

6-13-2019

Sensor System for Rescue Robots

Alexander Moran

Emir Kusculu

Follow this and additional works at: https://scholarcommons.scu.edu/elec_senior



Part of the [Electrical and Computer Engineering Commons](#)

Recommended Citation

Moran, Alexander and Kusculu, Emir, "Sensor System for Rescue Robots" (2019). *Electrical Engineering Senior Theses*. 47.
https://scholarcommons.scu.edu/elec_senior/47

This Thesis is brought to you for free and open access by the Engineering Senior Theses at Scholar Commons. It has been accepted for inclusion in Electrical Engineering Senior Theses by an authorized administrator of Scholar Commons. For more information, please contact rscroggin@scu.edu.

SANTA CLARA UNIVERSITY

Department of Electrical Engineering

I HEREBY RECOMMEND THAT THE THESIS
PREPARED UNDER MY SUPERVISION BY


Alexander Moran, Emir Kusculu

ENTITLED

Sensor System for Rescue Robots

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF

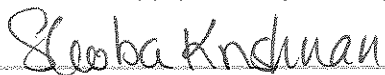
**BACHELOR OF SCIENCE IN
ELECTRICAL ENGINEERING**



Thesis Advisor(s) (Dr. Sally Wood)

13 June 2019

date



Department Chair(s) (Dr. Shoba Krishnan)

6/12/19

date

Sensor System for Rescue Robots

Alexander Moran, Emir Kusculu

Department of Electrical Engineering
Santa Clara University
2019

ABSTRACT

A majority of rescue worker fatalities are a result of on-scene responses. Existing technologies help assist the first responders in scenarios of no light, and there even exist robots that can navigate radioactive areas. However, none are able to be both quickly deployable and enter hard to reach or unsafe areas in an emergency event such as an earthquake or storm that damages a structure. In this project we created a sensor platform system to augment existing robotic solutions so that rescue workers can search for people in danger while avoiding preventable injury or death and saving time and resources. Our results showed that we were able to map out a 2D map of the room with updates for robot motion on a display while also showing a live thermal image in front of the system. The system is also capable of taking a digital picture from a triggering event and then displaying it on the computer screen. We discovered that data transfer plays a huge role in making different programs like Arduino and Processing interact with each other. Consequently, this needs to be accounted for when improving our project. In particular our project is wired right now but should deliver data wirelessly to be of any practical use. Furthermore, we dipped our feet into SLAM technologies and if our project were to become autonomous, more research into the algorithms would make this autonomy feasible.

Keywords: Thermal, LiDAR, Digital Camera, Sensor System, Rescue, Robot

Table of contents

ABSTRACT	1
List of Figures	4
List of Tables	5
Introduction	6
Problem Statement	6
Benefit	6
Existing Solutions	7
Objective	7
Background	8
Scope	10
Requirements and Specifications	11
System	12
General overview	13
Functional Block Diagram	13
About the Block Diagram	14
Data Acquisition	14
Data Analysis	15
Technologies Used	16
Thermal Sensor	17
Digital Camera	17
LiDAR	19
Overview	19
Possibilities	19
Testing	19
Thermal Camera Testing	22
Digital Camera Testing	23
LiDAR Testing	24
Results	26
Mapping	26
Thermal Imaging	26
Picture Taking	27
Delivery	28
Stakeholder Needs	28

Time	29
Risk Analysis	30
Ethical Analysis	31
Introduction	31
Ethical Purpose	31
Safety	31
Methods	32
Markkula Center for Applied Ethics	32
The Virtue Approach	32
The Common Good Approach	33
Sustainability	33
Pillar of Social Equity	33
8 Frugal Design Strategies	34
Environmental Impact	36
Solutions	37
Local Recycling	37
Other Technologies	38
Future Work	38
SLAM	38
Wireless	39
Lessons Learned	39
References	40
Appendices	43
A Sensor System	43
B Schematics	45
C LiDAR Code	46
D Imaging Code (Arduino)	49
E Imaging Code (Processing)	55
F Thermal Camera Code	59

List of Figures

Figure 1.....	6
Figure 2.....	13
Figure 3.....	15
Figure 4.....	16
Figure 5.....	20
Figure 6.....	21
Figure 7.....	21
Figure 8.....	22
Figure 9.....	24
Figure 10.....	25
Figure 11.....	25
Figure 12.....	27
Figure 13.....	27
Figure 14.....	29

List of Tables

Table 1.....	12
Table 2.....	13
Table 3.....	18

1. Introduction

1.1. Problem Statement

Rescue workers are constantly putting their safety and lives on the line, in particular, 42.6% of Firefighter deaths happen on-scene [1]. This is due in part to the need to enter unstable buildings that are on the brink of collapsing, buildings filled with toxic gases, or environments with temperatures the human body must avoid. The problem here is that rescue workers must enter these buildings in order to find people trapped in the building, and oftentimes the rescuers have no idea where or how many people are in the building. The same holds true for earthquake events. Buildings have collapsed and are structurally unstable; however, the rescue workers must still enter the environment to find a trapped person. A versatile sensor platform used with appropriate robotic transport could increase safety and also improve effectiveness for rescue workers.

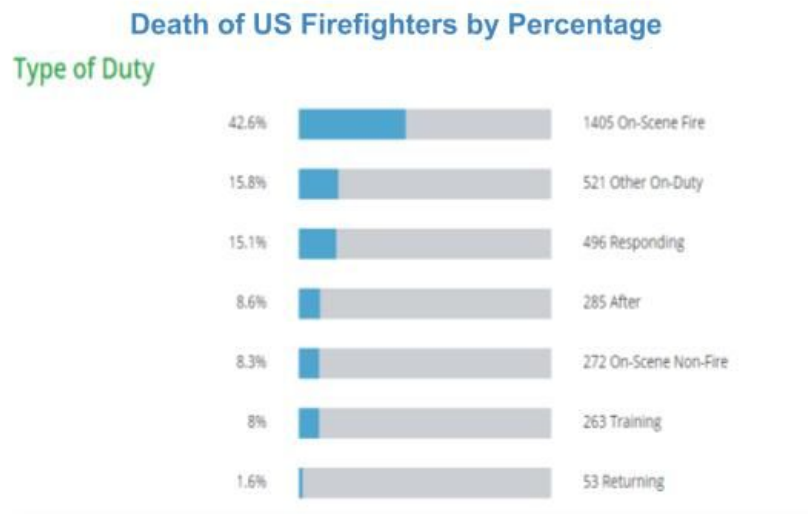


Figure 1: Death of US Firefighters by Percentage since 1990

1.2. Benefit

The goal for this sensor platform is to decrease the injury and death count of rescue workers on duty while also improving the effectiveness of their missions. For the cases where a

building's structural stability is compromised and it does in fact collapse, a rescue worker will not have to perish; instead the replaceable robot will be destroyed. Also, with an effective use of the first responders' time, they will be capable of rescuing even more lives of those stuck in a building.

1.3. Existing Solutions

There are many different technologies that were created to help first responders assist the community and even robots that are used in other situations that help survey an environment and relay the information back to the user. Simple but one helpful device known as LuminAid [2] was featured on an episode of the TV series *Shark Tank*. The creators inspiration was a result of the 2010 Haiti earthquake where people were left without reliable light sources for several weeks. LuminAID has a 22 hour LED that rescue workers can use in unseeable areas after the result of a disaster. It's solar rechargeable and is able to float - useful for flood scenarios. What if workers can't get into a spot or it is unsafe to do so? We address this problem in our objective.

Another device is the robot named "Scorpion" [3] which was deployed at the Fukushima Nuclear Plant. Since radioactivity is unsafe for people, robots are sent into the nuclear cores of the site to measure the radioactivity. Scorpion can be sent out and remotely controlled to enter areas unsafe for rescue workers. Unfortunately, Scorpion died 2 hours into a 10 hour mission, and this mission cost a total of \$100 million dollars. This is a huge cost to invest into a robot which means there is definitely a market for a sensor system that costs \$400-\$700 dollars that can be universally implemented. However, it would be meant for more common, less extreme 1 sigma disasters.

1.4. Objective

There is a need for a sensor system to augment existing robotic solutions to disaster situations. Our versatile system can be used on a variety of robots such as an RC car, drone, or snake. This sensor and imaging solution will ultimately have a wide range of applications and be portable to different mobile robotic platforms. However, the focus is on helping first responders

in search & rescue scenarios. Due to the small nature of these robots it will be able to get into the tight spaces of a collapsed building or even the small openings in case of a mine collapse. It will also be able to be used in different situations such as casual play and training games. It could have multiple uses in indoor robotics where autonomous robots need some kind of map of its environment. This mapping, is what we are trying to accomplish.

We are building a scanning and imaging solution that is able to map and describe its indoor environment and structure from the perspective of a mobile platform. It will also be able to determine if there is any life forms in these structures using heat sensing. We will use a rotating LiDAR sensor or an ultrasonic sensor to scan an environment structurally. These sensors will give us data on the size and shape of the larger room. Then we will use a thermal camera to search for signatures of life. We are not totally sure a thermal camera will be the best sensor but if we could reliably detect the unique heat signature of warm blooded life forms it would work. Then when one is detected a digital picture will be taken. Having recorded all this information we will then use a microcontroller to process and to relay it to a remote display or mobile platform. Then we could use this information in coordination with a robot that is small enough to fit into tight spaces to explore and determine if there is any signs of life. For testing we will combine this with a small push platform that we can use to explore environments. We plan to make our methodology versatile so it could be used with different robot types, like a snake robot. Our main purpose is to build the detecting component that could in theory be combined with any kind of robot to accomplish more specified tasks.

1.5. Background

After looking at the existing solutions, we can see that there is a need for a Sensor System designed for rescue robots. LuminAid [2] itself is only capable of providing light for a first responder, however it could not help if a first responder could not enter a space or if the space was too unsafe to enter. Time and resources would be wasted if a worker had to move unnecessary debris out of the way. Furthermore, if a place was unsafe to enter and the room were to collapse after a worker entered, it would create another issue in an already problematic rescue scenario. A robot like Scorpion [3] addresses an issue specifically designed to navigate small

spaces and unsafe areas, but a huge development cost of \$100 Million uses a huge amount of resources, thus the need for a universal Sensor Platform System around the cost of \$400-\$700 dollars would cut major costs. The only thing users would need is to design the robot meant to host of the system.

With this platform there are many different enabling technologies. Two big mapping technologies are Sonar and LiDAR. Sonar [4] makes use of an ultrasonic pulse which is sent out from the sensor and then the reflected pulse's time of return is measured to give data for mapping capabilities. Sonar is most notably used in autonomous driving cars such as Tesla and Waymo vehicles [5]. LiDAR [6] is similar to sonar in that it sends a signal and then measures the return of that signal, however, instead of the signal being sound, it uses light. Think of if you shot a laser beam at a mirror and measuring the time the reflection returns. LiDAR is normally implemented for SLAM (Simultaneous Location And Mapping) projects [7]. Such as a Roomba robot. SLAM will be discussed later in a later section.

Other enabling technologies include infrared [8] sensors. These are used in many different locations such as motion detectors for lights, thermal cameras, and obstacle avoidance. What infrared does is that it detects the infrared rays being emitted from different object in a sensor's line of sight. Two devices that will be discussed later that implement the use of infrared are the Kinect and the PIR (Passive Infrared) Sensor.

Given these technologies we will be capable of creating a sensor system that uses LiDAR or Sonar to create a map of the room our system is in. Existing rotating LiDARs are capable of creating a 2D representation of a room. In order to find mammalian life in a room, infrared sensors solutions can be used to find that warm human body temperature inside of a room.

In future iterations, LiDAR can be enhanced. LiDAR currently has existing 3D capabilities which can be found in the use of construction surveying. In fact, a SCU Senior project from 2018 used LiDAR to create a 3D image of a room. With 3D imaging, our sensor system will provide users with useful information that will allow them to make faster and more accurate decisions. One great benefit of LiDAR is its use in SLAM. By using proper SLAM algorithms, we can create a sensor system that will help turn any robot it is put on into an autonomous device that can search for people and navigate on its own [7].

