SCIENCE JOURNAL

ASSIGNMENT 5

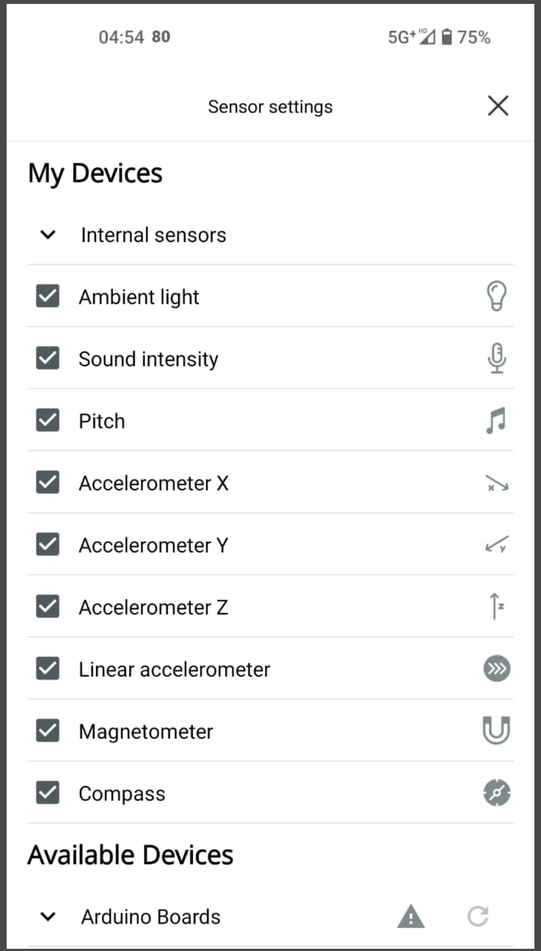
SENSORS

Science Journal, a mobile application developed by Google, provides a user-friendly interface to utilize the sensors on your smartphone in order to collect and analyze scientific data, allowing individuals to explore and better comprehend the scientific aspects of the world around them.

SENSORS

These are the sensors I found in my phone

- **Ambient light sensor**: A sensor that measures the amount of light in the surrounding environment and adjusts the screen brightness of your device accordingly.
- **Sound Intensity sensor**: A sensor that detects the intensity of sound waves in the surrounding environment, enabling you to measure the volume of sound.
- **Pitch sensor:** A sensor that determines the pitch of sound waves, allowing you to measure the frequency of a sound.
- **Accelerometer (X/Y/Z)**: A sensor that measures acceleration in the X, Y, and Z directions, providing information about the movement and orientation of your device.
- **Linear Accelerometer:** A variation of the accelerometer that measures linear acceleration, specifically in a straight line, without accounting for rotation.
- **Magnetometer:** A sensor that measures the strength and direction of magnetic fields in the surrounding environment.
- **Compass:** A variation of the magnetometer that determines the direction of north based on the Earth's magnetic field, enabling you to navigate using your device.

