Thermal Anomaly Detection using Computer Vision for Search and Rescue Drones

BME / ECE 499

August 5, 2024

University of Victoria

Group ID: 3

Faculty supervisor: Lin Cai

Co-supervisor (if any):

Team Information:

S. No.	Name	V Number
	Calum Clark	V00954036
	Vaishakh Vinod Menon	V00939019
	Joshua Ruiz Nowell	V00968407
	Swetha Anand	V00959863
	Kumudu Perera	V00932926

Acknowledgements

We would like to acknowledge and thank our project supervisor, Dr. Lin Cai, the Electrical and Computer Engineering Technical Staff, as well as our TA, and the CEWIL Canada organization for funding the project. We would also like to thank Bryan Bradford and Metchosin Search and Rescue for their ideas and support in obtaining videos to help train our computer vision model. Finally, we would like to thank Sana Shuja for her contributions to the ECE 499 class.

Table of Contents

Executive Summary

Search and Rescue operations rely heavily on human input in emergency situations. They use people on foot, sometimes trained animals and human operated drones. These methods can only cover so much distance and are prone to human error, resulting in missing people that might otherwise have been found. Our project focuses on implementing a computer vision model to detect thermal anomalies from a drone camera. We use OpenHD software on Raspberry Pi devices, along with a yolov8 based computer vision model to detect anomalies that are likely to have been caused by people. This will allow both human operated and autonomous drones to detect missing people in conditions that might otherwise be inaccessible or locations that could be easily missed. Over the course of this project, we were able to build a functioning video transmission system and computer vision model that can detect people with high accuracy. Integration of these systems was more complex than originally thought, and will require further development.

Introduction

The need for advanced search and rescue technology is increasing from limitations of current technology capabilities in remote and environmentally challenging locations. The objective of this project will meet the needs by creating a long range thermal video transmission and detection attachment for search and rescue drones. This system will utilize thermal cameras and a computer vision machine learning model programmed to detect humans. By using the effectiveness of thermal cameras and computer vision methods, our project aims to improve the accuracy and safety of current search and rescue missions.

The scope of the project includes transmitting thermal video data over a distance of three kilometers and processing the model's output to detect humans within one hour of the drone's return. This system is split into two elements: the hardware telemetry unit, and the machine learning computer vision model for human detection.

The following regulations are outlined by the Engineers and Geoscientists BC (EGBC) Code of Ethics, as referenced in **Appendix A**, that are directly applicable for this project [1]:

Protection of the Public: The aim of this project is to aid a team in search and rescue operations which directly helps to the safety and well-being of the public
Integrity: Maintaining transparency throughout the project, especially in terms of the system's capabilities and restrictions, need to be clearly communicated.
Accountability: By building a system that is built for the projection of the public, engineers need to be responsible for the reliability and accuracy of the system.

This project solution is intended to be used by search and rescue teams, which includes first responders, and disaster recovery management companies. The solution is designed for use in remote areas where there is low cellular service. To overcome this issue we focused on using a portable solution that is able to operate without using any external WiFi networks or service towers for human detection and location.

Objectives

The main objectives of the project are:

- Create and implement a video and data transmission system over ~3km
- Build, train and implement a computer vision model for detecting humans from thermal cameras
- Output images of human detection to some base station interface

- Integrate the video telemetry features with the computer vision functions
- Design the attachment to be lightweight and portable as to ensure the drones flight is not impacted

Design Specifications

The following section focuses on the design specifications needed to ensure the system met performance, reliability and useability requirements, while adhering to industry standards.

Data Transmission

The data transmission system of the drone attachment was designed with four main criteria. The range of transmission, the power requirements for proper transmission, the size of the attachment for the air and the ground unit, and the quality of the transmitted video, including data loss and transmission latency.

Firstly, the transmission system was designed for a range of 5 kilometers with a minimum range of 3km. This range allows for the drone and its operator to cover a large area in remote locations and still be able to communicate data and footage with the base station. It also helps to keep a large enough useful range in case of poor operating conditions like high winds, rain and with land obstructions. This was decided as an initial criteria and provided by the search and rescue organization to provide a large enough useful search grid.

