General Engineering Department

ROGUE

(Rescue Or Get U Escaped)

Final Design Report (FDR)

Project Team Members:

Caroline Barker
Yiqing Guo
Lucia Jeon
Lin Lin Jin

1. INTRODUCTION

1.1 Purpose of Project

The motivation for this project was to address the issue of emergency response systems by designing a robot that offers an alternative to the current routine that occurs during emergency situations. The created robot is able to independently navigate and traverse a given room, send the locations of disaster victims to emergency personnel, and provides a safer and more efficient alternative for rescue teams in locating victims.

1.2 Background of the Experiment

After sudden emergency disaster situations--such as earthquakes, tsunamis, storms, etc. There are often many people trapped under debris caused by damaged infrastructure from the disaster. In these high pressure and intense situations, a mere matter of hours can determine the difference between life and death for many of these victims. In these cases it is essential that life-saving search and rescue operations are efficient in locating victims and possess the skills to navigate these extremely dangerous landscapes especially when infrastructure may be disrupted or entirely destroyed. According to the UNDAC, disaster response organizations are usually deployed within “12-24 hours of an emergency” (UNDAC).

The wait of 12 to 24 hours in order for help to be deployed is much of a concern in such time sensitive situations. Because of this, it was proposed that providing an alternative to human rescue teams would be most beneficial in reaching the most amount of people in the quickest amount of time. In order to minimize the chance of endangering more people--such as those putting their life on the line in rescue teams--on dangerous landscapes,

a robot instead would navigate the terrain and pinpoint the location of discovered victims and send that location to emergency personnel. In this sense, the concept of ROGUE was developed.

2. Requirements

2.1 Physical Components

The required physical components consisted of a moving body in order to traverse the landscape. In order to achieve this, two wheels are on either side of the robot frame, as well as a small peg in the front to provide stability. In order for the Infrared sensor to sense people, there is a space between the body of the robot and the top cover so that the camera sensor can poke through. The top cover is also semi-transparent in order for the LED to radiate red in an emergency. Also located on the top cover of the robot is a button that is connected to the Arduino red board that acts as an automatic prompt of sending the location. Other physical components consisted of the microphone sensor which is mounted within the body of the robot in order to sense any local sounds, as well as the bluetooth module--which is also mounted within the body of the robot--and controls all communication between sending location to emergency services.

2.2 Software Components

Required software components consisted of an Arduino red board that controls all the sensors that ROGUE utilizes. The Arduino IDE was essential in programming each of the different sensors and other physical components: the microphone sensor, the infrared sensor, the button, the motor, the LED strip, and the ultrasonic sensor.

