Monitoramento de Ponto Cego

Utilizando uma placa microcontroladora Arduino Uno de baixo custo, o projeto tenta recriar um recurso visto apenas em carros de alto valor: luzes LED que alertam o motorista quando há outros veículos nos pontos cegos do carro. É um excelente projeto para aprender como funcionam os sistemas de sensores no mundo real.

Materiais necessários:

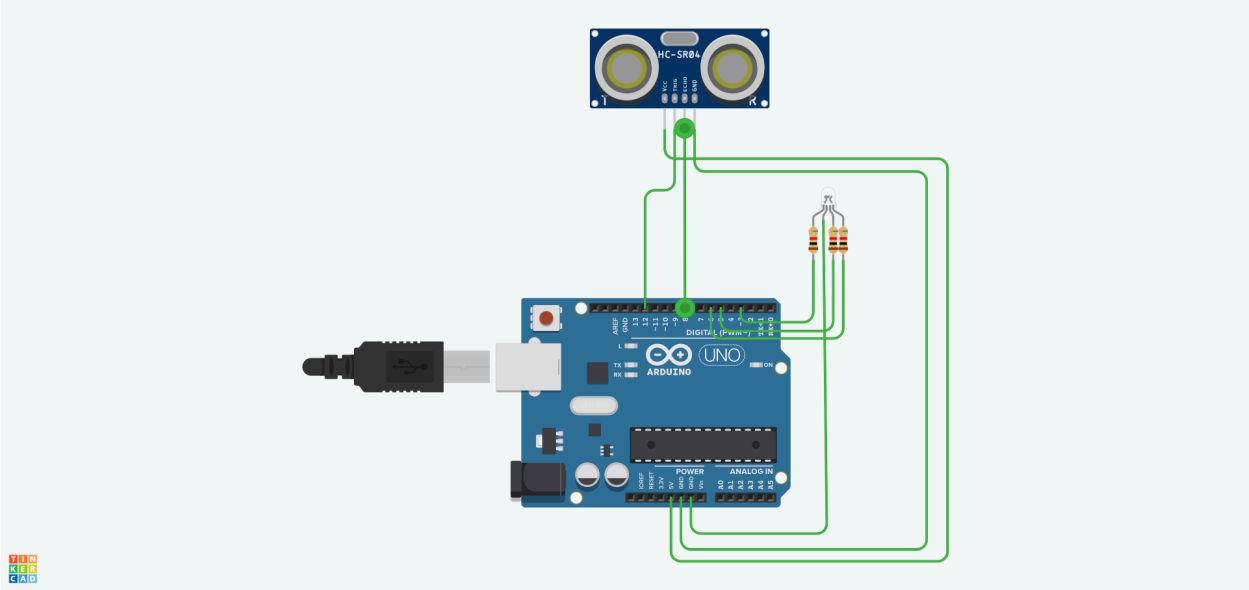
- 1 x Placa Uno R3 com cabo USB
- 1 x Protoboard(Opcional)
- 1 x Fita LED RGB (1 metro)
- 8 x Jumpers macho-fêmea
- 3 x Resistores
- 1 x Sensor Ultrasonico HC-SR04

Como funciona o sensor de monitoramento de ponto cego.

Lógica

1. Inicialize os pinos e dispare as distâncias.
2. Calcule a distância entre o host e o objeto.
3. Compare a distância calculada com a distância do disparo e pisque a cor de acordo.
4. No meu código, configurei a faixa de LED para piscar em laranja quando o objeto estiver a 30 cm e vermelho quando o objeto estiver a 10 cm. Quando nenhum objeto for detectado, a faixa de LED ficará azul. Você pode alterar as cores de acordo com sua preferência.

Pinagem



Código

```
//initializing the rgb strip pins
#define b 6 //blue
#define g 5 //green
#define r 3 //red

//initializing the ultrasonic sensor pins

#define echoPin 8
#define trigPin 12

//initializing the trigger distances of colour alerts
#define trigDist1 30
#define trigDist2 10

long duration;
int distance;
bool fade=true;

void setup() {
  Serial.begin(9600);
  pinMode(g,OUTPUT);
  pinMode(b,OUTPUT);
  pinMode(r,OUTPUT);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
}


void loop() {
  calDistance(); //calling the function that will calculate the distance
  if(distance<=trigDist1 && distance>trigDist2){ //stage 1 trigger distance
    fadedOrange(3);
    fade=true;
  }
  else if(distance<=trigDist2){ //stage 2 trigger distance
    fadedRed(3);
    fade=true;
  }
  else{
    fadeInBlue(); //idle
  }
}

void calDistance(){
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  distance = duration * 0.034 / 2; //calculating distance using the distance =
speed of sound * time taken/2
  Serial.println(distance);
```

```
}

void fadedRed(int val){ //function to get a faded red blink
    for(int i=255; i>0; i-=val){
      analogWrite(r, i);
      analogWrite(g, 255);
      analogWrite(b, 255);
      delay(5);
    }
    for(int i=0; i<255; i+=val){
      analogWrite(r, i);
      analogWrite(g, 255);
      analogWrite(b, 255);
      delay(5);
    }
}

void fadedOrange(int val){ //function to get a faded orange blink
    for(int i=255; i>0; i-=val){
      analogWrite(r, i);
      analogWrite(b, 255);
      analogWrite(g, i>230?i:230);
      delay(5);
    }
    for(int i=0; i<255; i+=val){
      analogWrite(r, i);
      analogWrite(b, 255);
      analogWrite(g,i<230?240:i);
      delay(5);
    }
}

void fadeInBlue(){ //Blue fade in
  if(fade==true){
    for(int i=255; i>0; i-=1){
        analogWrite(g, 255);
        analogWrite(r, 255);
        analogWrite(b, i);
        delay(5);
     }
  }
  fade=false;
  analogWrite(g, 255);
  analogWrite(r, 255);
  analogWrite(b, 0);
}
```