There are many different robots being developed for different scenarios. Two normal robots are RC cars and a drone. These would be best to start off with when implementing our system onto an actual robot. However, because our system is meant to be universal, we want our system to work with unique robots like a robotic snake [9]. There are many more robots created for different scenarios because every disaster scenario is different. A fire scenario would require a very rigid and robust robot that can keep the system safe from fire and falling debris. A flood scenario would render many robots obsolete so a boat robot and drone would need to be used. Whatever the scenario our system needs to work with the robot.

Our classes such as COEN 11 and 12 plus Mechatronics have prepared us to deal with Arduino coding and use of a microcontroller. However, we were not taught how to use different codes such as Python or Java, which were needed in order to make use of our LiDAR data and to use the software “Processing”. We needed to develop those missing coding languages. Furthermore, the biggest learning curve resided in letting the different programs communicate with each other.

In conclusion, to communicate with the users, we would make use of a display that is friendly to the eyes. Like the book “The Design of Everyday Things” tells us, there is nothing worse than putting in extra effort to understand what someone is showing you [10]. While our system runs, the data being displayed will be readable enough to understand what it is that is being shown without much thought. That way our sensor system experience will be seamless.

1.6. Scope

Our project will provide useful information with limited technology to our first responders. Our system will:

1. Determine physical dimensions of a space
2. Detect life signatures
3. Snap pictures when needed

Because there are so many different scenarios that can interfere with our sensor system, we will do a proof of concept assuming that we are in an earthquake scenario. This creates a more ideal environment for us as we do not need to worry about fire and smoke interfering with both our

thermal and LiDAR sensor. In a future iteration the sensor system will be wireless with a one hour battery life. As of now the system is wired to a computer.

1.7. Requirements and Specifications

This sensor system needs to augment existing robotic solutions to disaster situations. It must be versatile so it can be used on robots like an RC car, drone, or even a snake robot. The whole purpose is so that our sensor system can be sent into locations where the rescue worker can not reach or it is unsafe for them to enter. This makes efficient time of the responders' time and resources while also avoiding preventable injury or death since they are only dispatched when a person is found and they are not running around the building looking in spots that are unnecessary. We designed this project to be built for 1 sigma events. This means we want something low bandwidth but gives useful information, that is why resolutions are okay to be low. This will make our project more cost efficient, reusable, and versatile since higher grade sensors could be more bulky, specialized, and expensive.

We require our sensor system to work in about the size of an average classroom here in the school of engineering and the system needs to detect what is immediately in front of it to avoid collision. Thus a distance sensor with min/max range of 6 in to 8 m is needed. We require our map of the room to be at least 2D so an angular resolution of 1° will do it. Because someone will be driving the robot, our sensor system needs information fast. A sensor that provides 4000 samples/sec will do just the trick.

To find the heat signature of a person we will use a thermal sensor. This will serve as a detector rather than an imager, so we do not need an image that is high quality. Most cost efficient thermal sensors come in resolutions of 64 pixels, thus we chose this as our specification. This sensor needs a good view of everything in front of the robot, so the distance must cover a majority of the room and a degree of view that can see well above the platform. It needs to detect a human at least 6 m away and have a degree of view of at least 60 degrees.

To take pictures, the camera must provide visual comprehensible images that can see smaller details. A VGA size image which is 640x480 pixels will do it.

Furthermore, there will be a remote display to show us the readable data for the user and a mounting platform for the sensor system is required.

To elaborate more, the current model of our project is on a platform with caster wheels that can be moved manually and the micro controllers are attached to our computer via wires. In the next iteration, we would have it set on a remote controlled robot where the smart nature is in the user. Where they navigate and get pictures based off what the thermal picks up.

Aspect	Specification
Distance Sensor	<ul style="list-style-type: none">• Min/Max Range: 6 in - 8 m• Angular Resolution: 1°• Samples/sec: 4000
Lifeform Sensor	<ul style="list-style-type: none">• 64 Pixel Resolution• 60° view
Camera	<ul style="list-style-type: none">• VGA(640x480 pixel) resolution
Miscellaneous	<ul style="list-style-type: none">• Microcontroller with multiple serial pins• Remote Display• Mounting Platform

Table 1: Table of Project Specifications

2. System

In order to create a functional system, there needs to be three main parts to it. One, the Sensor Data Acquisition, gathers the data from our sensors and turns it into usable data. Two, Data Analysis, takes the gathered data and turns it into visually, human readable data. Third, a Display, is capable of actually visually representing that data. This process is shown in Figure 2.

Block	Overview
Sensor Data Acquisition	<ul style="list-style-type: none"> ● Gather data from sensors ● Send as useful data to computer
Data Analysis	<ul style="list-style-type: none"> ● Process data through respective programs ● Turn data into visually useful data
Display	<ul style="list-style-type: none"> ● Display processed data into “readable” data for first-responder

Table 2: System Parts Defined

2.1. General overview

Our Sensor System for Rescue Robots will be designed with the following functionality:

1. The robot will be able to scan the room it is in with LiDAR
2. While scanning the robot will be looking for thermal signatures
3. When a heat signature is detected, the robot will take a picture and send it to the driver

This is the scope of our project. We will create a sound base to leave the opportunity for our concept to be made more complex in the future. We would like our design to be integrated into robots that can navigate through rubble and detect humans in a smoky or fire rich environments.

2.1.1. Functional Block Diagram

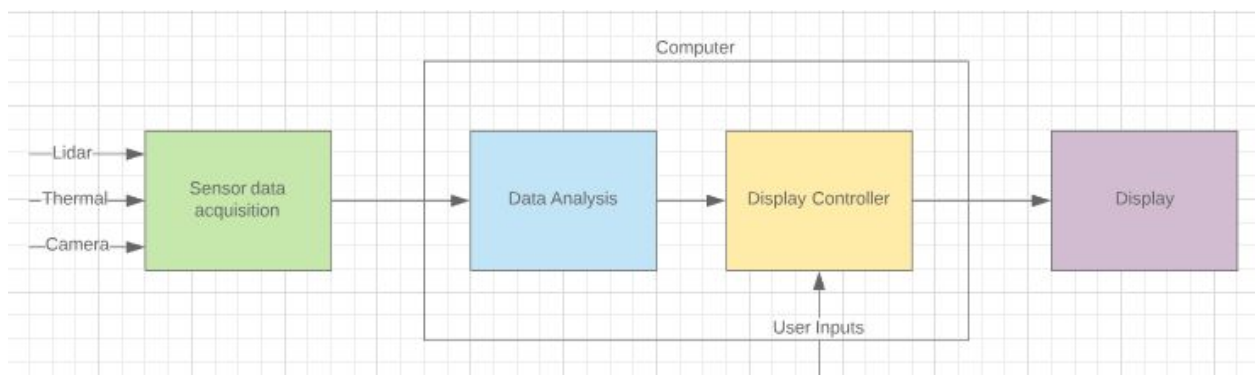


Figure 2: Functional Block Diagram of Sensor System

Looking at Figure 2, we can see the technologies used - LiDAR and thermal sensor and digital camera. Their data is collected on a microcontroller and interpolated in the Data Acquisition. The data is then sent via serial to the computer where software turns the data into useful information. On the computer we want a future iteration where users are given a Display Controller. They can decide the information and style of information given to them via User Input, likely a keyboard. Then finally, the useful data is displayed into actual visual information that is readable by the user of the system.

2.2. About the Block Diagram

2.2.1. Data Acquisition

Data Acquisition is the portion where readable data is sent from our microcontroller to the computer via serial port. We decided to go with the *Elegoo MEGA 2560* [15] (Product #: 191). for our microcontroller since it has 4 Serial pins, bigger RAM than the arduino for image decoding, and I2C [11] for the thermal, the image decoding in this part is what turns our JPEG into MCUs (Multicoded Units) so that the microcontroller can send in 8-bit raw data that can be analysed by our computer software. This block also contains an SD card which stores all the JPEG images taken by the camera. Our LiDAR and Thermal are constant and in parallel while our digital camera remains as a triggered event - which is when a heat signature is detected. In the position, the LiDAR is scanning every 2-3 seconds so we are getting new information of the 2D representation during every cycle. Our low-resolution camera picks up heat signatures and if something is detected then the digital camera is pointed to take a picture. When this picture is taken, the microcontroller breaks the image into a bitmap so the 8-bit data can be sent through the serial port into the computer

2.2.2. Data Analysis

Data Analysis is the portion where all of our code and software is. We are using softwares like Processing, Arduino IDE, and Python. Processing is what takes the image's MCUs and displays an image on the screen for the user. Our Python code is turning our LiDAR and Thermal Data into useful information so that it can be displayed on our screen. The LiDAR portion takes that past 5 sets of measurements and creates a map of each set. Different colors are used to represent different sets of data gathered. This allows the driver to know how the room view is changing with robot motion and where they are. The Arduino IDE is what helps these softwares communicate with the microcontrollers of the sensors so that we get the right data and are triggering the right events. Currently, we are actually streaming our thermal image, so the Python code is setup to display that for the user and compare it to their image obtained. The digital image itself takes about 30 seconds from trigger to fully displayed image with a VGA (640x480 Pixels) sized image.

The environmental map is updated constantly and as of right now so is the thermal image. Once this analysis is done we want the final product to have a display that resembles Figure 3.

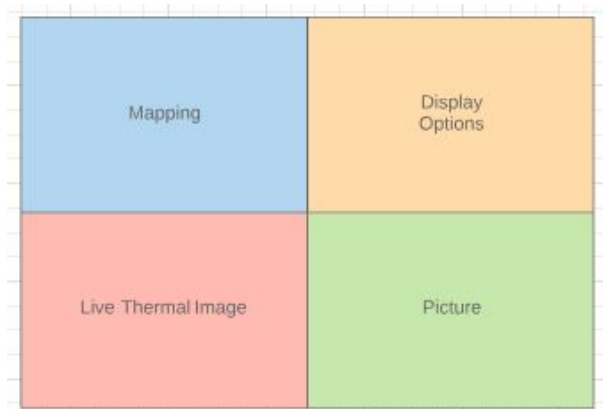


Figure 3: Display Style

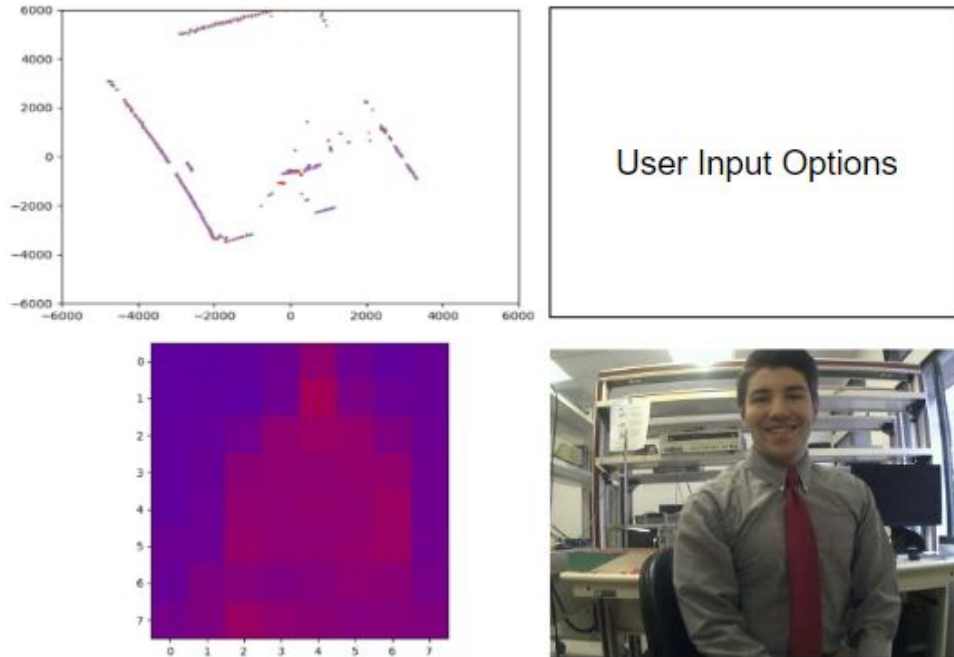


Figure 4: Example Display

2.3. Technologies Used

In order to develop the sensor system we specified, we used three main technologies. A LiDAR was used to gather the data points of the room so we could create a 2D map of the room. In order to detect a human's thermal signature, we decided to use a thermal sensor. Finally, to get a picture after the detection of a thermal signature we will be using a digital camera.