3. Procedures

3.1 Physical Construction

1. Initial design (pivoted, Figure 1, 2, 3)

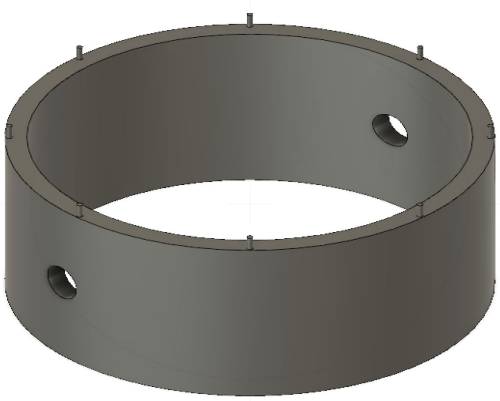


Figure 1. Initial design 1

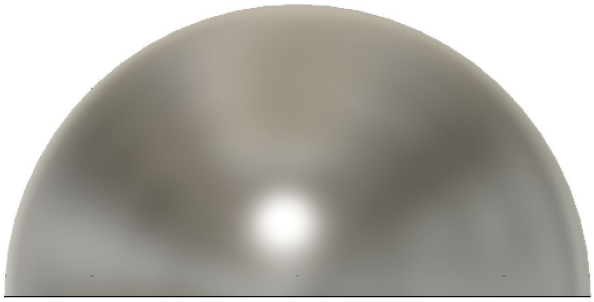


Figure 2. Initial design 2

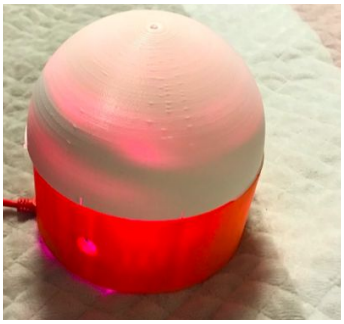


Figure 3. Initial design 3

2. Design now (Figure 4, 5, 6)

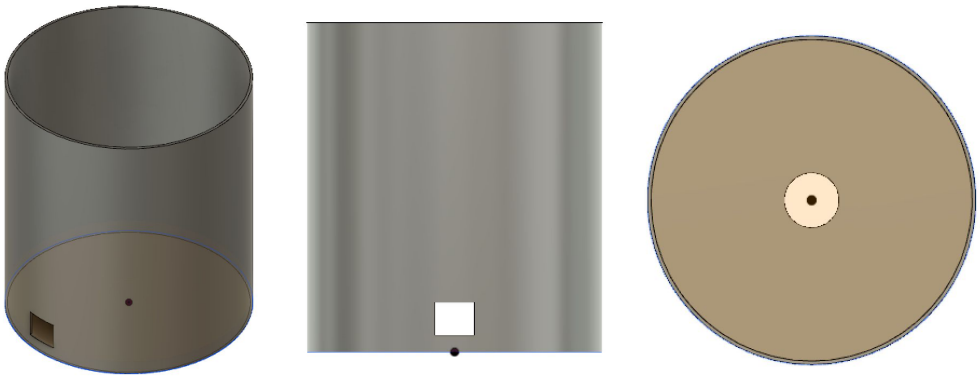


Figure 4. Design now 1

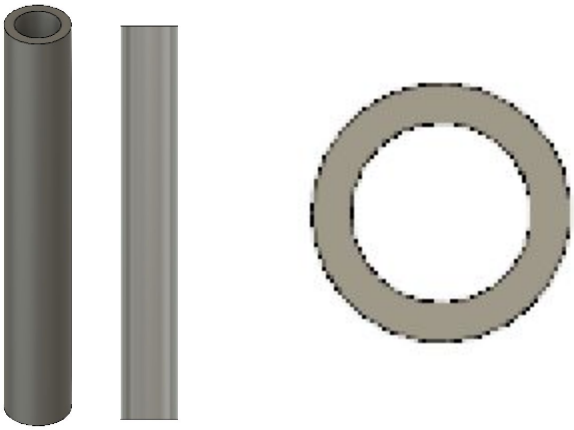


Figure 5. Design now 2

Figure 6. Design now 3

3.2 Software Setup

ROGUE largely depends on its motors and sensors. Please refer to the Appendix for the direct code that ROGUE uses.

One of the sensors utilized by ROGUE is the microphone sensor. Using the Arduino IDE interface, a voltage reading was calculated using the Arduino microphone input (MAX9814). Then using a series of equations--measuring amplitude and change in frequency--a decibel was calculated. After determining the decibel, the conditional would compare this reading to the average decibel reading of a fire alarm. Once the reading of the decibel is equal to or higher than the emergency decibel reading, the robot would then enter the emergency protocol. Appendix A demonstrates the implementation of the MAX9814 microphone sensor.

The second sensor utilized by ROGUE is the Infrared Sensor. After entering emergency protocol, ROGUE traverses the room to find victims from the disaster. Using the average temperature of the human body, the robot scans the room to determine if the temperature of an object matches. Once there is a match, ROGUE sends the object's position. See Appendix B for the implementation of the infrared module.

Another implementation if the Infrared Sensor is faulty or readings aren't exact is the inclusion of the button--also demonstrated in Appendix B. If a victim presses ROGUE'S button during an emergency situation, the location of ROGUE is automatically sent to emergency personnel. This feature is to added as a safety feature that is completely automatic and doesn't rely on a sensor.

Once ROGUE enters emergency protocol, it is programmed to radiate a red light to notify people of its presence. The inclusion of this LED strip is to make it easier for victims to draw attention to themselves if the robot is near. Thus victims can either better position themselves to be seen by ROGUE or be able to press ROGUE's emergency button if it draws near. See Appendix C for the LED strip coding.

3.3 Software Troubleshooting

When the code for the microphone sensor was first tested, the results were not as expected. Sounds were played at various volumes in order to test the functionality of the sensor using the code demonstrated in Appendix D. What was thought was supposed to be an easy calculation of the decibel reading from the sensor was in fact a voltage reading. However, the confusion it caused, delayed the creation of the finished product. A faulty sensor was first to blame for the inaccurate readings. However, it was determined that the sensor was functioning normally, the only issue was that the readings it was outputting was not readings that could be so easily transformed into decibels--they were in fact voltage readings. The converting to decibels required much more complicated and extensive equations, demonstrated in *readVcc()* method of Appendix A.

Other troubleshooting of code consisted of using a sensor and an indicator. Appendix C is the code used to troubleshoot the decibel readings and what happens when a decibel reading is at the average decibel reading of an alarm. The indicator of that noise level is the lighting up of the LED strip, demonstrating that the body of the code works.

