ODOT: Hazard Mitigation System for Tractor Mowers

Final Report

Dustin Erickson

Jacob Harvey

Zane Ott

2024

Oregon State University

Oregon Department of Transportation

Project Sponsor: Dr. Galvez, Oregon Department of Transportation

Instructor: Dr. Clemen, Oregon State University

Oregon State University
College of Engineering

DISCLAIMER

This report was prepared by Oregon State University (OSU) students as part of a university course requirement. While considerable effort has been put into the project, it is not the work of licensed engineers and has not undergone the extensive verification that is common in the profession. The information, data, conclusions, and content of this report should not be relied on or utilized without thorough, independent testing and verification. University faculty members may have been associated with this project as advisors, sponsors, or course instructors, but as such they are not responsible for the accuracy of results or conclusions.

ABSTRACT

Oregon Department of Transportation (ODOT) has enlisted the multidisciplinary capstone team to design and build a system that detects potential hazards for tractor mowers. The hazard mitigation system will alert operators to any obstacles or people near the mowers. The project comprises three stages: data collection, training, and final deployment. This capstone team focused on reviewing a previous prototype and designing a system for the first phase of the project. The system that the team designed can capture images from three color cameras and two infrared cameras mounted on several different points on the tractor. It is designed to work independent of the tractor operations and requires no external input from the operator. By the end of summer 2024, the image collection system is projected to capture 4 million images. These images will be used by a future team to train an image recognition model.

ACKNOWLEDGEMENTS

The Oregon State University Engineering Capstone Design Team would like to provide recognition to several individuals for assistance and guidance during the course of this project. Special thanks is granted to the project sponsor and ODOT Research Engineer Christhian Galves, ODOT Equipment Engineer Karl Raschkes, and the Corvallis/Salem maintenance facility for their help and hospitality during the team's frequent visits.

Gratitude is also extended to Dr. Layne Clemen of the OSU engineering department for his role as instructor in the 2023-2024 Engineering Capstone Design program. The team is additionally grateful for support granted by OSU and any relevant faculty members, buildings, and resources aiding in the completion of the ODOT Hazard Mitigation system development.

The ENGR 415 and 416 courses which facilitated the ODOT Hazard Mitigation System project is an Accreditation Board for Engineering and Technology (ABET) affiliated program. The application of fundamental engineering concepts and techniques applied to the Capstone design project are based on ABET accredited courses hosted by OSU. The team would like to acknowledge ABET's role in the project, as well as the countless professors and OSU faculty responsible for fostering a positive and high-level educational environment.

TABLE OF CONTENTS

1 BACKGROUND

1.1 Introduction

ODOT has commissioned Oregon State University to research and develop a "Hazard Mitigation System" (HMS) to solve a solution that has become worse in modern infrastructural and cultural conditions. This system's purpose is to warn the operator of any potentially hazardous objects or people in the path of their tractors. The tractors are equipped with large assemblies to enable them to mow a large area quickly. These assemblies are engineered to be robust when cutting anything from grass, wood, and potentially rock. However, there are items and refuse that pose a risk that can damage equipment installed on these tractors. People who are prone in grass who don't notice the tractor approaching are also a risk that can cause severe injury or death if they are not noticed by the operator.

To protect the tractor, the equipment installed on the tractor, the operator, and people in the vicinity of the tractor, a system to detect all objects and people around the operating platform will be designed to warn the operator so they may approach the hazard on a situational basis. This system has been determined to be developed using modern object detection employing computer vision and AI learning models to define these hazards and let the operator know that they are in their path.

1.2 Project Scope

The project sponsor has placed emphasis on creating a prototype that can easily be turned into a final product. There was no budget given, so the team was able to propose any changes to create a more effective system. This resulted in less emphasis on reusing any of the hardware and software from the first prototype. The following list of customer requirements are in order of priority:

- Must be as close to the final product as logically possible.
- Must be rated for a high dust environment.
- Must be water resistant.
- Must be resistant to heavy vibrations and impacts.
- Must be low power to prevent overheating tractor cabin.
- The system must be able to collect useful data for further development.
    - Must be able to capture people and large animals.
    - Needs to capture a large variety of man-made refuse.
    - Must be able to capture hazards from over 75 ft away.
- Must use as much hardware as possible from previous system listed as:
    - Cameras
        - FLIR Boson ADK 32° FOV
        - OAK-1 W POE
    - Software
        - Developed From "LSA Autonomy"
    - Control system
    - EVS 1100 Industrial Computer

2 DESIGN PROCESS

2.1 Customer and Engineering Requirements

The process in developing a design started defining the problem that needs to be solved, the team took data and information form ODOT engineers, operators, and maintenance personnel, as well as the previous design to general administrative guiding documentations that assists engineers in making fundamentally important decisions such as a House of Quality and a Pugh Matrix.

Before implementing our design, it is necessary to define the customer requirements and our engineering or technical specifications that have been previously defined in our project scope. Using the house of quality in Appendix D, the team was able to define the importance of specific elements of the final system that had the most importance. The team would develop the understanding needed in the importance of the cameras used for the system, qualities from the hardware such as the degree of water resistance, and dust resistance, as well as system requirements that are necessary for the system.

2.2 Major Design Requirements

There are many elements to this system both electrical and mechanical that require special consideration and development. For each of these elements, a Pugh Chart was developed to better understand how the team would go forward in our design approach. Appendix C shows a more granular view on what is required for the system to be defined in the development process. Everything in these Pugh charts was heavily considered as development of the design's progress.

2.3 Bill of Materials

Progression of the design process found the creation of a Bill of Materials for acquisition of relevant hardware that would be necessary for the team to complete the HMS. The electrical design was first developed from the functional models displayed in Appendix A**.**

After defining the electrical design the acquisitions of parts that were necessary for operation were defined in Appendix B. Defining what we already had from the previous system and what we needed to acquire for the newly developed and improved system.

2.4 Iterative Development

Design development after the initial designs progressed as an iterative development with multiple visits to ODOT's Salem Maintenance Facilities for dimensioning and discussions with operators and engineers as well as disassembling, measuring, and pitching methods of design and implementation for the system to mount. After these visits the mechanical engineering student team would use these measurements and ideas to generate multiple iterations and methods to mount the relevant system that would meet customer requirements and engineering specifications. While the mounting solutions were being developed, the electrical engineering student team worked on system viability, data acquisition methods, and the relevant code needed to operate the system seamlessly.

3 DESIGN PROPOSAL – First Term

3.1 Prototype Review

The previous system, displayed in Figure 3.0, was resolved to be underperforming after extensive review. It was determined that minimal hardware could be used in the new design. The control system used between 200 and 350 watts of power which was understood to be too energy intensive. The camera mounting solutions were rudimentary and maintenance intensive on the tractor, requiring parts of the tractor to be edited and machined to mount the cameras in their appropriate positions. The mounting of the control system was propped up behind the operator's seat which would be subject to damage and misuse. Another issue regarding the system was the lack of vibration dampening which could lead to components in the system being loosened and falling apart, leading to the system needing more regular maintenance.

