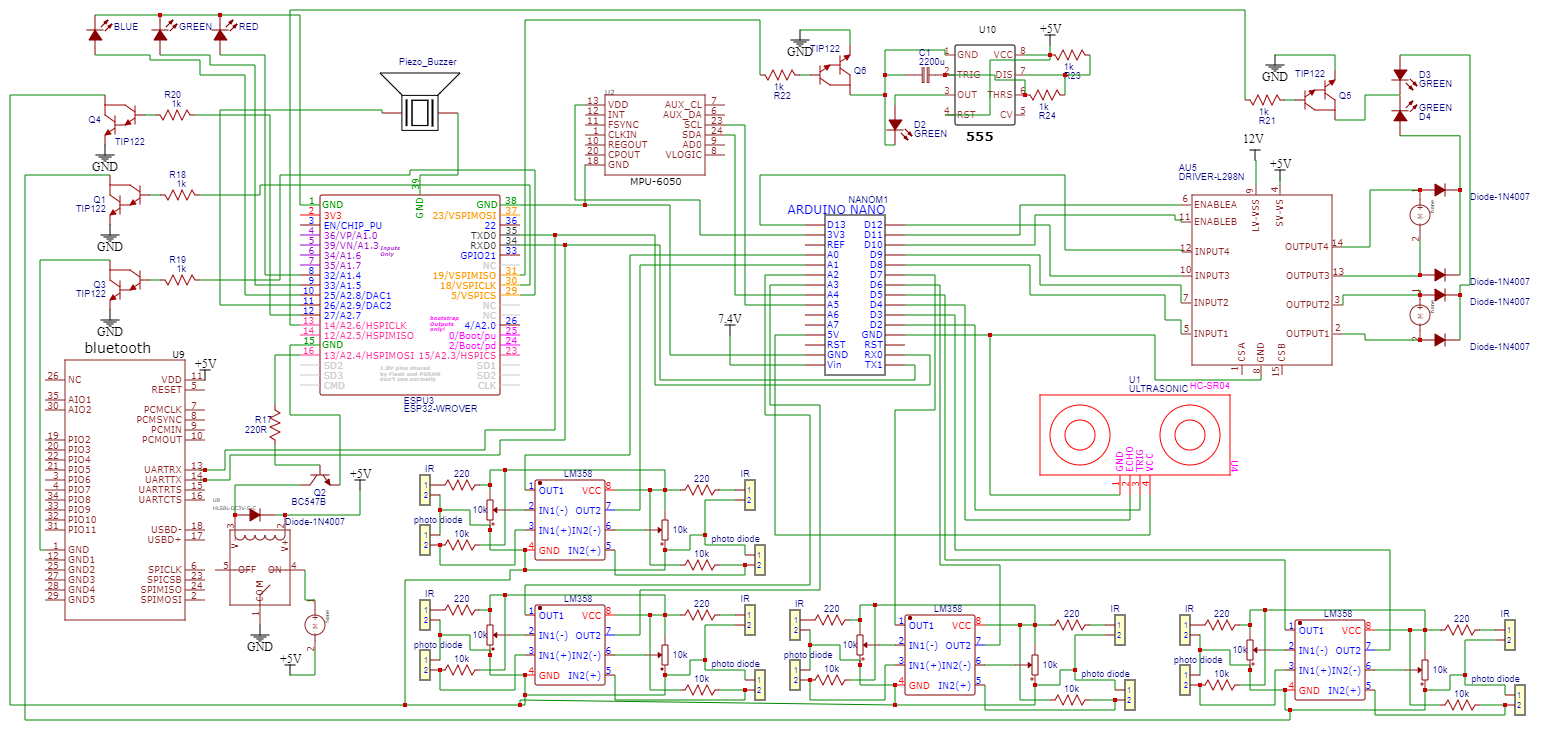
13.IR Receiver and Push button

14. Neo Pixel RGB LED\*3 units

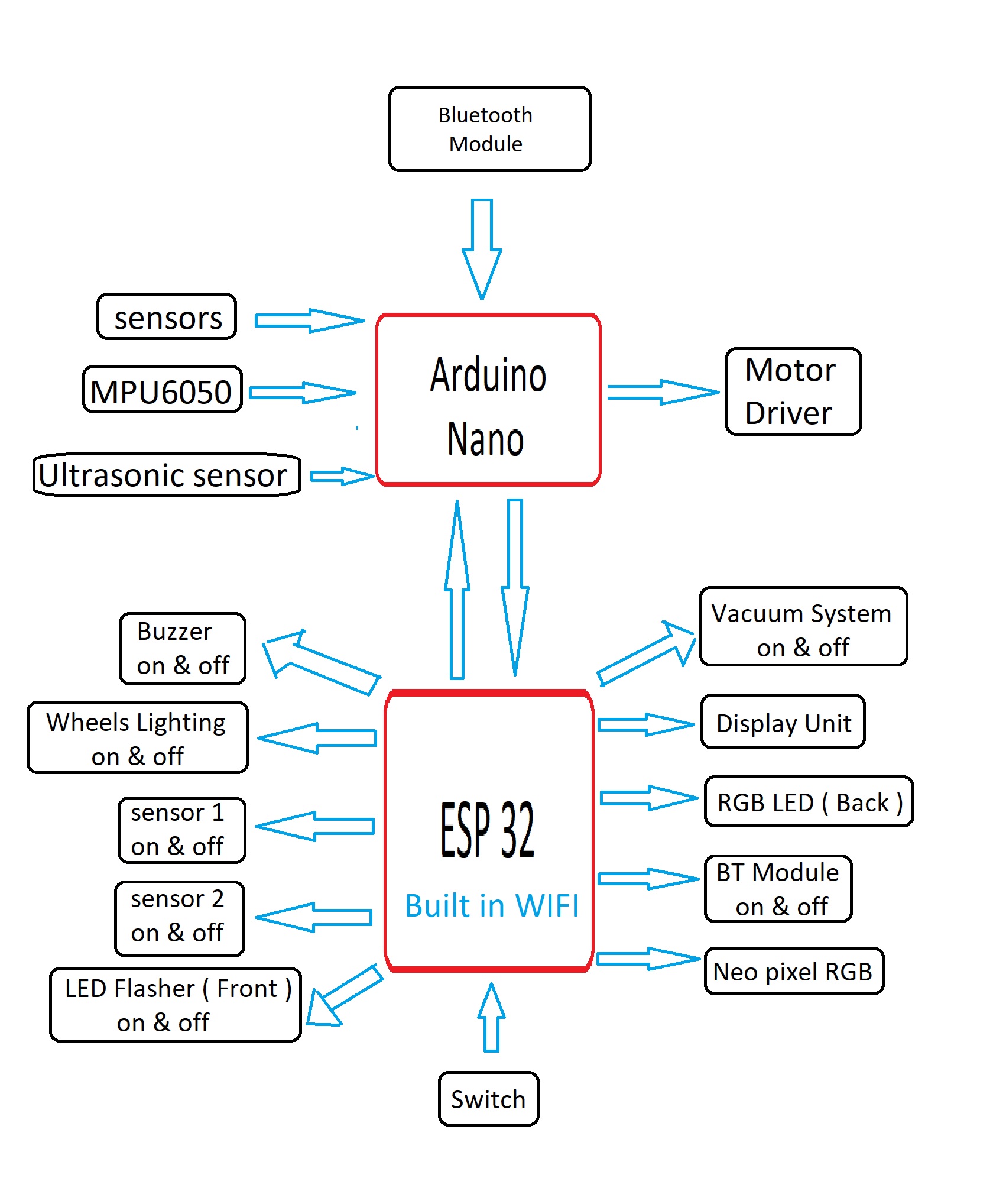
15.5mm RGB LED\*5

16.Battery ( 3.7V, 7.4V, 12V )

4.**CIRCUIT DIAGRAM** :



# **5.BLOCK DIAGRAM OF SMART VACUUM CLEANER**



# **6.WORKING PROCESS:**

It has a disk shape, sucks dirt via a retractable dustbin on top of which a cooling fan is mounted. The suction fan helps create vacuum that attracts dirt to the dustbin.The robot navigates with a front caster wheel and two rear wheels, and detects obstacles using the ultrasonic sensors.

# 7.**APPLICATIONS :**

Robotic Vacuum Cleaner is developed to make cleaning process easier especially for working people.

This Robotic Vacuum Cleaner is designed for specific area such as under beds.

# **8.Conclusion**:

A cheaper and user friendly Vacuum Cleaner robot can be developed with four different mode of controlling

( Manual, Autonomous mode,Object tracking Mode and Line Tracking ) using an Arduino Board and ESP 32 with more electronics functionality. Battery monitoring, self-charging, lighter body weight and to set alarm on/off time manually are the future scope of this project.

# **APPENDIX**

# **ESP 32 :**

#include <Wire.h>

#include <Adafruit\_GFX.h>

#include <Adafruit\_SSD1306.h>