4. Milestone and Final Product Requirements

4.1 Benchmark A Requirements

- Hardware:
    - Preliminary prototype of robot frame that contain battery and motor mounts. Must have space for Arduino board and wiring.

- Software:
    - Completed value proposition canvas, pseudocode of program (written out functions in the code, no syntax required) that encompass four functionalities of the robot.
- Have motor and sensor calculations made and items purchased.
- Rethink name for our product so that it's more marketable: We need to think about who we are selling our robot to.

4.2 Benchmark B Requirements

- Software: 3 out of 4 sensors coded to function
    - IR/thermal
    - Bluetooth: Interact with computer. Be able to send and receive date to and from computer.
    - Mic/sound: Pick up sound at a certain dB and set off alarm.
    - Ultrasonic: Rotate around and pick up distance readings.
- Hardware:
    - Almost-final CAD & print of robot frame
    - Electrical components fully wired

4.3 Final Submission Requirements

Finished Robot Body	Includes a final CAD model (parts and assembly completed and fully-defined), 3D-printed, and includes mounts, fasteners, holes, extra members to hold all others components, such as the wheels, motor, battery, etc.	25

All Sensors Coded & Wired	All four sensors programmed with Arduino and functioning properly. Also includes finished wiring/soldering of wires to the board.	25
Mapping Function	A fully-functioning and generally efficient mapping function. This includes the IR sensor swiveling around on an axis and being able to sense people and place their locations on a map. This includes the ultrasonic sensor swiveling on an axis and reading what objects are in the vicinity and also placing the locations on a map.	50

4.6 Human Resources and Training

Some resources and training attended and used were open labs and workshops, especially after multiple pivots of project idea. There was difficulty of deciding what kind of robot to make and how to start. Ideas were pivoted many times that even when it is already half the semester. Therefore, members of the group attended multiple open labs during the spring break to ask for help and suggestions. From the help of the RAD TA, the idea of an emergency rescue robot was finally decided and started.

5. Results

5.1 Benchmark A Results

- Hardware: Finished with the preliminary prototype of robot frame/body that has enough space for placing equipment. (Figure 7, 8)