Next, the power requirements of the attachment is such that it needs to be powered off the battery of the drone itself, without causing a large impact on the flight time of the drone. Depending on the size and power requirements of the drone itself, this can range from a 5V supply to upwards of 22V. Therefore, we built a system that functions off the minimum of 5V, with attached regulators for higher voltages, along with a current draw below 600mA. The current draw was decided by finding the lowest possible current from different systems compatible with transmission software including the Raspberry Pi Zero 2W or a Raspberry CM4 with EMMC (embedded multimedia card) with an Ochin CM4 carrier board. [2]

Third, the size of the attachment was designed such that most drones employed by search and rescue would be able to carry it, with minimal impact to flight data such as speed and battery life. The only requirement for the size of the ground unit was that it be reasonably small enough to be portable, with the ability to display video. As such, the smallest equipment capable of transmission on the air unit was employed. In addition, an enclosure was designed for the air unit that is engineered to integrate a video and data transmission system capable of reaching a range of 5km. This enclosure was designed using SolidWorks as shown in **Figure 1** and is able to mount onto the drone that can be used. The design looks like a box with dimensions of 60mm x 70mm x 120mm which accommodates a thermal camera, the communication modules and the Raspberry Pi Zero 2W. The following enclosure was 3d printed so that the unit is lightweight and has a durable structure as seen in **Figure 2**. At the same time we made sure to even the weight distribution to maintain a balanced and stable flight so that drag can be reduced. We added small ventilation slots to the enclosure so that it follows aerodynamic principles which ensures no additional drag will be created.

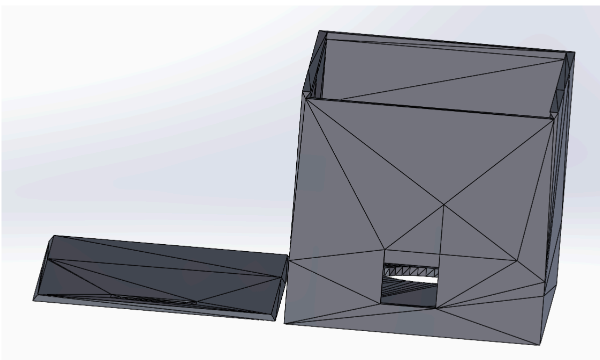

Figure 1: Solidworks Design

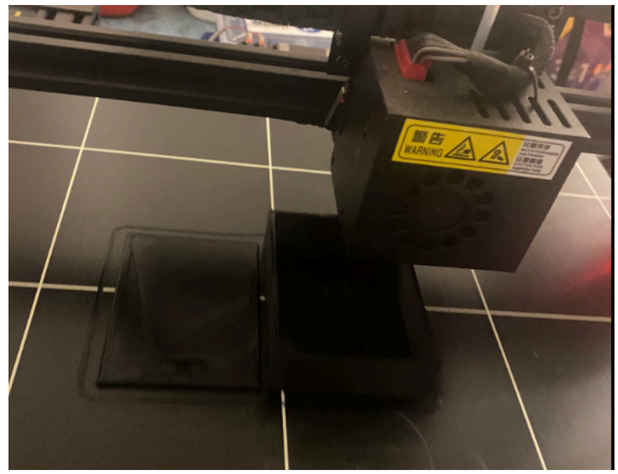

Figure 2: 3D Printed Model of the Enclosure

Fourth, the quality of video transmission needed to be high enough that the computer vision model would be able to identify humans, and the latency low enough that the location of people could still be easily found. It was determined that a total latency, including the computer vision analysis, of less than 5 seconds would be ideal, with minimal lag and a loss of less than 10% from 20m away in open area would be acceptable for the prototype.

Overall, the data transmission system needed to have a range of 5km in ideal conditions, capabilities to handle a variety of supply voltages with minimum current draw, a small volume and weight on the air unit with the ground unit displaying an output video. Finally, the transmission system needs to be capable of transmitting video with a latency of less than 1s and 10% loss from a distance of 20m. These specifications are for the design prototype, while further improvements will be made for a fully functional device being used in the field.

Computer Vision Model

The model was designed and trained to achieve a detection accuracy of over 85%. High accuracy allows for reliable detection of missing humans or those in distress. By being able to detect them more accurately, there is a much higher chance for the system to pinpoint their location. An accuracy of 85% or higher was chosen to benchmark the models accuracy metric because it ensures that the model can detect humans in challenging and remote locations while still being realistic to the systems efficiency. By aiming for high accuracy, we set the standard for the system's effectiveness to be increased, allowing it to maximize its usefulness in real world applications.

Another benchmark for the project was the need for the model to maintain high detection accuracy in noisy thermal images, which may include temperature variations, artifacts, and trees from the environment. This ensured accuracy and reliability were adhered to industry standards. This can be measured using metrics like precision which should be higher than 60% and a score of F1 and mean Average Precision (mAP) closer to 1.0 for better precision and recall [3] .

In terms of scalability, the model should be able to operate in different modes such as grayscale and color to accommodate various thermal camera outputs and enhance detection capabilities.

In terms of training and testing dataset, the optimal epoch is around 100-150 to train at. The model must also be trained and tested on a diverse dataset with different input modes to ensure better performance evaluation. As a result, the model must be trained to 100-150 epochs with each training set in 16 batches [3].

Social Impact

The goal of this project was to hopefully create a meaningful device or application to further advance and aid in a specific area of society. The project that was chosen was discussed not only among team members but with real industry workers to provide a solution that would impact society in a positive way

The thermal rescue drone is intended to benefit the search and rescue efforts in society to ultimately introduce an effective and practical way in locating and rescuing individuals that may find themselves in a dangerous situation.

