```
//** VEHICLE LINE TRACKING PROGRAM **//

#include <WiFi.h>   //include wifi library
#include <ESPAsyncWebServer.h>  //include ESP32 Arduino web server library

#define in1 26      //Right motor pole 1
#define in2 25      //Right motor pole 2
#define in3 33      //left motor pole 1
#define in4 32      //left motor pole 2
#define ena 18      //Right motor speed
#define enb 5       //Left motor speed
#define left 4      //IR sensor Left
#define right 14    //IR sensor Right

//Define some variables to help control digital input values //
int no = 0;
int point = 20;
int l,r,el,er;
int mask=0,mask2=0,mask3=0,mark1=0,mark2=0;
int x=0,y=0,ox=0,oy=0,nx=0,ny=0,pos=0;
int state=0;            //Orientation state (1=Forward, 2=Reverse,
3=Right,     4=Left) Zero is initial state

/////// Set up & configuration wifi network ///////////
const char *ssid = "KARAM&KERANA";
const char *password = "20132013";

//Specifies the pins connected to each button
const int ledPins[] = {2, 15, 12, 19, 21, 32, 34, 35, 27};
//the total number of Buttons
const int numLeds = 9;
//Assign server port number
AsyncWebServer server(80);
////////////////////////// ** End Of Declarations ** //////////////////////////

void setup()
{
//Assign port_mode to each pin
  pinMode(in1,OUTPUT);
  pinMode(in2,OUTPUT);
  pinMode(in3,OUTPUT);
  pinMode(in4,OUTPUT);
  pinMode(ena,OUTPUT);
  pinMode(enb,OUTPUT);
  pinMode(left,INPUT);
```

```
  pinMode(right,INPUT);
  analogWrite(ena,50);  //initial speed for right wheels
  analogWrite(enb,50);  //initial speed for left wheels
  delay(3000);    //Time Delay 3sec.

////////////////////////////////////////////////

  Serial.begin(115200);
  // Set up ESP32 as an access point
  WiFi.softAP(ssid, password);
  Serial.print("Access Point IP Address: ");
  Serial.println(WiFi.softAPIP());
////////////////////////////////////////////////

//GPIO pins for Buttons
  for(int i = 0; i < numLeds; i++)
{
    //Set up the activity/status for each button
        pinMode(ledPins[i], OUTPUT);
        digitalWrite(ledPins[i], LOW);
}

  server.on("/", HTTP_GET, [](AsyncWebServerRequest *request)
{
    //Declare html variable & initiate Table with 3X3 buttons at the center of
the web-page
    String html = "<html><body><center><table  border='1' width='30%'>";
    //Title
        html += "<tr bgcolor='blue'><th colspan='3'><u>Select
Location</u></th></tr>";
     // Row 1
        html += "<tr><td><a href='/turnOn?point=0'>POINT02</a></td>";
        html += "<td><a href='/turnOn?point=1'>POINT12</a></td>";
        html += "<td><a href='/turnOn?point=2'>POINT22</a></td></tr>";
    // Row 2
        html += "<tr><td><a href='/turnOn?point=3'>POINT01</a></td>";
        html += "<td><a href='/turnOn?point=4'>POINT11</a></td>";
        html += "<td><a href='/turnOn?point=5'>POINT21</a></td></tr>";
    // Row 3
        html += " <tr><td><a href='/turnOn?point=6'>POINT00</a></td>";
        html += "<td><a href='/turnOn?point=7'>POINT10</a></td>";
        html += "<td><a href='/turnOn?point=8'>POINT20</a></td></tr>";

        html += "</center></table></body></html>";
   //End of web-page set up
```

```
        request->send(200, "text/html", html);
});
  // Handle buttons turn-on requests
  server.on("/turnOn", HTTP_GET, [](AsyncWebServerRequest *request)
{
  // point variable will be equal to the pressed button number in
  // the web-page & converted to integer.
        point = request->arg("point").toInt();
        request->send(200, "text/plain", "OK");
    });
  // Start server
  server.begin();
}

void loop()
{
 // point value should be between 0-8 to make If-statement true.