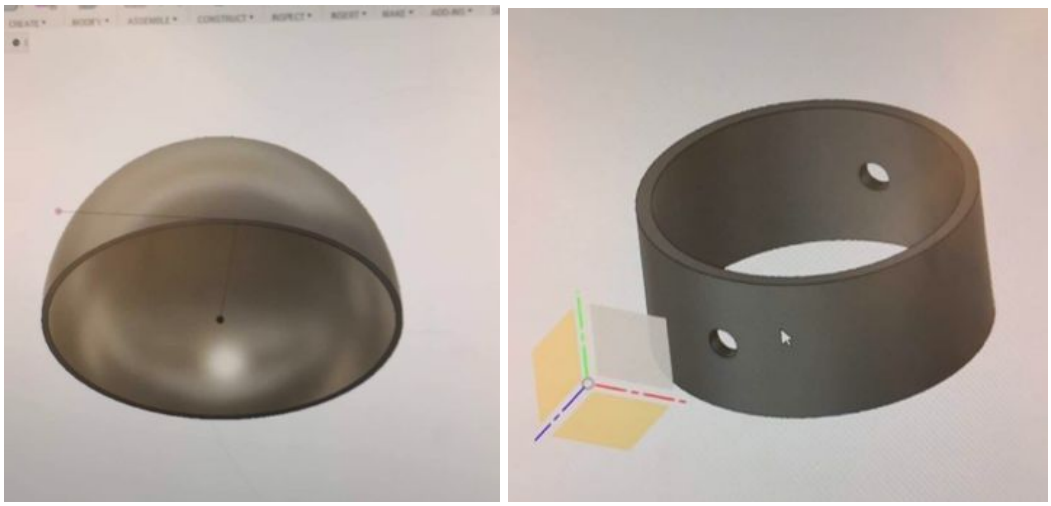


Figure 7. CAD drawings

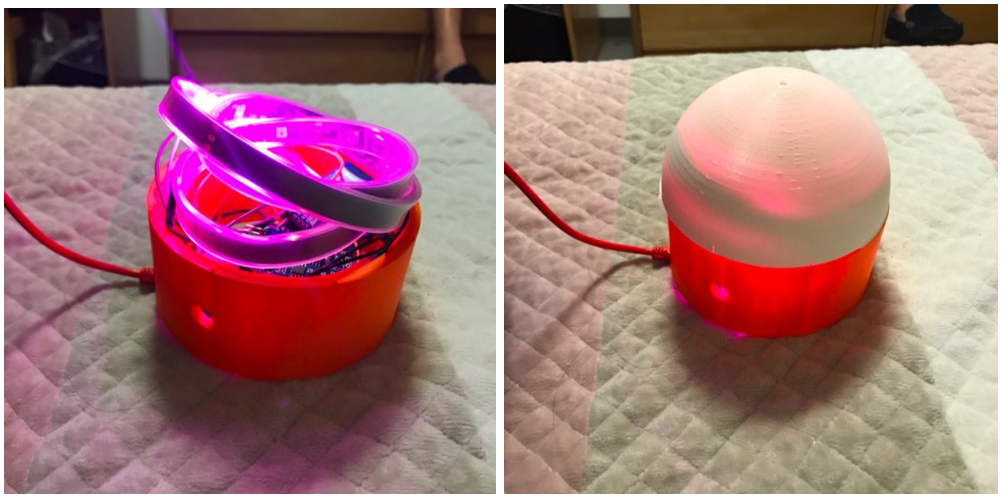


Figure 8. 3D printed models

- Software: The Value Proposition Canvas was completed and pseudocode of program was written and demonstrated the basic concept of the product. (Figure 9)

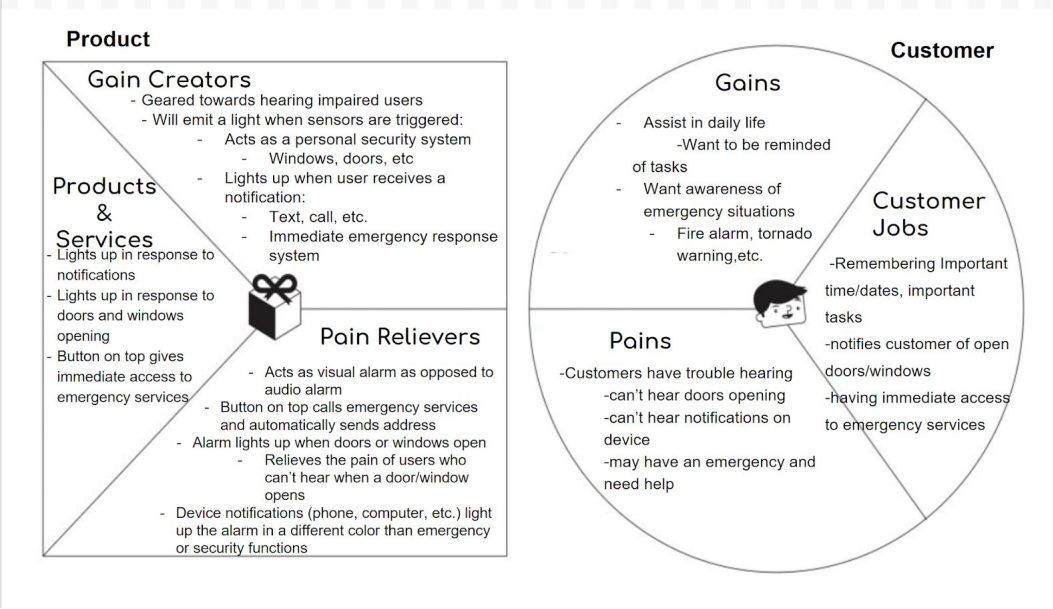


Figure 9. Value Proposition Canvas

- -Bluetooth and LED connected to Arduino board and coded. (Figure 10, 11)

