Rear End Collision Sensor

Hannah Schiffmacher

1 Hardware Component Prototype

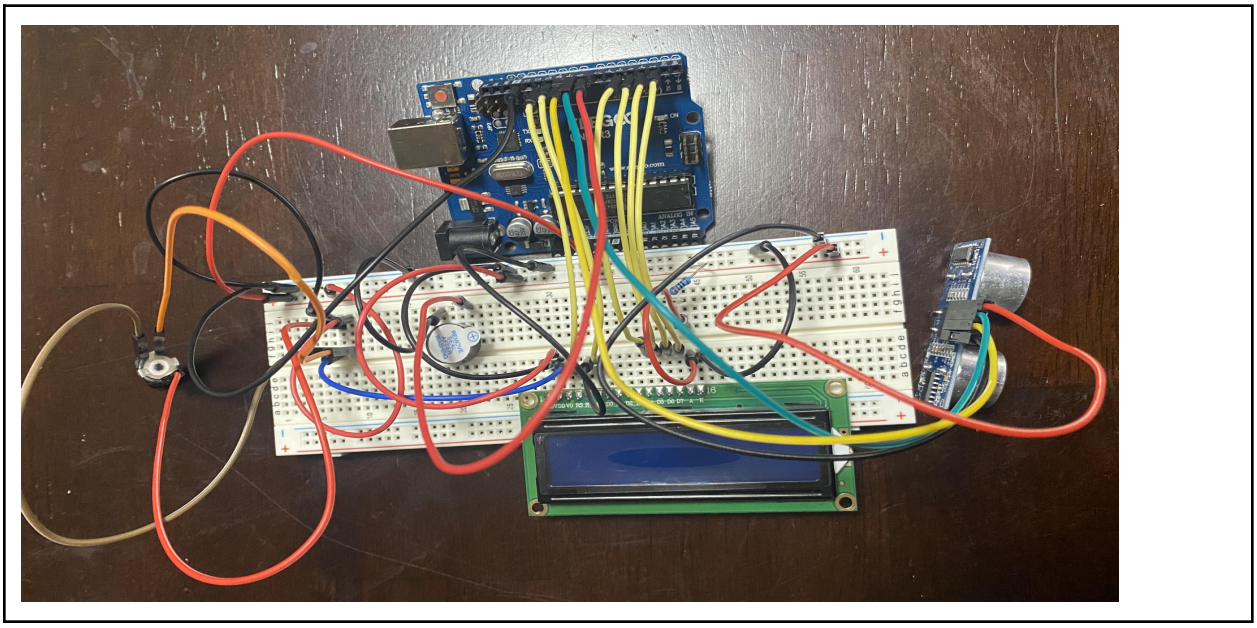
Elegoo Uno R3
Microcontroller board that supports the rest of the hardware

Ultrasonic Sensor
This measures the distance an object is away from the sensor

LCD Display
This displays the messages according to the distance from the ultrasonic sensor. For instance, if the distance is small, “WARNING” will be displayed. If a car is coming towards the sensor quickly and the distance is changing rapidly, “INCOMING CAR” will be displayed.

Potentiometer
This controls the contrast of the display.

Active Buzzer
This also provides output. Depending on the circumstance, the buzzer will beep. For instance if there is a “WARNING” it will beep 3 times. If it is “INCOMING CAR” the buzzer will rapidly beep five times. This is to simulate the warning systems in cars.



The yellow wires connect the LCD to the Uno, the orange and brown wire connect the potentiometer to the uno, the turquoise and yellow wire connect the ultrasonic sensor to the Uno, the blue wire connects the LCD to the potentiometer, and the red and black wires connect all of the components to the negative and positive channels on the breadboard to the Uno.