Furthermore, we used several communication standards that will be referenced throughout the text. The connection to the microcontrollers, computer, LiDAR used USB [12]. The thermal camera used I2C [11] connection. The digital camera and microcontroller both used Serial [12] in sending out data. Our camera also saved images in JPEG [13] format. Finally, we had extra pins on the digital camera for NTSC [14].

2.4. Thermal Sensor

When first looking for a thermal sensor we looked at multiple solutions. One such solution was a Kinect and the other was a PIR sensor. These both are solutions used in devices that help detect the presence and movement of mammalian life.

The Kinect uses “depth mapping” to identify objects. How this works is that infrared sensors taking information from in front of the kinect create a 3D depth map inside it. Based on this depth map, the kinect looks for joints in order to identify a person in the image and reacts accordingly. The Kinect is also readily available for about \$15-\$20 with many guides online. However, in an emergency scenario people can be lying on the ground unconscious. This means no joints will show in the depth map and we would skip past someone still alive in a building

A PIR Passive Infrared Sensor is cheap, affordable, and easy to implement. However, it is used in motion detection applications. Again, if someone is unconscious on the ground, we would not be able to detect them and would thus skip past someone still alive and conscious.

The sensor we decided to go with is the *AMG8833 IR Thermal Camera* [15] (Product #: 3538). It has a -20°C-80°C range which is able to detect a human body temperature of 97°F (36.1°C) - 99°F(37.2°C). It reaches all of our specs as well. It is 8x8 pixels which is at least 64 pixel resolution and has a 60°x60° angle of view which covers a great view of what’s in front of the sensor system. The description also says it can detect a human heat signature up to 7 m away which is above what we originally desired. This thermal sensor has an I2C Interface which means our microcontroller MUST have this capability.

2.5. Digital Camera

There are many digital cameras out there on the market. Adafruit has a wide selection of small and affordable decisions such as *Mini Spy Camera with Trigger*, *Spy Camera for Raspberry Pi*, and *Adjustable Pi Camera*. However, these all were created for use with a Raspberry Pi and we wanted something more versatile in case we need to make use of different softwares and micro controllers.




Mini Spy Camera with Trigger	
Spy Camera for Raspberry Pi	
Adjustable Pi Camera	
Problem:	<ul style="list-style-type: none"> • All are designed for Raspberry Pi, not universal

Table 3: List of Various Digital Cameras

The digital camera chose was the *TTL Serial JPEG Camera* [15] (Product #: 397). This camera has three different resolutions of photos, the highest being VGA (640x480 Pixels) which reaches our requirements. Every step down gives us half the resolution of the high one. The great thing about this camera is that it has a 60° angle of view which matches up with the thermal sensor. So when something is detected, we will see the object that was detected for sure. Furthermore, this camera gives us opportunity for future design decisions as it has motion detection capabilities and is constantly streaming video. The way an image is taken is by taking a snapshot of the stream. With NTSC pins, we can hook up a monitor to the sensor system if necessary. Because this camera sends serial data, we need a microcontroller with multiple serial capabilities to integrate all the sensors.

2.6. LiDAR

2.6.1. Overview

The LiDAR can measure point distances using light beams and the time it takes to reflect back to the sensor. There are multiple types of LiDARs. Basic ones are only able to measure one dimensionally(1D). More sophisticated LiDARs can measure 2D by rotating the sensor. Even more sophisticated sensors can measure 3D. We believed it was essential to have a LiDAR sensor because it would allow us to get the overview of the environments layout. The closed environments in our use scenarios could be mapped using an accurate LiDAR sensor. These maps could later help the user navigate through the environment more efficiently.

2.6.2. Possibilities

As complexity of the sensor increases so does the cost of the sensor. Out of the options we chose to use a 2D LiDAR. After we determined the dimension of our sensor we had a few more choices to make regarding the range, data speed, angle resolution, and software interface.

3. Testing

In order to do our testing, we decided to focus on testing each unit first. This ensures that future problems we run into is a result of the integration and not the units themselves. Then once each unit is working, we then did testing on the integration aspects of the project.

- Gather Thermal image data and display live image
 - By looking at a live image, we can tell if a hot object enters in view of the thermal camera
- Write and read a file onto an SD card
 - Ensures there is no issue with the memory card itself
- Take an image and store onto SD card

- We can read directly from SD card when we plug it into the computer, this ensures proper image writing and any further problems is in the processing
- Use spacebar as a trigger to take picture and process image onto computer
 - Ensures are image and processing can be accomplished through a triggered event
- Mapped the inside of a small square box
 - Ensure that the data being displayed does indeed show a box

By the end of the testing we built our platform in the Maker Lab. We did this by laser cutting acrylic into a box and a platform. The platform was drilled and then caster wheels were attached. The box was glued together minus one side and this is where we placed our Microcontrollers. The Lidar was placed on top so that there would be nothing in the way of its scan and the thermal and digital camera were attached to the front side of the box. In Figure 5 we can see the overall design while in Figure 6 we see how the front of the system was set up and in Figure 7 we can see the inner wiring of the box containing the microcontrollers.

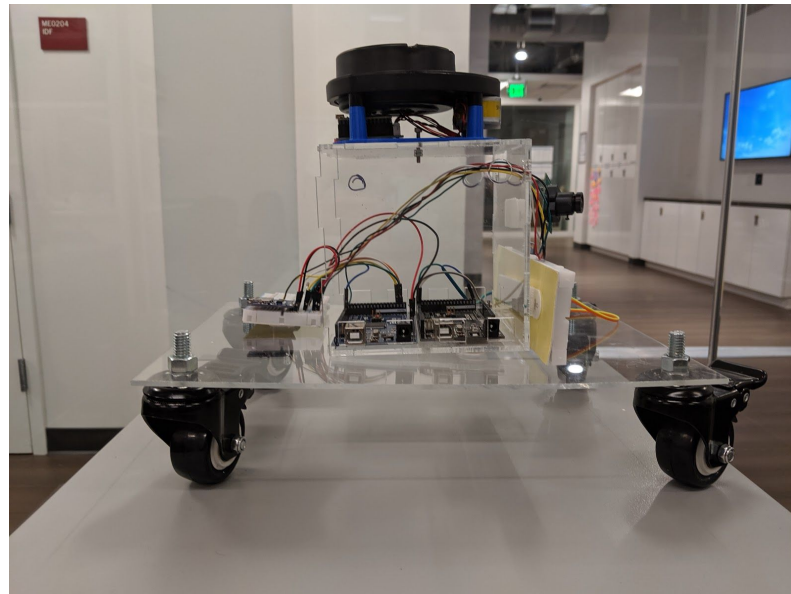


Figure 5: Side of Sensor System

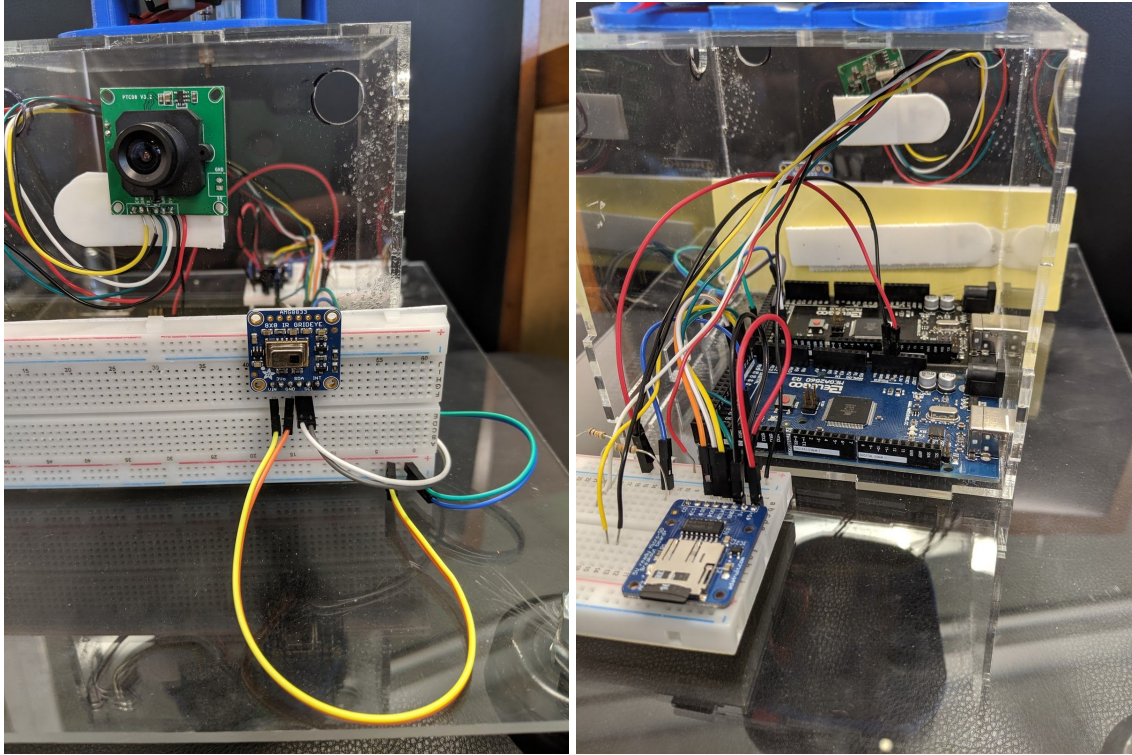


Figure 6 and 7: Close up of the Circuitry and Front of Sensor System

3.1. Thermal Camera Testing

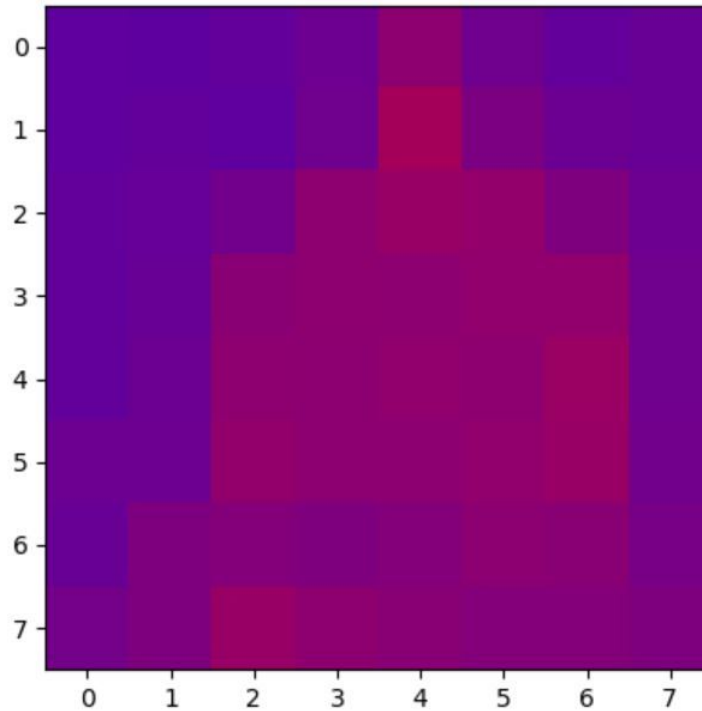


Figure 8: Thermal Image of Hand Entering View

Our thermal Camera uses I2C communication and to communicate with it we needed to utilize an Arduino and the AdaFruit Library in order to Serial stream the data. The data is in the form of an 8x8 array of floats. The array is in the pixel resolution and the data resolution itself is 8 bits. We had to write Python code to image that data as well. This was primarily to see if the readings passed our threshold temperature. The thermal Camera was intended to be used as a detector and not an imager. But in order to easily determine if this happened we decided to image it. So our Python code is able to take the serial Streaming data and image it as an 8x8 pixel with a color gradient. To determine if it worked we experimented but trying to detect the heat signature of someone standing in front of it in different distances. We were able to detect someone at 3m.

3.2. Digital Camera Testing

The preliminary test was to make sure the camera could take photos. Originally we just wanted the image to be sent directly from the camera to the computer display. This proved to be quite difficult as we tried to use Python to do the process but we would only get a fractal image rather than human readable data. Instead we decided to go with a store first, display after method. The advantage with this is that with future iterations we can give the user ability to shuffle through all the photos for future reference which could be quite useful for documentation and double checking purposes.