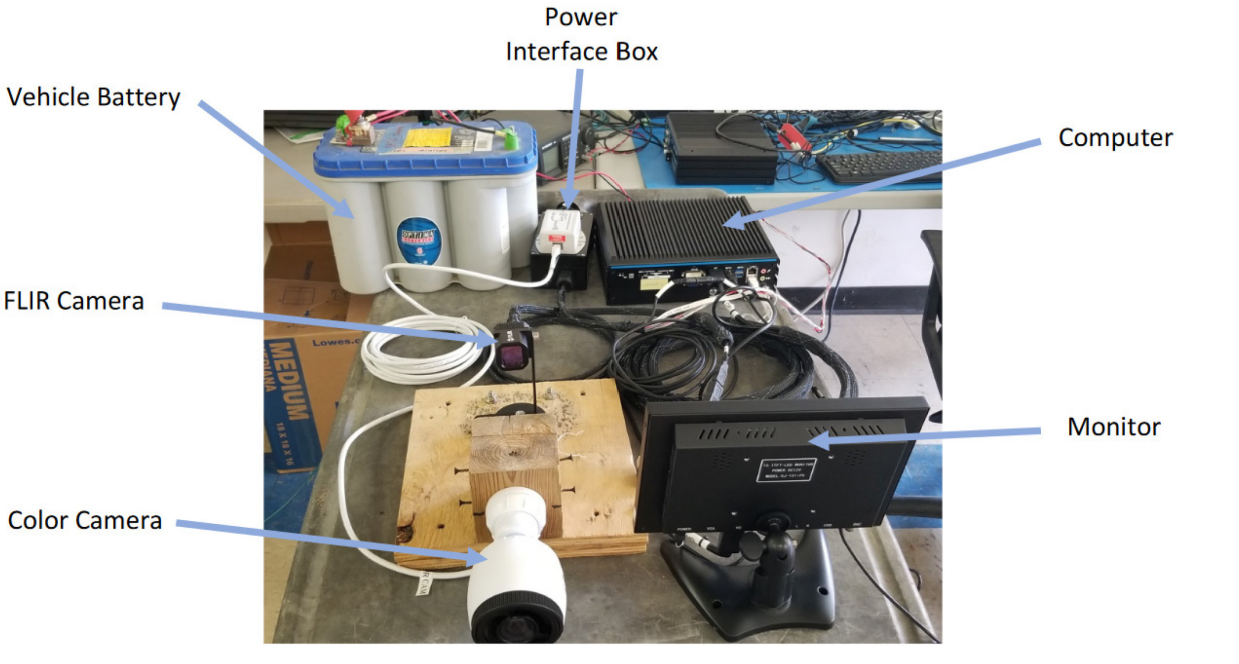


Figure 3.0: Prototype built by LSA Autonomy

After the review, it was determined that the FLIR Boson ADK infrared cameras and the OAK-1 W POE color cameras would be suitable for the new system. The FLIR cameras are thermal imaging cameras that are best used with objects that emit heat or take heat from the environment. These cameras will benefit the system by detecting people and large animals even if they are slightly obstructed by other objects in the environment. The OAK-1 W POE cameras are ideal for the system because they are designed for object detection. These cameras are 4K color cameras that record at 30 frames per second with a 150° field of view. This will ensure that all objects can be detected at a reasonable range while maintaining a wide field of view. Besides the cameras, none of the other electrical hardware was suitable for the final design.

The previous control system was power intensive, so different computer hardware was required to reduce the power drain on the tractor's battery. Another aspect noticed about the previous

system was the large amount of space it took up in the tractor's cab. Although not a generated customer requirement, it was deemed necessary by the team to shrink the footprint of the control system down to free the operator and maintenance personnel from a cumbersome system. The previously designed hardware was defined by the functional model displayed in Figure 3.1.

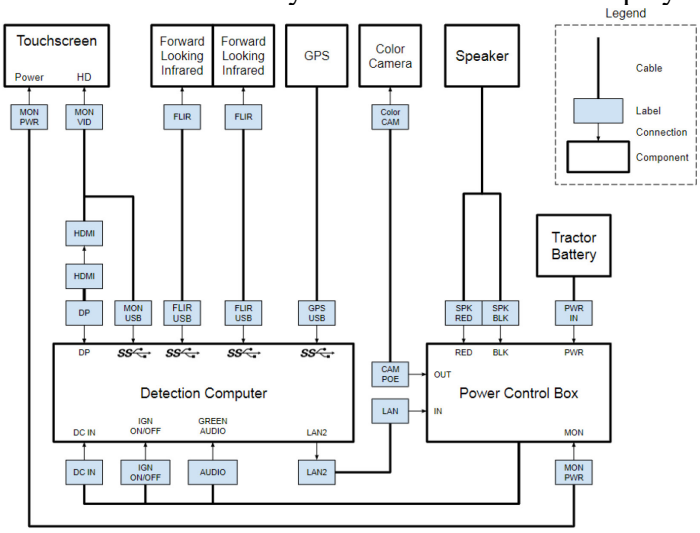


Figure 3.1: Old System Block Model

3.2 Electrical Design

System level changes had to be made to the existing prototype to reduce power, size, and cost. The core change was switching the existing industrial computer for a Nvidia Jetson Nano. The Jetson was chosen as it drastically reduces the power consumption of the system to 10 watts from the current system requirement of 200 watts. The Jetson Nano also reduces the footprint of the control surface to a more manageable and portable system. An additional benefit of the Nano is that it is optimized for computer vision applications. A 2TB SD card, a battery backup, and a voltage converter are all required to run the Jetson Nano in the developed configuration.

Additionally, the camera system was observed to be changed for an IP67 machine learning and object tracking compatible camera that was tested and mounted on the test platform tractor in all previous system tests. This version of the prototype used a Luxonis OAK-1 PoE camera, and it is working as expected. The intended use-case for this camera model is marketed towards camera vision and obstacle detection applications, and it also has excellent documentation for open-source software, aiding in its customizability for the required tasks of the system. The OAK-1 W PoE has a wide field of view, which will be used to see more of the environment per camera. For the 3 ethernet cameras to work, a PoE switch will be required, which will be supplied power via a 48v converter. A detailed block diagram can be found in Appendix A.

3.3 Enclosure Design

All hardware required to connect power to the cameras and compile object recognition training data will be housed inside of an enclosure, which is intended for mounting inside the tractor cab in front/above the operator, outside of the driver's view. The majority of ODOT's mowing tractors feature an unused radio compartment above the steering wheel, and while the internal space in the compartment varies between models, this opening will be used to mount the main

electronics enclosure. This allows the camera and power cables to be routed into the existing wire conduit. An image of the mounting location can be seen in Figure 3.2.

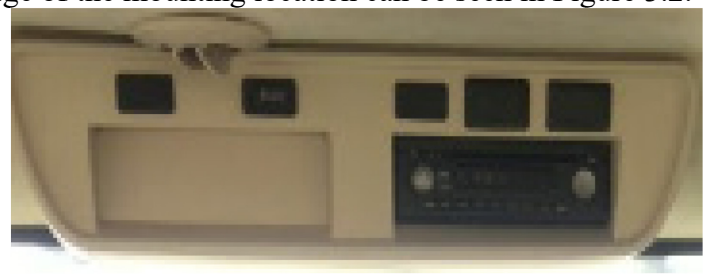


Figure 3.2: Electrical Accessory Slot

The initial layout of the enclosure placed emphasis on routing the ethernet cables directly into the ethernet switch. The rest of the cables would snake through the enclosure to their respective terminations. The initial design seen in Figure 3.3 was based on the datasheet dimensions of each component. This layout changed many times in several weeks to accommodate the real-world size of each component. Additionally, the enclosure's outer dimensions would need to be increased to allow for wiring inside the enclosure.

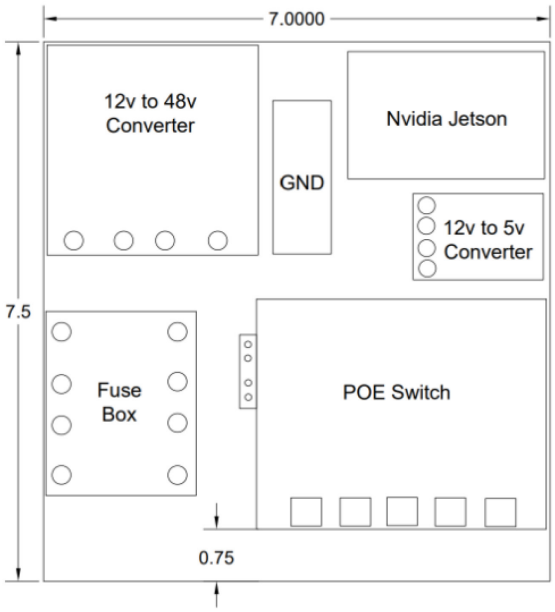


Figure 3.3: First Electrical Enclosure 2D Layout

The biggest challenge with the electronics enclosure design was controlling the ingress of dust while maximizing airflow over the NVIDIA Jetson board. Providing filtered, constant positive air pressure to the case using a 120mm computer case fan ensured minimal dust ingress inside the enclosure. There are several modular options for dust filters built to interface with standard 120mm fans, and therefore the group will determine a reasonable accommodation based on durability, cost, and performance.

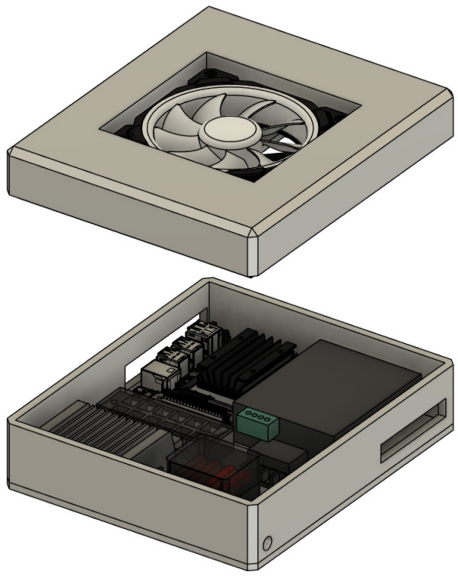


Figure 3.4: First Electrical Enclosure 3D Model

Modeled in Figure 3.4, all electrical components are displayed in the footprint needed to fit in the radio slot. After many iterations with the defined electrical hardware, placement of the hardware was able to fit in the restricted footprint defined by the team. Another consideration when going forward with the design was vibration control. It was essential to ensure the system would be sturdy and rigid when mounted. The enclosure would be easy to manufacture, and all ports will have rated protections installed to ensure a clean and functional system.

3.4 Camera Mounting Design

The cameras for the system had to be mounted in a position that is secure and covers the necessary parts of the tractor. Additionally, the number of cameras must be determined for adequate coverage.

Figure 3.6a: Light Mounting Solution and Figure 3.6b: Door Handle Mounting Solution.