Having a drone that can easily fly and survey a large portion of land while transmitting crucial data such as video and GPS location would help rescue teams immensely when attempting a rescue that is time critical or in a dense area. With the help of the thermal camera and machine learning model, identification of humans is made significantly easier. A human may miss important information in a video but with the help of a pre-trained model that can easily and accurately detect humans there would be less error in video analysis and in turn save human lives.

Literature Survey

The following section breaks down our project into two parts for analysis - a hardware and a software aspect.

Video and Data Transmission

The two aspects of data transmission were identified to be selecting the camera, and the method of transmission. The first thing to decide was the type of IR camera to be used, since the resolution and frame rate will impact the amount of data being sent and stored, as well as the time required for analysis. OpenHD is compatible with most thermal USB cameras, however, for the sake of this analysis, we will consider the FLIR One Pro, Infiray T2 Pro and Hti HT-301. The FLIR One Pro has a cost of $500 and resolution of 160x120 at 9Hz. It connects via USB-C and has a battery life of approximately an hour [4]. The Infiray T2 Pro is $470 with a resolution of 256x192 at 25Hz [5]. Finally, the Hti HT-301 is more expensive at about $960 with a resolution of 384x288 at 25Hz [6]. The Hti HT 301 and Infiray T2 Pro are both USB-C or micro-USB connections and can be powered directly by the connection. Given the above resolutions, the FLIR One Pro would produce a video size of 1.5GB, while the Infiray T2 Pro and Hti HT-301 would produce files of 6GB and 27GB respectively [5]. For thermal USB cameras, OpenHD is compatible with cameras exceeding 327 000 pixels at 60Hz, so is sufficient for all cameras.

As illustrated in Table 1, the criteria for table selection is limited to price, size, resolution and frame rate, and compatibility with OpenHD software. The price was given the most weight due to limited budget and resources. The size was then given a weighting due to the camera being airborne on the drone with a limited carrying capacity. Next was the resolution and frame rate. It is important to obtain high quality images, however, high resolution and frame rate correlate proportionally to increased analysis time. Finally, compatibility was given a low weighting as all selected cameras satisfied the requirements. Table 2 implements the criteria weightings for each camera with FLIR One Pro, Infiray T2 Pro and Hti HT-301 receiving ratings of 67, 80, and 61 respectively. This analysis indicates the Infiray T2 Pro is the selected camera.

Table 1: Criteria Weightings for Camera Analysis

Criteria	Weight	5	4	3	2	1
Price	35	<$500	<$600	<$700	<$900	>$900
Size	30	<1oz	<2oz	<3oz	<4oz	>4oz
Resolution/ Frame rate	25	>85000 30Hz	>85000 >20Hz	>75000 >10Hz	>75000 9Hz	<75000 9Hz
Compatibility	10	Works with open HD		No OpenHD, can save		No OpenHD, no saving

Table 2: Camera Analysis Based on Criteria

Criteria	FLIR One Pro	Infiray T2 Pro	Hti HT-301
Price	4	5	1
Size	4	3	4
Resolution/ Frame Rate	1	3	4
Compatibility	5	5	5
Total	67	80	61

Next, based on research, there are 3 possible solutions for transmitting data. The first is to save the data on an SD card and analyze it once the drone has returned to ground. The second method is to implement OpenHD, a video transmission software that modifies the configurations of wifi cards for long range video transmission. While other long range video transmission softwares exist, such as EZ-Wifibroadcast and WFB-NG, their ease of use and compatibility with updated Raspberry Pi's, which are needed for the analysis, was such that OpenHD is the only viable option [7]. The third method would be to access the video from the drone controller for analysis. However, we are attempting to make an attachment that can be implemented on a variety of search and rescue drones and therefore this solution would require more time and research than is possible in the scope of this project, not to mention the regulations involved. For the sake of this analysis, we will focus on the implementation of OpenHD and saving data until return to base. Given that OpenHD is able to transmit data over this distance, and video can then be analyzed on the ground, the only difference between the systems is that for OpenHD, analysis can be completed during flight time but it requires parts for a long distance connection. The cost of parts only needed for OpenHD includes Raspberry Pi Zero 2W ($15), 2 wifi dongles ($15 each), 2 5V BECs ($11 each). A Raspberry Pi 4B+ is used for the analysis, however, a laptop or computer would be required for analysis in either case, so we leave this out of the analysis. Despite the additional price for using Open HD of $67.00, in Search and Rescue situations, time is much more important so OpenHD is the obvious choice.

