**ARDUINO CODE:**

#include <LiquidCrystal.h>

#include <Servo.h>

Servo servo1;

#define NO\_SAMPLES 100u

#define THRESHOLD 15u

// Color Thresholds

#define IDLE\_RED\_THRESHOLD 190u

#define IDLE\_GREEN\_THRESHOLD 160u

#define IDLE\_BLUE\_THRESHOLD 75u

#define RED\_RED\_THRESHOLD 315u

#define RED\_GREEN\_THRESHOLD 575u

#define RED\_BLUE\_THRESHOLD 495u

#define GREEN\_RED\_THRESHOLD 140u

#define GREEN\_GREEN\_THRESHOLD 550u

#define GREEN\_BLUE\_THRESHOLD 520u

#define BLUE\_RED\_THRESHOLD 20u

#define BLUE\_GREEN\_THRESHOLD 500u

#define BLUE\_BLUE\_THRESHOLD 375u

// Sensor Connections

const byte S0 = 8;

const byte S1 = 9;

const byte S2 = 10;

const byte S3 = 11;

const byte Sout = 12;

const int RS = 13, EN = 7, D4 = 2, D5 = 3, D6 = 4, D7 = 5; //lcd connection

LiquidCrystal lcd(RS, EN, D4, D5, D6, D7);

uint16\_t sumOfSamples = 0;

uint16\_t redData = 0;

uint16\_t greenData = 0;

uint16\_t blueData = 0;

int r\_count =0;

int g\_count =0;

int b\_count =0;

int idle\_flag =1;

int red\_flag =1;

int green\_flag =1;

int blue\_flag =1;

int servoPin=6;

void setup()

{

servo1.attach(servoPin);

pinMode(S0, OUTPUT);

pinMode(S1, OUTPUT);

pinMode(S2, OUTPUT);

pinMode(S3, OUTPUT);

pinMode(Sout, INPUT);

// Setting frequency-scaling to 20%

digitalWrite(S0,HIGH);

digitalWrite(S1,LOW);

Serial.begin(9600);

lcd.begin(16, 2); // set up number of columns and rows

lcd.setCursor(0, 0); // move cursor to (0, 0)

}

void loop()

{

int i = 0;

// Apply Red Filter

digitalWrite(S2,LOW);

digitalWrite(S3,LOW);

delay(1);

sumOfSamples = 0;

for (i=0; i < NO\_SAMPLES; i++)

{

// Reading the Pulse Width

sumOfSamples += pulseIn(Sout, LOW);

}

redData = sumOfSamples/NO\_SAMPLES;

Serial.print("R = ");

Serial.print(redData);

delay(100);

// Apply Green Filter

digitalWrite(S2, HIGH);

digitalWrite(S3, HIGH);

delay(1);

sumOfSamples = 0;

for (i=0; i < NO\_SAMPLES; i++)

{

// Reading the Pulse Width

sumOfSamples += pulseIn(Sout, LOW);

}

greenData = sumOfSamples/NO\_SAMPLES;

Serial.print(" G = ");

Serial.print(greenData);

delay(100);

// Apply Blue Filter

digitalWrite(S2, LOW);

digitalWrite(S3, HIGH);

delay(1);

sumOfSamples = 0;

for (i=0; i < NO\_SAMPLES; i++)

{

// Reading the Pulse Width

sumOfSamples += pulseIn(Sout, LOW);

}

blueData = sumOfSamples/NO\_SAMPLES;

Serial.print(" B = ");

Serial.print(blueData);

Serial.println(" ");

delay(100);

if( isIdle() )

{

// Turn-Off All Led's

Serial.println("All Led's Off");

idle\_flag =0;

red\_flag =1;

green\_flag=1;

blue\_flag=1;

}

if( isRed() )

{

// Turn-On Red Led

Serial.println("Red Led:");

if (red\_flag ==1)

{

idle\_flag =1;

red\_flag =0;

green\_flag=1;

blue\_flag=1;

r\_count = r\_count+1;

Serial.print(" : Red value=");

Serial.print(r\_count);

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("RED");

lcd.setCursor(0, 1);

lcd.print("R=");

lcd.print(r\_count);

lcd.print(" G=");

lcd.print(g\_count);

lcd.print(" B=");

lcd.print(b\_count);

servo1.write(60);

}

}

if( isGreen() )