The team decided that 2 FLIR cameras and 3 color cameras were necessary to cover a large enough field of view. The FLIR cameras only have a 32 degree field of view so together they have a combined 64 degree field of view. Color cameras would be placed around the tractor to maximize the area seen by the cameras. To do this with 150 degree view for each camera, the team has settled on three OAK-1 Cameras to mount. When defining the first place to mount the cameras with operators in the field, the headlights on top of the tractor as seen in **Figure 3.6a** seemed to be defined as a potential location as they have direct access to wiring inside the tractor and they are installed by a single M8 1.25 X 2" bolt each. Using this same mounting placement, a housing was modeled to hold one OAK-1 Camera and both of the FLIR cameras.

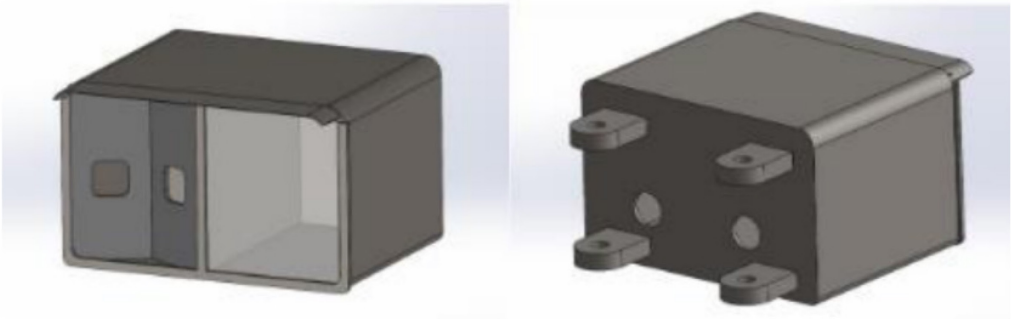


Figure 3.7a: Camera Housing Font Revision 1 and Figure 3.7b: Camera Housing Back Revision 1

Looking at Figure 3.7a, at the front of the picture, the OAK-1 camera would go into the polycarbonate glass covered unit on the right to protect the camera from any impacts, dust, and water. Vibration dampening would also go into the unit where it is housed, but further research and design is needed to figure out that mounting solution. The FLIR cameras on the other hand because they cannot view through translucent material, they would be placed into the faceplate on the left.

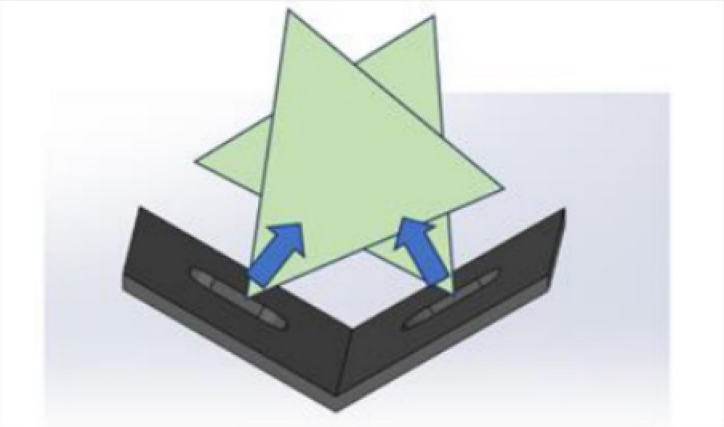


Figure 3.8: FLIR Faceplate Mounting

Figure 3.8 illustrates the simulated FLIR cameras' positions to define a wider field of view so they can detect hazards in a wider area (as represented by the green triangle). This is achieved using the faceplate mounting solution in its current configuration. The Team and onsite operators agreed that the rest of the cameras would be better served inside the tractor cab to protect them from environmental abuse. It was decided that the other two color cameras were to be mounted on the inside handles of the doors as shown in Figure 3.6b**.**

4 Design Solution - Second Term

4.1 Electrical Design

No major changes were implemented to the electrical system after the original design proposal. The final wiring diagram can be seen in Appendix E. The three minor changes to the system were:

- The fuse box was removed and replaced with a single large fuse. Each wire was upsized to accommodate the full load rating of the 5 amp fuse.
- A resistor divider was implemented between the 12v power supply and ground. This acted as a voltage detection circuit for the GPIO pins on the Jetson Nano.
- A parallel resistor and capacitor were placed on the Jetsons power supply pins to allow a remote power on.

This design solution works very well. The system can start up when the tractor is turned on, and it can shut itself off when the tractor turns off. Additionally, the electrical components are very small which allows the system to fit in a small enclosure. Finally, the system requires little power; a power consumption study was conducted and it only consumed 22.8 watts at peak operation. Each of these benefits fit the customer requirements and engineering specifications.

4.2 Software Design

The software design changed many times as the project progressed. The final product utilizes several key files working together to capture images, save them, and manage powering off the Jetson Nano. Upon start up, the linux kernel runs a service file. The service file starts an initialization file. The initialization file handles input and output detection, and runs the camera program as a subprocess. The main file is what handles capturing and saving the images. Within the main file, are several imported classes and libraries to assist in capturing images. The file names are as follows:

- mainprogram.service
- init.py
- main.py
- flirCap.py
- colorCap.py
- datalog.py

A diagram representing the code hierarchy can be seen in Appendix F, and the code can be seen in Appendix G. The main program works by putting each camera in a thread, and putting that thread in an infinite loop. That loop captures an image every five seconds and stores it. Upon storing the image, the datalog associated with each camera is incremented. This allows for the system to keep track of the name of each image which prevents it from overwriting images. Finally, upon detection of power loss, the init.py script will send a termination signal to the main program. Once it is safe to exit, the program will turn off the Jetson Nano. This prevents the Jetson from killing the uninterruptible power supply battery.

4.3 Enclosure Design

In order to ensure electronics hardware associated with the data collection system is properly

protected and controlled, an enclosure was designed to help secure all componentry and provide a sealed, clean environment for sensitive hardware. Because the data collection system will serve as a standalone prototype, the electronics enclosure is manufactured using 3D printing technology.

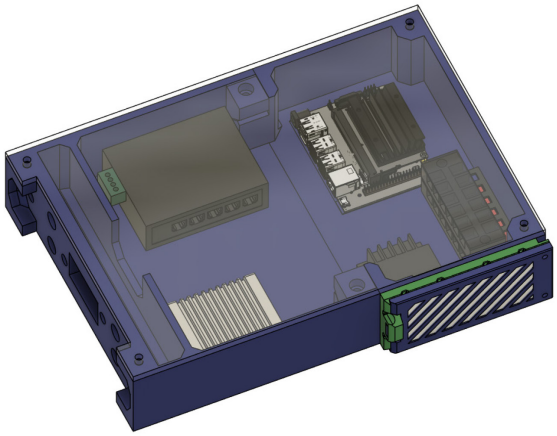


Figure 4.1: CAD Design of the Data Collection System Hardware Enclosure

Before a design for the enclosure was determined, the team deliberated on where to locate the enclosure mounting hardware such that it would remain stable and out of the way of the operator. The team decided to utilize the overhead radio deck opening for mounting the enclosure. Based on recommendations from ODOT to avoid drilling holes where possible, the team designed a mounting bracket that would provide a rigid connection to the unused radio deck directly in front of the operator. Four bolts threaded into steel tabs were used to lock the bracket into place, clamping onto the radio deck and allowing for seamless installation of the hardware enclosure, without the need for drilling through existing panels.

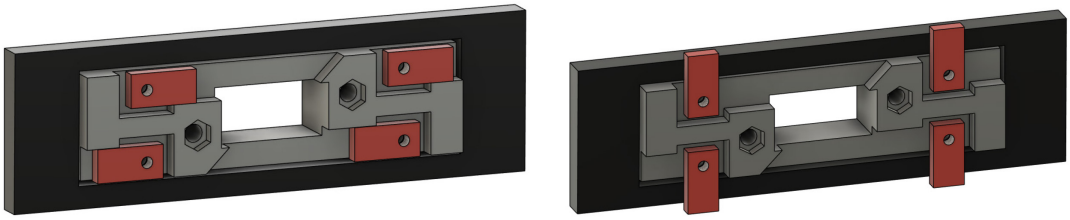


Figure 4.2: CAD Design of the radio deck mounting bracket, with clamps disengaged (left) and engaged (right)

After determining suitable locations for each component of the system, the hardware enclosure was designed and 3D printed using a carbon fiber PETG filament due to the material's suitable temperature resistance and dimensional stability. The team settled on a two piece enclosure design with an acrylic lid, which bolts directly up to the radio deck mounting bracket.

Figure 4.3: Prototype hardware enclosure, mounted to radio deck bracket

Although this design proved to be relatively sturdy once installed, ODOT's maintenance crew raised concerns over the placement directly above and in front of the driver's seat, as it obstructed the operator's view. Agreement between the team and ODOT Engineers to relocate the box was reached soon after, and the hardware will be placed in a small overhead storage compartment adjacent to the operator. The mounting bracket intended for mounting the hardware enclosure to the radio deck will be evaluated for securing a display/monitor at a later time, when ODOT is ready to develop a finalized Hazard Detection System and such equipment is deemed necessary.