Video Analysis

Thermal imaging drones are increasingly becoming essential in search and rescue(SAR) operations due to their ability to detect human heat signatures in low visibility conditions. Artificial Intelligence (AI) on the other hand is used to significantly advance the current needs of SAR drones by increasing processing times, detecting human heat signatures and providing close to real-time data to rescue teams which is crucial in rescue missions. The integration of drones with AI eliminates the need for manual labor and can be used to analyze vast amounts of data in limited time instead.

AI-powered drones, such as the ones being developed at the University of Colorado-Boulder, are built to analyze patterns and report the data. The drones use deep learning algorithms for object detection in challenging environments such as forests [8]. Similarly, another AI model was developed by Jan-Hendrik Ewers at the University of Glasgow who used historical SAR data to create probability maps to improve the chances of finding a person. By analyzing artificial neural networks and comparing them to a viewer's mind, the model is replicated to autonomously perform like a human rescuer which ensures accuracy as well as speed [9]. This AI model, tested against traditional search patterns, showed that the AI driven pattern found the missing person 19% of the time, compared to the 8% traditionally. Although the two above show research teams and their progress, the statistics suggest a huge market for improvement.

Team Duties & Project Planning

All team members would contribute and meet to discuss project planning and research topics. Various team meetings were held to update on progress and any changes that needed to be done for the initial project road map.

Each team member had been assigned a critical deliverable that is essential for the project's success. The two main components that are essential for this project are the thermal video transmission and the computer vision model designed to detect humans. After the team had been divided into its two sub teams the deliverables were assigned as follows:

Calum: Focused on the design and implementation of the video transmission system. Performed research on the various types of video transmission software currently in use. Opted for using OpenHD software for the software and began research and purchasing of its components and implementation. Performed component assembly with soldering and got the basic OpenHD software running successfully with video transmission. Worked with Kumudu to find a way to modify the OpenHD code, allowing for the computer vision model to be implemented.

The main challenge we faced was the modification of OpenHD software. The code was structured such that detailed documentation was needed to properly be able to execute the code due to the convoluted nature of its dependencies and security protocols. Without such documentation, the modification of the runnable code proved to be impossible, leading to the eventual failure of integration between OpenHD, and the computer vision model. Another issue we had was the technical nature of the soldering and assembly. Due to the minuscule size of the solders in a USB port, the equipment had to be handled very carefully and repaired several times, which may have contributed to a decrease in the efficiency of the final prototype.

Kumudu: Worked with Calum and focused on the hardware and video transmission component, aiming to design and implement a system capable of covering at least 5km. We chose OpenHD as the main software as it would transmit high definition video over a long range. At the same time Kumudu assisted Calum with soldering the components together and getting them to work together. Another task that Kumudu focused on was building an enclosure for the air unit as it had to be attached to the drone. The box shaped design was created with an opening for the camera and a switch which was used to on/off the air unit.

Some difficulties that Kumudu came across was figuring out the material to build the enclosure. The design was printed using a 3D printer which did not use the most appropriate material to build the enclosure. At the same time some ventilation slots had to be added to the enclosure to reduce air resistance so that the drone would fly without any additional drag.

Josh: Started on the first implementation of the machine learning model. A custom dataset was needed to train a pre-made model using YOLOv8. Once the dataset was acquired it required setting up an environment where files and code could be shared across team members for easier access and collaboration. Initially a google drive was created to hold training and validation data sets as well as provide a location for testing videos and outputs. A GitHub repository was then created to allow for better version control and coding contributions.

One specific challenge faced was ensuring the dataset being used had split up the files correctly into training and validation categories with, but also maintained the corresponding naming of the labeled data text file and the image itself. Josh created a python script to effectively split the large amounts of data into correct locations while retaining the correct naming in the process. Once an initial model and a way to test results was created by Josh he was able to hand off the project to Swetha and Vaishakh to further improve the efficiency of the model. Research for implementing a live model detection application was then done and attempted to hopefully have detections done in real time. FInally the creation and population of the team website was made in GitHub pages using html and CSS.

Swetha: Worked with Josh and Vaishakh to build, test and implement the computer vision model. Upon Josh's initial code to use the YOLOv8 model. Vaishakh and I trained the model with the dataset found online using josh's code. We ran around 300 epochs in total to create a model with high accuracy and a up to standard PR curve. Once the dataset was sent from the company, we turned the footage into a dataset, custom labeled it, and used the dataset to train a new model. The new model was then tested on videos sent by the company as well as from youtube. Upon building our model, I updated the code to get metrics about the model's accuracy such as PR curves, box losses, precision, recall, false positive rate, and mAP.