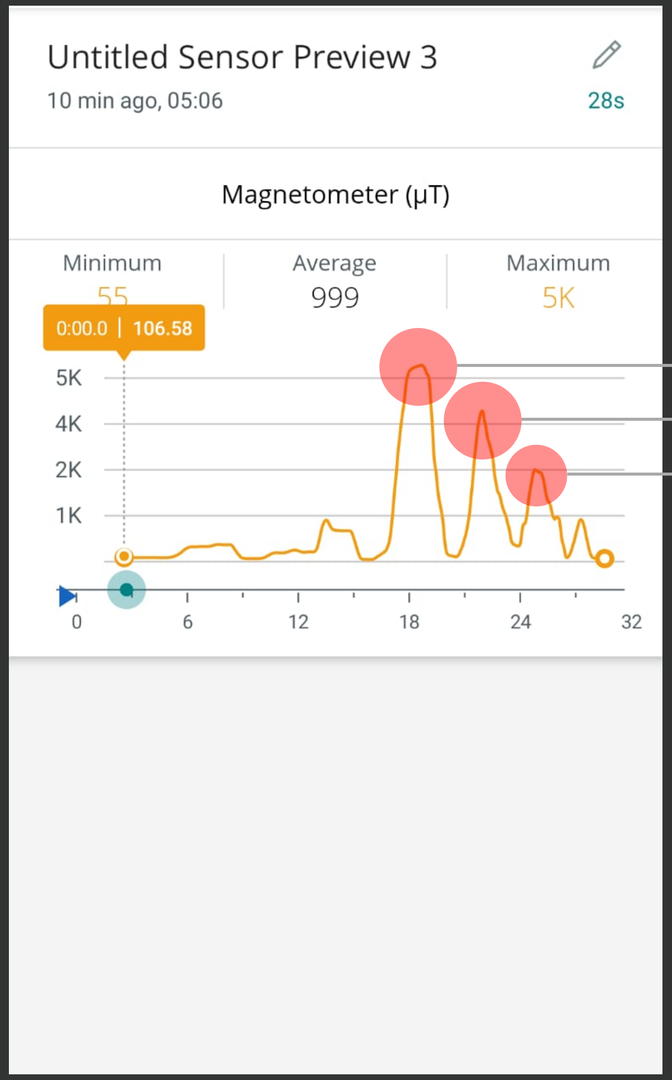


EXPERIMENTS

I started off by exploring the magnetometer which seemed like an interesting sensor.
I tried finding things that I felt would react in an interesting way with the sensor, and found that the ends of the lid and body of my laptop had a magnetic closing mechanism, which helps it to stay shut. And sure enough, it spiked the readings by quite a bit