{

// Turn-On Green Led

Serial.println("Green Led");

if (green\_flag ==1)

{

idle\_flag =1;

red\_flag =1;

green\_flag=0;

blue\_flag=1;

g\_count = g\_count+1;

Serial.print(" : Green Value=");

Serial.print(g\_count);

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("GREEN");

lcd.setCursor(0, 1);

lcd.print("R=");

lcd.print(r\_count);

lcd.print(" G=");

lcd.print(g\_count);

lcd.print(" B=");

lcd.print(b\_count);

servo1.write(120);

}

}

if( isBlue() )

{

// Turn-On Blue Led

Serial.println("Blue Led");

if (blue\_flag ==1)

{

idle\_flag =1;

red\_flag =1;

green\_flag=1;

blue\_flag=0;

b\_count = b\_count+1;

Serial.print(" : Blue Value=");

Serial.print(b\_count);

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("BLUE");

lcd.setCursor(0, 1);

lcd.print("R=");

lcd.print(r\_count);

lcd.print(" G=");

lcd.print(g\_count);

lcd.print(" B=");

lcd.print(b\_count);

servo1.write(180);

}

}

if((isRed()==false) && (isGreen()==false) && (isBlue()==false) )

{

lcd.clear();

lcd.setCursor(0, 0);

lcd.print(" Unknown Color");

lcd.setCursor(0, 1);

lcd.print("R=");

lcd.print(r\_count);

lcd.print(" G=");

lcd.print(g\_count);

lcd.print(" B=");

lcd.print(b\_count);

servo1.write(0);

idle\_flag =1;

red\_flag =1;

green\_flag=1;

blue\_flag=1;

}

delay(2000);

}

//end of loop function

boolean isIdle( void )

{

boolean status = false;

if ( ((IDLE\_RED\_THRESHOLD-THRESHOLD) < redData)

&& (redData < (IDLE\_RED\_THRESHOLD+THRESHOLD)) )

{

if ( ((IDLE\_GREEN\_THRESHOLD-THRESHOLD) < greenData)

&& (greenData < (IDLE\_GREEN\_THRESHOLD+THRESHOLD)) )

{

if ( ((IDLE\_BLUE\_THRESHOLD-THRESHOLD) < blueData)

&& (blueData< (IDLE\_BLUE\_THRESHOLD+THRESHOLD)) )

{

status = true;

}

}

}

return status;

}

boolean isRed( void )

{

boolean status = false;

if ( ((RED\_RED\_THRESHOLD-THRESHOLD) < redData)

&& (redData < (RED\_RED\_THRESHOLD+THRESHOLD)) )

{

if ( ((RED\_GREEN\_THRESHOLD-THRESHOLD) < greenData)

&& (greenData < (RED\_GREEN\_THRESHOLD+THRESHOLD)) )

{

if ( ((RED\_BLUE\_THRESHOLD-THRESHOLD) < blueData)

&& (blueData< (RED\_BLUE\_THRESHOLD+THRESHOLD)) )

{

status = true;

}

}

}

return status;

}

boolean isGreen( void )

{

boolean status = false;

if ( ((GREEN\_RED\_THRESHOLD-THRESHOLD) < redData)

&& (redData < (GREEN\_RED\_THRESHOLD+THRESHOLD)) )

{

if ( ((GREEN\_GREEN\_THRESHOLD-THRESHOLD) < greenData)

&& (greenData < (GREEN\_GREEN\_THRESHOLD+THRESHOLD)) )

{

if ( ((GREEN\_BLUE\_THRESHOLD-THRESHOLD) < blueData)

&& (blueData< (GREEN\_BLUE\_THRESHOLD+THRESHOLD)) )

{

status = true;

}

}

}

return status;

}

boolean isBlue( void )

{

boolean status = false;

if ( ((BLUE\_RED\_THRESHOLD-THRESHOLD) < redData)

&& (redData < (BLUE\_RED\_THRESHOLD+THRESHOLD)) )

{

if ( ((BLUE\_GREEN\_THRESHOLD-THRESHOLD) < greenData)

&& (greenData < (BLUE\_GREEN\_THRESHOLD+THRESHOLD)) )

{

if ( ((BLUE\_BLUE\_THRESHOLD-THRESHOLD) < blueData)

&& (blueData< (BLUE\_BLUE\_THRESHOLD+THRESHOLD)) )

{

status = true;

}

}

}

return status;

}