Because the standard microcontroller memory has no room to store the image data, we implemented an *Arduino SD Card Breakout Board*. This threw in the SD Card coding library . To ensure the SD card worked we ran sample code to write and read from the SD card. Then we worked to write a JPEG picture onto the card, take the card out, and then connected to our computer to display the image. After a few tweaks we saw our first JPEG file on the SD card. We moved to the processing portion. We managed to have access to some code which displayed images one by one on the SD card. This was not ideal, so we made the code display the image that was taken during a triggered event. Using the spacebar as a trigger, we made it so that Processing would work with the Arduino IDE. After many iterations, we finally managed to trigger the camera to take a picture and Processing to take that exact picture to display it on the users screen.

3.3. LiDAR Testing

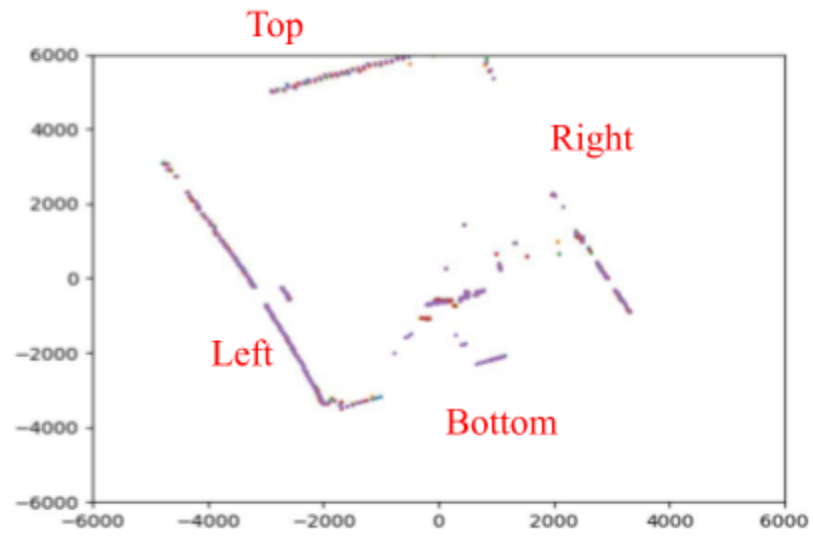


Figure 9: Conference Room Scan With 5 Previous Measurements



Figure 10 and 11: Images of Room Scanned

To test our LiDAR we initially need to test if the communication to the LiDAR worked. Our LiDAR uses Serial port Communication with a Get Surreal board as an interface. The Get Surreal board is able to Serial Stream data in the Form of “A, Angle , Distance , Signal Strength”. The “A” is the start of the line, Angle is in degrees of 0-360, Distance is from 0-6000, and Signal Strength is also 0-2000. For our test we needed to write Python code that could turn the

radial data into a 2D map, and then plot the subsequent set of points on a graph. From this graph we experimented to see if we could observe recognizable features of the room we were in. This is mainly how we made sure that our system worked.

4. Results

4.1. Mapping

After setting up the code and Sensor we were able to map an average size room consistently. There were some issues with our demonstration. It was very difficult to process the data properly if the LiDAR was moved. To remedy this we decided to show multiple data scans at the same time. This allowed us to properly analyze subsequent scans. Overall it worked as we designed it with our limited resources. And there is always more functionality we can add to get more out of our system.

4.2. Thermal Imaging

For thermal imaging we decided to experiment with different temperature gradients. We were able to get a low detection of -8°C and a highest temperature of 80°C . We were able to get a continuous stream of data and were able to detect a human's heat signature from approximately 2m-3m. With this reading we were able to prompt a message to let the user trigger the digital camera to confirm if the origin of the heat signature.

4.3. Picture Taking



Figure 12: First Official Image Taken Directly from SD card



Figure 13: Successful Trigger Image Straight to Processing

After examination of time, we learned a full VGA image of 640x480 pixels took over 30 seconds to process. This was not ideal as we want an image to be almost immediately visible at the point of being taken. We decided to drop the quality to 320x240 pixels and our time dropped to about 10 seconds of

processing. Considering Figure 13 is 320x240, this loss in quality is acceptable for the better speed output of the image.

5. Delivery

5.1. Stakeholder Needs

Functional:

The essence of our project is that it will be a self-navigable robot with few sensors attached to it. It needs to be small enough to fit through tight space and easily carried so that important rescue supplies will not be sacrificed when packing emergency vehicles or bags to the brim. Our project will feature an RC car as our robot and two sensors. One sensor is needed for reading and collecting data on the environment and the second one will be used to detect the presence of a mammalian life form. We will use LiDAR and infrared respectively. Both sensors will be interfaced with a Raspberry Pi that has the capability of sending out information in regards to the presence of a person.

Financial:

We want our device to be widely used not only in first world countries but third worlds as well. Especially in events of a catastrophe the less money spent on rescue tools means more money for foods and essentials. We will attempt to find the most cost efficient parts available, but our starting goal is to keep it under \$750. We would prefer people to not worry about the cost and focus more on using it to keep the rescue workers safe.

Technical:

In order for this system to function properly, we will need the sensors to be compatible. We will look for sensor that have well documented specifications and interfaces like I2C and avoid any sensors that require additional hardware. We want the whole system to be integrated on one microcontroller. The battery should be capable of running the robot for at least an hour. Smaller emergency situations usually don't last too long. Finally, the way to communicate information should be simple by sending out a signal and location of a robot if a person is detected.

Societal:

The point of this project is to save the lives of rescue workers and avoid any unnecessary deaths for them. They are the real heroes of society and with their lives intact they can continue to save more people in the long run. Our project will help maintain the work force that save us in our most dire moments.

5.2. Time

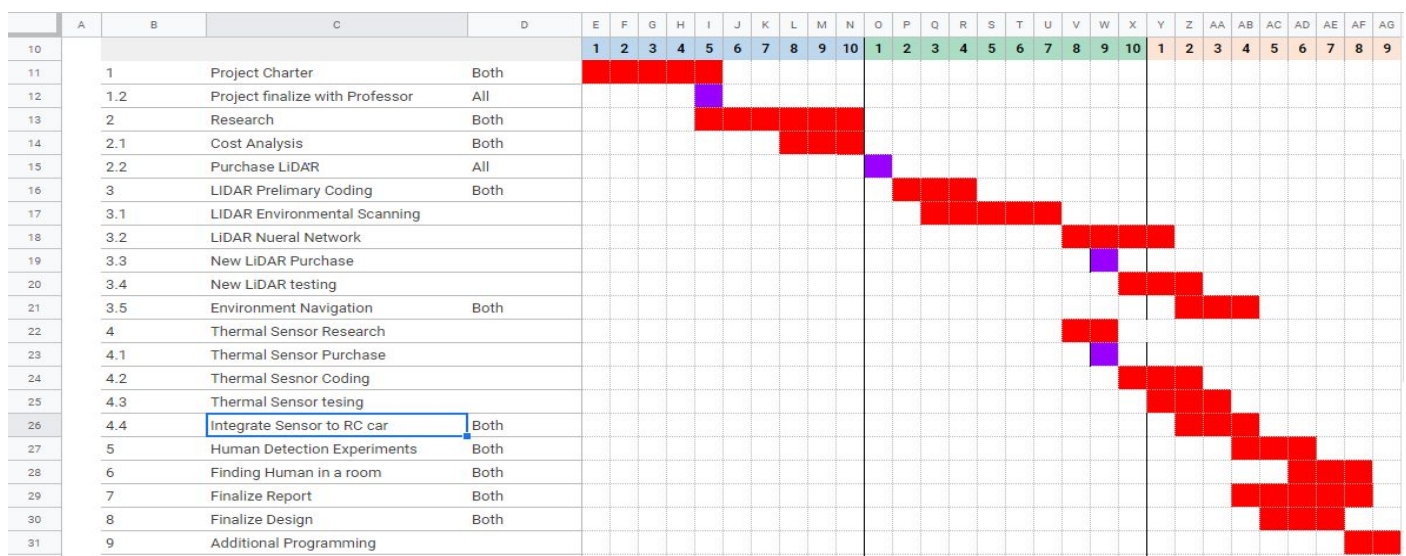


Figure 14: Final Gantt Chart

We ran into a few mishaps with our project, so the timeline did fall behind in some locations. In particular, we expected the imaging to be a quick image that was immediately uploaded to the computer but after some research we discovered we needed to deliver the image from the microcontroller as MCUs. This meant an inclusion to an SD card which needed test time itself.

Along with the LiDAR, it took some trouble shooting to get it working and a bunch of new code to analyze and read the data. Furthermore, we dipped our feet into researching info on SLAM which pushed back the time and integrating the sensors so that they work as one cohesive unit.

5.3. Risk Analysis

One major risk analysis to note is bandwidth. In the event of a catastrophe, there can be a loss of communication towers, so using exterior wireless communications like a cellular tower could be a huge problem. We want data to be sent rapidly and 100 percent of the time. This why local wireless will be the most optimal communication method to the sensor system, so navigating the system remotely far away from the disaster is not recommended. There are many local wireless communication methods as well. We recommend a local cellular connection.

Another note is that our our current iteration loads an image onto a computer pixel MCU by MCU (a bundle of pixels). This means a user would have to wait a full 15-30 seconds before the picture fully loads, costing valuable time, this delay should be noted and accounted for in time crucial situations.

Finally, this sensor system is only as rugged as the robot it is put on. If a client were to use this in a fire without necessary heat protection, then the circuitry could malfunction. It is important to design a robot for its respective environment and then attach our sensor system.

6. Ethical Analysis

6.1. Introduction

As engineers we need shape the world around us like any other profession. For this reason we need to know and appreciate ethics. Due to this we will discuss the ethical consideration we took in planning our Senior Project. These are the different ethical examinations we made of our project. Both the main purpose and the methods we choose to pursue is coherent with multiple ethical theories. Our ethical purpose stems with the ethical theory of utilitarianism, providing the greatest good for the most amount of people. You can see this in the fact that our project is trying to provide relief to rescue workers who may enter danger. While our methods are meant to work integrated with the deontological ethical theory, rule based ethics, the rules we choose to follow are the IEEE code of ethics and The Universal Declaration of Human Rights. We choose to do so to ensure the highest standard of safety.

6.2. Ethical Purpose

The main purpose of our project is to build a device that can image and locate people inside earthquake debris. This project is going to help those who are in most desperate need. If we look at the statistics of Preventable Disaster Deaths or PDDs for short we see that a lot of these are due to the fact that we are unable to locate those that are in need. The next would be organization of the first responders. Our project plans on helping improve both of these. The first by helping locate people inside tight and unreachable places and bringing them supplies. The second by helping increase information of the location.

6.3. Safety

Due to the fact that our project will be on used in situations were speed and accuracy is important, our project needs to be reliable. Not only this, but due to the fact that our project

might also be used in conditions where infrastructure might be damaged, we will need our project to be self sufficient. The next important thing is that our project must not harm the environment and the people within it.

6.4. Methods

We want our project to be in the highest of ethical standards. This is why we have done a deep ethical examination of our projects and plan on strict methods to ensure that our project is successful in accomplishing these standards. We plan on using extensive simulations to make sure our project is working on par for the difficult situations it might be used in. We then plan on consulting both industry specialist and academics to make sure that our project is operative.

6.5. Markkula Center for Applied Ethics

The Markkula Center for Applied Ethics has a framework for ethical decision making. Within this framework are 5 standards that help to make these difficult ethical decisions. The five are as follows: The Utilitarian Approach, The Rights Approach, The Fairness and Justice Approach, The Virtue Approach, and The Common Good Approach. We decided to go with the last two listed for our approaches.

6.5.1. The Virtue Approach

In this approach, we are meant to make decisions based upon actions that are consistent with the idea that this will provide for the full development of humanity[16]. This is where things like compassion and generosity kick in to ensure we care making the best decisions possible for the people around us. In our project we use this approach in the following way. Firefighters are out there protecting their community and risking their lives, yet they perish for doing something so just. To show our compassion and care we want to minimize the suffering they face when they do so much to protect us. Thus this project is meant to minimize their time in danger and stop preventable injury or death.