Areas where the sensor spiked

EXPERIMENTS



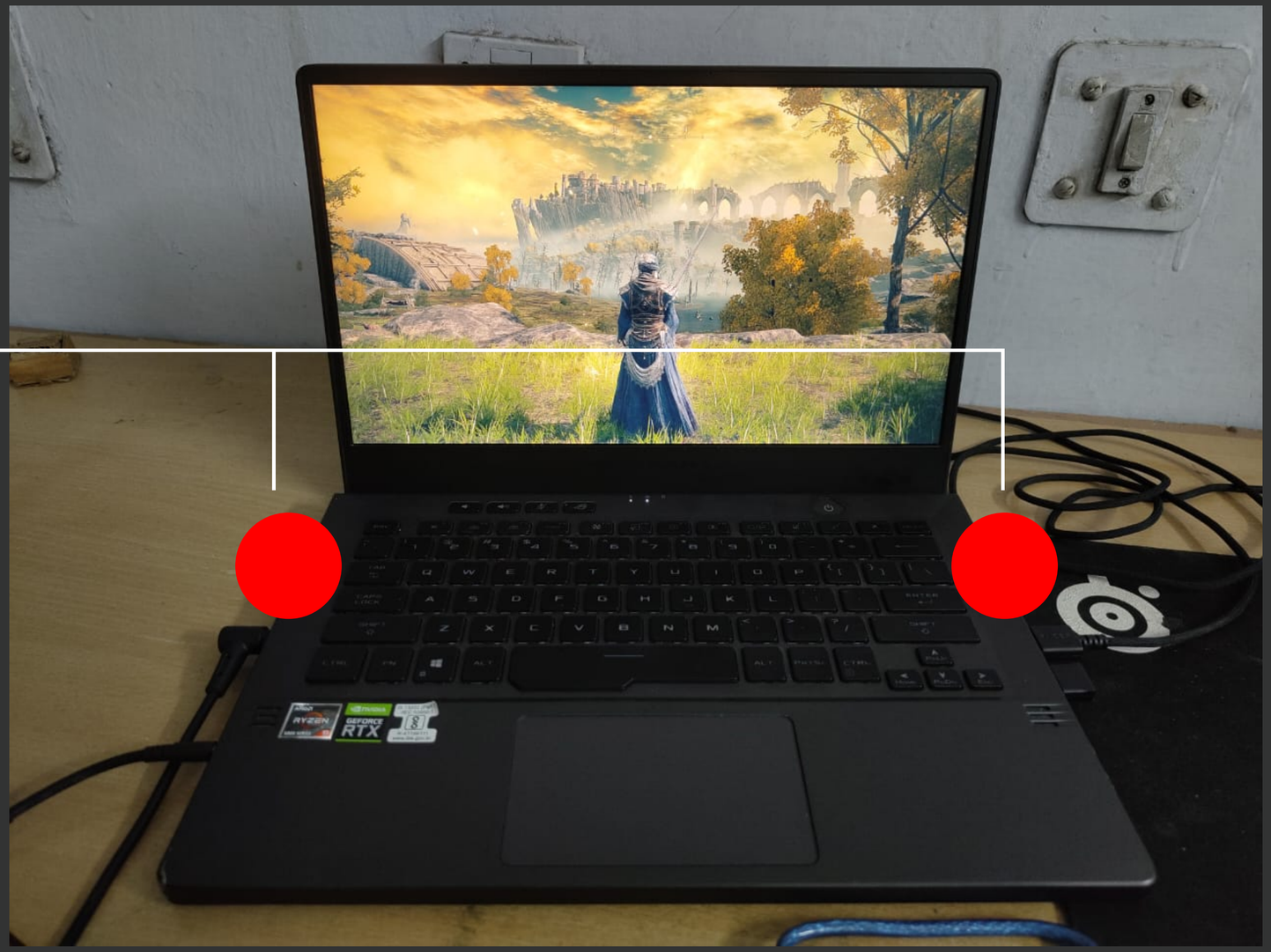
Peaks when brought close to the magnets

Areas where the sensor spiked

EXPERIMENT 2

Fan noise pressure test

Exhaust Vents



EXPERIMENT 2

Fan noise pressure test

I also decided to test the sound intensity sensor to measure the changes in the fan noise of my laptop during as it increases while running a GPU/CPU stress test.

EXPERIMENT 2

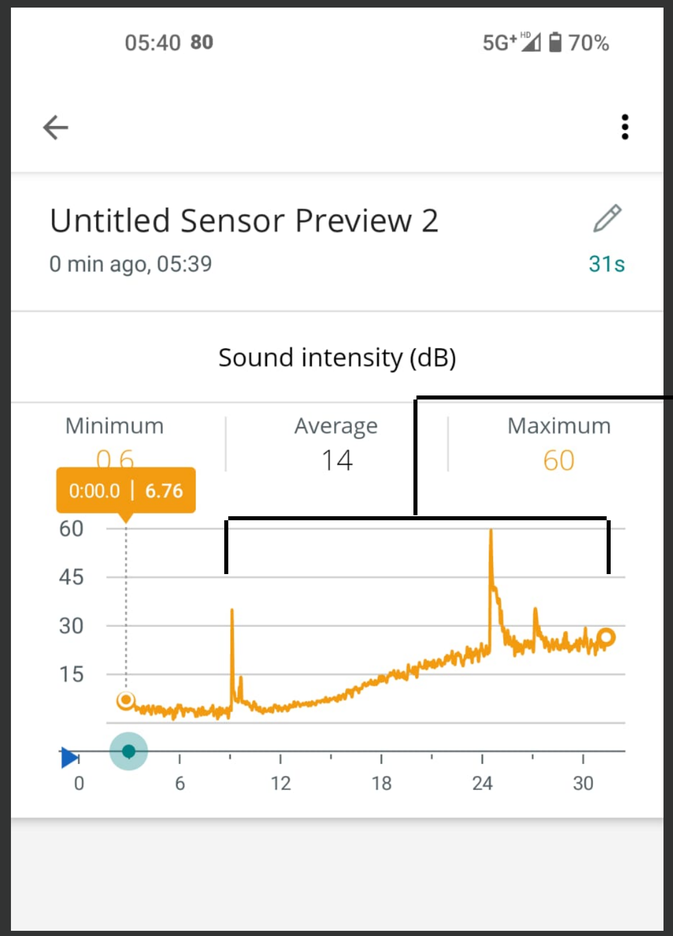
Fan noise pressure test

To do this I ran an extremely demanding program on my system and measured the fan noise near the exhaust vents.

As the temperature of the GPU/CPU started throttling, the fans ramped up in an attempt to cool the system down