Vaishakh: worked closely with Swetha to take the base model that Josh had created and enhance its ability in human detection through training and testing. Once the model was made, Vaishakh and Swetha worked collaboratively to find specific datasets that would output the best results with respect to the application of the model. After finding the first applicable dataset, Vaishakh was partnered with Swetha to train the model on this dataset to ensure accurate human detection. After testing the model on test footage found online, the challenge faced was to increase the accuracy of the model and remove false positives and double detections. By implementing Non-maximum suppression into the code, some double directions were removed helping to increase accuracy. To achieve maximum accuracy, Vaishakh worked with Swetha to take the footage sent by the partnered search and rescue company, Metchosin Search and Rescue, and use the footage to create a custom dataset. They labeled over 700 frames and trained the model again on the custom dataset to increase accuracy as it would be trained on footage that closely resembles the application it would face. At the end, final testing of the model revealed its metrics to show a precision of 98%.

Design Methodology & Analysis

The following section outlines the comprehensive approach taken to design, develop, and analyze our data transmission system and human detection AI model.

Data Transmission

The development of the data transmission and computer vision integration steps can be broken down into 4 design methodology phases.

1. Problem Definition

   We began by defining the main objective of the data transmission team, which was to be able to transmit video over long distances with low latency, and attempt the integration of video analysis. The scope was limited to transmitting in outdoor areas similar to those that might be encountered in search and rescue operations, and over distances shorter than 100m, due to the unavailability of drone equipment for testing and a large amount of interference in the city.

2. Research phase

   The next step involved performing research on the various types of video transmission currently being used in the market. It was found that there were a few different software companies, however, the only one with the sufficient range, quality, and within our budget was OpenHD software, which is based on Raspberry Pi operating systems. We also performed research on the types of equipment needed for this setup, the possible sizes and any additional requirements.

   The results of this methodology were the analysis of various applicable software sources based on criteria such as possible range, video quality, power requirements and user interface. Eventually, this led to a discussion and plans to implement the software. We also purchased the necessary components based on the analysis. One thing that could have been improved was further research into the actual code itself and how to implement changes into the software.

3. Design

   The design at this stage was performed mostly based on the research phase, with additional elements of SolidWorks for the enclosure, analysis of power requirements based on battery sizes and drawing of wiring diagrams for the setup. The results were detailed diagrams of the component assembly, along with a power analysis and minimum requirements for the drone battery.

4. Assembly and Integration

   The assembly phase was fully completed by following the diagram from above with soldering and other wiring techniques. Various fragile solder were reinforced with hot glue, and interference was minimized around the camera and wifi cards with copper tape. Overall, the assembly phase of the design was fairly quick, albeit with a few minor repairs.

   The integration process involved modifying the OpenHD code in an attempt to pipe OpenHD video through the computer vision model, and out to the OpenHD interface. This methodology of viewing the code and attempting to modify it into usable code was unsuccessful. The main reason was a lack of time, however, by performing more research and attempting to implement the new code earlier, we might have seen more success.

Computer Vision Model

Similar to data transmission, the computer vision model was broken down into 6 design steps.

In the first phase, we defined the problem, which is to develop an AI model that can detect humans in thermal images as close to real-time processing as possible. The scope, however, was limited to only analyzing images in forests and not fire. As a result, our following datasets would need to be tailored to humans in forests as well.

In the second phase, we collected our data and pre-processed it before implementing it into our code. The first step to this was to find data sets - we were able to find a dataset that had over 3000 labeled images. However, to increase accuracy of our model, we also created a custom dataset of over 700 images. The custom dataset needed to be converted from videos to a frame per second then labeled. Once the datasets were ready, we wrote code to convert them into grayscale. This made our model adaptable to any type of thermal input.

In our third phase, we compared various YOLO models for our base model. Through literature review through the progress report, we decided to use YOLOv8 as the base detection model. YOLOv8 was chosen as it is easy to integrate into computer vision projects and yields accurate results with high speed. YOLOv8 being the latest iteration, is known for its fast processing times which makes it ideal for applications with quick processing times such as search and rescue.

Once we had a database and a base model, we trained to make our first model as our fourth phase. The model was trained three times with 150 epochs each time for better accuracy and reliability. Each batch size should be 16 for maximum storage optimization. The model was trained on specific datasets that would be most comparable to the real world aspects the model would be applied in. The datasets had a range of humans filmed using thermal cameras on drones in different locations such as open fields, roads, dense forest areas which helped keep the model robust and ready for diverse locations. By utilizing custom datasets and pre-processing techniques such as Non-Maximum Suppression, the final prototype model could yield a precision of 98%.

Additionally, while training, we created benchmarks to measure the model's accuracy and to measure improvement from each round of training in our fifth phase. Using the TensorBoard library, the model was analyzed to find confusion matrices, PR curves, box losses, precision and recall. The same library was used to ensure the metrics were consistently being evaluated throughout the testing phase.

The objective of the deployment and testing phase is to test the model in real-world scenarios through the use of thermal cameras to quantify the models performance. The scope of this testing involves using footage from thermal cameras attached onto drones flying in locations such as dense forests and open fields. By running the model on the footage received, we can quickly quantify and analyze the models ability to detect humans form the footage.These results will provide the necessary information needed to make any needed adjustments to the model's code to perform better. Once the changes have been made, further testing on the model will be conducted again until the model reaches a level of acceptable precision and recall through more optimization.

