```
//Fully working inputs and outputs
// v2 synced haptics and light peltier, desensitized haptics
//  Variables
int PulseSensorPurplePin = 1;        // Optical Sensor PURPLE WIRE connected to
ANALOG PIN 5
//int LED13 = 13;   //  The on-board Arduion LED

int period = 1000;
unsigned long time_now = 0;

int Signal;                // holds the incoming raw data. Signal value can range
from 0-1024
int Threshold = 513;// average signal from optical
int upper = 700;    // upper and lower thresholds
int lower = 250;
bool state = true;

const int haptic = 2; // make sure to wire 2 as + and GROUND
const int haptic1 = A0;
const int trigPin = 3;
const int echoPin = 4;
int RGBLED_RedPin = 13;
int RGBLED_GreenPin = 12;
int RGBLED_BluePin = 11;
int PWMValue_RedPin = 0;
int PWMValue_GreenPin = 0;
int PWMValue_BluePin = 0;

// defines variables
long duration;
int distance;


// The SetUp Function:
void setup() {
  pinMode(LED_BUILTIN,OUTPUT);         // for testing
  pinMode(haptic, OUTPUT);
  pinMode(haptic1, OUTPUT);
   Serial.begin(9600);          // Set's up Serial Communication at certain speed.
  pinMode(7,OUTPUT);  // POSITIVE //peltier 1
  pinMode(8,OUTPUT); // NEGATIVE
  pinMode(9,OUTPUT); // PWM
  pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
  pinMode(echoPin, INPUT); // Sets the echoPin as an Input
  pinMode(RGBLED_RedPin, OUTPUT); //rgb LED
```

```
  pinMode(RGBLED_GreenPin, OUTPUT);
  pinMode(RGBLED_BluePin, OUTPUT);
  pinMode(6,OUTPUT);  // POSITIVE //peltier 2
  pinMode(5,OUTPUT); // NEGATIVE
  pinMode(10,OUTPUT); // PWM
}


// The Main Loop Function
void loop() {

  Signal = analogRead(PulseSensorPurplePin);  // Read the PulseSensor's value.

  // Clears the trigPin
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
// Sets the trigPin on HIGH state for 10 micro seconds
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
// Reads the echoPin, returns the sound wave travel time in microseconds
duration = pulseIn(echoPin, HIGH);
// Calculating the distance
distance= duration*0.034/2;

if (distance < 12) {
  PWMValue_RedPin = 255;
  PWMValue_GreenPin = 0;
  PWMValue_BluePin = 0;
   digitalWrite(8, LOW);          //peltiers HOT
        digitalWrite(7, HIGH);
        analogWrite(9,70);
        digitalWrite(6, LOW);
        digitalWrite(5, HIGH);
        analogWrite(10,70);


} else if (distance < 40 && distance > 11) {
  PWMValue_RedPin = 0;
  PWMValue_GreenPin = 255;
  PWMValue_BluePin = 0;
   digitalWrite(8, LOW); // PELTIERS HOT
   digitalWrite(7, HIGH);
   analogWrite(9,70);
   digitalWrite(6, LOW);
```

```
   digitalWrite(5, HIGH);
   analogWrite(10,70);

} else if (distance > 39) {
  PWMValue_RedPin = 0;
  PWMValue_GreenPin = 0;
  PWMValue_BluePin = 255;
  digitalWrite(8, HIGH);  //PELTIERS COLD
  digitalWrite(7, LOW);
  analogWrite(9,70);

  digitalWrite(6, HIGH);
  digitalWrite(5, LOW);
  analogWrite(10,70);
}
// second leg is anode
 analogWrite(RGBLED_RedPin, PWMValue_RedPin);
 analogWrite(RGBLED_GreenPin, PWMValue_GreenPin);
 analogWrite(RGBLED_BluePin, PWMValue_BluePin);
// Prints the distance on the Serial Monitor
Serial.print("Distance: ");
Serial.println(distance);

//cleans up optical signal
   if(Signal > upper && state == true ){                            // If the
signal is above "550", then "turn-on" Arduino's on-Board LED.
     digitalWrite(LED_BUILTIN,HIGH);
     digitalWrite(LED_BUILTIN,LOW);
     state = false;
     Serial.println(state);
   } else if (Signal < lower && Signal > 500 && state == true) {
     digitalWrite(LED_BUILTIN,HIGH);                //  Else, the sigal must be
below "550", so "turn-off" this LED.
     state = false;
     Serial.println(state);
   } else { digitalWrite(LED_BUILTIN,LOW);
     state = true;
     Serial.println(state);
   }
if(millis() > time_now + period && state == false){ //how often should it check,
modify period
        time_now = millis();
        Serial.println("Hello");
        //haptic
        digitalWrite(haptic,HIGH);
```

```
        digitalWrite(haptic1,HIGH);
        delay(150);
        digitalWrite(haptic,LOW);
        digitalWrite(haptic1,LOW);
        /*wave right
        digitalWrite(haptic,HIGH);
        delay(150);
        digitalWrite(haptic,LOW);


        digitalWrite(haptic1,HIGH);
        delay(150);
        digitalWrite(haptic1,LOW);

        /wave left
         digitalWrite(haptic1,HIGH);
        delay(150);
        digitalWrite(haptic1,LOW);


        digitalWrite(haptic,HIGH);
        delay(150);
        digitalWrite(haptic,LOW);

        //peltier


        /*digitalWrite(8, LOW);
        digitalWrite(7, HIGH);
        analogWrite(9,70);
        digitalWrite(6, LOW);
        digitalWrite(5, HIGH);
        analogWrite(10,70);*/
        state = true;
        delay(200); // wait until signal is read again

     Serial.println(state);
   }

/* away
digitalWrite(8, HIGH);
digitalWrite(7, LOW);
analogWrite(9,50);
 */
```

```
delay(10);


}
```