Figure 4.4: Overhead storage compartment for relocation of hardware (compartment removed)

4.4 Camera Mounting Design

When mounting the cameras to the tractor it is important that there is full coverage and line of sight to areas that are common areas as discussed by operators that are frequently damaged or prone to oversight by operators that may cause risks. To determine this an investigation was done to determine placement of these cameras.

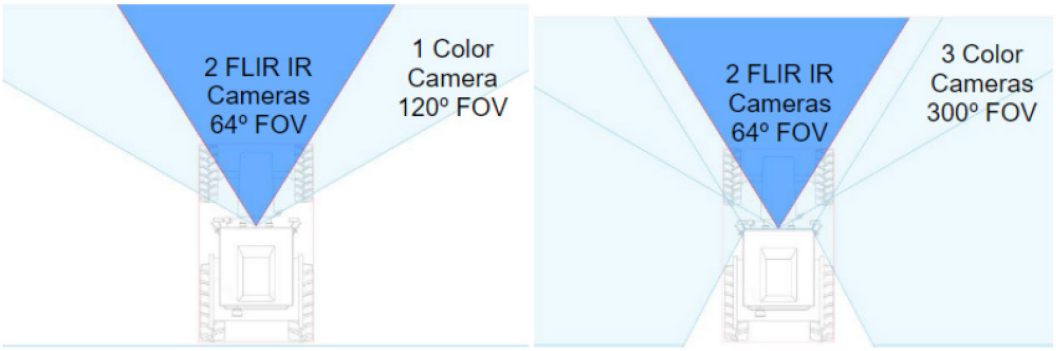


Figure 4.5: Coverage of Designated Cameras around Tractor

With the investigation of the mounting placements and coverage needed on the tractor, the finalized designs for the system could be implemented.

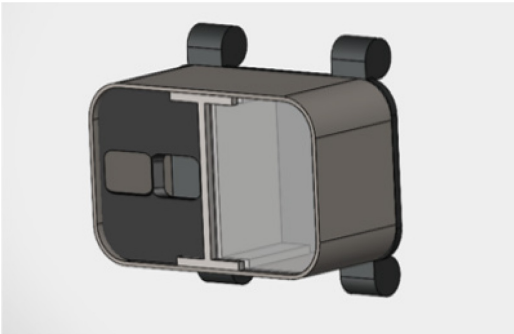


Figure 4.6: Finalized Front Camera Mount

The front facing camera was designed by the same principle of the first iteration, however, an easy way to transfer the mounting fixture for the top headlights of the tractor to below the camera mount was printed from a solid infill of simple PLA plastic and was manufactured to fit the fixtureing for the cameras.

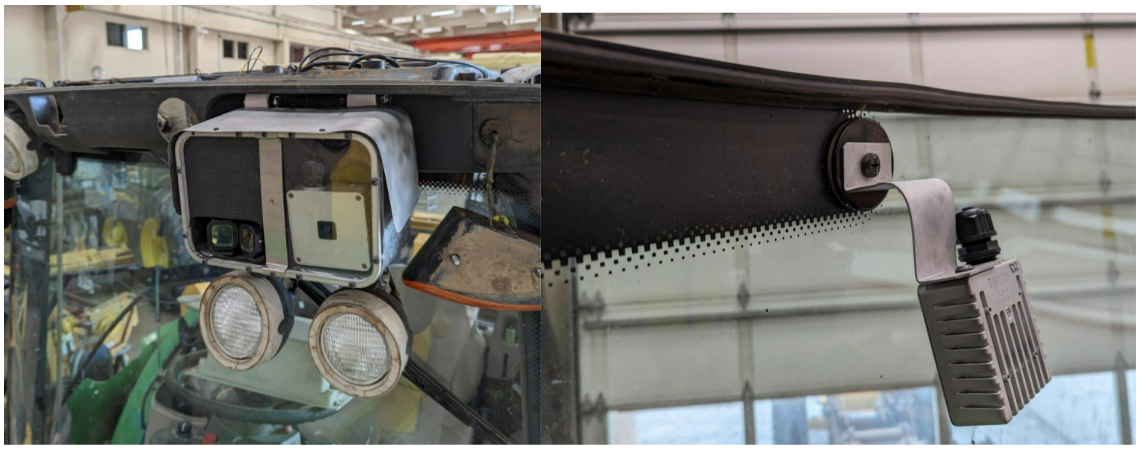


Figure 4.7: Front Facing Cameras and Figure 4.8: Side Mounted Cameras

The general assemblies and prototype was manufactured to use relevant hardware and mounting locations to mount the cameras in a way that can be easily edited and removed for replacements or iterations so the development of the system can be streamlined. However, due to concerns in

line of sight being slightly obscured by the operator and the mounting of the headlights to a non tested fixture point, the team edited it's mounting locations for the front facing cameras to inside the cowling of the tractor. This drastically deviated from one of the initial core customer requirements stated by the sponsor, however, understanding that feelings change and functionality is the prime motivator for this system, the team edited its mounting strategies and worked to implement a system that would work as displayed in **Figure 4.9**.

Figure 4.9: New Front Facing Camera Locations

4.5 Project Results

From the thorough design iterations, open communication, and the support of OSU, ODOT, and instructors and fellow students. The project resulted in a success in terms of the system build and implementation to begin collecting data for future iterations of the system. One of the stretch goals for the system was to collect data and start on the necessary steps needed to start training the machine learning algorithm to aid in future development of the project, but the completion of the system as it stands to start taking pictures during the normal operation of the tractor will develop more data than what the team could generate with the little insight given by what the operators see on a day to day basis.

The team has successfully implemented a system that will turn on without operator involvement as they turn on the tractor that takes useful pictures to develop a large amount of data over its summer time operations. This data bank will aid in developing a machine learning model that can successfully identify hazards that may cause danger to the equipment, the operator, or the public.

As it stands, there has been design changes against the current mounting strategies for the cameras and the enclosure design, however, the team has put in a concerted effort to make sure that ODOT will have a working system ready for operation before they finish the maintenance of the test bed tractor that will be put into service for Spring of 2024 despite these changes and criticisms to the finalized designs.

5 LOOKING FORWARD

While the OSU Capstone team provided a functioning data collection system for ODOT to use throughout the 2024 highway maintenance season, there are several areas of the design that will need to be re-evaluated and possibly adjusted before implementation. The design team has highlighted several key design considerations that will help ODOT maximize the effectiveness of the data collection system during operation.

Mounting of the hardware enclosure originally designated to utilize the radio deck in front/above the operator was deemed to be too much of a visual obstruction, as well as a potential danger regarding head injury if struck against the sharp corners of the enclosure. Because of this, the Capstone team discussed relocation of the hardware to a built-in compartment adjacent to and above the driver's seat, which will be executed by Christhian Galves of ODOT. The original enclosure will be used as a layout regarding bolt patterns of hardware, and will help assist in the relocation of all necessary electronics hardware. For future use regarding a finalized HMS, the Capstone team found that a more modular mounting system should be investigated, since the storage compartment that will be used to relocate the electronics varies between model years of ODOT's tractors. Design cues from the original enclosure such as wiring layout and air filter design will likely remain the same as the designs used for the original box.

The Camera mounting bracket mounted on the front of the Tractor cab was also deemed by maintenance to obstruct the operator's view, and relocation of the two flir cameras and the front-facing color camera has already been completed by the Capstone team. Old mounting hardware from the previous team was used to place the cameras behind existing cowling which had been previously modified for the cameras. The OSU Capstone Design team believes that while this solution will suffice for the data collection system, a lower-profile mounting bracket that does not require cutting or drilling plastic shrouds on the tractor may be more suitable for a larger-scale installation project once ODOT is ready to manufacture a final object detection system. The side mounted cameras were deemed to be sufficient in their current locations, and will remain mounted to the top of each cab door.

Performance of the cameras will need to be evaluated once the tractor equipped with the data collection system is operational, which ODOT anticipates will be around the end of March/beginning of April 2024. Additional vibration damping solutions may need to be considered to maximize image quality if deemed necessary, as well as solutions for automatically cleaning dust/chaff buildup on the externally mounted cameras and cab windows. The data collection system enclosure is currently designed to use a filtered 4010 cooling fan, which may also need to be evaluated for performance during heat wave conditions in the warmer summer months. Due to the high cost of the FLIR infrared cameras used in this system, the Capstone team recommends an in-depth investigation into the benefit of the infrared cameras, and whether these will be necessary for further development of a hazard mitigation system. Other comparable infrared camera solutions should also be researched, to help minimize the total cost of a fleetwide installation of Hazard Mitigation Systems in the future.

6 CONCLUSIONS

After an extensive review of the previous HMS design and the iterative design carried out by the Oregon State University Multidisciplinary Capstone Design Team, and the build and implementation of the system being installed onto field equipment, the team has demonstrated a successful test bed to develop the hazard mitigation system in its totality. Although there were some last minute design iterations due to concerns from maintenance personnel and ODOT engineers, the team took it in stride and redesigned our implementation of the system due to those concerns and successfully set ODOT up for a successful run for the upcoming operating season.