6.5.2. The Common Good Approach

In this approach, the belief is that life in community is good in itself and the actions we decide to take should reflect that [16]. This pushes for the idea that we should have compassion for everyone in the community, even the vulnerable. Because Firefighters fight to help us, if we are able to help and protect them, they can in turn continue to protect us and the entire community. By helping to preserve the life of one person, their actions contribute to preserving the lives of the community around us.

7. Sustainability

7.1. Pillar of Social Equity

There are three Pillars of Sustainability. Ecological Protection, Economic Development, and Social Equity. The focus of our Sensor System for Rescue Robot is the Social Equity aspect. The reason being is that we are after the safety and protection for the lives of the people that serve to protect our community.

Rescue workers are put in danger every day. Fire fighters are going head first to the scene of a fire, police are going head first into the scene of a crime, and medical responders are going towards the scenes of both. The unfortunate part about this is that on-scene fatalities account for many deaths for rescue workers. In particular, 42.6% of firefighter fatalities occur on-scene at a fire. The social injustice in this scenario is that firefighters are protecting their community but they perish. A valuable member of the community being lost and unable to help another person caught in danger.

Although every member of society is making a mark on the community with their work, they are not constantly risking their lives on the job. The wealthiest class of people have the safest jobs. They work in buildings and have a nice desk to sit at with a comfy chair to make themselves comfortable. This itself is stark contrast to the lower class. People that work in machinery, construction, and even farms task greater health risks having to constantly go to work. Not only can they get seriously injured by the tools they use but as time progresses their

bodies begin to degrade. Even with this stark contrast, there are many precautions businesses and their workers can take to maintain their health. Rescue workers do not have the luxury of the high class or the low class. When there is danger, they go straight to it. Then if the unfortunate occurrence of them losing their life does occur, then it puts society as a whole at risk since there is one less person out there who can help in our time of need.

Our objective with this project is to prevent any unnecessary deaths for rescue workers losing their life in the line of duty. Our robot was shaped with the sole purpose of avoiding to needlessly risk a rescue workers live in the event of an emergency or catastrophe. Rather than send a Firefighter into a building after following an earthquake or scene of a massive fire, we want our robot to go in first to precisely decide a course of action for these honorable members of society. The robot is sent into a structurally unsound building and seeks out any human life forms. Pictures are taken and sent to someone outside the building to determine if someone is trapped inside. With this information, rescue workers can be sent to an exact room so that they can get into the building and out as fast as possible so that their spending minimal time in an unsafe building.

With our project, we hope to provide the Social Equity to rescue workers. They deserve it for their noble deeds after all. Ethically, it is great that we are working to protect the people of our society who risk their lives for the safety of others. Moreover by us focusing on helping to keep the rescue workers safe, then it only benefits society as a whole since we retain the number of heroes who can continue to help people with their long Life.

7.2. 8 Frugal Design Strategies

In order for our project to be used with the result we want, we first need to realize how beneficial our project should be. We have done this by looking at the 8 Frugal Design Strategies and which one we will use now and could include in the future.

Ruggedization: Our project is meant to be used in situations where the building is not structurally sound. Ours in particular is designed to be used for earthquake scenarios

where there is no fire, so with more time we would want our to be able to be hit with falling debris but maintain integrity.

Affordability: Cost is always a factor for anything implemented in a company and especially with a government branch like rescue workers. Our device uses easily purchased sensors that anyone at home could buy and make themselves, making this a cost effective product.

Simplification: Our project is designed to be quick and to the point. There is no use for fancy interfaces. We simply want to take pictures and send data.

Adaption: Our project is to simply show the capabilities of what our robot system can do. It is meant to be expanded upon so that people can integrate them into robots of all different sizes.

Renewability: Each microcontroller's lifespan is responsible for a CO2 emission of a car driving one mile, which becomes a huge amount when 49.3 Billion are sold worldwide [18]. To reduce our carbon footprint, we would look towards local materials, primarily at microcontrollers that are no longer used but could be reused.

Reliance on local materials, manufacturing: Our parts were mostly ordered new off of online shopping. However, if we were ever to mass produce our project, we would want to contact local businesses or recycling centers to ask for any microcontrollers they have no use for.

User-Centric Design: The project is meant to be used during high stress situations, so we need a very simple interface. The only action a rescue worker would need to take is to navigate the car with a controller. The robot itself will take pictures and send it to the driver on a screen

Lightweight: Because rescue workers already have to carry so much gear on them, we do not want to add so much extra weight they would be discouraged on using it. Our project is designed with few sensors and a simplistic setup so that it does not intrude on the gear that must already be carried.

The strategies that we are currently implementing into our project is Simplification, User-centric Design, and Adaption. This is because we want to address the “Usability” aspect of “Professional Issues and Constraints”. The biggest obstacle of introducing any new technology to any old business is that people need to be convinced about change. If our project has very few sensors with no bells and whistles and requires very simple user inputs, rescue workers will better understand it and be more inclined to use it for the benefits it provides of protecting their lives. Moreover, we know different emergency scenarios call for different robots. An RC car would not be ideal for every building, so implementing our system on a drone or snake robot would increase the effectiveness. With a simple system, we provide an easy use experience and opportunity to expand our system’s capabilities

7.3. Environmental Impact

Two materials that are essential to our project are Microcontrollers and Silicon. The lifespan of a single Microcontroller going to a consumer or business is equal to 390 g of CO₂, which is car driving one mile [17]. It was estimated by IC Insights that in 2019 we would have 49.3 Billion microcontrollers sold in 2019 worldwide [18]. To conceptualize this, the environmental impact would be equal to 1.49 Million people, $\frac{3}{4}$ of Santa Clara County’s Population [19], traveling to San Francisco from the city of Santa Clara for work, and then back every day of the year. Silicon itself is responsible for 6.3% of this CO₂ emissions, however that is only scratching the surface on Silicon. This material is found in our infrared, our camera, our LiDAR, and every other electronic device you can think of. Silicon is responsible for 37.3% of the water waste for a Microcontroller [17]. This totals to about 13.31 million gallons of water

wasted each day of the year. That means we are filling a little over 20 olympic size swimming pools every single day and are just emptying it.

7.4. Solutions

When looking at the CO₂ emissions of a Microcontroller, 81% of them come from the production phase [17]. The best course of action to prevent this environmental impact is if we avoid it all together. For our design, to make it environmentally friendly, it would be ideal to go to a electronic recycling facility and retrieve working microcontrollers they may already have. Either way, we should use a microcontroller that has been used and neglected rather than purchase a new one. In terms of controlling the silicon aspect of all our electronics, we would do well to seek a vendor whose manufacturing process are aiming for net zero impact, but here we could also seek our thermal camera, digital camera, and LiDAR from a recycling facility as this would not expend new water to create our device.

7.5. Local Recycling

TDR Electronic Recycling in San Jose, CA and Recology in San Francisco, CA are both locations we can recycle electronics. TDR looks to see if the item can be reused and sold in a store. If not it is de-manufactured, broken into its component parts and sent for metal recovery [20]. Although Recology does not have their own store to resell electronics, they are very community driven. They have ran programs to make Google Mountain View campus more sustainable but one notable program is the Recology Artist in Residence (AIR) program. Recology use a lot of waste material and donate it to local artists so they can make their masterpieces from this [21]. Although not used to create new electronics, it gets the community involved with the idea of sustainability.

7.6. Other Technologies

Biogas is an emerging technology in India. This technology shows perfectly how the frugal design strategy of Affordability and Renewability can take place. India's energy demand is roughly 80% dependent on crude oil imports which is very pricey, but because of their plethora of their waste biomass, India can create about 49 billion m³ per year of Biogas [22]. This makes it an exceptional way to save India money, especially those in poorer communities who can reap the benefits of locally produced Biogas. Since the Biogas itself is made from waste material, it is India's attempt at making a circular green economy. Furthermore, because Biogas is produced locally, then that means that this technology also satisfies the Reliance on Local Materials, Manufacturing strategy.

Soapens is a technology that serves to help hygiene of developing countries. As many as 3.5 million children worldwide die each year from diarrhea and respiratory infections [23]. This is due to the lack of hygiene. The Frugal Design Strategy used most notably here is the User-centric design. Their reasoning was that kids like to draw, but they do not take the time out of the day to wash their hands. Sometimes the teachers have to take them individually to the bathroom. With Soapens, kids are encouraged to draw on their hand with the soap since it is second nature for them to do so. The teacher can then easily keep track of who washed their hands and who did not since the colored soap will still be on their hands. Even better, the other Frugal Strategy used is Lightweight design. These Soapens can fit easily into the child's backpack and can be taken out for an activity during the day.

8. Future Work

8.1. SLAM

We believe that if we can add SLAM to our platform solution it could provide important data to first responders. We could utilize the LiDAR data to create a 2D map of the environment. This map will then allow the user to determine the best routes to navigate the environment. To map an environment multiple LiDAR scans are needed. Subsequently these

LiDAR scans must be put together according to the positional difference of the robot. This could be done with accurate odometry on the chassis of the platform or it could be done by algorithms on the LiDAR scans. We researched these algorithms and concluded that it would require extensive programming to implement.

8.2. Wireless

Our use scenario requires the access to low infrastructure debris areas and to explore these areas. This is why utilizing a wireless connection will greatly increase our mobility. For the communication method on the other hand we would require the use of low frequency in order to be able to transmit through walls and from underground. It would also need to be a stand alone system that does not require infrastructure like WiFi to work in natural disaster situations. Even with a low bit-rate communication we would be able to communicate out sensor data because we intentionally designed our platform to work with low bandwidth.

9. Lessons Learned

Starting this project we initially had an idea. Our idea was a platform system that could scan the room it was in. But we really did not know of all the challenges that we had to overcome. One of the first things we learned was that there was not a single communication protocol amongst electronic devices. Every part we looked had different communication protocol and so we decided to look into all the primary ones. These were UART, Serial, I2C and MISO/MOSI.

Secondly we had to learn how to code in different languages to bring together each component. We used PYTHON, JAVA, and C. This coding greatly improved our abilities. Because we were able to practice them in this project.