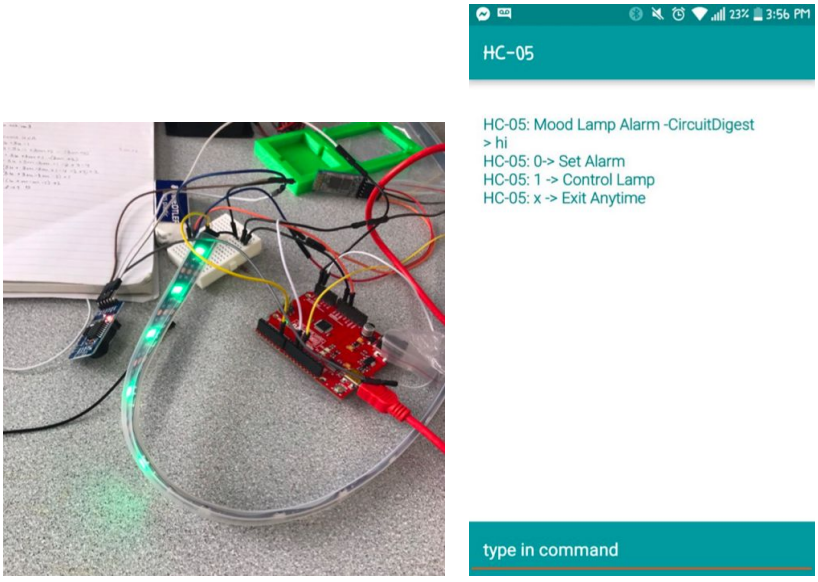


Figure 10. Circuits and Bluetooth screen

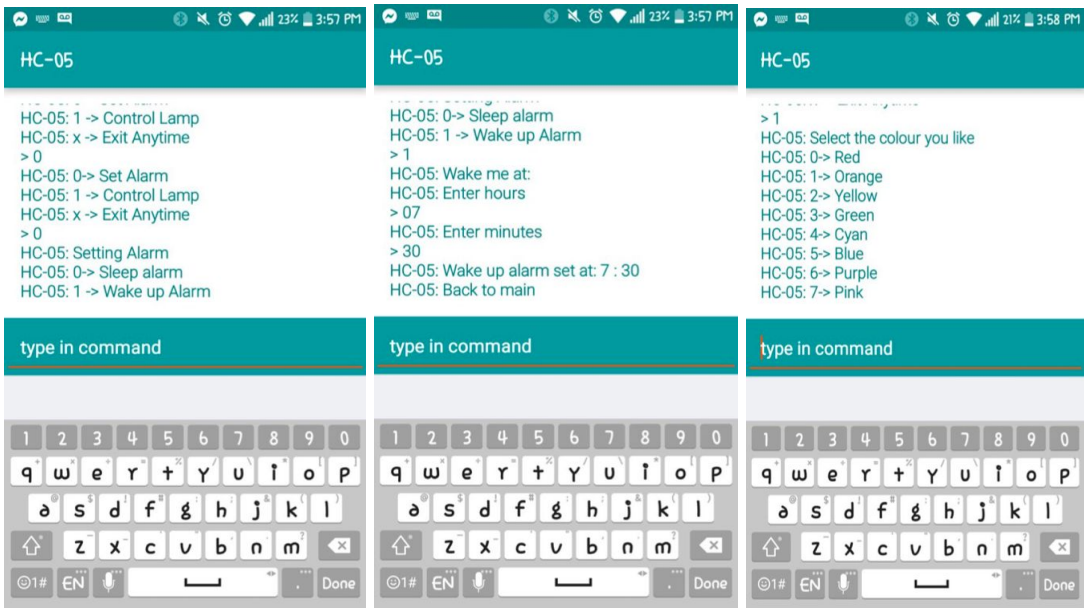


Figure 11. Bluetooth screens

- Motor and sensor calculations made and items purchased.

Project Cost Estimate	
Equipment Cost Estimate	
Ultimaker 3D Print	$0
Arduino Board	$0
Arduino Speaker	$0
IR Remote/Reciever on Arduino	$0
Real Time Clock Module for Arduino	$6
HC-05 Bluetooth Module for Arduino	$9
LDR	$0
LEDs	$10
100k Resistors	$0
Smartphone	NA
Labor Cost Estimate	
$50 an hour, 4 members, 60 hours per member	$12,000
TOTAL	$12,025

- The name of the robot was changed.
- Overall grade: 100%

5.2 Benchmark B Results

- Sensors are coded to function (Figure 12)

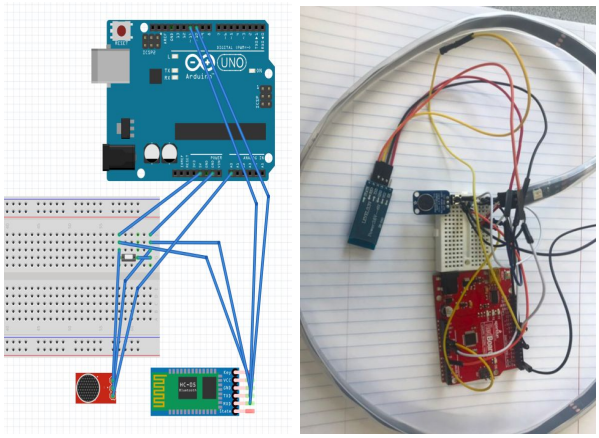


Figure 12. Circuit schematics

- Almost-final CAD & print of robot frame (Figure 13)