Safety was the first and foremost priority of this system as it is a system to increase the safety for the operator, the equipment, and the public. If our system kept the operator from successfully operating and doing their duties in the safest means necessary, then we had to pivot and make those changes and reiterate to make sure that our approach was keeping that in mind.

Designing in a multidisciplinary capacity, combining electrical, mechanical, and software solutions to create a system that would function properly, safely, reliably, and intuitively, the team was able to use engineering principles to ensure that our electrical systems would be low power and would not breach any kind of consumer code as well as making sure that hardware and mounting was properly calculated to have a sufficient safety factor as to ensure the system would not break or become a hazard to the equipment or the operator under normal operating conditions.

The Hazard Mitigation System has been installed onto a field tractor in an ODOT maintenance yard and is preparing for operation. The team has successfully implemented this system and will be on standby to hear any news or improvements for the system as it is developed.

7 REFERENCES

[1] “Home,” Oregon Department of Transportation : Home : State of Oregon, https://www.oregon.gov/odot/pages/index.aspx (accessed Mar. 10, 2024).

[2] “College of Engineering,” Oregon State University College of Engineering, https://engineering.oregonstate.edu/ (accessed Mar. 10, 2024).

[3] “Home,” ABET, https://www.abet.org/ (accessed Mar. 10, 2024).

[4] “Flir ADKTM,” FLIR ADK Thermal Vision Automotive Development Kit | Teledyne FLIR, https://www.flir.com/products/adk/?vertical=automotive&segment=oem (accessed Mar. 10, 2024).

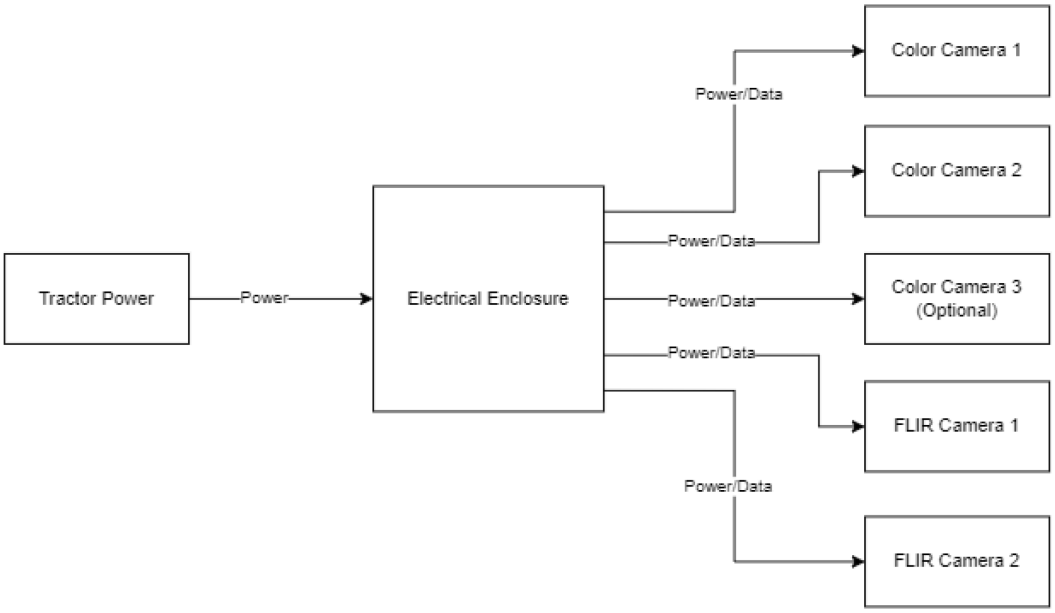
[5] “Oak-1 W Poe,” Luxonis, https://shop.luxonis.com/products/oak-1-w-poe?variant=43854167048415 (accessed Mar. 10, 2024).

[6] “Buy the latest Jetson Products,” NVIDIA Developer, https://developer.nvidia.com/buy-jetson (accessed Mar. 10, 2024).

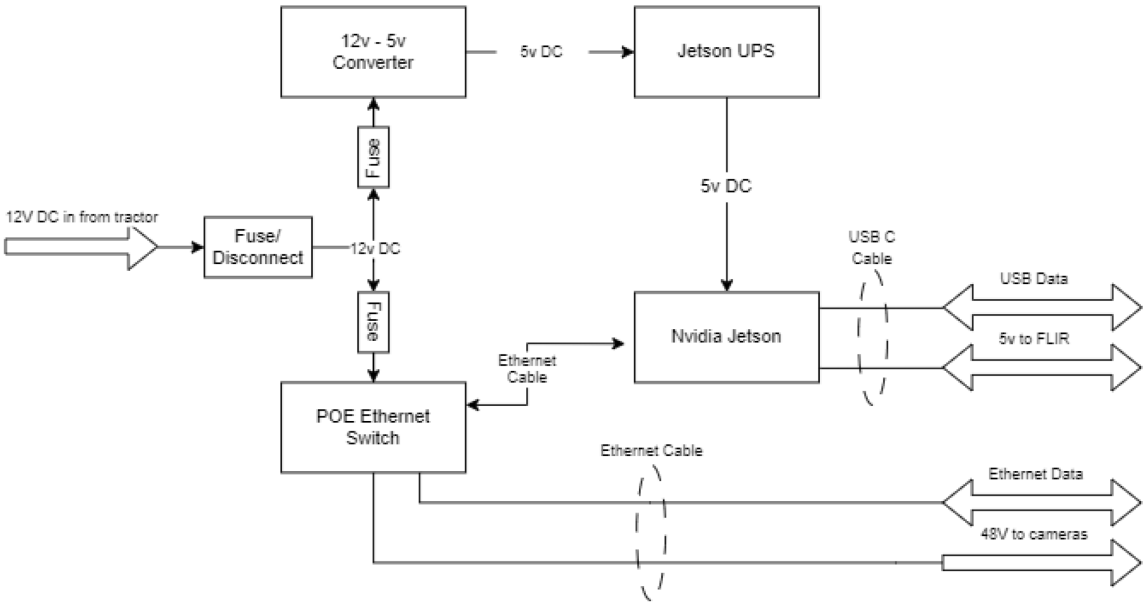
8 APPENDICES

8.1 Appendix A: Electrical Block Diagram

System Level Diagram



Electrical Enclosure Diagram



8.2 Appendix B: Bill of Materials

Bill of Materials (Electrical)

Item	Quantity	Price Each	Total Price	Came with Previous System
NVIDIA Jetson Nano	1	$149.00	$149.00	No
NVIDIA Jetson Nano UPS	1	$27.99	$27.99	No
2TB SD Card	1	$30.00	$30.00	No
TRENDnet 5-Port Industrial Switch	1	$105.95	$105.95	No
DC Voltage Regulator 12-5v	1	$12.99	$12.99	No
DC Voltage Regulator 12-48v	1	$39.99	$39.99	No
ZOOKOTO 12V 4 Way Car Auto Blade Fuse Box	1	$10.99	$10.99	No
Joinfworld 35A Terminal Block	1	$8.99	$8.99	No
FLIR Boson ADK 32 degree FOV	2	$3469.00	$6938.00	Yes
OAK-1 W PoE	3	$349.00	$1047	Yes/ Require 3 More for Full System Coverage

Total: $8370.90

8.3 Appendix C: Pugh Charts

Key

Symbols	Relationshipo	Value
+	Better than baseline	1
0	About the same	0
−	Worse than baseline	-1

Guide:

1. Before you start, collecto the two sets of data.
2. Insert the criteria on the left hand column.
3. Insert the alternatives on the top row.
4. Work through the matrix and indicate how the criteria will affect the various alternatives being considered.
5. Review the completed matrix to make the best decision for your situation.

Note: you need only to fill the white, blue and yellow cells.