#include <WiFi.h>

#include <WiFiClient.h>

#include <BlynkSimpleEsp32.h>

#include <FastLED.h>

#include <IRremote.h>

#define BLYNK\_PRINT Serial

#define OLED\_RESET 4

#define R\_channel 0

#define G\_channel 1

#define B\_channel 2

#define pwm\_Frequency 5000 // pwm frequency

#define pwm\_resolution 8 // 8 bit resolution

#define sensor1 27

#define sensor2 18

#define wheels\_led 14

#define front\_led 19

#define buzzer 26

#define input1\_pb 35

#define input2\_IR 34

#define LEDG 33

#define LEDR 32

#define LEDB 25

#define bluetooth 5

#define motor 16

#define LED\_PIN 17

#define NUM\_LEDS 3

int motor\_app\_data, bt\_app\_data, Front\_rgb\_led\_app\_data, Front\_led\_app\_data, wheels\_led\_app\_data,qwe,temp;

int r, g, b, x, y, z, sensor1\_d, sensor2\_d, motor\_on\_off\_data, bt\_out\_data, wifi\_inp, disp\_inp, c, c\_app;

char rC;

char auth[] = "xkevKTbSwiAfFTw3ooO5tUlpUyn\_CRu5"; //Authentication Code sent by Blynk to Mail ID

char ssid[]="Balu";

char pass[]="9494004099";

String s\_1\_inp, s\_2\_inp, motor\_s\_app\_data, BT\_\_data\_inp, terminal\_out\_data;

String Front\_led\_s\_data, wheels\_led\_s\_data, rgbb\_d\_inp, rgbf\_d\_inp;

bool pb;

uint8\_t hue = 0;

WidgetTerminal terminal(V1);

Adafruit\_SSD1306 display( 128, 64, &Wire, OLED\_RESET);