EXPERIMENT 2

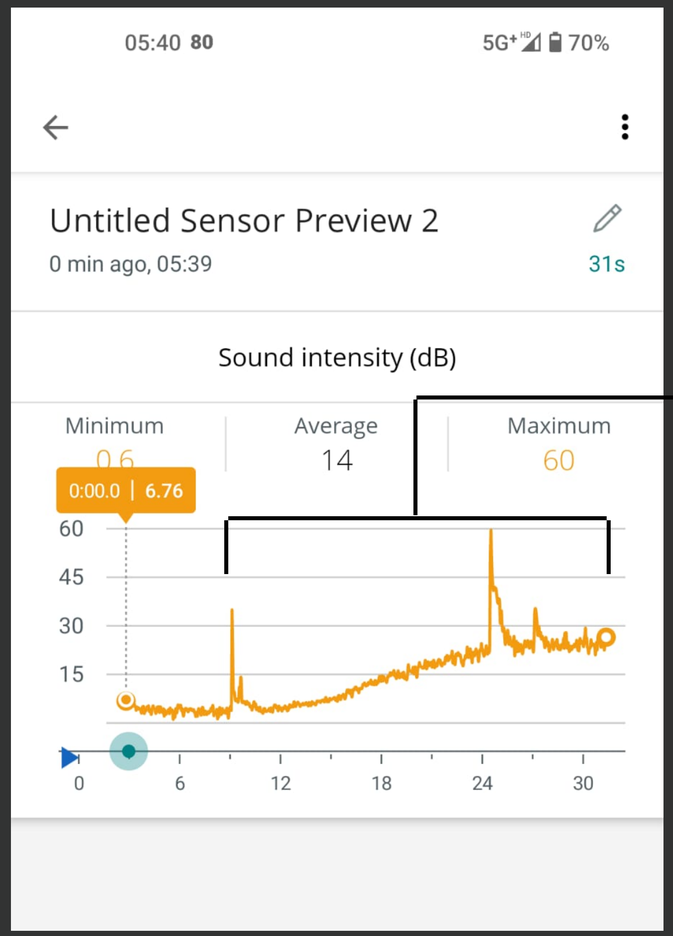
Fan noise pressure test



Gradual increase in fan noise

EXPERIMENT 2

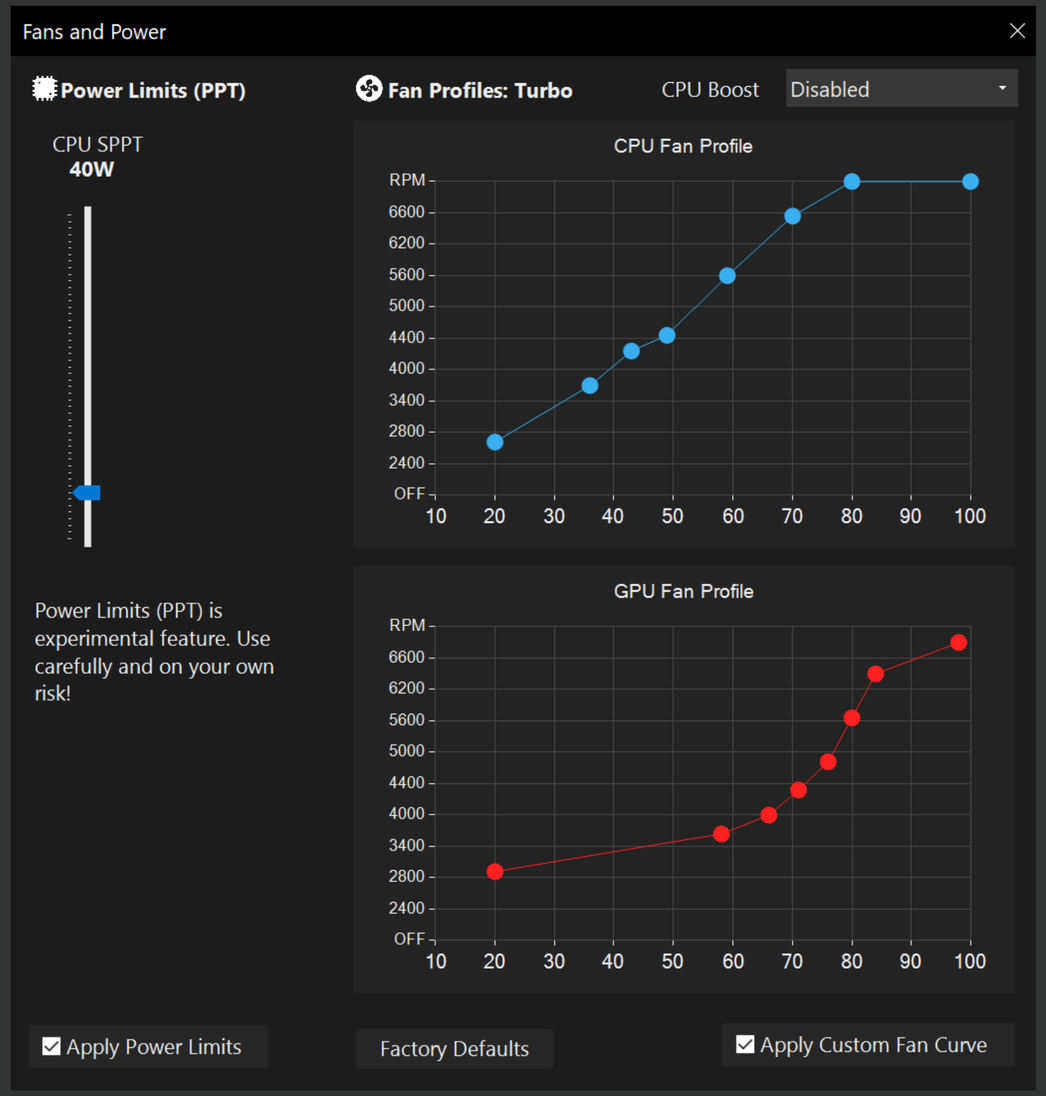
Fan noise pressure test

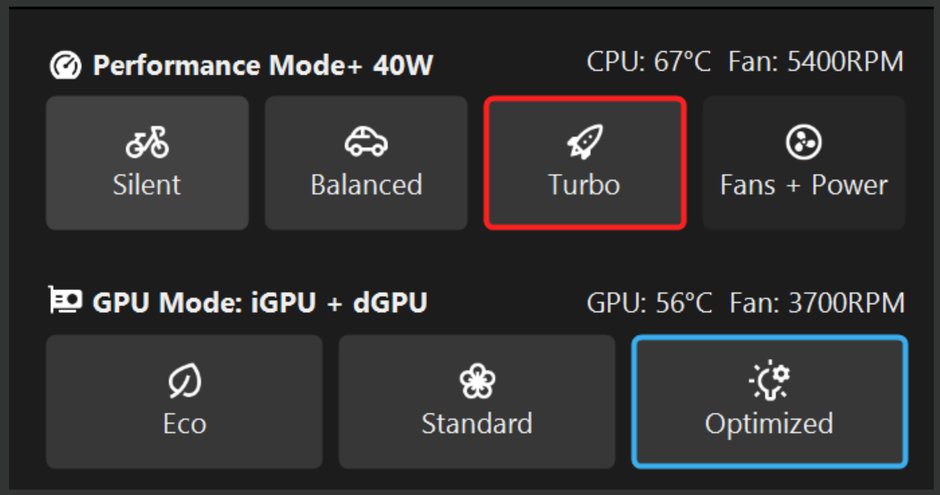


Gradual increase in fan noise

EXPERIMENT 2

Fan noise pressure test





I also monitored the fan curves and speed on the system