Design & Prototype

The following section focuses on the final design of our prototype.

Data Transmission

The video transmission assembly implements OpenHD software to transmit video over long range. It has two main sections; the air and ground units. The air unit consists of a Raspberry Pi Zero 2W, GPS unit, RTL8812AU wifi chip and an Infiray T2 Search thermal camera. These components are connected through a USB Hub. The ground unit uses a Raspberry Pi 4B+, a wifi card with the RTL8812AU chip and a 7" display for outputting the resultant video.

OpenHD functions by capturing video from the camera on the air unit, transmitting it to the ground unit, then to the OpenHD app, and decoding and outputting the video. Transmission is done by reconfiguring the wifi cards to communicate on a 5.8GHz frequency which has less interference than the 5GHz band used for wifi transmissions.

Overall, the prototype for video transmission satisfies the base objectives for the project with video transmission capable of approximately 100m before video loss and latency becomes too high. For a functional device, further modifications are required in the design by reducing wiring, increasing the quality of the wifi card and usb hub, and increasing the efficiency of the antennae.

Computer Vision Model

Table 3 below compares the final prototype against our initial design specifications to evaluate project performance:

Table 3: Evaluation Metrics vs Final Performance

Criterion	Target	Purpose	Benchmark Rationale	Final Prototype
Detection Accuracy	Achieve detection accuracy of over 85%	Ensure reliable detection of missing humans or those in distress	An accuracy of 85% or higher ensures the model can detect humans in challenging locations while maintaining system efficiency	Achieved detection accuracy of 96%
Robustness to Noise	Maintain high detection accuracy in noisy thermal images	Ensure accuracy and reliability in varied environmental conditions	Precision should be higher than 60%, with an F1 score and mAP closer to 1.0	Precision: 98%, F1 score: 0.981, mAP: 0.993
Mode Compatibility	Ensure the model is scalable to operate in different modes	Accommodate various thermal camera outputs and enhance detection capabilities	Consistent performance across different modes (grayscale and color)	Our model converts the video to grayscale to analyze and outputs the video in the same mode as the input.

Optimal Epochs	Train the model for 100-150 epochs	Ensure the model is sufficiently trained without overfitting	Monitor training and validation loss over epochs to confirm optimal performance within 100-150 epochs	Trained for 150 epochs every round to obtain the most optimized model.
Batch Size	Use a batch size of 16	Balance training efficiency and memory usage	Stable and converging loss metrics	Batch size of 16 used
Diverse Dataset	Train and test the model on a diverse dataset	Ensure better performance evaluation and generalization	Consistent high accuracy and reliability	Trained and tested on diverse datasets - custom and pre-labeled datasets

Figure 3 shows a working model through a video from Metchosin Search and Rescue. The video was fed into our model and the output resulted in the following image:

Figure 3. Human Detection on Metchosin SAR Footage

In order to achieve the results above, we used YOLOv8 as our base model. The model is known for its exceptional performance in object detection tasks. Since the model uses a deep CNN as its backbone, its speed and accuracy makes it highly suitable for scenarios where quick decision-making is crucial. In order to train our model, we created a google drive folder with test videos, data sets, labels and a folder for output videos. We mounted our google drive to our google colab to access the google drive storage. Once the folder was accessed, we ran the code in **Appendix C** to access the Ultralytics library with the YOLO model for our base training.

Once we trained and tested the model, it created two files - best.ty and last.pt. Where, best.pt is the best model of all the epochs. Using our best.pt file, we then configured our code to analyze a video using the model.

Once the best model is saved, we can test the model using test footage. The code in **Appendix D** sets the path that the test footage should be saved in and reads from that directory to use the test footage. There is also a path specified to output the video after the model has been run on the footage. Then the code uses OpenCV to open the video file from the specified path and check it to see if the file exists and is accessible. The code then reads the video dimensions to create an object with the same dimensions to output the video by using OpenCV functions cv2.VideoCapture and cv2.VideoWriter.

The main preprocessing happens when each frame of the video is converted to grayscale for the model to run on and then back to its original format once outputting the test video. The model is processed on every frame and tries to detect humans within that frame. If the model analyzes a positive detection, it will output the bounding box coordinates, the confidence score, and class label as per YOLOv8's model standard output information. This information is used by the program to draw the corresponding bounding box on the frame, and annotate the class label only if the confidence score given by the model is above the threshold level specified in the script. This thresholding is used and necessary to increase the confidence that the detection is a true positive (correct human detection). Once all the frames have been run on the model, the video is then saved into the path specified earlier in the code.