CRGB leds[NUM\_LEDS];

const unsigned char myBitmap [] PROGMEM = {

0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xf0, 0x0f, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0x00, 0x00, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xf8, 0x00, 0x00, 0x1f, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xe0, 0x00, 0x00, 0x07, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0x80, 0x00, 0x00, 0x01, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0x00, 0x00, 0x00, 0x00, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xfc, 0x00, 0x00, 0x00, 0x00, 0x3f, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xf8, 0x00, 0x00, 0x00, 0x00, 0x1f, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xf0, 0x00, 0x00, 0x00, 0x00, 0x0f, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xe0, 0x00, 0x00, 0x00, 0x00, 0x07, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xc0, 0x00, 0x00, 0x00, 0x00, 0x03, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xc0, 0x00, 0x00, 0x0f, 0xfc, 0x01, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0x87, 0xe0, 0x0f, 0x9f, 0xfe, 0x01, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0x07, 0xf0, 0x0f, 0x9f, 0xfc, 0x00, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0x07, 0xf0, 0x0f, 0x9f, 0xf8, 0x00, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xfe, 0x07, 0xf8, 0x0f, 0x9e, 0x00, 0x00, 0x7f, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xfe, 0x07, 0xf8, 0x0f, 0x9c, 0x00, 0x00, 0x7f, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xfc, 0x07, 0xfc, 0x0f, 0x98, 0x00, 0x00, 0x3f, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xfc, 0x07, 0xfe, 0x0f, 0x98, 0x00, 0x00, 0x3f, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xfc, 0x07, 0xfe, 0x0f, 0x98, 0x00, 0x00, 0x3f, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xfc, 0x07, 0xdf, 0x0f, 0x98, 0x00, 0x00, 0x1f, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xf8, 0x07, 0xcf, 0x0f, 0x88, 0x00, 0x00, 0x1f, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xf8, 0x07, 0xcf, 0x8f, 0x8a, 0x7e, 0x00, 0x1f, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xf8, 0x07, 0xc7, 0xcf, 0x02, 0xff, 0x80, 0x1f, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xf8, 0x07, 0xc7, 0xce, 0x20, 0xff, 0xc0, 0x1f, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xf8, 0x07, 0xc3, 0xee, 0x38, 0x7f, 0xe0, 0x1f, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xf8, 0x07, 0xc3, 0xe3, 0x20, 0x0f, 0xf0, 0x1f, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xfc, 0x07, 0xc1, 0xf3, 0xc0, 0x07, 0xf0, 0x1f, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xfc, 0x07, 0xc0, 0xd3, 0x80, 0x03, 0xf0, 0x3f, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xfc, 0x07, 0xc0, 0x12, 0x00, 0x03, 0xf0, 0x3f, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xfc, 0x07, 0xc0, 0x2e, 0x18, 0x03, 0xf0, 0x3f, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xfe, 0x07, 0xc0, 0x1c, 0x1c, 0x07, 0xf0, 0x7f, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xfe, 0x07, 0xc0, 0x03, 0x1f, 0xbf, 0xe0, 0x7f, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0x07, 0xc0, 0x1f, 0x1f, 0xff, 0xe0, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0x07, 0xc0, 0x1f, 0x1f, 0xff, 0xc0, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0x87, 0xc0, 0x0f, 0x1f, 0xff, 0x01, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0x80, 0x00, 0x00, 0x0f, 0xfe, 0x01, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xc0, 0x00, 0x00, 0x00, 0x00, 0x03, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xe0, 0x00, 0x00, 0x00, 0x00, 0x07, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xf0, 0x00, 0x00, 0x00, 0x00, 0x0f, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xf8, 0x00, 0x00, 0x00, 0x00, 0x1f, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xfc, 0x00, 0x00, 0x00, 0x00, 0x3f, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0x00, 0x00, 0x00, 0x00, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0x80, 0x00, 0x00, 0x01, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xe0, 0x00, 0x00, 0x07, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xf8, 0x00, 0x00, 0x1f, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xfe, 0x00, 0x00, 0x7f, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xf8, 0x1f, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,

0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff

};

void setup() {

pinMode( sensor1, OUTPUT);

pinMode( sensor2, OUTPUT);

pinMode( wheels\_led,OUTPUT);

pinMode( front\_led, OUTPUT);

pinMode( buzzer, OUTPUT);

pinMode( bluetooth, OUTPUT);

pinMode( motor, OUTPUT);

pinMode( input1\_pb, INPUT);

pinMode( input2\_IR, INPUT);

pinMode( LEDR, OUTPUT);

pinMode( LEDG, OUTPUT);

pinMode( LEDB, OUTPUT);

Serial.begin( 9600 );

WiFi.begin( ssid, pass);

Blynk.config( auth ); //WIFI Name and Password

FastLED.addLeds<WS2812B, LED\_PIN, GRB>( leds, NUM\_LEDS);

FastLED.setBrightness( 100 );

ledcAttachPin( LEDR, R\_channel);

ledcAttachPin( LEDG, G\_channel);

ledcAttachPin( LEDB, B\_channel);

ledcSetup( R\_channel, pwm\_Frequency, pwm\_resolution);

ledcSetup( G\_channel, pwm\_Frequency, pwm\_resolution);

ledcSetup( B\_channel, pwm\_Frequency, pwm\_resolution);

Front\_led\_s\_data="OFF"; wheels\_led\_s\_data="OFF"; rgbb\_d\_inp="OFF"; rgbf\_d\_inp="OFF";

s\_1\_inp="OFF"; s\_2\_inp="OFF"; motor\_s\_app\_data="OFF"; BT\_\_data\_inp="OFF"; c=0;

Front\_rgb\_led\_app\_data = 1;qwe=1,temp=9;

if(!display.begin( SSD1306\_SWITCHCAPVCC, 0x3C )) // Address 0x3C for most of these displays, if doesn't work try 0x3D

{ Serial.println( F( "SSD1306 allocation failed" )); for(;;); }

display.clearDisplay(); //for Clearing the display

display.setTextColor( SSD1306\_WHITE );

display.drawBitmap( 0, 0, myBitmap, 128, 64, WHITE);