INPUT/OUTPUT

OVERVIEW

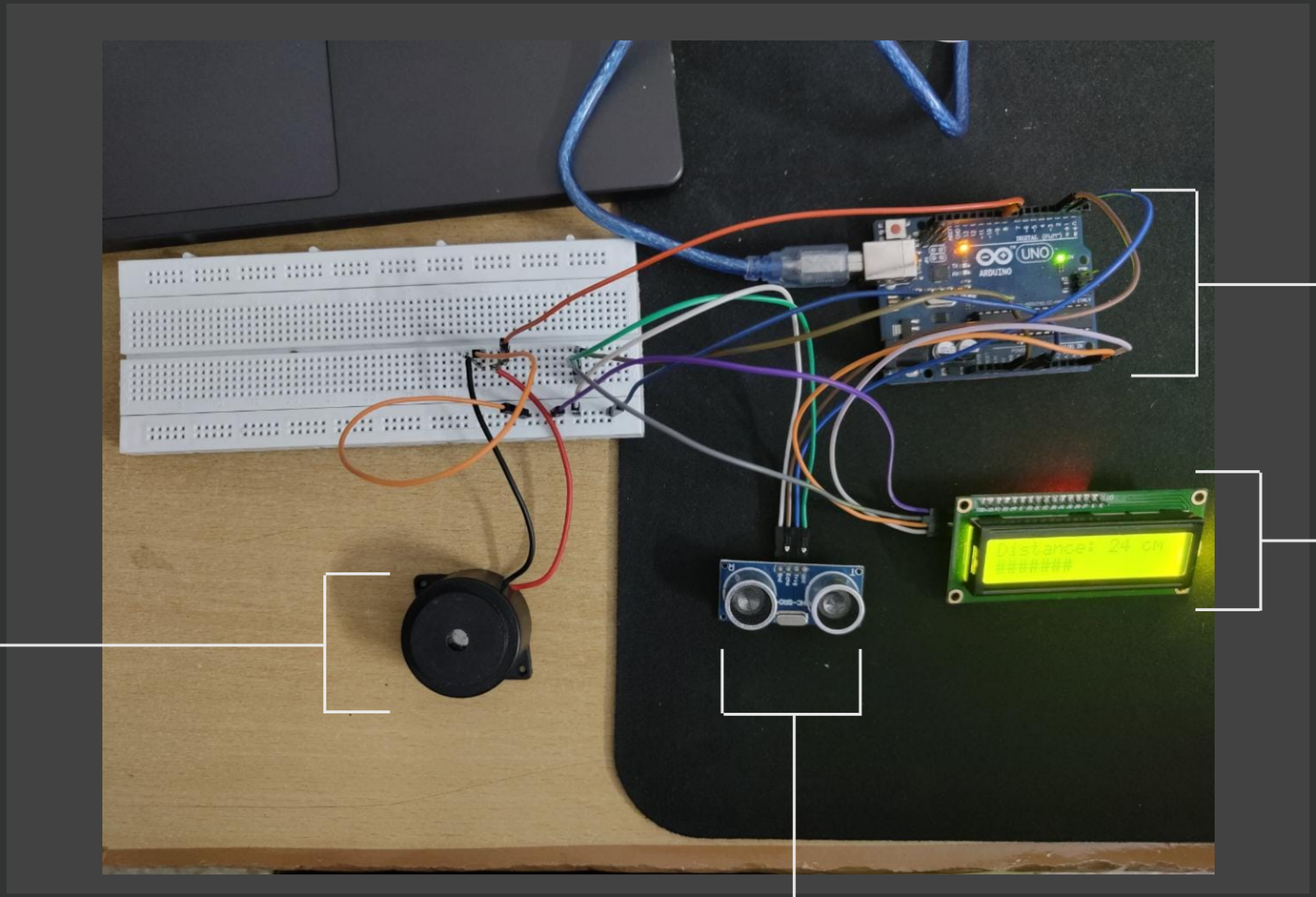
Proximity Alarm

The following combination of components acts as a sort of proximity alarm that can be applied to multiple use cases.

One use case that I can readily think of is a the mechanism of a car’s proximity sensor, that alerts the driver when the vehicle rolls close to a wall or obstruction