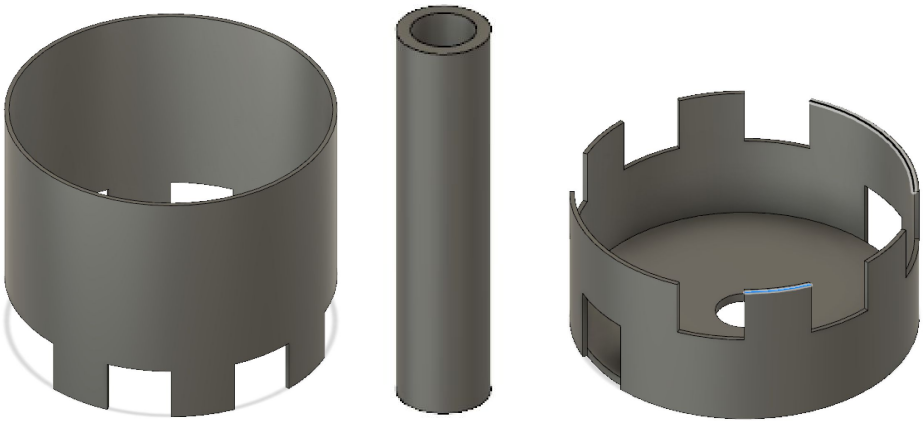


Figure 13. CAD drawings

- Electrical components fully wired
- Overall grade: 100%

5.3 Difficulties Experienced

The main problem encountered was to brainstorm an innovative robot that can solve a real life problem. The initial topic given was a prompt-based topic, “A Friend to Clean”. Basically, it is a roomba that also functions like a “friend”. Since roomba is already out the market, the idea was then changed to an educational robot. However, the topic was taken by another and the topic had to be pivoted once more. The idea changed to an alert robot that assist people in daily life. It responds to movement, lights up and acts like an alarm, and follows commands through an app. However, it was not innovative enough and the topic was once again pivoted. Then, the idea was changed to a robot that functions like a security and notification system that aimed to help hearing impaired people. The idea was also pivoted because it was too general and not innovative. However, from this idea, the idea of emergency assistance was zoomed in and finally came up of an idea of making an emergency robot that tracks and rescues people in emergency situations.

6. Conclusion

6.1 Results of Project

As a result, we achieved our goal of creating an emergency response robot by integrating a variety of sensors with a mobile robotic body. These components allow the robot to detect humans, obstacles, and its location in order to fulfill its function of finding people that are trapped or lost in case of an emergency. Physically, the robot is capable of holding all the relevant components, and is mobile on its two wheels. Software-wise, the robot’s various sensors - thermal, microphone, and ultrasonic - are all coded to take in data from acceptable ranges. In addition, bluetooth is present in the robot’s software to facilitate the transfer of data from the

robot to emergency services. Currently, we are working on the 3D printing of the finished design, as well as the coding that efficiently define coordinates of the room.

6.2 Future Improvements

Different functions can be added to our robot to improve the operation. First, camera can be added to take picture or video of the person inside the room. Since we are only sending coordinates of the people inside, it will be a better solution to also utilize images of the inside to better inform the situation. Also, it can provide a live-feed camera that can be accessed by the user. Second, rather than detecting the alarm of the sound, we can have better method to detect emergency. For example, an earthquake detector or smoke detector can be utilized rather than microphone. Third, instead of using an in-built map, we can use the mapping function to traverse from one end of the room to the exit of the room. In order to do so, a sensor that detects the distance from the wall that incorporates three-dimensional space is needed.

7. Works Cited

"Emergency Handbook." *UNHCR*, The UN Refugee Agency, 24 Apr. 2018, emergency.unhcr.org/entry/258567/search-and-rescue-response-and-coordination-natural-disasters.