2 Software Component

```
#include <LiquidCrystal.h>
const int trigPin = 9;
const int echoPin = 10;
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
const int buzzerPin = 8;
// https://docs.arduino.cc/learn/electronics/lcd-displays
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
const int tailgating_distance = 6;
long duration;
int distance;
int distance2 = -1;
int i = 0;
int distance_diff = 0;
bool b_switch = true; // turn buzzer on and off

void warning(bool b_switch) {
  // print warning to lcd
  lcd.print("WARNING");
  // buzzer output
  if (b_switch) {
    for (int i = 0; i < 3; i++) {
      buzz();
      delay(200);
    }
  }
  // delay for five seconds
  delay(5000);
  // clear lcd
  lcd.clear();
}

void incoming(bool b_switch) {
  // print to lcd
  lcd.print("INCOMING CAR");
  // buzzer output
  if (b_switch) {
    for (int i = 0; i < 5; i++) {
      buzz();
      delay(100); // Adjust the delay between buzzes if needed
```

```
    }
  }
  // delay for five seconds that programs starts again
  delay(5000);
  // clear lcd
  lcd.clear();
}


void buzz() {
  // Generate a buzz by toggling the buzzer pin
  digitalWrite(buzzerPin, HIGH);
  delay(100); // Adjust the duration of the buzz if needed
  digitalWrite(buzzerPin, LOW);
}

void setup() {
  // put your setup code here, to run once:
  lcd.begin(16, 2);
  // lcd.print("Distance");

  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(buzzerPin, OUTPUT);
  Serial.begin(9600);
}

void loop() {
  // put your main code here, to run repeatedly:
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  distance = duration * 0.034 / 2;     // centimeters

  // checks if sensor bugs out
  if (distance > 100) {
    return;
  }
// Prints distance in Serial Monitor
  Serial.print("Distance: ");
  Serial.println(distance);

  distance_diff = distance2 - distance;

  // incoming fast
  if (distance_diff > 7 && distance < 30) {
    incoming(b_switch);
  }
```

```
  // tailgating
  if (distance < tailgating_distance) {
    warning(b_switch);
  }

// tracks distance over time
  if (i % 12 == 0) {
        distance2 = distance;
  }

  i += 1;
}
```

3 Tests and Patterns

There were many things I needed to consider before writing the functions for the software. I needed to figure out a “tailgating” distance that was scaled to the size for this project. I also need to figure out the speed at which an incoming car should be considered a warning. I used the ultrasonic sensor to figure out these distances and speeds. I found that the max distance the sensor can detect is around 70 centimeters, and it is pretty accurate in regards to recording distances of moving objects. However, I noticed that objects that are less than five centimeters away will throw the sensor off and record a distance of over 1000 centimeters but this was fixed with a simple if statement.

I also originally wanted to display the distance in real-time on the LCD Display but I found it took too much time and the display would be buggy because the screen would not completely clear. Therefore, I made it so that it only displays messages when there are warnings.