OVERVIEW

Circuit



OVERVIEW

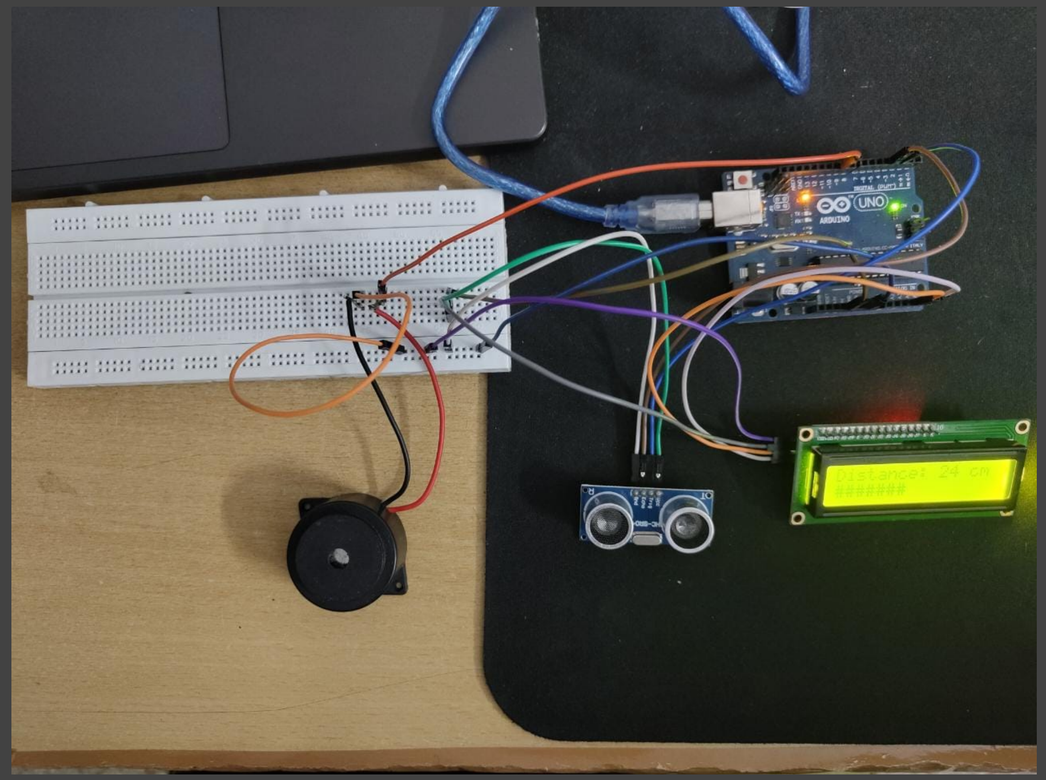
Circuit

Components/Workings

I2C Display: Displays a bar, which increases/decreases in proportion to the distance of an object to the ultrasonic sensor. The displayed distance value is also simplified and rounded off so that it is easier to understand