References

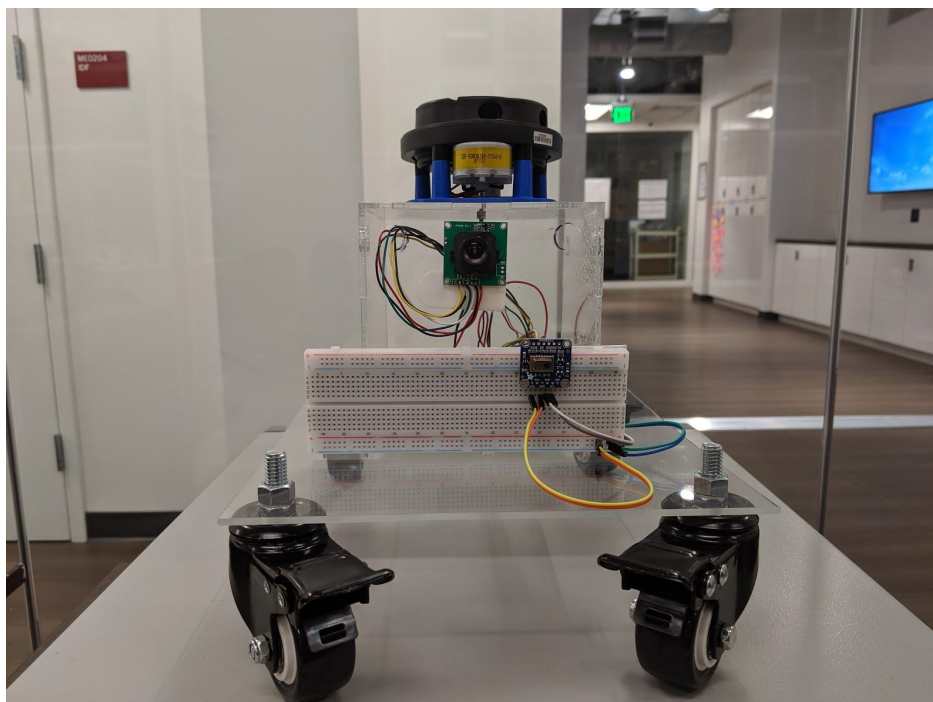
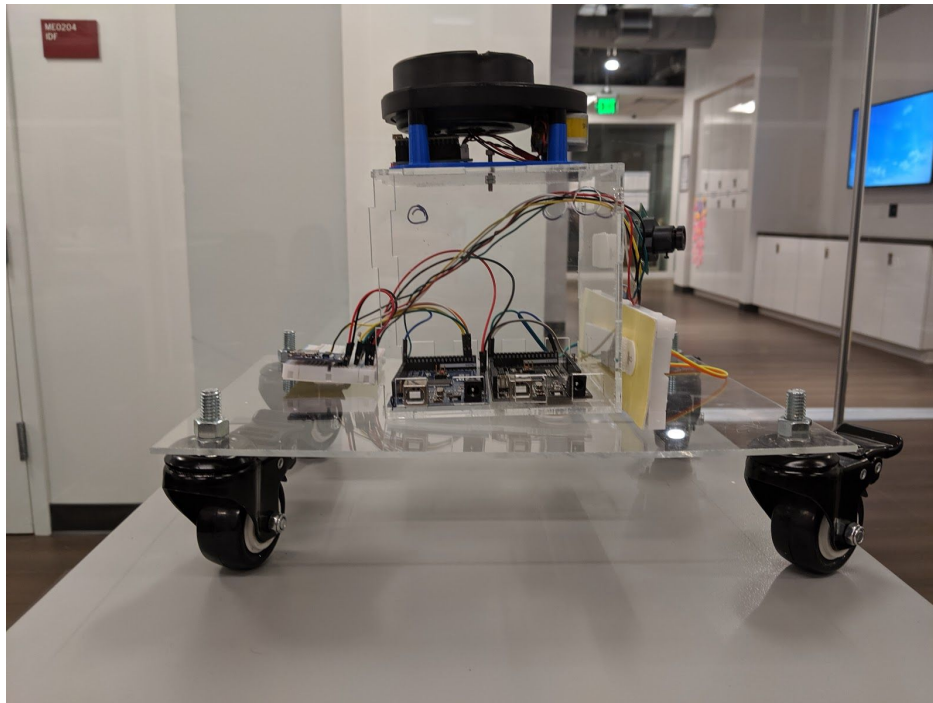
- [1] U. Administration, "Summary incident report", *Apps.usfa.fema.gov*, 2019. [Online]. Available: <https://apps.usfa.fema.gov/firefighter-fatalities/fatalityData/incidentDataReport>. [Accessed: 12- Jun- 2019]
- [2] "LuminAID Solar Lanterns and Solar 2-in-1 Phone Chargers", *LuminAID*, 2019. [Online]. Available: <https://luminaid.com/>. [Accessed: 12- Jun- 2019]
- [3] R. Cheng, "Inside Fukushima: Standing 60 feet from a nuclear disaster - Video", *CNET*, 2019. [Online]. Available: <https://www.cnet.com/videos/inside-fukushima-standing-60-feet-from-a-nuclear-disaster/>. [Accessed: 12- Jun- 2019]
- [4] R. Burnett, "Understanding How Ultrasonic Sensors Work - MaxBotix Inc.", *MaxBotix Inc.*, 2019. [Online]. Available: <https://www.maxbotix.com/articles/how-ultrasonic-sensors-work.htm>. [Accessed: 12- Jun- 2019]
- [5] S. SEVEN, İ. C. DİKMEN, T. KARADAĞ, H. G. BAKIR and T. ABBASOV, "Design and implementation of ultrasonic sonar system for autonomous vehicles," *2018 International Conference on Artificial Intelligence and Data Processing (IDAP)*, Malatya, Turkey, 2018, pp. 1-4.
- [6] D. Lv, X. Ying, Y. Cui, J. Song, K. Qian and M. Li, "Research on the technology of LIDAR data processing," *2017 First International Conference on Electronics Instrumentation & Information Systems (EIIS)*, Harbin, 2017, pp. 1-5.
- [7] K. Song *et al.*, "Navigation Control Design of a Mobile Robot by Integrating Obstacle Avoidance and LiDAR SLAM," *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Miyazaki, Japan, 2018, pp. 1833-1838.
- [8] R. T. Valadas, A. R. Tavares, A. M. d. Duarte, A. C. Moreira and C. T. Lomba, "The infrared physical layer of the IEEE 802.11 standard for wireless local area networks," in *IEEE Communications Magazine*, vol. 36, no. 12, pp. 107-112, Dec. 1998.
- [9] M. Simon, "This Robot Snake Means You No Harm, Really", *WIRED*, 2019. [Online]. Available: <https://www.wired.com/story/this-robot-snake-means-you-no-harm-really/>. [Accessed: 12- Jun- 2019]
- [10] D. Norman, *The Design of Everyday Things*. New York: BasicBooks, 2006.

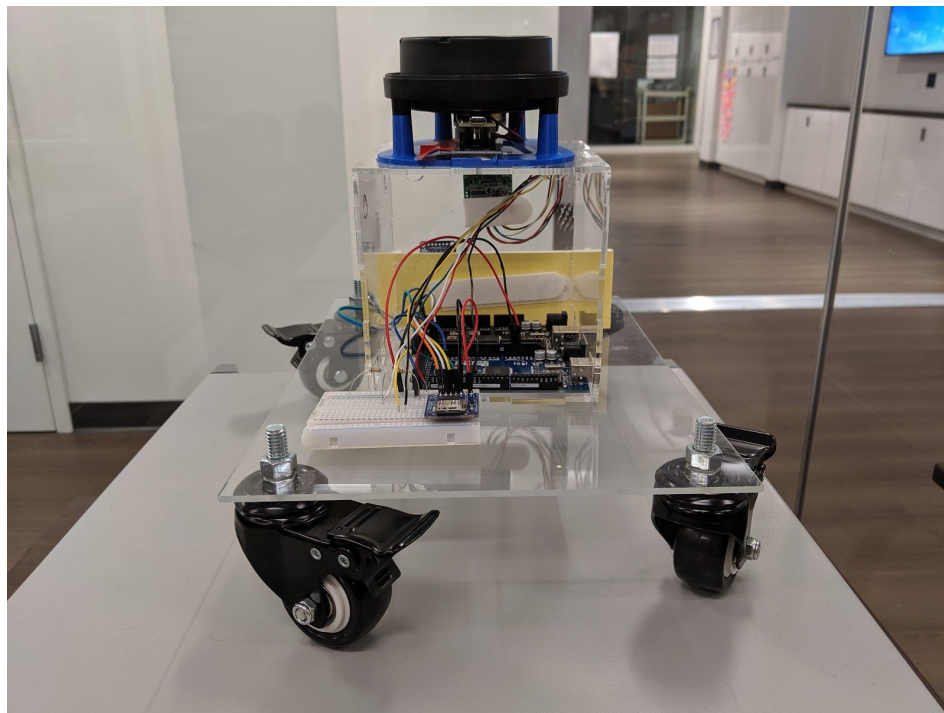
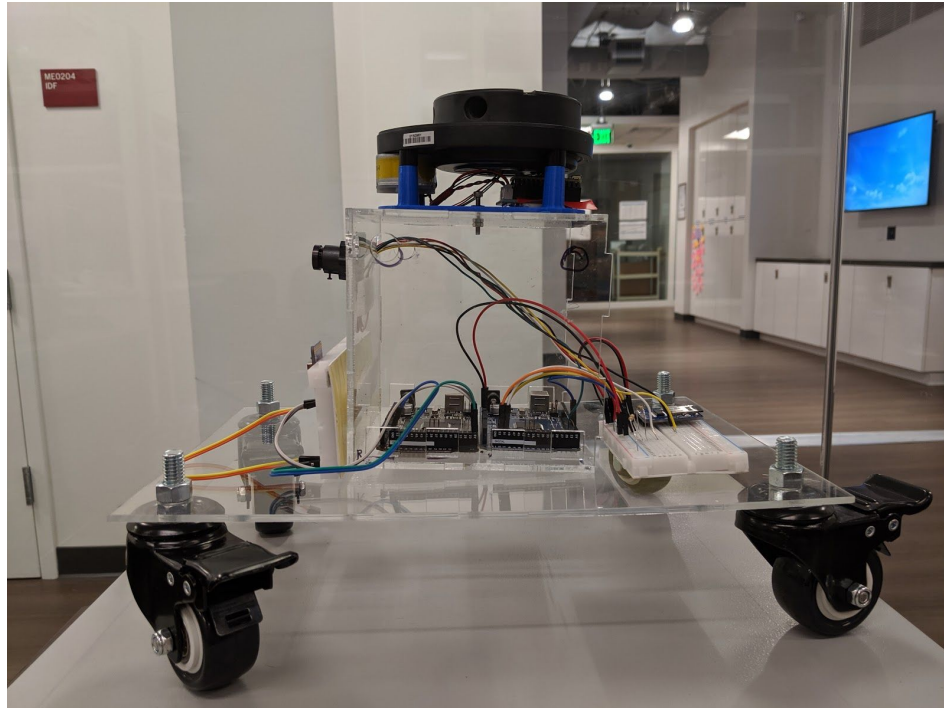
- [11] K. B. Bharath, K. V. Kumaraswamy and R. K. Swamy, "Design of arbitrated I2C protocol with DO-254 compliance," *2016 International Conference on Emerging Technological Trends (ICETT)*, Kollam, 2016, pp. 1-5.
- [12] IEEE Standard for a High-Performance Serial Bus - Redline," in *IEEE Std 1394-2008 (Revision of IEEE Std 1394-1995) - Redline* , vol., no., pp.1-1074, 21 Oct. 2008
- [13] H. -. Hsin, "Texture segmentation in the joint photographic expert group 2000 domain," in *IET Image Processing*, vol. 5, no. 6, pp. 554-559, Sept. 2011.
- [14] RP 167:1995 - SMPTE Recommended Practice - Alignment of NTSC Color Picture Monitors," in *RP 167:1995* , vol., no., pp.1-7, 11 Feb. 1995
- [15] A. Industries, "Adafruit Industries, Unique & fun DIY electronics and kits", *Adafruit.com*, 2019. [Online]. Available: <https://www.adafruit.com/>. [Accessed: 12- Jun- 2019]
- [16] S. University, "A Framework for Ethical Decision Making", *Scu.edu*, 2019. [Online]. Available: <https://www.scu.edu/ethics/ethics-resources/ethical-decision-making/a-framework-for-ethical-decision-making/>. [Accessed: 12- Jun- 2019]
- [17]A. ST and S. Priorities, "Footprint of a Microcontroller - STMicroelectronics", *St.com*, 2019. [Online]. Available: https://www.st.com/content/st_com/en/about/st_approach_to_sustainability/sustainability-priorities/sustainable-technology/eco-design/footprint-of-a-microcontroller.html#. [Accessed: 25- Mar- 2019].
- [18]IC Insights, "Microcontroller Unit Shipments Surge but Falling Prices Sap Sales Growth", *Icinsights.com*, 2015. [Online]. Available: <http://www.icinsights.com/news/bulletins/Microcontroller-Unit-Shipments-Surge-But-Falling-Prices-Sap-Sales-Growth/>. [Accessed: 25- Mar- 2019].
- [19]US Census Bureau, "Search Results", *Census.gov*, 2017. [Online]. Available: https://www.census.gov/search-results.html?q=santa+clara+county+population&page=1&state=Geo=none&searchtype=web&cssp=SERP&_charset_=UTF-8. [Accessed: 25- Mar- 2019].
- [20]TDR, "Free Ewaste Pick up and Drop off - Electronic Recycling", *TDR Electronic Recycling*, 2019. [Online]. Available: <https://tdrelectronicrecycling.com/>. [Accessed: 25- Mar- 2019].

- [21]Recology, "Cultural Impact - Recology", *Recology*, 2019. [Online]. Available:
<https://www.recology.com/cultural-impact/#art-of-recology>. [Accessed: 25- Mar- 2019].
- [22]V. Vijay and P. Ghosh, "Biological Waste to Biogas: A Solution to Self-Sustainable Energy in Rural India", *engineeringforchange.org*, 2018. [Online]. Available:
<https://www.engineeringforchange.org/news/biological-waste-biogas-solution-energy-self-sustainable-rural-india/>. [Accessed: 25- Mar- 2019].
- [23]R. Goodier, "Shh.. Playing with Soap Actually Prevents Disease and Childhood Mortality", *engineeringforchange.org*, 2017. [Online]. Available:
<https://www.engineeringforchange.org/news/soapen-playing-soap-prevents-disease-childhood-mortality/>. [Accessed: 25- Mar- 2019].