   if (point >= 0 && point < numLeds)
{
      // Explicit conditions for each Button
          if (point == 0)
      {
              nx = 0;
              ny = 2;
              no = 1;
            }
      else if (point == 1)
            {
              nx = 1;
              ny = 2;
              no = 1;
            }
       else if (point == 2)
            {
              nx = 2;
              ny = 2;
              no = 1;
            }
       else if (point == 3)
            {
              nx = 0;
              ny = 1;
              no = 1;

            }
```

```
      else if (point == 4)
            {
              nx = 1;
              ny = 1;
              no = 1;
            }
      else if (point == 5)
            {
              nx = 2;
              ny = 1;
              no = 1;
            }
      else if (point == 6)
            {
              nx = 0;
              ny = 0;
              no = 1;
          }
      else if (point == 7)
            {
              nx = 1;
              ny = 0;
              no = 1;
                  }
      else if (point == 8)
            {
              nx = 2;
              ny = 0;
              no = 1;
            }
    }

    if (no == 1)  // if true, means one of the web-page buttons has been pressed.
{
          calc();
          point =20 ; /* point value should be assigned by any NO greater than 8,
            to avoid enrolls in the above first If-statement without
                          button press*/
      }
}
void calc()
{
  x= nx-ox;         //nx & ny denote the selected location
  y= ny-oy;         // ox & oy are the previous location
```

```
  if(x>0 && y>0)
{
        exy();       // Execute y-motion (Move forward on Y-axis)
        state = 3;   // initial orientation value to turn right
        orientation();
        exx();       // Execute x-motion (Move forward on X-axis)
        state = 4;   //
        orientation();
        no=0;
    }
else if(x<0 && y<0)
    {
        mark2=1;
        getvalues();
        state=2;
        orientation();
        exy1();
        state=3;
        orientation();
        exx1();
        state=3;
        orientation();
        mark2=0;
        no=0;
    }
  if(x==0 && y>0)
    {
        getvalues();
        exy();
      no=0;
    }
else if(x==0 && y<0)
    {
        getvalues();
        state=2;
        orientation();
        exy1();
        state=2;
        orientation();
        no=0;
    }
else if(y==0 && x>0)
    {
        getvalues();
        state=3;
```

```
        orientation();
        exx();
        state=4;
        orientation();
        no=0;
    }
else if(y==0 && x<0)
    {
        getvalues();
        state=4;
        orientation();
        exx1();
        state=3;
        orientation();
        no=0;
    }
else if(x>0 && y<0)
    {
        getvalues();
        state=3;
        orientation();
        exx();
        state=3;
        orientation();
        mark1=1;
        exy1();
        state=2;
        orientation();
        mark1=0;
      no=0;
    }
else if(x<0 && y>0)
    {
        getvalues();
        exy();
        state=4;
        orientation();
        mark1=1;
        exx1();
        state=3;
        orientation();
        mark1=0;
      no=0;
    }
}
```

```
void getvalues()   // Get values from IR sensors
{
  // The return value is 0 for Black or 1 for White
l = digitalRead(left);
    r = digitalRead(right);
}
void compare()
{
  if(l==0 && r==0)        // means IR sensors in white line
{
        forward();
    }
  if(l==1 && r==0)     // left IR in black, Right IR in white= turn left
{
        Tleft();
    }