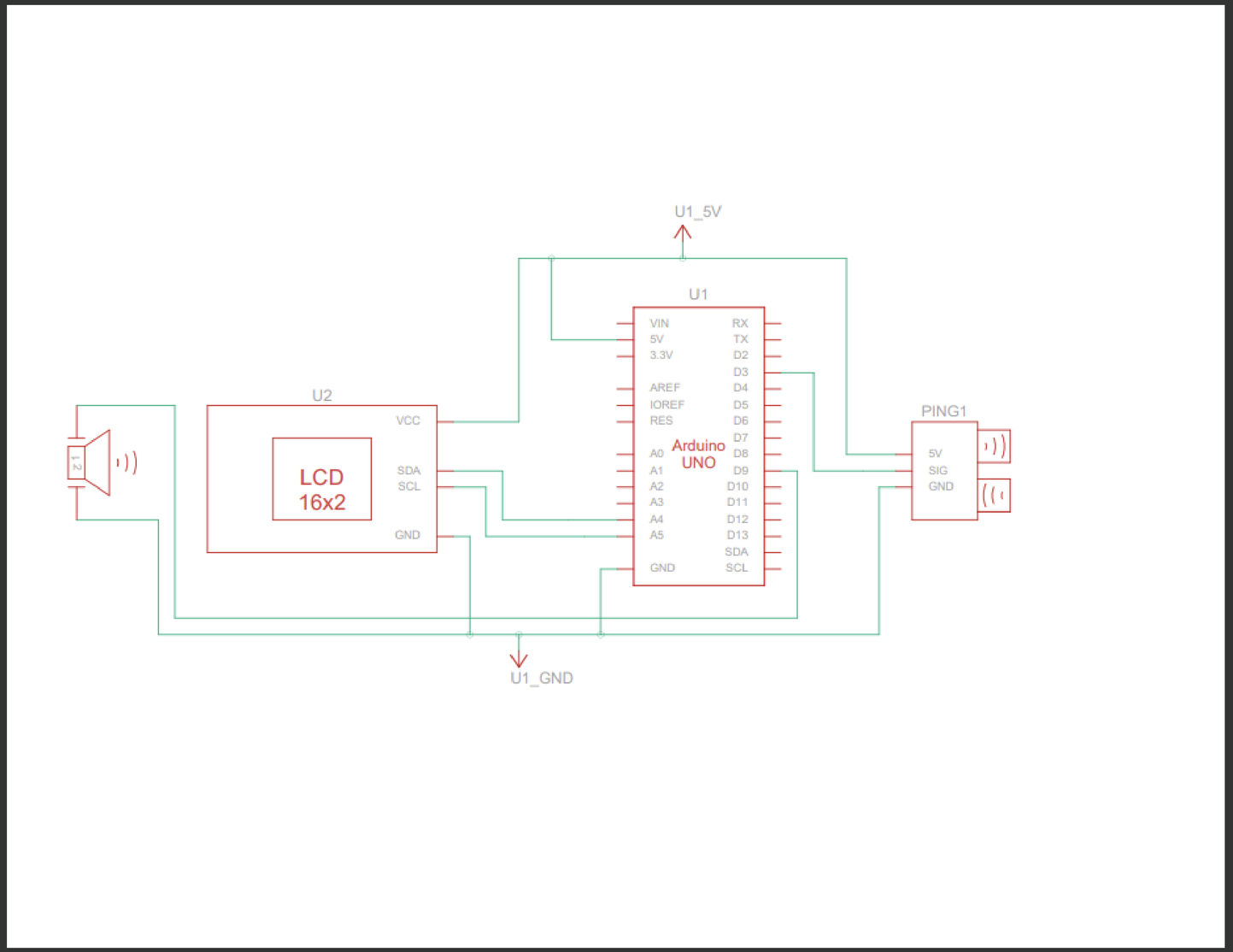
Ultrasonic Sensor: Measures distance from any object in its proximity