// display.drawBitmap( x position, y position, bitmap data, bitmap width, bitmap height, color);

display.display();

delay(3000);

display.clearDisplay();

display.setCursor( 15, 18); display.setTextSize(2);

display.print(F("NS Ideas"));

display.setCursor( 21, 43); display.setTextSize(1);

display.print(F("...Presents..."));

display.display();

delay(2000);

display.clearDisplay();

display.setCursor( 34, 1); display.setTextSize(2);

display.print(F("SMART"));

display.setCursor( 27, 23); display.setTextSize(2);

display.print(F("VACCUM"));

display.setCursor( 21, 44); display.setTextSize(2);

display.print(F("CLEANER"));

display.display();

delay(3000);

display.clearDisplay();

display.setCursor( 5, 5); display.setTextSize(1);

display.print(F("Initializing......"));

display.setCursor( 27, 22); display.setTextSize(1);

display.print(F("Arduino Nano"));

display.setCursor( 30, 32); display.setTextSize(1);

display.print(F("All Sensors"));

display.setCursor( 21, 42); display.setTextSize(1);

display.print(F("WI-FI Settings"));

display.setCursor( 8, 52); display.setTextSize(1);

display.print(F("Bluetooth Settings"));

display.display();

delay(2000);

display.clearDisplay();

display.setCursor( 0, 20); display.setTextSize(1);

display.print(F("Calculating gyro offset, do not move MPU6050 or VACCUM CLEANER"));

display.display();

delay(2000);

}

void menu( int mode\_disp\_data){

display.clearDisplay();

display.setTextSize(1);

display.setCursor( 30, 0);

display.print("Select Mode");

if ( mode\_disp\_data == 0 ) { display.setCursor( 5, 14); display.print(">> Rest"); }

else { display.setCursor( 30, 14); display.print("Rest"); }

if ( mode\_disp\_data == 1 ) { display.setCursor( 5, 24); display.print("> 1.Auto"); }

else { display.setCursor( 18, 24); display.print("1.Auto"); }

if ( mode\_disp\_data == 2 ) { display.setCursor( 5, 34); display.print("> 2.Object Tracking"); }

else { display.setCursor( 18, 34); display.print("2.Object Tracking"); }

if ( mode\_disp\_data == 3 ) { display.setCursor( 5, 44); display.print("> 3.Line Tracking "); }

else { display.setCursor( 18, 44); display.print("3.Line Tracking"); }

if ( mode\_disp\_data == 4 ) { display.setCursor( 5, 54); display.print("> 4.Manual"); }

else { display.setCursor( 18, 54); display.print("4.Manual"); }

display.display();

}

void mode\_( int mode\_disp\_data, int wifi\_disp\_data){

display.clearDisplay();

display.setTextSize(3);

if ( mode\_disp\_data == 0 ) { display.setCursor( 30, 16); display.print("REST"); }

if ( mode\_disp\_data == 1 ) { display.setCursor( 30, 16); display.print("AUTO"); }

if ( mode\_disp\_data == 2 ) { display.setTextSize(2);

display.setCursor( 29, 5); display.print("OBJECT");

display.setCursor( 14, 28); display.print("TRACKING");}

if ( mode\_disp\_data == 3 ) { display.setTextSize(2);

display.setCursor( 35, 5); display.print("LINE");

display.setCursor( 14, 28); display.print("TRACKING");}

if ( mode\_disp\_data == 4 ) { display.setCursor( 11, 16); display.print("MANUAL"); }

if ( wifi\_disp\_data == 1 ) { display.setTextSize(1);display.setCursor( 3, 55);

display.print("...WIFI-CONNECTED...");

}

else if(wifi\_disp\_data==0){ display.setTextSize(1);display.setCursor(5,55);

display.print(".WIFI-NOT CONNECTED.");

}

display.display();

}

void display\_oled\_disp1( String s1\_disp\_data, String s2\_disp\_data, String Motor\_disp\_data, String Motor\_display\_data){

display.clearDisplay();

display.setTextSize(1);

display.setCursor( 44, 0); display.print("STATUS");

display.setCursor( 3, 18); display.print("MOTOR");

display.setCursor( 90, 18); display.print(":");

display.setCursor( 100, 18); display.print(Motor\_disp\_data);

display.setCursor( 3, 29); display.print("BLUETOOTH");

display.setCursor( 90, 29); display.print(":");

display.setCursor( 100, 29); display.print(Motor\_display\_data);

display.setCursor( 3, 40); display.print("SENSOR 1");

display.setCursor( 90, 40); display.print(":");

display.setCursor( 100, 40); display.print(s1\_disp\_data);

display.setCursor( 3, 51); display.print("SENSOR 2");

display.setCursor( 90, 51); display.print(":");

display.setCursor( 100, 51); display.print(s2\_disp\_data);

display.display();

}

void display\_oled\_disp2( String frontLED\_disp\_data, String downLED\_disp\_data, String rgbBack\_disp\_data, String rgbFront\_disp\_data ){

display.clearDisplay();

display.setTextSize(1);

display.setCursor( 44, 0); display.print("STATUS");

display.setCursor( 3, 18); display.print("FRONT LED'S");

display.setCursor( 90, 18); display.print(":");

display.setCursor( 100, 18); display.print(frontLED\_disp\_data);

display.setCursor( 3, 29); display.print("WHEEL'S LED");

display.setCursor( 90, 29); display.print(":");

display.setCursor( 100, 29); display.print(downLED\_disp\_data);

display.setCursor( 3, 40); display.print("RGB BACK");

display.setCursor( 90, 40); display.print(":");

display.setCursor( 100, 40); display.print(rgbBack\_disp\_data);

display.setCursor( 3, 51); display.print("RGB FRONT");

display.setCursor( 90, 51); display.print(":");

display.setCursor( 100,51); display.print(rgbFront\_disp\_data);

display.display();

