[Ultrasonic Security System]

ECEN 106 – Computer Systems Final Project
[Victor Santana], [12/9/2023]

<u>Explanation: What my project does:</u>

My project is an Ultrasonic Security System. This very simple Arduino Project is divided into 6 easy steps. This simple Arduino UNO device will use the following components: an Arduino UNO, jumper wires, LEDs (1- Green, 1- Yellow, and 1- Red), a breadboard, an ultrasonic sensor - HC -SRO4, 220 Ohm resistors (5), a buzzer, and the Arduino IDE code.

The Arduino is programmed to give three different types of signals depending on how close the object is to the sensor. Once on, the project will emit ultrasonic waves that will detect how close the object could be. Depending on how close the object is, the Arduino will turn on the different types of LEDs. Each one work as following: First, if the object is in a “safe” distance, the green LED will be ON. Second, if the object is in a “not so safe" distance, the yellow LED will be ON. Lastly, if the object is “not in a safe distance” both the buzzer and the red LED will be ON.

<u>Explanation: How my project works:</u>

[Describe how your project is able to do what it does. For example, what do the input and/or output devices do? How are they connected? What does the software do?]

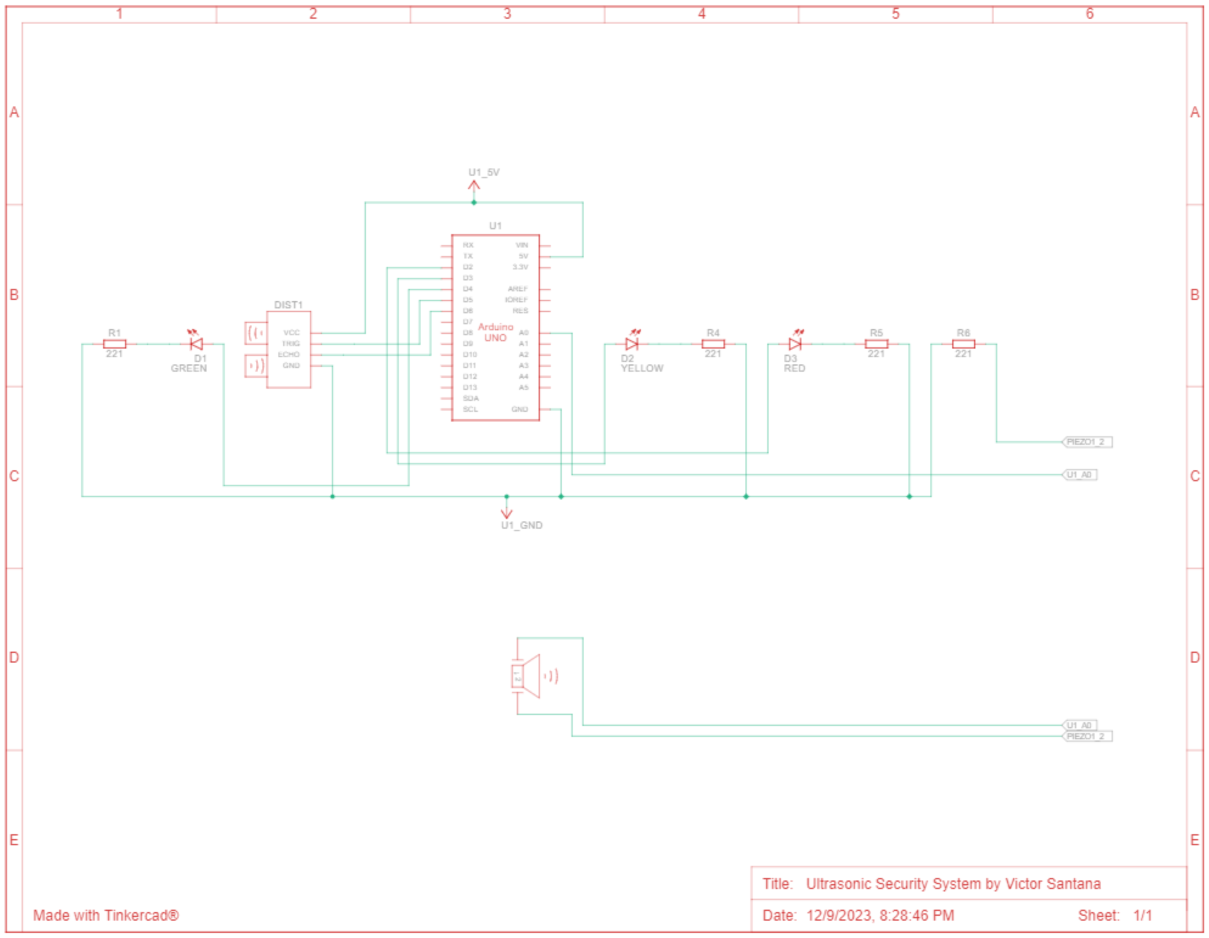
The Arduino UNO will give voltage, control (with the software added) and manage the instructions of every component in the project. Very simple. The software will assign and specify the parameters and roles of each of the components with the information fed by the ultrasonic system, which will be described below. The breadboard will have all the components connected to it, so voltage and GND can be evenly distributed.