Integration

The integration of the computer model into the Raspberry Pi itself was not completed due to the complexity of OpenHD code and the lack of time given for the project. However, to make the video process in real-time, we planned to configure a GStreamer pipeline to receive encoded RTP streams from the OpenHD air unit, decode the video, then re-encode it at the receiver's end to then proceed video directly to the AI model. We planned to then implement another pipeline to output the analyzed video to the OpenHD user interface. This process could have been automated to start and stop as new RTP streams become available, keeping video latency and processing times as low as possible. This stage of the project ended up being much more than was expected for the scope and timeframe of the project and never proceeded to the testing phase.

Testing & Validation

Data Transmission

The test plan for the video transmission section of the project involved testing transmission loss and delay for the system from various distances, in an open field and in forested area. It was found that for distances less than 20m, video latency was less than 0.5s, while loss was less than 5%. For distances of more than 50m, video latency was closer to 2s, with losses upwards of 10%. The objectives for the prototype were that for distances of 20m, the latency should be less than 5s with data loss under 10%. Both of these objectives were successfully met in the testing phase when supplied with the recommended Raspberry Pi 4B+ power supply for the ground unit and a standard 9V battery for the air unit. Figure 4 shows a test transmission from approximately 15m away with transmissions from air to ground at a speed of 4.7MBit/s with a loss of 0%.

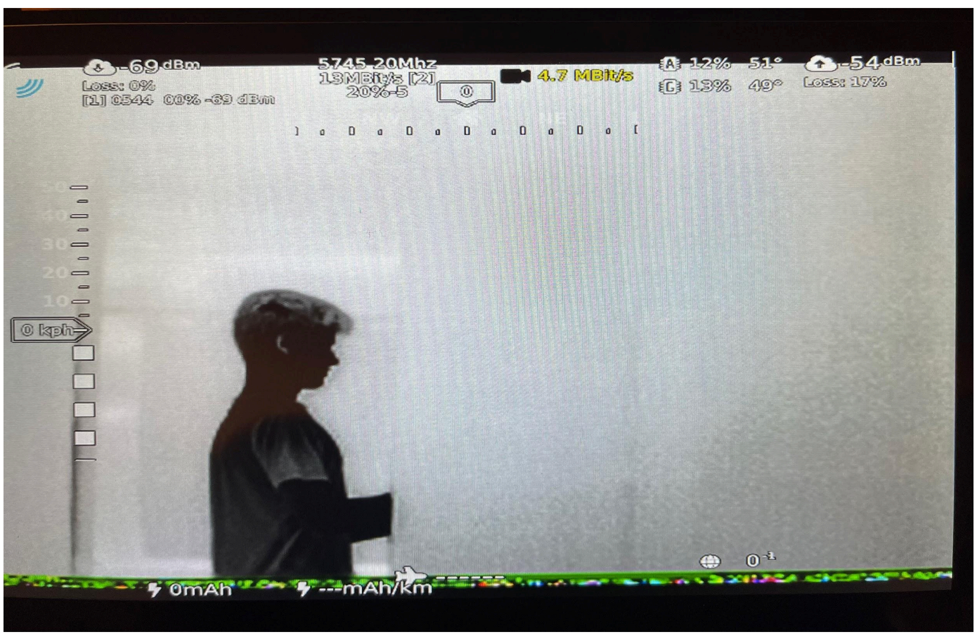

Figure 4: Test for video transmission

For a proper functioning device, further testing is required with a drone including longer range and different height variations. This is important because the directionality of the signal could have an impact on the strength of video transmission due to the orientation of the antenna and the air unit's wifi card.

Computer Vision Model

The main focus for testing the model was to validate its ability to detect humans. The testing was done under different conditions for the environmental conditions, type of thermal footage, and location. The testing was initially done of footage of a thermal camera drone operating in a human field with a test subject running through. The footage was then used to test the models performance in detection by testing on Google Colabs processor itself. Google Colab Pro allowed us access to faster GPUs and more RAM, which was able to handle the testing done by the model. **Figure 5, 6** shows the results of the model run on the footage used for testing. After testing, **Figure 7** presents the models confusion matrix, which depicts the models performance in terms of true positive, false positive, true negative, and false negative. The matrix is able to highlight and showcase to us the areas in which the model is best and where it may be weaker, allowing for specific adjustments of the model.