}

void switch\_on\_off(bool sensor1\_out\_data,bool sensor2\_out\_data,bool wheels\_led\_out\_data,bool front\_led\_out\_data,bool bluetooth\_out\_data,bool motor\_out\_data){

digitalWrite( sensor1, sensor1\_out\_data);

digitalWrite( sensor2, sensor2\_out\_data);

digitalWrite( wheels\_led, wheels\_led\_out\_data);

Blynk.virtualWrite( V35, wheels\_led\_out\_data);

digitalWrite( front\_led, front\_led\_out\_data);

Blynk.virtualWrite( V34, front\_led\_out\_data);

digitalWrite( bluetooth, bluetooth\_out\_data);

Blynk.virtualWrite( V32, bluetooth\_out\_data);

digitalWrite( motor, motor\_out\_data);

Blynk.virtualWrite( V31, motor\_out\_data);

}

void RGB\_Color( int i, int j, int k) {

ledcWrite( R\_channel, i);

ledcWrite( G\_channel, j);

ledcWrite( B\_channel, k);

}

void rgb\_led1(){//for wifi not connected

for(int i = 0; i < NUM\_LEDS; i++) { leds[i] = CRGB::Red; FastLED.show();

delay(100); leds[i] = CRGB::Black;

}

for(int i = NUM\_LEDS - 1; i >= 0; i--) { leds[i] = CRGB::Red; FastLED.show();

delay(100); leds[i] = CRGB::Black;

}

}

void rgb\_led2(){//mode changed

for (int i = 0; i < NUM\_LEDS; i++) { leds[i] = CHSV(hue + (i \* 10), 255, 255); }

EVERY\_N\_MILLISECONDS(15){ hue++; } FastLED.show();

}

void rgb\_led3(){//bluetooth mode

EVERY\_N\_MILLISECONDS(50) { leds[0] = CHSV(160, random8(), random8(100, 255));

for (int i = NUM\_LEDS - 1; i > 0; i--) { leds[i] = leds[i - 1]; }} FastLED.show();

}

void rgb\_led4(){//for wifi connected

for(int i = 0; i < NUM\_LEDS; i++) { leds[i] = CRGB::Green; FastLED.show();

delay(100); leds[i] = CRGB::Black; }

for(int i = NUM\_LEDS - 1; i >= 0; i--) { leds[i] = CRGB::Green; FastLED.show();

delay(100); leds[i] = CRGB::Black; }

}

void rgb\_led5(){//auto

for(int i = 0; i < NUM\_LEDS; i++) { leds[i] = CRGB::Orange; FastLED.show();

delay(100); leds[i] = CRGB::Black; }

for(int i = NUM\_LEDS - 1; i >= 0; i--) { leds[i] = CRGB::Orange; FastLED.show();

delay(100); leds[i] = CRGB::Black; }

}

void rgb\_led6(){//odject tracking mode

for(int i = 0; i < NUM\_LEDS; i++) { leds[i] = CRGB::Gray; FastLED.show();

delay(100); leds[i] = CRGB::Black; }

for(int i = NUM\_LEDS - 1; i >= 0; i--) {leds[i] = CRGB::Gray; FastLED.show();

delay(100); leds[i] = CRGB::Black; }

}

void rgb\_led7(){//line tracking mode

for(int i = 0; i < NUM\_LEDS; i++) { leds[i] = CRGB::Yellow; FastLED.show();

delay(100); leds[i] = CRGB::Black; }

for(int i = NUM\_LEDS - 1; i >= 0; i--) { leds[i] = CRGB::Yellow; FastLED.show();

delay(100); leds[i] = CRGB::Black; }

}

void rgb\_led8(){//rest stop

for(int i = 0; i < NUM\_LEDS; i++) { leds[i] = CRGB::White; FastLED.show();

delay(100); leds[i] = CRGB::Black; }

for(int i = NUM\_LEDS - 1; i >= 0; i--) {leds[i] = CRGB::White; FastLED.show();

delay(100); leds[i] = CRGB::Black; }

}

BLYNK\_WRITE(V31){

motor\_app\_data = param.asInt();

motor\_on\_off\_data = motor\_app\_data;

if( motor\_app\_data == 1 ) motor\_s\_app\_data = "ON";

else motor\_s\_app\_data = "OFF";

}

BLYNK\_WRITE(V32){

bt\_app\_data = param.asInt();

bt\_out\_data = bt\_app\_data;

if( bt\_app\_data == 1 ){ BT\_\_data\_inp = "ON"; bt\_out\_data = 1; }

else {BT\_\_data\_inp = "OFF"; bt\_out\_data = 0; }

}

BLYNK\_WRITE(V33){

Front\_rgb\_led\_app\_data = param.asInt();

}

BLYNK\_WRITE(V34){

Front\_led\_app\_data = param.asInt();

if( Front\_led\_app\_data == 1 ) Front\_led\_s\_data = "ON";

else Front\_led\_s\_data = "OFF";

}

BLYNK\_WRITE(V35){

wheels\_led\_app\_data = param.asInt();

if( wheels\_led\_app\_data == 1 ) wheels\_led\_s\_data = "ON";

else wheels\_led\_s\_data = "OFF";

}

BLYNK\_WRITE(V11){

r = param[0].asInt();

g = param[1].asInt();

b = param[2].asInt();

RGB\_Color(r,g,b);