The LEDs work in the following way. First, the green LED (connected to pin D4): if the object is in a “safe” distance (between 55 and 200 inches), only the green LED will turn ON. Second, the yellow LED (connected to pin D3): if the distance is between 15 and 55 inches, only the yellow LED will turn ON. Lastly, the red LED (connected to pin D2): if the distance is less than 15 inches, only red the LED will turn ON.

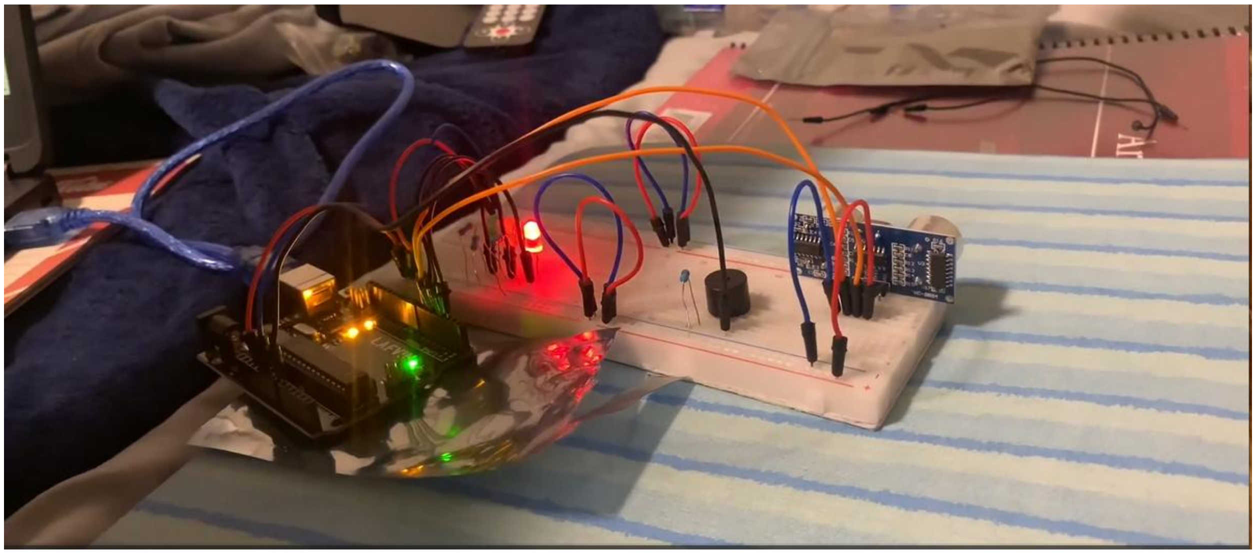
The buzzer (connected to pin A0) works as follows: if the distance is less than 8 inches, both the red LED and the buzzer will be ON. Each of the LEDs and the buzzer have a 220-ohm resistor connected to the GND lines in the breadboard. The ultrasonic sensor will emit ultrasonic waves to detect objects within a 200 inches area and will send the information to the Arduino, so it can properly manipulate the LEDs.

Hardware: Wiring diagram

I re-created the diagram in my own TinkerCad account.