      else if(l==0 && r==1)  //  Right IR in black, left IR in white= turn Right
    {
        Tright();
    }
  if(l==1 && r==1 && mask==0)    //  Right IR in black, left IR in black= stop
    {
        delay(300);
        off();
        mask=1;        // to stop increment (pos++), will not enter again
            // unless it move to white line
     if(y>0 && x<0 && mark1==1||y<0 && x>0 && mark1==1)
         {
             pos--;
         }

    else if(x<0 && y<0 && mark2==1||y<0 && x==0||x<0 && y==0)
    {
         pos--;
    }
    else pos++;
     forward();
    }
  if(l==0 && r==0)        // both IR sensors in white line
    {
       mask=0;
    }
}
void exy()      // execute y, if its value is positive
{
```

```
  loop7:
  if (y > pos)  //pos= position (increment by 1 (pos++) after each intersection)
  {
    getvalues();
    compare();
  }
  else if (y == pos)
  {
    off();
    pos=0;
    oy=abs(ny);
    goto loop8;
  }
  goto loop7;
  loop8:
  oy=abs(ny);
  pos=0;
}
void exx()
{
  loop9:
  if(x>pos)
  {
    getvalues();
    compare();
  }
  else if(x==pos)
  {
    off();
    pos=0;
    ox=abs(nx);
    goto loop10;
  }
  goto loop9;
  loop10:
  ox=abs(nx);
  pos=0;
}
void exy1()   // Execute -y (for negative values of y) different process of y
{
  loop7:
  if (y < pos)      // y= -
  {
    getvalues();
    compare();
```

```
  }
  else if(y == pos)
  {
    off();
    pos=0;
    oy=abs(ny);
    goto loop8;
  }
  goto loop7;
  loop8:
  oy=abs(ny);
  pos=0;
}

void exx1() // Execute -x (for negative values of x) different process of x
{
  loop9:
  if(x < pos)
  {
    getvalues();
    compare();
  }
  else if(x == pos)
  {
    off();
    pos=0;
    ox=abs(nx);
    goto loop10;
  }
  goto loop9;
  loop10:
  ox=abs(nx);
  pos=0;
}
void inertia() // this function is to initiate the car
{
  analogWrite(ena,5);
  analogWrite(enb,5);
  digitalWrite(in1,LOW);
  digitalWrite(in2,HIGH);
  digitalWrite(in3,LOW);
  digitalWrite(in4,HIGH);
  delay(50);
}
void forward()
```

```
{
  if(mask2==0) // just at the begin (after each intersection)
    {
      inertia();
      mask2=1;
    }
  analogWrite(ena,35);
  analogWrite(enb,35);
  digitalWrite(in1,LOW);
  digitalWrite(in2,HIGH);
  digitalWrite(in3,LOW);
  digitalWrite(in4,HIGH);
}
void Tleft() // turn left
{
  analogWrite(ena,5);
  analogWrite(enb,4);
  digitalWrite(in1,LOW);
  digitalWrite(in2,HIGH);
  digitalWrite(in3,HIGH);
  digitalWrite(in4,LOW);
}
void Tright() // turn right
{
  analogWrite(ena,4);
  analogWrite(enb,5);
  digitalWrite(in1,HIGH);
  digitalWrite(in2,LOW);
  digitalWrite(in3,LOW);
  digitalWrite(in4,HIGH);
}
void off() // stop
{
  mask2=0; // to trigger inertia function when car move again
  digitalWrite(in1,LOW);
  digitalWrite(in2,LOW);
  digitalWrite(in3,LOW);
  digitalWrite(in4,LOW);
}
void orientation()
{
  if (state==2)
  {// to reverse Orientation
    Tleft180();
    off();
```

```
    state=0;
  }else if(state==3)
  {
    Tright90();
    off();
    state=0;
  }else if(state==4)
  {
    Tleft90();
    off();
    state=0;
  }
}
void Tleft180()
{
  analogWrite(ena,170);
  analogWrite(enb,240);
  digitalWrite(in1,LOW);
  digitalWrite(in2,HIGH);
  digitalWrite(in3,HIGH);
  digitalWrite(in4,LOW);
  delay(500);
}
void Tleft90()
{
  analogWrite(ena,250);
  analogWrite(enb,110);
  digitalWrite(in1,LOW);
  digitalWrite(in2,HIGH);
  digitalWrite(in3,HIGH);
  digitalWrite(in4,LOW);
  delay(315);
}
void Tright90()
{
    analogWrite(ena,140);
    analogWrite(enb,240);
    digitalWrite(in1,HIGH);
    digitalWrite(in2,LOW);
    digitalWrite(in3,LOW);
    digitalWrite(in4,HIGH);
    delay(270);
}
```