Appendix A

The following is the coded microphone sensor.

```
#define PIN 6
#define N_LEDS 16

Adafruit_NeoPixel pixels = Adafruit_NeoPixel(N_LEDS, PIN, NEO_GRB + NEO_KHZ800);

const int MIC = A0;
float dbValue;

void setup(){
  pinMode(MIC, INPUT);
  Serial.begin(9600); //baud rate at 9600 -> serial port trasnferring 9600 bits/sec
  pixels.begin();
}

void loop(){
  float ref_volt = float(readVcc())/1000.0;
  float dbValue = (analogRead(MIC)/1024.0)*ref_volt*50.0;
  Serial.println(dbValue);

  if (dbValue > 150){
    Serial.println("I am in the loop");
    pixels.setPixelColor(1, pixels.Color(200, 0, 0));
    pixels.show();
  }
  else{
  pixels.setPixelColor(1, pixels.Color(0, 200, 0));
  pixels.show();
  }
  delay(300 );
}


// read voltage to ensure ADC converts properly
long readVcc(){
   // Read 1.1V reference against AVcc
  // set the reference to Vcc and the measurement to the internal 1.1V reference
  #if defined(__AVR_ATmega32U4__) || defined(__AVR_ATmega1280__) || defined(__AVR_ATmega2560__
    ADMUX = _BV(REFS0) | _BV(MUX4) | _BV(MUX3) | _BV(MUX2) | _BV(MUX1);
  #elif defined (__AVR_ATtiny24__) || defined(__AVR_ATtiny44__) || defined(__AVR_ATtiny84__)
    ADMUX = _BV(MUX5) | _BV(MUX0);
  #elif defined (__AVR_ATtiny25__) || defined(__AVR_ATtiny45__) || defined(__AVR_ATtiny85__)
    ADMUX = _BV(MUX3) | _BV(MUX2);
  #else
    ADMUX = _BV(REFS0) | _BV(MUX3) | _BV(MUX2) | _BV(MUX1);
  #endif

  delay(2); // Wait for Vref to settle
  ADCSRA |= _BV(ADSC); // Start conversion
  while (bit_is_set(ADCSRA,ADSC)); // measuring

  uint8_t low  = ADCL; // must read ADCL first - it then locks ADCH
  uint8_t high = ADCH; // unlocks both

  long result = (high<<8) | low;

  result = 1125300L / result; // Calculate Vcc (in mV); 1125300 = 1.1*1023*1000
  return result; // Vcc in millivolts
```

Appendix B

The following code demonstrates the implementation of the Infra-Red sensor as well as the button.

```
#include <Wire.h> // Include Wire.h - Arduino I2C library
#include <SparkFunMLX90614.h> // Include IR thermometer library

IRTherm temp; // Create an IRTherm object called temp

void setup() {
  // put your setup code here, to run once:
  temp.begin(); // Initialize I2C library and the MLX90614
  temp.setUnit(TEMP_F); // Set units to Farenheit (alternatively TEMP_C or TEMP_K)
  Serial.begin(9600);
}

void loop() {
  // put your main code here, to run repeatedly:
  if (temp.read()) // Read from the sensor
  { // If the read is successful:
    float ambientT = temp.ambient(); // Get updated ambient temperature
    float objectT = temp.object(); // Get updated object temperature
    Serial.println("Ambient: " + String(ambientT));
    Serial.println("Object: " + String(objectT));
    Serial.println();
  }
  delay(500);
}
```