Hardware: Photo(s)



Software: Arduino sketch

Here is the fixed Arduino sketch that I found in the comment section:

```
int trigPin = PD5; // Sensor Trip pin connected to Arduino pin D5
int echoPin = PD6; // Sensor Echo pin connected to Arduino pin D6
int redLED = PD2; // Red LED connected to pin D2
int yellowLED = PD3; // Yellow LED connected to pin D3
int greenLED = PD4; // Green LED connected to pin D4
int buzzer = A0; // Buzzer connected to Analogue pin A0
long TempDistance = 0; // A variable to store the temporary distance
int counter = 0; // Counter value to check if the object has stopped moving
void setup() {
Serial.begin(9600);
pinMode(trigPin, OUTPUT);
pinMode(echoPin, INPUT);
pinMode(redLED, OUTPUT);
pinMode(greenLED, OUTPUT);
pinMode(yellowLED, OUTPUT);
pinMode(buzzer, OUTPUT);
}
void loop() {
long duration, Distance;
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
duration = pulseIn(echoPin, HIGH);
Distance = (duration/2) / 74; // Distance in Inches
if(counter < 20){ // Do the rest if the car is still moving
if (Distance > 200) { // Nothing in the garrage
turnThemAllOff();
}
if ((Distance > 55) && (Distance <= 200)) { // Turn on Green LED
digitalWrite(greenLED, HIGH);
digitalWrite(yellowLED, LOW);
digitalWrite(redLED, LOW);
noTone(buzzer);
}
if ((Distance > 15) && (Distance <= 55)) { // Turn on Yellow LED
digitalWrite(yellowLED, HIGH);
digitalWrite(redLED, LOW);
digitalWrite(greenLED,LOW);
noTone(buzzer);
}
if (Distance <= 15) { // Turn on Red LED
digitalWrite(redLED, HIGH);
digitalWrite(greenLED,LOW);
digitalWrite(yellowLED, LOW);
noTone(buzzer);
}
if (Distance < 8) { // Item is way to close - start the buzzer
tone(buzzer, 500);
}
}
if ((Distance == TempDistance) || ((Distance+1) == TempDistance) || ((Distance-1)
== TempDistance)){
if(counter >= 20){ // Turn off the lights if the object hasn't moved for 20cycles
(no change in distance)
Serial.println("No movement detected, turning off the lights");
turnThemAllOff();
} else {
counter++;
}
} else {
counter = 0; // Reset counter if there is a movement
}
TempDistance = Distance;
Serial.print(Distance);
Serial.println(" inches");
Serial.print("Counter : ");
Serial.println(counter); delay(500); }
// Function to turn the LEDs off
void turnThemAllOff(){
digitalWrite(redLED, LOW);
digitalWrite(greenLED,LOW);
digitalWrite(yellowLED, LOW);
noTone(buzzer);
}
```

<u>Software: Citation</u>

"Ultrasonic Security System," by Krepak. Arduino Project Hub:
https://projecthub.arduino.cc/Krepak/ultrasonic-security-system-a6ea3a

The original code was not working, so I used a fixed version that I found in the comments from the same page (I tried search for the name of the user, but I could not find it). To get the hardware to work, I just had to change the original pin locations to the ones described in the software sketch. After I made those modifications, the projected worked without issues.

Discussion:

[Include here a one-paragraph discussion. For example, you may write about of the problems you had, what you learned, what else you would do if you had more time and/or skills, etc.]

I had two noticeable problems while working with this project. First, my Arduino UNO was not connecting to my laptop. For troubleshooting I tried to restart my laptop, use a different cord to connect my Arduino and use some of the pre-made sketch from the IDE. After testing, I noticed that the wiring was in the incorrect places, and it possibly made my Arduino go crazy. Thankfully, I re-wired everything, and this fixed the issue. Lastly, the Arduino sketch. The code was not good because of how everything was so misplaced and defined incorrectly, which made the code non-functional. Thankfully, there were some users that shared fixed versions of the code, and I used the code that seemed to work the best for me. What I learned from this projected is that not everything will go as expected, and that I should be prepared to solve as many possible non-desired outcomes as possible.