}

BLYNK\_WRITE(V1){

terminal\_out\_data = param.asStr();

if( terminal\_out\_data == "stop" ){ c=0; }

else if( terminal\_out\_data == "rest"){ c=0; }

else if( terminal\_out\_data == "auto"){ c=1; }

else if( terminal\_out\_data == "object tracking"){ c=2; }

else if( terminal\_out\_data == "line tracking"){ c=3; }

else if( terminal\_out\_data == "manual"){ c=4; }

else if( terminal\_out\_data == "object"){ c=2; }

else if( terminal\_out\_data == "obj"){ c=2; }

else if( terminal\_out\_data == "line"){ c=3; }

else if( terminal\_out\_data == "bluetooth"){ c=4; }

else if( terminal\_out\_data == "manual"){ c=4; }

}

BLYNK\_WRITE(V3){

c\_app = param.asInt(); c = c\_app; }

void loop() {

if(WiFi.status()!=WL\_CONNECTED){ Serial.println("Not connected"); rgb\_led1(); wifi\_inp=0;qwe=1;}

else {

Serial.println("Connected"); Blynk.run(); wifi\_inp=1;

if((Front\_rgb\_led\_app\_data==1)&&(qwe==1)){ rgb\_led4(); rgb\_led4(); qwe=0;}

else {

leds[0] = CRGB::Black; FastLED.show();

leds[1] = CRGB::Black; FastLED.show();

leds[2] = CRGB::Black; FastLED.show();

}

}

pb = digitalRead(input1\_pb);

if( pb == HIGH ){ c++; if(c>4) c=0; Blynk.virtualWrite(V3,c);}

if(Serial.available()){

rC = Serial.read();

if( rC == 'a'){c=1;}//auto mode

else if( rC == 'b'){ c = 2; }//human tracking mode

else if( rC == 'c'){ c = 3; }//line tracking mode

else if( rC == 'd'){ c = 4; }//bluetooth mode

else if( rC == 's'){ c = 0; }//stop

Blynk.virtualWrite( V3, c);

if(c>5){ c = 0;}

}

if(c!=temp){rgb\_led2();delay(200);rgb\_led2();delay(200);rgb\_led2();}

temp = c;

switch(c){

case 0 : disp\_inp = 0; //rest mode 27

Serial.println('s');

sensor1\_d = 0; sensor2\_d = 0; bt\_out\_data = 0; motor\_on\_off\_data = 0;

s\_1\_inp="OFF"; s\_2\_inp="OFF"; motor\_s\_app\_data="OFF"; BT\_\_data\_inp="OFF";

terminal.println("REST MODE");

c=5;

break;

case 1 : disp\_inp=1;//auto mode

Serial.println('a');

sensor1\_d=1; sensor2\_d=0; bt\_out\_data=0; motor\_on\_off\_data=1;

s\_1\_inp="ON"; s\_2\_inp="OFF"; motor\_s\_app\_data="ON"; BT\_\_data\_inp="OFF";

terminal.println("AUTO MODE");

c=5;

break;

case 2 : disp\_inp=2;//human tracking mode

Serial.println('b');

sensor1\_d=1; sensor2\_d=0; bt\_out\_data=0; motor\_on\_off\_data=1;

s\_1\_inp="ON"; s\_2\_inp="OFF"; motor\_s\_app\_data="ON"; BT\_\_data\_inp="OFF";

terminal.println("OBJECT TRACKING MODE");

c=5;

break;

case 3 : disp\_inp=3;//line tracking mode

Serial.println('c');

sensor1\_d=0; sensor2\_d=1; bt\_out\_data=0; motor\_on\_off\_data=1;

s\_1\_inp="OFF"; s\_2\_inp="ON"; motor\_s\_app\_data="ON"; BT\_\_data\_inp="OFF";

terminal.println("LINE TRACKING MODE");

c=5;

break;

case 4 : disp\_inp=4; //bluetooth mode ||anual

Serial.println('d');

sensor1\_d=0; sensor2\_d=0; bt\_out\_data=1; motor\_on\_off\_data=1;

s\_1\_inp="OFF"; s\_2\_inp="OFF"; motor\_s\_app\_data="ON"; BT\_\_data\_inp="ON";

terminal.println("BLUETOOTH MODE ");

c=5;

break;

}

if(Front\_rgb\_led\_app\_data==1){

if(c==0)rgb\_led8();

else if( c == 1)rgb\_led5();

else if( c == 2)rgb\_led6();

else if( c == 3)rgb\_led7();

else if( c == 4)rgb\_led3();

}

switch\_on\_off(sensor1\_d,sensor2\_d,wheels\_led\_app\_data,Front\_led\_app\_data,bt\_out\_data,motor\_on\_off\_data);

if( z>=0 && z<2 ){ menu(disp\_inp); }

else if( z>=2 && z<6){ mode\_(disp\_inp,wifi\_inp); }

else if( z>=6 && z<8){ display\_oled\_disp1(s\_1\_inp,s\_2\_inp,motor\_s\_app\_data, BT\_\_data\_inp); }

else if( z>=8 && z<10){ display\_oled\_disp2(Front\_led\_s\_data,wheels\_led\_s\_data,rgbb\_d\_inp,rgbf\_d\_inp); }

x = millis();

if( x-y >= 1000 ){ y=x; z++; if( z>10 ){ z=0; } y=x; }

}

# Arduino Nano Code :

#define IN2 13

#define IN1 12

#define IN4 9

#define IN3 8

#define E2 10

#define E1 11

#define a4 7

#define a5 6

#define a6 5

#define echoPin 2

#define a7 3

#define trigPin 4

#define TRIGGER\_PIN A1

#define ECHO\_PIN A0

#define MAX\_DISTANCE 100

#include<NewPing.h>

#include <Wire.h>

#include <MPU6050\_light.h>

MPU6050 mpu(Wire);

unsigned long currentTime;

unsigned long previousTime=0;

const int eventInterval=50;

int q=0;

char rC;

int c = 3,timex,now;

int speedE1=250;

int speedE2 =speedE1-70 ;

unsigned long timer = 0;

unsigned int dist;

float temp ,t,Z;

bool ir1, ir2, ir3, ir4, ir5, ir6, ir7, ir8;

long duration, average;

long aver[3];