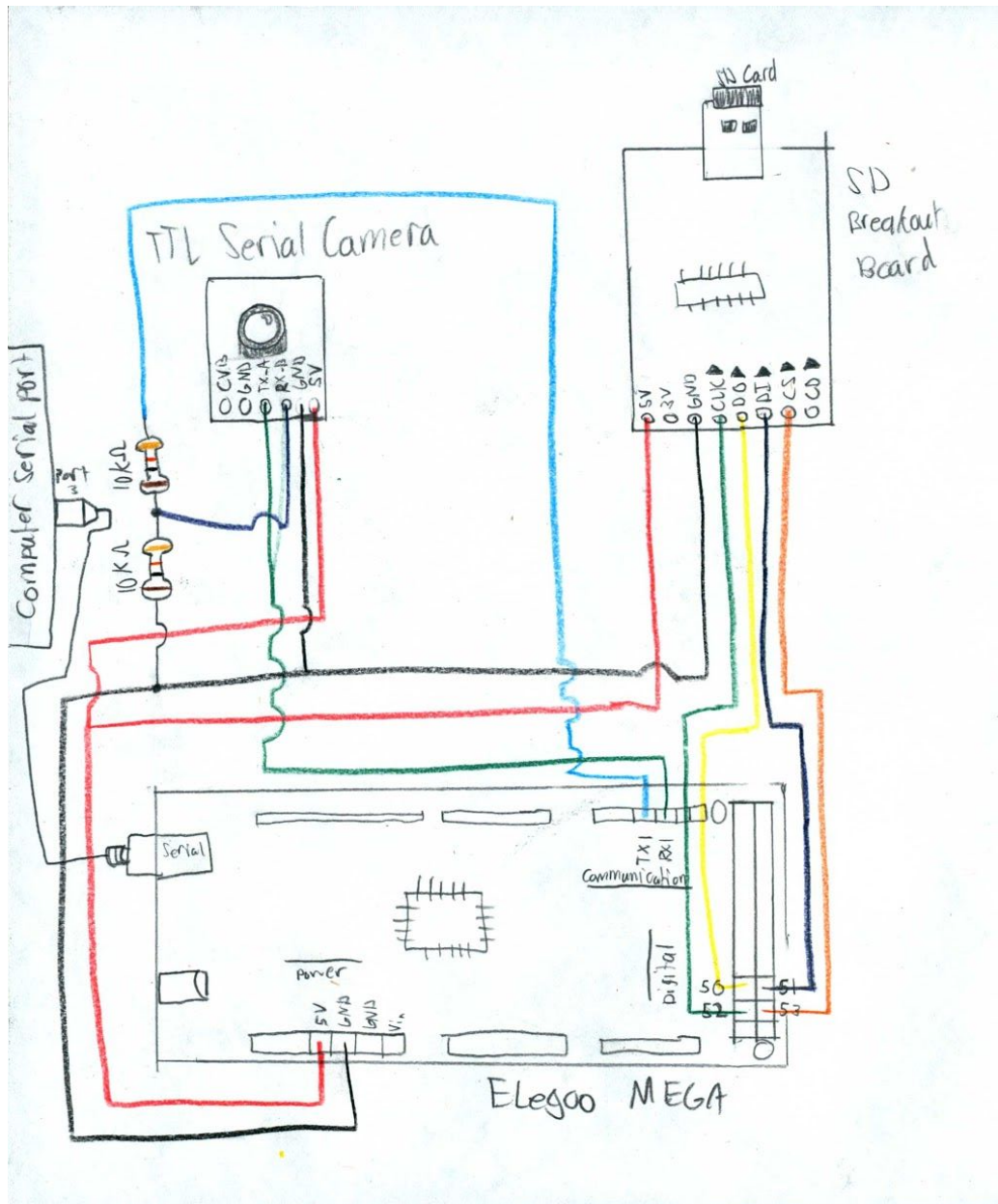
Appendices

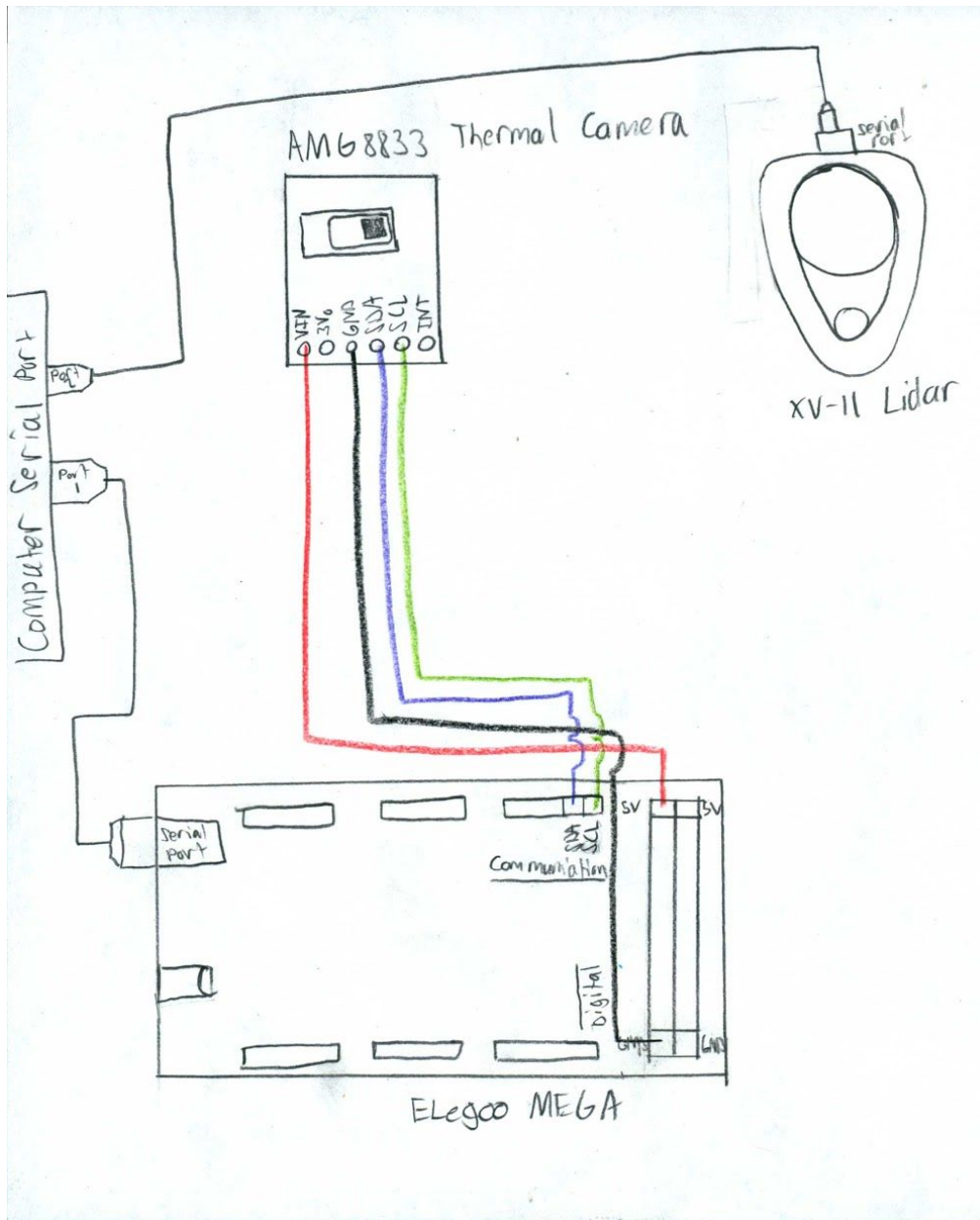
A Sensor System





B Schematics





C LiDAR Code

```
port1 = serial.Serial('COM6' , 115200)

print(port1.name)
print("Serial Started \n")

def getData():

    # port1.write((s + '\n').encode('utf-8'))
```



```

# EMPTY BUFFER
port1.reset_input_buffer()

i = 0
while i < 1000:
    line = port1.readline()
    if(line[0:3] == ("A,0").encode('utf-8')):
        print(line)
        break
    i += 1

stream = line
for x in range(720):
    stream += port1.readline()
print(stream)
return stream

def data2dict( stream ):

    #PHASE 1: Serial STRING -> INT ARRAY (360) 0:??? 1:??? 3:??? 4:???
    variable_array = stream.split("\r\n".encode('utf-8'))
    info_dict = {}    # key = angle value: [distance,signal,count]
    for i in range(0,len(variable_array)):
        current_line = variable_array[i]
        if(len(current_line) == 0):
            continue
        else:
            angle = 0
            distance = 0
            signal = 0
            line_reading = current_line.split(','.encode('utf-8')) # A , ??? , ??? I, ??? S
            if(line_reading[0] == 'A'.encode('utf-8')):
                if(len(line_reading) >= 2 and False == ('A'.encode('utf-8') in line_reading[1])):
                    angle = int( line_reading[1] )
                    if(len(line_reading) >= 3 and line_reading[2] != 'I'.encode('utf-8') and line_reading[2] !=
'S'.encode('utf-8') and False == ('A'.encode('utf-8') in line_reading[2])):
                        print(line_reading[2])
                        distance = int( line_reading[2] )
                        if(len(line_reading) >= 4 and line_reading[3] != 'S'.encode('utf-8') and False ==
('A'.encode('utf-8') in line_reading[3])):
                            signal = int(line_reading[3])
                            if(angle in info_dict):
                                d = info_dict[angle][0]

```

```

        s = info_dict[angle][1]
        c = info_dict[angle][2]
        info_dict[angle] = [(d*c + distance)/(c+1),(s*c + signal)/(c+1),c+1]
    else:
        info_dict[angle] = [distance,signal,1]
    else:
        continue

print(info_dict)
return info_dict


def polar2cart( info_dict ):
    x_values = []
    y_values = []

    for i in info_dict.keys():
        x_values.append(info_dict[i][0] * np.cos(i * np.pi / 180)) # rads
        y_values.append(info_dict[i][0] * np.sin(i * np.pi / 180))

    return (np.array(x_values),np.array(y_values))


fig = plt.figure(1)
plt.ion()
plt.show()

i = 0

while True:
    stream = getData()
    info_dict1 = data2dict( stream )
    x,y = polar2cart(info_dict1)
    plt.xlim(-6000,6000)
    plt.ylim(-6000,6000)
    plt.scatter(x,y,1.5)
    plt.pause(1)

```

```

if(i >= 5):
    fig.clear()
    i = 0
i += 1

```

D Imaging Code (Arduino)

```

#include <Adafruit_VC0706.h>
#include <SPI.h>
#include <SD.h>
#include <JPEGDecoder.h>

#define chipSelect 53

Adafruit_VC0706 cam = Adafruit_VC0706(&Serial1);

void setup() {
    pinMode(13, OUTPUT);

    // When using hardware SPI, the SS pin MUST be set to an
    // output (even if not connected or used). If left as a
    // floating input w/SPI on, this can cause lockuppage.
    #if !defined(SOFTWARE_SPI)
    #if defined(__AVR_ATmega1280__) || defined(__AVR_ATmega2560__)
        if(chipSelect != 53) pinMode(53, OUTPUT); // SS on Mega
    #else
        if(chipSelect != 10) pinMode(10, OUTPUT); // SS on Uno, etc.
    #endif
    #endif

    Serial.begin(115200);
    // Serial.println("VC0706 Camera snapshot test");

    // see if the card is present and can be initialized:
    if (!SD.begin(chipSelect)) {
        Serial.println("Card failed, or not present");
        // don't do anything more:
        return;
    }
}