Buzzer: Sounds an alarm when any object crosses the threshold distance to the car to alert the driver



OVERVIEW

Circuit



OVERVIEW

Code

sketch_apr13a.ino

```ino
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

#define TRIG_PIN 2
#define ECHO_PIN 3
#define DISPLAY_COLS 16

LiquidCrystal_I2C lcd(0x27, DISPLAY_COLS, 2);

void setup() {
  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
  lcd.begin();
  lcd.backlight();
}

void loop() {
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);

  long duration = pulseIn(ECHO_PIN, HIGH);
  int distance = duration * 0.034 / 2;

  int barLength = map(distance, 0, 50, 0, DISPLAY_COLS);

  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Distance: ");
```

```ino
  pinMode(ECHO_PIN, INPUT);
  lcd.begin();
  lcd.backlight();
}

void loop() {
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);

  long duration = pulseIn(ECHO_PIN, HIGH);
  int distance = duration * 0.034 / 2;

  int barLength = map(distance, 0, 50, 0, DISPLAY_COLS);

  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Distance: ");
  lcd.print(distance);
  lcd.print(" cm");

  lcd.setCursor(0, 1);
  for (int i = 0; i < barLength; i++) {
    lcd.print("#");
  }

  delay(100);
}
```

OVERVIEW

sketch_apr13a.ino

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

#define TRIG_PIN 2
#define ECHO_PIN 3
#define DISPLAY_COLS 16

LiquidCrystal_I2C lcd(0x27, DISPLAY_COLS, 2);

void setup() {
  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
  lcd.begin();
  lcd.backlight();
}

void loop() {
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);

  long duration = pulseIn(ECHO_PIN, HIGH);
  int distance = duration * 0.034 / 2;

  int barLength = map(distance, 0, 50, 0, DISPLAY_COLS);

  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Distance: ");
```

Code

- The libraries Wire.h and LiquidCrystal_I2C.h are included for I2C communication and for controlling a 2x16 LCD display.
- Two constants are defined: TRIG_PIN and ECHO_PIN, representing the pins used to send a trigger signal and receive an echo signal, respectively. DISPLAY_COLS represents the number of columns of the LCD display.
- An instance of the LiquidCrystal_I2C class is created with the address of the LCD display and the number of columns.
- In the setup() function, the TRIG_PIN is set as an output pin, ECHO_PIN is set as an input pin, and the LCD display is initialized and turned on.
- In the loop() function, the TRIG_PIN is first set to LOW, then after a short delay, set to HIGH and then back to LOW again. This sends a 10 microsecond pulse to the ultrasonic sensor.

OVERVIEW

Code

- The duration of the pulse sent by the ultrasonic sensor is then measured using the pulseIn() function. The distance is then calculated using the speed of sound (approximately 34 cm/ms), divided by 2 as the signal has to travel to and from the obstacle.
- The distance is then mapped to a range of values between 0 and DISPLAY_COLS, where DISPLAY_COLS is the number of columns of the LCD display.
- A bar graph is created on the second row of the LCD display to represent the distance measurement. The number of "#" characters displayed corresponds to the mapped distance value.
- The first row of the LCD display shows the measured distance in centimeters.
- The loop() function then waits for 100 milliseconds before repeating the measurement and display process.

```
12    pinMode(ECHO_PIN, INPUT);
13    lcd.begin();
14    lcd.backlight();
15  }
16
17  void loop() {
18    digitalWrite(TRIG_PIN, LOW);
19    delayMicroseconds(2);
20    digitalWrite(TRIG_PIN, HIGH);
21    delayMicroseconds(10);
22    digitalWrite(TRIG_PIN, LOW);
23
24    long duration = pulseIn(ECHO_PIN, HIGH);
25    int distance = duration * 0.034 / 2;
26
27    int barLength = map(distance, 0, 50, 0, DISPLAY_COLS);
28
29    lcd.clear();
30    lcd.setCursor(0, 0);
31    lcd.print("Distance: ");
32    lcd.print(distance);
33    lcd.print(" cm");
34
35    lcd.setCursor(0, 1);
36    for (int i = 0; i < barLength; i++) {
37      lcd.print("#");
38    }
39
40    delay(100);
41  }
42
```

OVERVIEW

Media

DEMO

https://youtu.be/fnfrxfDUrrA

Source Code

https://github.com/shagore/ard5/blob/main/assn5.ino