Problem/Situation: Mounting locations for the electronic assembly

		Alternatives						
Criteria	**Direct From Fusebox**	**12v Accessory Outlet**	**Serpentine Belt Driven**	**Solar Panel**	**Kinetic Generator**	**N/A**	**Totals**	**Rank**
Continous Power	0	**0**	**0**	**–**	**0**		-1	6
Simple Installation	0	**+**	**–**	**–**	**–**		-2	8
Cheap	0	**0**	**–**	**–**	**–**		-3	9
Resilient	0	**0**	**0**	**–**	**0**		-1	6
	0						0	
	0						0	
	0						0	
	0						0	
	0						0	
	Totals	1	-2	-4	-2			
	Rank	1	2	4	2			

Problem/Situation: Electronics enclosure cooling system

		Alternatives						
Criteria	**No Cooling System**	**Computer Case Fan**	**Air-Air Heat Exchanger**	**Air-Liquid Heat Exchanger**	**Refrigerent System**	**Closed Loop Liquid**	**Totals**	**Rank**
Maintains air temperature of < 140 F	0	+	+	+	+	+	5	1
Cheap	0	0	−	−	−	−	-4	7
Easy to install	0	0	−	−	−	−	-4	7
Low Poer	0	0	−	−	0	0	-2	6
Prevents Dust	0	−	0	0	0	0	-1	5
Resilient	0	0	−	−	−	−	-4	7
	0						0	
	0						0	
	0						0	
	Totals	1	-2	-4	-2			
	Rank	1	2	4	2			

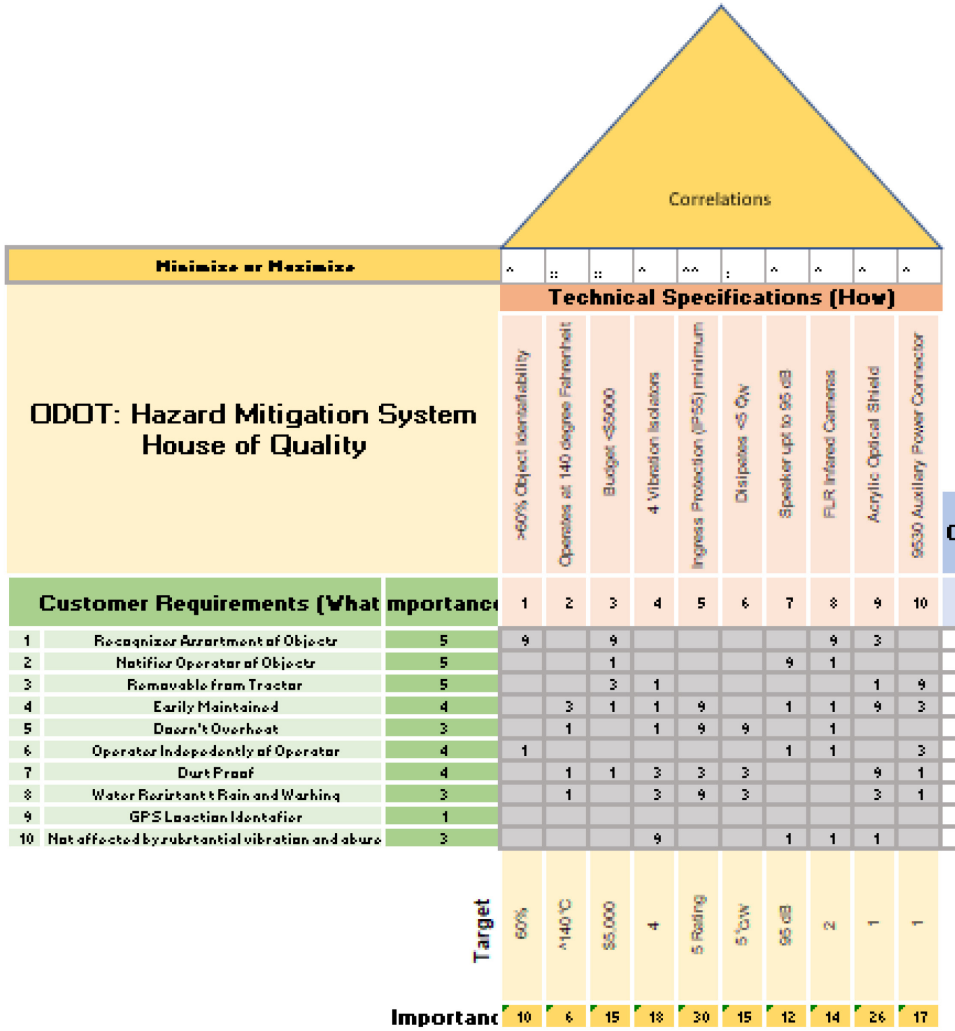
Problem/Situation: Data storage hardware

		Alternatives						
Criteria	**External Hard Drive**	**USB Data Storage**	**Solid-State Drive**	**Auxiliary Computer**	**N/A**	**N/A**	**Totals**	**Rank**
Must be able to store >2TB of data	0	0	0	+			5	1
Cheap	0	+	−	−			-4	7
Resilient	0	0	0	−			-4	7
Store data for a long period of time	0	+	−	−			-2	6
Small size	0	+	+	−			-1	5
Fast	0	0	+	0			-4	7
	0						0	
	0						0	
	0						0	
	Totals	1	-2	-4	-2			
	Rank	1	2	4	2			

Problem/Situation: Mounting locations for the electronic assembly

		Alternatives						
Criteria	**Behind Seat**	**Tractor Roof**	**Adjacent to Steering Wheel**	**Outside Cab Window**	**Under-carriage**	**N/A**	**Totals**	**Rank**
Protects assembly from being crushed	0	+	+	+	−		3	1
Protects assembly from dust	0	−	0	−	−		1	3
Protects assembly from heat	0	+	0	+	+		3	1
Easy access	0	0	+	+	−		1	3
Easy to build mount	0	0	0	0	−		3	1
Out of operators way	0	+	−	−	+		1	3
Close to camera system	0	+	0	+	−		3	1
	0						1	3
	0						3	1
	Totals	3	1	2	-3			
	Rank	1	3	2	4			

8.4 Appendix D: House of Quality



House of Quality (graphic)

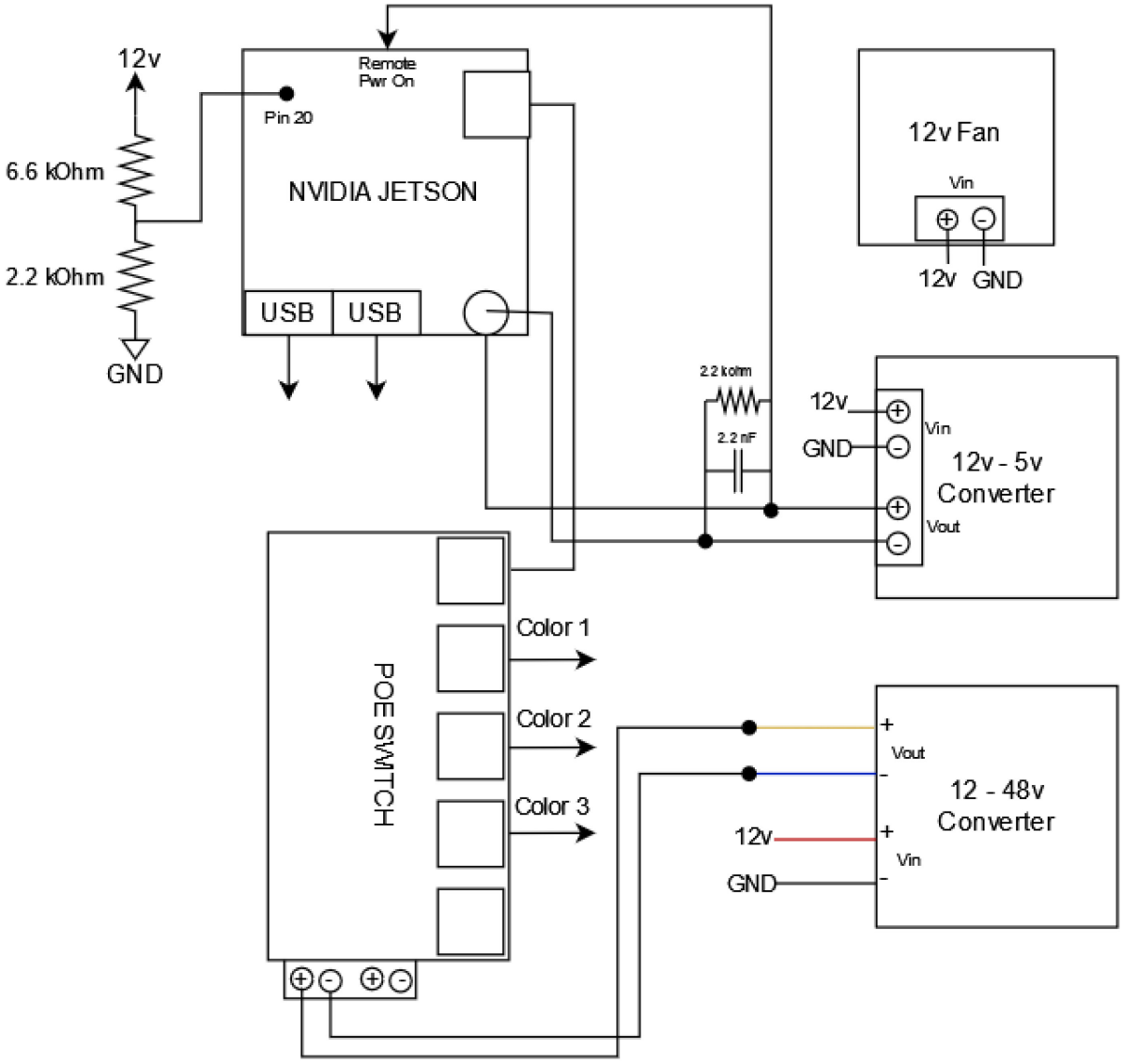
House of Quality (table)