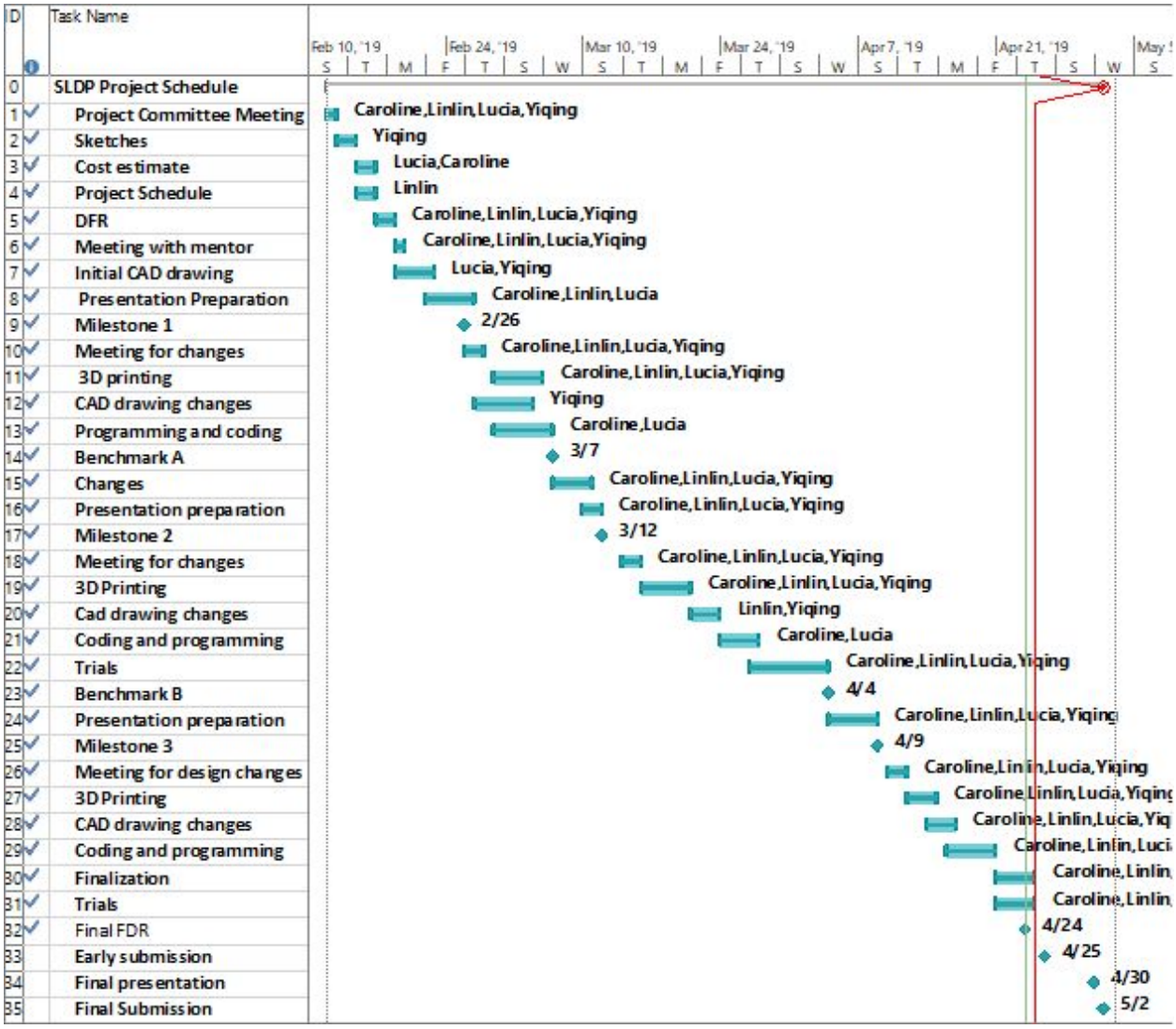
Appendix C

The following code demonstrates the implementation of the LED strip.

```
#include <Adafruit_NeoPixel.h>
#include <SoftwareSerial.h>

// Bluetooth connected to 10 and 11
SoftwareSerial BTserial(10, 11);
const int LEDPin = 6;
int LEDValue;

const int buttonPin = 1;
int buttonState = LOW;

const int micPin = A0;
int micState;

//Adafruit_NeoPixel pixels = Adafruit_NeoPixel(16, 6, NEO_GRB + NEO_KHZ800);
void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  //pinMode(buttonPin, INPUT);
  pinMode(micPin, INPUT);
  pixels.begin();=,
}

void loop() {
  // put your main code here, to run repeatedly:
  int adc, dB;
  adc = analogRead(micPin); //read in initial pin
  Serial.println(adc);
  dB = (adc + 83.2073) / 11.003; //getting the decibal value

  Serial.println(dB);

  if (dB > 80){
    pixels.setPixelColor(1, pixels.Color(200, 0, 0));
    pixels.show();
  }
  else{
    pixels.setPixelColor(1, pixels.Color(0, 200, 0));
    pixels.show();
  }

  delay(700);
}
```

Additional documents:

Project schedule from Milestone 3:



Cost estimate:

Project name:					
Labor Cost Estimate Breakdown Table					
DESCRIPTION	**AMOUNT**		**UNIT PRICE**		**TOTAL**
Software Manager	60	Hrs	$ 50.00	/ Hr	$ 3,000.00
Hardware Engineer	60	Hrs	$ 50.00	/ Hr	$ 3,000.00
Project Manager	60	Hrs	$ 50.00	/ Hr	$ 3,000.00
Program Manager	60	Hrs	$ 50.00	/ Hr	$ 3,000.00
PROJECT COST ESTIMATE					**$ 12,000.00**

Equipment Cost Estimate Breadown Table					
DESCRIPTION	**AMOUNT**		**UNIT PRICE**		**TOTAL**
Ultimaker 3D Print	1	Unit(s)	$ -	/ Unit	$ -
Arduino Microphone	1	Unit(s)	$ 8.00	/ Unit	$ 8.00
Arduino Board	1	Unit(s)	$ -	/ Unit	$ -
Arduino Speaker	1	Unit(s)	$ 8.00	/ Unit	$ 8.00
Arduino Oscillator	1	Unit(s)	$ 7.00	/ Unit	$ 7.00
HC-05 Bluetooth Module	1	Unit(s)	$ -	/ Unit	$ -
LED Strip	1	Unit(s)	$ -	/ Unit	$ -
100k Resistors	2	Unit(s)	$ -	/ Unit	$ -
Smartphone/app	1	Unit(s)	$ -	/ Unit	$ -
EQUIPMENT COST ESTIMATE					**$ 23.00**

GRAND TOTAL FOR ALL COST ESTIMATES:	**$12,023.00**