NewPing sonar(TRIGGER\_PIN, ECHO\_PIN, MAX\_DISTANCE);

void setup(){

Serial.begin(9600);

pinMode(A0, INPUT);

pinMode(A1, INPUT);

pinMode(A2, INPUT);

pinMode(A3, INPUT);

pinMode(a4, INPUT);

pinMode(a5, INPUT);

pinMode(a6, INPUT);

pinMode(a7, INPUT);

pinMode(trigPin, OUTPUT);

pinMode(echoPin, INPUT);

pinMode(E1, OUTPUT);

pinMode(E2, OUTPUT);

pinMode(IN4, OUTPUT);

pinMode(IN3, OUTPUT);

pinMode(IN2, OUTPUT);

pinMode(IN1, OUTPUT);

}

void Bot(bool IN\_1,bool IN\_2,int speedE1,bool IN\_3,bool IN\_4,int speedE2){

digitalWrite(IN1,IN\_1);

digitalWrite(IN2,IN\_2);

analogWrite(E1,speedE1);

digitalWrite(IN3,IN\_3);

digitalWrite(IN4,IN\_4);

analogWrite(E2,speedE2);

}

void midObj(){

ir6 = digitalRead(a5);

ir5 = digitalRead(a4);

if(ir6==HIGH){

mpu.update(); Z=mpu.getAngleZ();

temp = Z-88;

rotation(temp);

Bot(HIGH,LOW,200,HIGH,LOW,200);

mpu.update(); Z=mpu.getAngleZ();

temp=Z+88;

rotation(temp);

Bot(HIGH,LOW,200,HIGH,LOW,200);

mpu.update(); Z=mpu.getAngleZ();

temp=Z+88;

rotation(temp);

Bot(HIGH,LOW,200,HIGH,LOW,200);

mpu.update(); Z=mpu.getAngleZ();

temp=Z-88;

rotation(temp);

Bot(HIGH,LOW,200,HIGH,LOW,200);

}

else if(ir5==HIGH){

mpu.update(); Z=mpu.getAngleZ();

temp = Z+88;

rotation(temp);

Bot(HIGH,LOW,200,HIGH,LOW,200);

mpu.update(); Z=mpu.getAngleZ();

temp=Z-88;

rotation(temp);

Bot(HIGH,LOW,200,HIGH,LOW,200);

mpu.update(); Z=mpu.getAngleZ();

temp=Z-88;

rotation(temp);

Bot(HIGH,LOW,200,HIGH,LOW,200);

mpu.update(); Z=mpu.getAngleZ();

temp=Z+88;

rotation(temp);

Bot(HIGH,LOW,200,HIGH,LOW,200);

}

}

void rotation(float temp ){

timex = 6;

now=millis();

while(true){

if((millis()-now)>timex){break;}

if((millis()-timer)>1){ mpu.update(); Z=mpu.getAngleZ(); timer = millis(); }

if ( Z <= temp && Z >= temp+4 ){ Bot(LOW,LOW,0,LOW,LOW,0);}

else if ( Z< temp+4 ){ Bot(HIGH,LOW,255,LOW,HIGH,255); }

else { Bot(LOW,HIGH,255,HIGH,LOW,255); }

}

}

void loop(){

int c = 3;

if(Serial.available()){

rC = Serial.read();

if(rC == 'a'){c=0;Serial.println('a');}//auto mode

else if(rC == 'b'){c=1;Serial.println('b');}//human tracking mode

else if(rC == 'c'){c=2;Serial.println('c');}//line tracking mode

else if(rC == 'd'){c=3;Serial.println('d');}//bluetooth mode

else if(rC == 'e'){c=4;Serial.println('e');}//stop

else if(c>4){c=0;}

}

currentTime = millis();

if(currentTime - previousTime>= eventInterval){

if(Serial.available()){

Serial.read();

}

previousTime = currentTime;

}

switch(c){

case 0 :

dist= sonar.ping\_cm();

if (dist == 0) {dist = 40;}

ir1 = digitalRead(A0);

ir2 = digitalRead(A1);

if ( dist > 13 && ir1 == false && ir2 == false ){Bot(HIGH,LOW,200,HIGH,LOW,200); }

dist= sonar.ping\_cm();

if (dist == 0) { dist = 40; }ir1 = digitalRead(A0);ir2 = digitalRead(A1);

if ( dist < 13 && ir1 == true && ir2 == true ){

ir3 = digitalRead(A2); ir4 = digitalRead(A3);

if ( ir3 == true && ir4 == false ){ Bot(LOW,LOW,200,LOW,LOW,200); }

ir3 = digitalRead(A2); ir4 = digitalRead(A3);

if ( ir4 == true && ir3 == false ){ Bot(LOW,HIGH,200,HIGH,LOW,200); }

}

dist= sonar.ping\_cm();

if (dist == 0) {dist = 40;}

ir1 = digitalRead(A0); ir2 = digitalRead(A1);

if ( dist<5 && ir1==false && ir2==false ){ /\*midObj();\*/ }

dist = sonar.ping\_cm();

if (dist == 0) {dist = 40;}

ir1 = digitalRead(A0); ir2 = digitalRead(A1);

if ( ir1==false || (dist<5 && ir2==true )){

temp = Z-45;//slightly left

rotation( temp );

Bot(HIGH,LOW,200,HIGH,LOW,200);