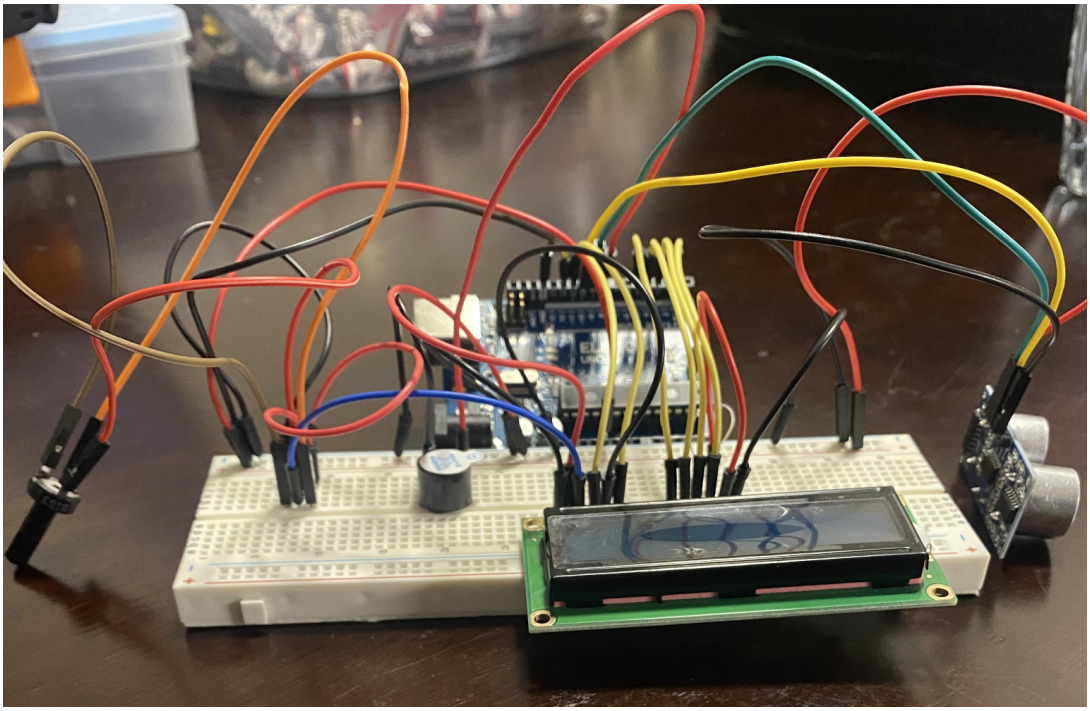
4 Photos

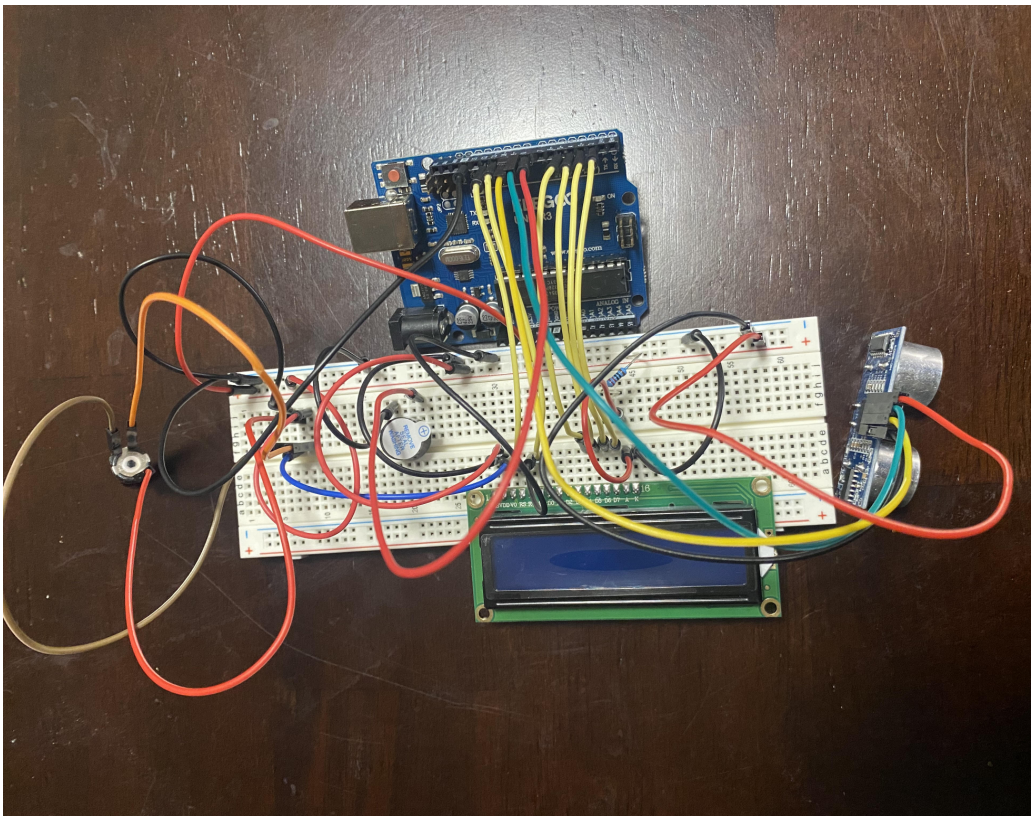
Software Prototype:

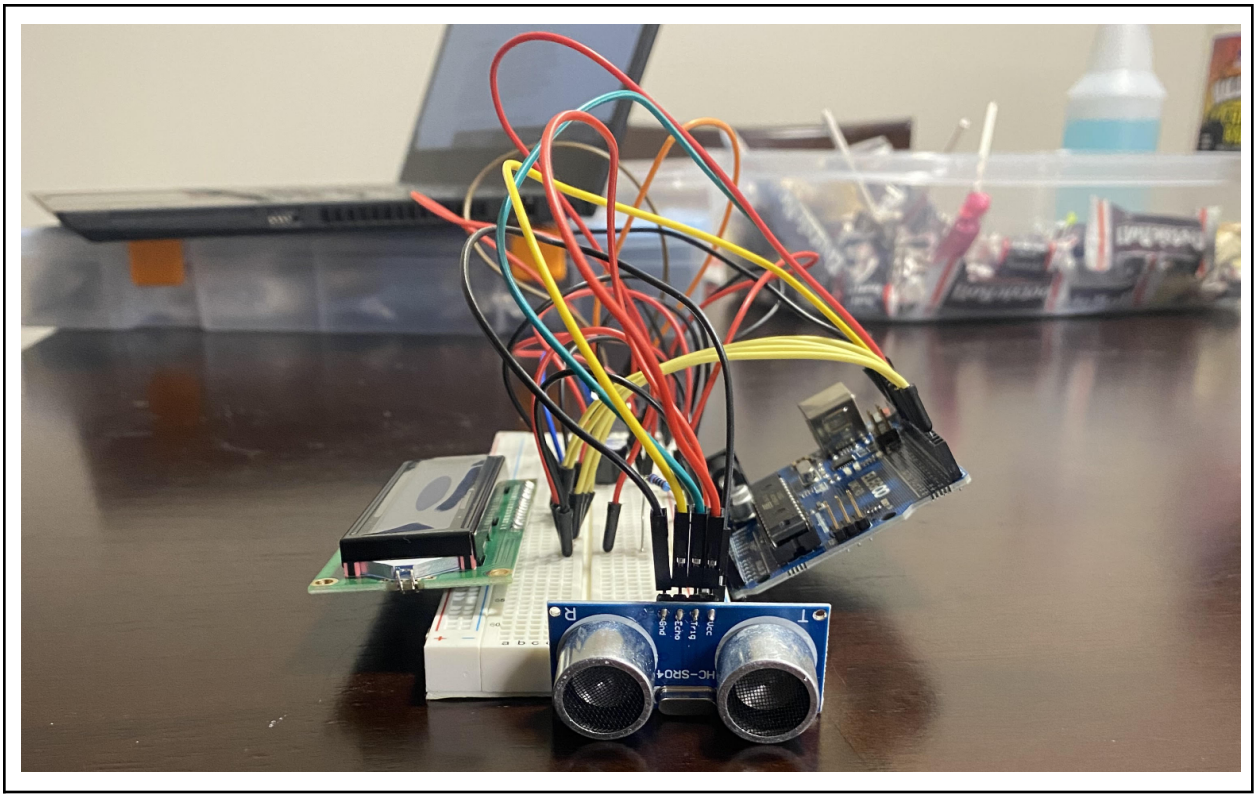
For code see section 2.

The warning() and incoming() functions both incorporate printing to the LCD and activating the buzzer. The buzz() function makes the buzzer beep one time, so I just call it multiple times in the other functions to get more output. A new distance is calculated in centimeters each time in the loop.

Hardware Prototype:







5 Observations and Notes

In my original design for the prototype, I wanted to use two ultrasonic sensors and one infrared sensor. I had thought that the ultrasonic sensors would not be the most accurate, but after testing I discovered that one was indeed enough and I did not even have to use an infrared sensor.

My next goal is to implement a way to detect if the driver is backing into a car and then output a warning similar to the tailgating warning. I also want to optimize the way I'm finding an incoming car's speed and try to make it more accurate.

6 References

LCD Libraries and Set-Up: https://docs.arduino.cc/learn/electronics/lcd-displays
Ultrasonic Sensor Set-Up:
https://howtomechatronics.com/tutorials/arduino/ultrasonic-sensor-hc-sr04/
Active Buzzer: https://www.youtube.com/watch?v=gj-H_agfd6U