			Correlations													
	Minimize or Maximize		^	::	::	^	^^	:	^	^	^	^				
	ODOT: Hazard Mitigation System House of Quality		**Technical Specifications (How)**													
			>60% Object Identafiability	Operates at 140 degree Fahrenheit	Budget <$5000	4 Vibration Isolators	Ingress Protection (IP55)	Disipates <5 C°/W	Speaker upt to 95 dB	FLR Infared Cameras	Acrylic Optical Shield	9530 Auxilary Power Connector	**Competitive Assessment**			
	Customer Requirements (What)	**Importance**	1	2	3	4	5	6	7	8	9	10	1	2	3	4
1	Recognizes Assortment of Objects	5	9		9					9	3		3	2	5	4
2	Notifies Operator of Objects	5			1				9	1			5	5	5	5
3	Removable from Tractor	5			3	1					1	9	1	1	1	1
4	Easily Maintained	4		3	1	1	9		1	1	9	3	2	3	5	1
5	Doesn't Overheat	3		1		1	9	9		1			1	1	1	1
6	Operates Indepedently of Operator	4	1						1	1		3	5	4	3	4
7	Dust Proof	4		1	1	3	3	3			9	1	1	1	2	1
8	Water Resistant t Rain and Washing	3		1		3	9	3			3	1	1	1	1	1
9	GPS Loaction Identafier	1											1	1	2	5
10	Not affected by substantial vibration and abuse	3				9			1	1	1		1	1	1	1
		Target	60%	^140 °C	$5,000	4	5 Rating	5 °C/W	95 dB	2	1	1				
	Importance		10	6	15	18	30	15	12	14	26	17				

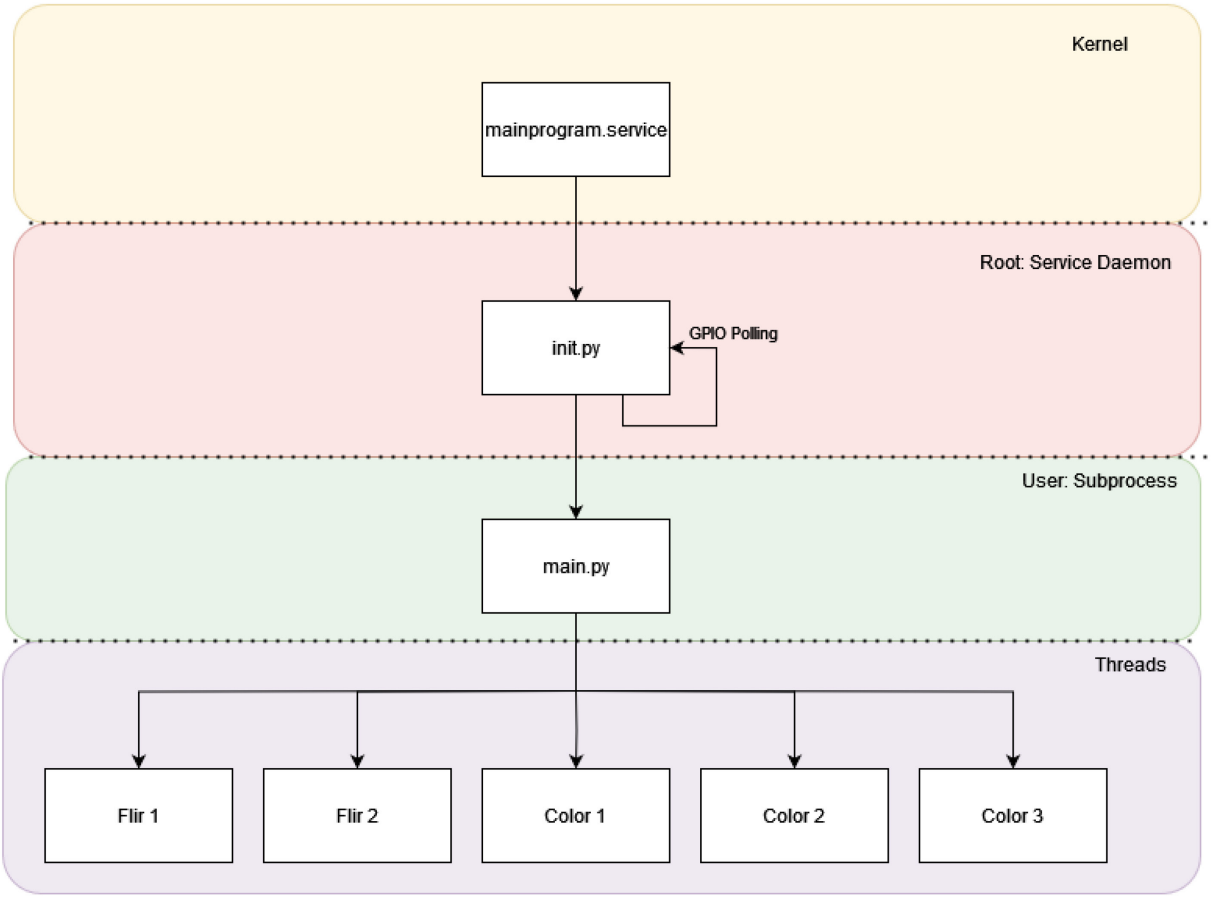
Keys:
Correlations:
^^ - Strong Positive
^ - Positive
:: - Strong Negative
: - Negative

Relationships:
Strong = 9
Medium = 3
Weak = 1

8.5 Appendix E: Electrical Wiring Diagram



8.6 Appendix D: Software Flow Chart



8.7 Appendix F: Software

8.7.1 mainprogram.service

```
[Unit]
Description=Starts main program on startup
After=network.target

[Service]
Type=oneshot
User=root
ExecStart=/usr/bin/python3 /home/beaver1/init.py
[Install]
WantedBy = multi-user.target
```

8.7.2 init.py

```
#!/usr/bin/python3

import Jetson.GPIO as gpio
import time
import sys
import subprocess
import os

venv = '/home/beaver1/.virtualenvs/depthAI/bin/python3'
script = '/home/beaver1/MAIN_PROGRAM/main.py'
args = [venv, script]
time.sleep(20)
p1 = subprocess.Popen(args)
print("Script started...")

input_pin = 20
def main():
    input_cntr = 0
    gpio.setmode(gpio.BCM)

    gpio.setup(input_pin, gpio.IN)

    print("Start Demo Now")
    try:
        while True:
            if (input_cntr == 2):
                print("Power Loss Detected!")
                p1.terminate()
                os.system("shutdown now-h")
```

```
        elif (gpio.input(input_pin) == 0):
            input_cntr = input_cntr + 1
        else:
            input_cntr = 0
            time.sleep(0.5)
    finally:
        gpio.cleanup()
        print("Program Exit")

main()
```