}

dist= sonar.ping\_cm();

if (dist == 0) {dist = 40;}

ir1 = digitalRead(A0);

ir2 = digitalRead(A1);

if ( ir2==false || dist<5 && ir1==true ){

temp = Z+45;//slightly left

rotation( temp );

Bot(HIGH,LOW,200,HIGH,LOW,200);

}

ir5 = digitalRead(a4);

if (ir5 == false){ Serial.print("backRight");

Bot(LOW,HIGH,200,LOW,HIGH,200);

temp = Z+45;//slightly Right

rotation( temp );

Bot(HIGH,LOW,200,HIGH,LOW,200);}

ir6 = digitalRead(a5);

if (ir6 == false){ Serial.print("backLeft");

Bot(LOW,HIGH,200,LOW,HIGH,200);

temp = Z-45;//slightly left

rotation( temp );

Bot(HIGH,LOW,200,HIGH,LOW,200);}

break;

case 1 :

dist= sonar.ping\_cm();

if (dist == 0) {dist = 40;}

Serial.print(dist);

Serial.print(" ");

Serial.print(ir1);

Serial.print(" ");

Serial.println(ir2);

if(((dist>=7)&&(dist < 14))||dist>=30){ Bot( LOW, LOW, 0, LOW, LOW, 0); }

dist= sonar.ping\_cm();

if (dist == 0) {dist = 40;}

ir2 = digitalRead(A1);

if((ir2==1) && (dist>=15)){ Bot(HIGH, LOW,255, LOW,HIGH,100); } //slightly right

dist= sonar.ping\_cm();

if (dist == 0) {dist = 40;}

ir1 = digitalRead(A0);

if((dist>=15)&&(ir1==1)){ Bot( LOW,HIGH,100,HIGH, LOW,255); } //slightly left

dist= sonar.ping\_cm();

if (dist == 0) {dist = 40;}

if(dist>=25 && dist<=29){ Bot(HIGH,LOW,255,HIGH,LOW,255); } //stright with high speed

dist= sonar.ping\_cm();

if (dist == 0) {dist = 40;}

if(dist>=20 && dist<=24){ Bot(HIGH,LOW,200,HIGH,LOW,200); } //stright with medium speed

dist= sonar.ping\_cm();

if (dist == 0) {dist = 40;}

else if(dist>=15 && dist<=19){ Bot(HIGH,LOW,150,HIGH,LOW,150); } //stright with low speed

ir1 = digitalRead(A0); ir2 = digitalRead(A1);

if((ir2==0) && (ir1==1)) { Bot(LOW,HIGH,100,HIGH,LOW,255); } //left

ir1 = digitalRead(A0); ir2 = digitalRead(A1);

if((ir2==1)&&(ir1==0)) { Bot(HIGH,LOW,255,LOW,HIGH,100); } //right.

dist= sonar.ping\_cm();

if (dist == 0) {dist = 40;}

if((dist>=0 && dist<=7)|| (dist>=0 && dist<=8)){ Bot(LOW,HIGH,255,LOW,HIGH,255); } //back

dist= sonar.ping\_cm();

if (dist == 0) {dist = 40;}

ir1 = digitalRead(A0);

if((dist<=6)&&(ir1==1)){ Bot( LOW,HIGH,100, LOW,HIGH,255);} //slightly back left

dist= sonar.ping\_cm();

if (dist == 0) {dist = 40;}

ir1 = digitalRead(A0); ir2 = digitalRead(A1);

if(dist<=6 && (ir2==1)){ Bot( LOW,HIGH,255, LOW,HIGH,100);} //stright back right

break;

case 2 :

dist= sonar.ping\_cm();

if (dist == 0) {dist = 40;}

if(dist <=15) {

Bot( LOW, LOW, 0, LOW, LOW, 0);

delay(500);

mpu.update(); Z=mpu.getAngleZ();

temp = Z+45;//turnRight

rotation( temp );

Bot(HIGH, LOW,200,HIGH, LOW,200);

delay(500);

mpu.update(); Z=mpu.getAngleZ();

temp = Z-45;//turnLeft

rotation( temp );

Bot(HIGH, LOW,200,HIGH, LOW,200);

delay(500);

mpu.update(); Z=mpu.getAngleZ();

temp = Z-45;

rotation( temp );

Bot(HIGH, LOW,200,HIGH, LOW,200);

delay(500);

mpu.update(); Z=mpu.getAngleZ();

temp = Z+45;

rotation( temp );

}

ir7= digitalRead(a6); ir8= digitalRead(a7);

if (ir7==HIGH && ir8==HIGH) { Bot(HIGH, LOW,100,HIGH, LOW,100); }

ir7= digitalRead(a6); ir8= digitalRead(a7);

if ( ir7==HIGH && ir8== LOW ) { Bot( LOW,HIGH,100,HIGH, LOW,100);}

ir7= digitalRead(a6); ir8= digitalRead(a7);

if ( ir7== LOW && ir8==HIGH ) { Bot( HIGH,LOW,100, LOW,HIGH,100); }

ir7 = digitalRead(a6);

ir8 = digitalRead(a7);

if ( ir7== LOW && ir8== LOW ) { Bot( LOW, LOW, 0, LOW, LOW, 0);}

break;

case 3 : rC=Serial.read();

if(rC=="F"){ Bot(HIGH, LOW,200,HIGH, LOW,200);}

if(rC=="B"){ Bot( LOW,HIGH,200, LOW,HIGH,200);}

if(rC=="R"){ Bot( LOW, LOW,200, LOW, LOW,200);}

if(rC=="L"){ Bot( LOW,HIGH,200,HIGH, LOW,200);}

if(rC=="S"){ Bot(HIGH, LOW,200, LOW,HIGH,200);}

break;

case 4 : Bot( LOW, LOW, 0, LOW, LOW, 0);

break;

}

}