```

```

// Try to locate the camera
if (!cam.begin()) {
//   Serial.println("Camera Found:");
// } else {
//   Serial.println("No camera found?");
    return;
}

/*
// Print out the camera version information (optional)
char *reply = cam.getVersion();
if (reply == 0) {
    Serial.print("Failed to get version");
} else {
    Serial.println("-----");
    Serial.print(reply);
    Serial.println("-----");
}
*/

// Set the picture size - you can choose one of 640x480, 320x240 or 160x120
// Remember that bigger pictures take longer to transmit!

//cam.setImageSize(VC0706_640x480);    // biggest
cam.setImageSize(VC0706_320x240);    // medium
//cam.setImageSize(VC0706_160x120);    // small

// You can read the size back from the camera (optional, but maybe useful?)
uint8_t imgsize = cam.getImageSize();
// Serial.print("Image size: ");
// if (imgsize == VC0706_640x480) Serial.println("640x480");
// if (imgsize == VC0706_320x240) Serial.println("320x240");
// if (imgsize == VC0706_160x120) Serial.println("160x120");

// Wait for the PC to signal
while(!Serial.available());

// Serial.println("Snap in 3 secs...");
delay(3000);

if (! cam.takePicture())
    return;
//   Serial.println("Failed to snap!");

```

```

// else
//   Serial.println("Picture taken!");

// Create an image with the name IMAGExx.JPG
char filename[13];
strcpy(filename, "IMAGE00.JPG");
for (int i = 0; i < 100; i++) {
    filename[5] = '0' + i/10;
    filename[6] = '0' + i%10;
    // create if does not exist, do not open existing, write, sync after write
    if (!SD.exists(filename)) {
        break;
    }
}

// Open the file for writing
File imgFile = SD.open(filename, FILE_WRITE);

// Get the size of the image (frame) taken
uint16_t jpglen = cam.frameLength();
// Serial.print("Storing ");
// Serial.print(jpglen, DEC);
// Serial.print(" byte image.");

int32_t time = millis();
pinMode(8, OUTPUT);
// Read all the data up to # bytes!
byte wCount = 0; // For counting # of writes
while (jpglen > 0) {
    // read 32 bytes at a time;
    uint8_t *buffer;
    uint8_t bytesToRead = min(32, jpglen); // change 32 to 64 for a speedup but may not work with all
    setups!
    buffer = cam.readPicture(bytesToRead);
    imgFile.write(buffer, bytesToRead);
    if(++wCount >= 64) { // Every 2K, give a little feedback so it doesn't appear locked up
//   Serial.print('.');
        wCount = 0;
    }
    //Serial.print("Read "); Serial.print(bytesToRead, DEC); Serial.println(" bytes");
    jpglen -= bytesToRead;
}
imgFile.close();

```

```

    time = millis() - time;
    // Serial.println("done!");
    // Serial.print(time); Serial.println(" ms elapsed");

    //Decoding

    // Open the root directory
    // char filename[13];
    char myFile[13];
    // strcpy(filename,"IMAGE00.JPG");
    myFile[0] = '/';
    for(int i = 0; i<13;i++){
        myFile[i+1]=filename[i];
    }
    //filename[0] = "/";
    /*filename = strcat(filename, "IMAGE00.JPG",strlen("IMAGE00.JPG"));

    File root = SD.open(myFile);
    //Increment to first JPEG
    // root.openNextFile();
    // root.openNextFile();

    // Wait for the PC to signal
    // while(!Serial.available());

    // Send all files on the SD card
    // while(true) {
        // Open the next file
        File jpgFile = root;

    // // We have sent all files
    // if(!jpgFile) {
    //     break;
    // }

    // Decode the JPEG file
    JpegDec.decodeSdFile(jpgFile);

    // Create a buffer for the packet
    char dataBuff[240];

```

```

// Fill the buffer with zeros
initBuff(dataBuff);

// Create a header packet with info about the image
String header = "$ITHDR,";
header += JpegDec.width;
header += ",";
header += JpegDec.height;
header += ",";
header += JpegDec.MCUSPerRow;
header += ",";
header += JpegDec.MCUSPerCol;
header += ",";
header += jpgFile.name();
header += ",";
header.toCharArray(dataBuff, 240);

// Send the header packet
for(int j=0; j<240; j++) {
    Serial.write(dataBuff[j]);
}

// Pointer to the current pixel
uint16_t *pImg;

// Color of the current pixel
uint16_t color;

// Create a data packet with the actual pixel colors
strcpy(dataBuff, "$ITDAT");
uint8_t i = 6;

// Repeat for all MCUs in the image
while(JpegDec.read()) {
    // Save pointer the current pixel
    pImg = JpegDec.pImage;

    // Get the coordinates of the MCU we are currently processing
    int mcuXCoord = JpegDec.MCUx;
    int mcuYCoord = JpegDec.MCUy;

    // Get the number of pixels in the current MCU
    uint32_t mcuPixels = JpegDec.MCUWidth * JpegDec.MCUHeight;

```

```

// Repeat for all pixels in the current MCU
while(mcuPixels--) {
    // Read the color of the pixel as 16-bit integer
    color = *pImg++;

    // Split it into two 8-bit integers
    dataBuff[i] = color >> 8;
    dataBuff[i+1] = color;
    i += 2;

    // If the packet is full, send it
    if(i == 240) {
        for(int j=0; j<240; j++) {
            Serial.write(dataBuff[j]);
        }
        i = 6;
    }

    // If we reach the end of the image, send a packet
    if((mcuXCoord == JpegDec.MCUSPerRow - 1) &&
        (mcuYCoord == JpegDec.MCUSPerCol - 1) &&
        (mcuPixels == 1)) {

        // Send the pixel values
        for(int j=0; j<i; j++) {
            Serial.write(dataBuff[j]);
        }

        // Fill the rest of the packet with zeros
        for(int k=i; k<240; k++) {
            Serial.write(0);
        }
    }
}
// }

// Safely close the root directory
root.close();
}

// Function to fill the packet buffer with zeros

```



```

void initBuff(char* buff) {
    for(int i = 0; i < 240; i++) {
        buff[i] = 0;
    }
}

```

```

void loop() {
}

```

E Imaging Code (Processing)

```

// Import the library
import processing.serial.*;

Serial port;

void setup() {
    // Set the default window size to 200 by 200 pixels
    size(200, 200);

    // Set the background to grey
    background(#888888);

    // Set as high framerate as we can
    frameRate(1000000);

    // Start the COM port communication
    // You will have to replace "COM30" with the Arduino COM port number
    port = new Serial(this, "COM6", 115200);

    // Read 240 bytes at a time
    port.buffer(240);
}

// String to save the trimmed input
String trimmed;

// Buffer to save data incoming from Serial port
byte[] byteBuffer = new byte[240];

// The coordinate variables

```

```

int x, y, mcuX, mcuY;

// A variable to measure how long it takes to receive the image
long startTime;

// A variable to save the current time
long currentTime;

// Flag to signal end of transmission
boolean received = false;

// Flag to signal reception of header packet
boolean headerRead = false;

// The color of the current pixel
int inColor, r, g, b;

// Image information variables
int jpegWidth, jpegHeight, jpegMCUSPerRow, jpegMCUSPerCol, mcuWidth, mcuHeight, mcuPixels;

// This function will be called every time any key is pressed
void keyPressed() {
  // Send something to Arduino to signal the start
  port.write('s');
  println("signal sent");
  // Start the timer
  startTime = millis();
}

// This function will be called every time the Serial port receives 240 bytes
void serialEvent(Serial port) {
  // Read the data into buffer

  port.readBytes(byteBuffer);

  // Make a String out of the buffer
  String inString = new String(byteBuffer);

  // Detect the packet type
  if(inString.indexOf("$ITHDR") == 0) {
    // Header packet

```

```

// Remove all whitespace characters
trimmed = inString.trim();

// Split the header by comma
String[] list = split(trimmed, ',');

// Check for completeness
if(list.length != 7) {
    println("Incomplete header, terminated");
    while(true);
} else {
    // Parse the image information
    jpegWidth = Integer.parseInt(list[1]);
    jpegHeight = Integer.parseInt(list[2]);
    jpegMCUSPerRow = Integer.parseInt(list[3]);
    jpegMCUSPerCol = Integer.parseInt(list[4]);

    // Print the info to console
    println("Filename: " + list[5]);
    println("Parsed JPEG width: " + jpegWidth);
    println("Parsed JPEG height: " + jpegHeight);
    println("Parsed JPEG MCUs/row: " + jpegMCUSPerRow);
    println("Parsed JPEG MCUs/column: " + jpegMCUSPerCol);

    // Start the timer
    //startTime = millis();
}

// Set the window size according to the received information
surface.setSize(jpegWidth, jpegHeight);

// Get the MCU information
mcuWidth = jpegWidth / jpegMCUSPerRow;
mcuHeight = jpegHeight / jpegMCUSPerCol;
mcuPixels = mcuWidth * mcuHeight;

} else if(inString.indexOf("$ITDAT") == 0) {
    // Data packet

    // Repeat for every two bytes received
    for(int i = 6; i < 240; i += 2) {
        // Combine two 8-bit values into a single 16-bit color
        inColor = ((byteBuffer[i] & 0xFF) << 8) | (byteBuffer[i+1] & 0xFF);
    }
}

```

```

// Convert 16-bit color into RGB values
r = ((inColor & 0xF800) >> 11) * 8;
g = ((inColor & 0x07E0) >> 5) * 4;
b = ((inColor & 0x001F) >> 0) * 8;

// Paint the current pixel with that color
set(x + mcuWidth*mcuX, y + mcuHeight*mcuY, color(r, g, b));

// Move onto the next pixel
x++;

if(x == mcuWidth) {
    // MCU row is complete, move onto the next one
    x = 0;
    y++;
}

if(y == mcuHeight) {
    // MCU is complete, move onto the next one
    x = 0;
    y = 0;
    mcuX++;
}

if(mcuX == jpegMCUSPerRow) {
    // Line of MCUs is complete, move onto the next one
    x = 0;
    y = 0;
    mcuX = 0;
    mcuY++;
}

if(mcuY == jpegMCUSPerCol) {
    // The entire image is complete
    received = true;
}
}
}
}

void draw() {
    // If we received a full image, start the whole process again

```

```

if(received) {

    // Reset coordinates
    x = 0;
    y = 0;
    mcuX = 0;
    mcuY = 0;

    // Reset the flag
    received = false;

    // Measure how long the whole thing took
    long timeTook = millis() - startTime;
    println("Image receiving took: " + timeTook + " ms");
    println();
}
}

```

F Thermal Camera Code

```

port1 = serial.Serial('COM9' , 9600)

''' PART 1: Get Data '''
def getData() :

    port1.reset_input_buffer()
    line = port1.readline()
    f_line = line.split("\r\n".encode('utf-8'))
    s_line = f_line[0].split(",".encode('utf-8'))

    data_arr = []

    if( len(s_line) == 65 ):
        m = 0
        for i in range(0,8):
            q = []
            for j in range(0,8):
                q.append(float(s_line[m]))
                m += 1

```

```

        data_arr.append(q)

    return data_arr

''' PART 2: Interpolate data '''
def interData( data_arr ):
    return NULL
''' COLOR '''
def color(data):
    max_temp, min_temp = 80, 15

    s = (max_temp - min_temp)/255
    s = 8

    if(data < min_temp):
        blue = 255
        red = 0
        green = 0
    if(data > min_temp and data < max_temp):
        red = s * data
        blue = 255 - s*data
        green = 0
    if(data > max_temp):
        red = 255
        blue = 0
        green = 0

    return int(red),int(green),int(blue)

''' PART 3: Data 2 img (show data) '''
def imgData( data_arr ):
    ''' 0 C - 100 C '''
    ''' 8x8 '''

    ''' 800x800 '''
    img_array = []
    if(len(data_arr) == 0):
        return 0

```

```
W,H = 400,400
```

```
img1 = Image.new('RGB', (8 , 8))
```

```
pixels1 = img1.load()
```

```
print((data_arr))
```

```
for i in range(0,8):
```

```
    for j in range(0,8):
```

```
        red,green,blue = color(data_arr[i][j])
```

```
        pixels1[i , j] = (red, green , blue) #(int(6.5 * data_arr[i][j]), 10 , int(255 - (6.5 * data_arr[i][j])))
```

```
#img = img1.resize((W,H), Image.BICUBIC)
```

```
plt.imshow(img1)
```

```
plt.pause(1)
```

```
fig.clear()
```

```
fig = plt.figure(1)
```

```
plt.ion()
```

```
plt.show()
```

```
while True:
```

```
    imgData(getData())
```