8.7.3 main.py

```
#!/home/beaver1/.virtualenvs/depthAI/bin/python3

import colorCap as cCap
import flirCap as fCap
import time
import threading
import datalog as log
import depthai as dai

#Important variables - Note: The flir ports like to randomly change
name1 = 'flir1'
name2 = 'color1'
name3 = 'flir2'
name4 = 'color2'
name5 = 'color3'

path1 = '/home/beaver1/MAIN_PROGRAM/data/images/flir1'
path2 = '/home/beaver1/MAIN_PROGRAM/data/images/color1'
path3 = '/home/beaver1/MAIN_PROGRAM/data/images/flir2'
path4 = '/home/beaver1/MAIN_PROGRAM/data/images/color2'
path5 = '/home/beaver1/MAIN_PROGRAM/data/images/color3'
logpath = '/home/beaver1/MAIN_PROGRAM/data/datalogs'

port1 = 0
port2 = "169.254.1.222"
port3 = 1
port4 = "169.254.1.221"
port5 = "169.254.1.223"

#Creates an instance of each datalog, flir camera, and color camera
flir1 = fCap.FlirCapture(name1, path1, port1)
flir2 = fCap.FlirCapture(name3, path3, port3)
```

```
color1 = cCap.ColorCapture(name2, path2, port2)
color2 = cCap.ColorCapture(name4, path4, port4)
color3 = cCap.ColorCapture(name5, path5, port5)

data_flir1 = log.Datalog(logpath, name1)
data_flir2 = log.Datalog(logpath, name3)
data_color1 = log.Datalog(logpath, name2)
data_color2 = log.Datalog(logpath, name4)
data_color3 = log.Datalog(logpath, name5)

#Sync_int will count number of cameras and start capturing images. A watchdog timer may be
added to allow for
#capture when a camera fails
#Exit flag is used to kill the threads
sync_int = 0
exit_flag = False
num_of_cams = 5

#Main Flir loop - should add to class when refactor
#Inputs(flirCap, number of image from csv, data log instance)
def flirLoop(flir, num, log):
    #Read in sync bit and add 1 for the camera instance
    global sync_int
    global num_of_cams
    sync_int = sync_int + 1         #Adds a camera to sync bit
    #Loop for waiting to start
    while True:
        if sync_int == num_of_cams:
            print("Start loop!")
            break

    #Main Loop
    while True:
        if exit_flag == True:       #Checks exit bit
            break
        else:
            flir.saveStill(num)       #Saves image
            log.utable([str(num)])    #Logs image number
            log.wtable()
            print(flir.name + " Captured!")
            flirDiscard(flir)       #Starts discard loop(see read me)
            num = num + 1           #Increments image number

#See read me for detailed explination of WHY this is needed
def flirDiscard(flir):
    tic = time.perf_counter()       #Initial time
```

```
    timer = tic + 5                #Goal time
    while (tic < timer):
        flir.discardFrame()        #Discards frame
        tic = time.perf_counter()  #Updateds current time

#Function to read in image from color camera "message"
def captureImage(device, cam, num, log):
    try:
        img = device.getOutputQueue("still").get()
        cam.save(num, img)          #Saves image
        log.utable([str(num)])      #Logs image number
        log.wtable()
        print("Color Captured!")
    except:
        sys.exit("    Failed!")

#Main colorCamera loop
def colorLoop(cam, num, log):
    global sync_int
    with dai.Device(cam.pipeline, cam.device_info) as device:
        #Start sync loop
        sync_int = sync_int + 1
        while True:
            if sync_int == num_of_cams:
                print("Start loop!")
                break
        while True:
            if exit_flag == True:   #Checks exit flag
                break
            else:
                captureImage(device, cam, num, log)
                num = num + 1

#Reads in the next image number for the CSV
def readCSV(log):
    num = log.rIter() #log.rIter() returns an int
    num = num + 1
    return num

def cleanup():
    exit_flag = True
    #Wait for both threads to stop
    p1.join()
    p2.join()
    p3.join()
```

```
    #Print updated csv table and then write it to the csv
    print(data_color1.table)
    print(data_flir1.table)
    print(data_flir2.table)
    data_flir1.wtable()
    data_color1.wtable()
    data_flir2.wtable()
    print("   Done!")


flir1_init_num = readCSV(data_flir1)
flir2_init_num = readCSV(data_flir2)
color1_init_num = readCSV(data_color1)
color2_init_num = readCSV(data_color2)
color3_init_num = readCSV(data_color3)

#Main program
if __name__ =="__main__":
    try:
        #Creates threads for color and flir loops
        p1 = threading.Thread(target=flirLoop, args=(flir1,flir1_init_num, data_flir1))
        p2 = threading.Thread(target=colorLoop, args=(color1, color1_init_num, data_color1))
        p3 = threading.Thread(target=flirLoop, args=(flir2,flir2_init_num, data_flir2))
        p4 = threading.Thread(target=colorLoop, args=(color2, color2_init_num, data_color2))
        p5 = threading.Thread(target=colorLoop, args=(color3, color3_init_num, data_color3))
        #Start the threads
        p1.start()
        p2.start()
        p3.start()
        p4.start()
        p5.start()

        #Main loop -- This checks for a keyboard interrupt and changes the exit flag.
        while True:
            time.sleep(30)
            sync_int = num_of_cams

    except:
        cleanup()
        print("Finished")


safe_exit.register(cleanup
```

8.7.4 colorCap.py

```
#Contains a class that initiates a depthAI camera.
#Worked as of 12/24/23 - Dustin
#To Do: Add features if possible. Works in this state if multithreading is used to run multiple at one time.

import cv2
import depthai as dai
import os
import sys

#path = '/home/derickson/Pictures/Color_Images/Color1'
#name = "color1"

class ColorCapture:
    def __init__(self, name, path, ip):
        print("Color Camera Initilizing...")
        self.name = name
        self.path = path
        self.ip = ip

        # Start defining a pipeline
        self.pipeline = dai.Pipeline()

        # Define a source - color camera
        self.cam = self.pipeline.create(dai.node.ColorCamera)

        # Script node
        self.script = self.pipeline.create(dai.node.Script)
        self.script.setScript("""
        import time
        ctrl = CameraControl()
        ctrl.setCaptureStill(True)
        while True:
            time.sleep(5)
            node.io['out'].send(ctrl)
        """)

        # XLinkOut
        self.xout = self.pipeline.create(dai.node.XLinkOut)
        self.xout.setStreamName('still')

        # Connections
        self.script.outputs['out'].link(self.cam.inputControl)
        self.cam.still.link(self.xout.input)
```

```
        self.device_info = dai.DeviceInfo(self.ip)
        print("Color Camera Intilizied")
    def infLoop(self, num):
        # Connect to device with pipeline
        print("Entered method")
        with dai.Device(self.pipeline, self.device_info) as device:
            while True:
                print("Entered Loop")
                try:
                    img = device.getOutputQueue("still").get()
                    cv2.imwrite(os.path.join(self.path,self.name +'_' + str(num) + '.jpg'),
img.getCvFrame())
                    print("Color Captured!")
                    num+=1
                except KeyboardInterrupt:
                    sys.exit("    Interrupted!")

    def save(self, num, img):
        cv2.imwrite(os.path.join(self.path,self.name +'_' + str(num) + '.jpg'), img.getCvFrame())
```

8.7.5 flirCap.py

```
#Contains a class that initiates an openCV camera. Also has functions specifically for two
cameras.
#Error handling is used to allow one camera to work if one fails.
#Worked as of 12/22/23 - Dustin
#To Do: Continue testing and debugging
import cv2
import os
import time
import sys

class FlirCapture:
    def __init__(self, name, path, port):
        self.name = name
        self.path = path
        self.port = port
        self.videoCap = cv2.VideoCapture(self.port)

    #Show video method
    def showVideo(self):
        while(True):
            ret,frame = self.videoCap.read()
            cv2.imshow(self.name, frame)
            if(cv2.waitKey(1) & 0xFF == ord('q')):
```

```
            self.videoCap.release()
            cv2.destroyAllWindows()

    #Show image method
    def showStill(self):
        ret,frame = self.videoCap.read()
        cv2.imshow(self.name, frame)
        while(ret):
            if(cv2.waitKey(1) & 0xFF == ord('q')):
                cv2.destroyAllWindows()
                ret = False

    #Save image method(takes a number as input to iterate image names)
    def saveStill(self, num):
        ret,frame = self.videoCap.read()
        cv2.imwrite(os.path.join(self.path,self.name +'_' + str(num) + '.jpg'), frame)

    #Every frame must be read. Use this method to prevent frame buffering.
    def discardFrame(self):
        ret,frame = self.videoCap.read()

#-------------------------------------------------------------------------------#
#Functions for using two FLIR cameras

#Show two images at the same time
def show2(f1, f2):
    try:
        f1.showStill()
        f2.showStill()
    except KeyboardInterrupt:
        sys.exit("    Keyboard Interrupt")
    except:
        print("Failed to show 2 FLIR images")

#Save two images
def save2(f1, f2, x):
    try:
        f1.saveStill(x)
        f2.saveStill(x)
    except KeyboardInterrupt:
        sys.exit("    Keyboard Interrupt")
    except:
        print("Failed to save 2 FLIR images")

#Discard two images
def dis2(f1, f2, deltaT):
```

```
    try:
        tic = time.perf_counter()
        timer = tic + deltaT
        while (tic < timer):
            f1.discardFrame()
            f2.discardFrame()
            tic = time.perf_counter()
    except KeyboardInterrupt:
        sys.exit("    Keyboard Interrupt")
    except:
        print("Failed to discard 2 FLIR images")

#Counting loop that takes two cameras, an amount of time, and a number of images as input
def countloop(f1, f2, iter, time):
    for i in range(iter):
        save2(f1, f2, i)
        dis2(f1, f2, time)

#Infinite loop that takes two cameras and an amount of time as input
def infloop(f1,f2, time):
        i = 0
        while True:
            save2(f1, f2, i)
            print("Captured!")
            dis2(f1, f2, time)
            i = i + 1
```

8.7.6 datalog.py

```
import os
import csv

class Datalog:
        def __init__(self, path, name):
        self.path = path
        self.filename = os.path.join(self.path, name + ".csv")
        self.table = self.rtable()

        def rfields(self):
        fields = self.table[0]
        return fields

        def rtable(self):
        with open(self.filename, 'r') as file:
        csvFile = csv.reader(file)
```

```
table = []
for lines in csvFile:
        table.append(lines)
return table

def rIter(self):
table = self.table
if len(table)==1:
return 0
else:
lastRow = table[len(table)-1]
iter = int(lastRow[0])
return iter

def utable(self, data):
table = self.table

table.append(data)

self.table = table

def wtable(self):
with open(self.filename, 'w') as csvfile:
# creating a csv writer object
csvwriter = csv.writer(csvfile)
# writing the data rows
csvwriter.writerows(self.table)

def clear(self):
header = self.rfields()
with open(self.filename, 'w') as csvfile:
csvwriter = csv.writer(csvfile)
csvwriter.writerow(header)
self.rtable()

def print(self):
for row in self.table:
print(row)
```