Figure 5: Human Detection from Model

Figure 6: Another Example of Human Detection

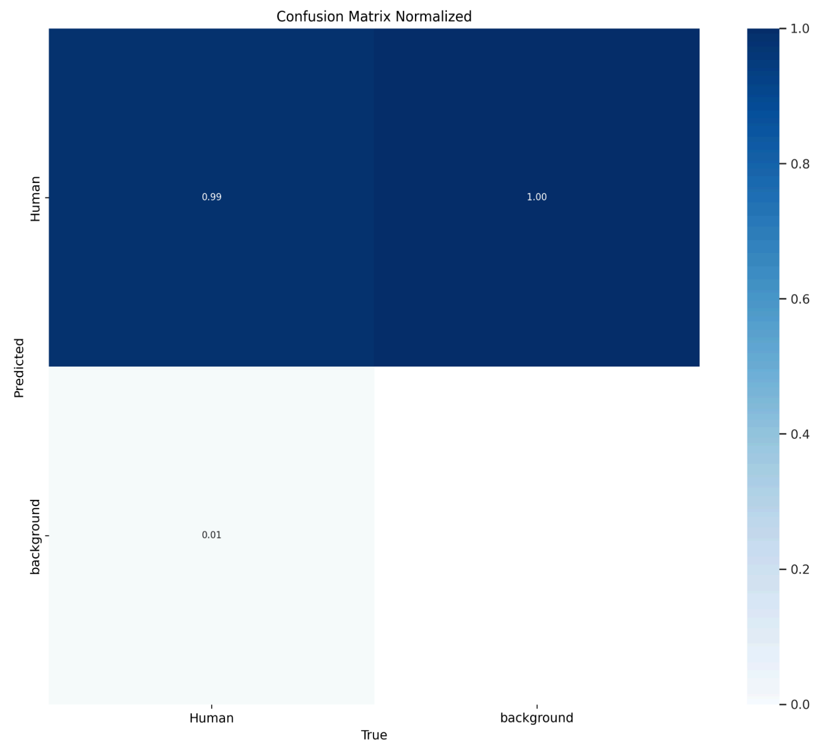

Figure 7: Confusion Matrix Result of Model

Cost Analysis

The creation of this project required specific equipment that is not readily available in homes resulting in the procurement of technologies and items that would cost money. The following **Table 4** indicates the items needed along with a short explanation and the corresponding costs to complete the project.

Table 4: Item and Cost Breakdown of Project

Item	Explanation	Cost
Raspberry Pi 4B+	A ground unit was needed to communicate and receive telemetry data from the air unit	$150.00
Raspberry Pi Zero 2W	A smaller air unit microcontroller was needed to send information and data to the ground unit	$35.00

GPS Module	A GPS module was needed to provide longitude and latitude information	$50.00
Thermal Camera	Was needed to provide thermal vision video to help in analysis	$450.00
USB Hub	Used to interface between USB port of Raspberry Pi Zero and GPS Unit, Thermal Camera and Wifi Card	$10.00
Wifi Adapters	One each for air and ground units were needed to provide the necessary frequency for video transmission	$40.00
2x 5V Voltage regulator	Needed to adapt power from the standard battery to a 5V output for the raspberry pi zero 2w and for air wifi adapter	$20.00
16GB SD cards	Needed to upload OpenHD images to air and ground units as well as to store images and recorded	$15.00
7” display	Used to display video for ground unit	$80.00
Google Colab Pro	Was needed to effectively and efficiently train and test machine learning model	$20.00
Adapters	Needed for connections not readily available	$60.00
Other	Miscellaneous small costs such as batteries and wiring	$50.00
Hours Worked	Hours worked by the team to cover the creation and implementation of the computer vision model and video transmission	$25.00/hour * ~400 hours total $10, 000
Total	Total costs of everything	$10, 980.00

As can be seen many different items were needed to create this project, especially the larger more expensive components such as the thermal camera and Raspberry Pi 4B+ were crucial in connecting different components of the project together. Given that this product is solely for use in Search and Rescue and other emergency operations, the return on investment can only be measured in lives saved, and in the amount of people it takes to search a designated area. Therefore, a proper return on investment calculation is impossible until the results of the product are seen in the field. In addition, additional grants would likely be provided by the government, countering some of the costs of development.

The majority of costs came from the research and development into the design and development of the computer vision model and video transmission software. Now that that’s been completed, and after further development into the integration and upgrades have been performed, the costs of the project will be greatly reduced. Time spent on each device should be reduced to no more than 10 hours on its implementation. On the other hand, costs for the equipment itself will increase slightly as higher quality products are implemented to achieve the best results possible for each device.

Conclusion & Recommendations

The development of this project will improve the overall techniques in search and rescue technology. With the shortcomings of current methodologies especially in remote and dense environments, this project offers a robust solution that increases the accuracy and efficiency of search and rescue operations. The integration of thermal cameras with a machine learning model allows for real-time human detection over large distances, improving response time and mission analyses compared to those with more primitive systems.

This approach is unique and modern, giving a great start for the future creations that will develop in search and rescue operations. The impact of this project on the safety and efficiency of emergency responses shows how our solution can make a large difference in critical situations.

With the results produced from this iteration of this project it is shown to be very promising. Further funding and exploration should be invested in developing this project to reach its full potential. With such a high accuracy of detection from the model and seamless video transmission the project was able to achieve it is clear that this solution for search and rescue is well worth the time and effort in implementing for real life application.

Recommendations for the future will include integrating a real time analyses and detection of a machine learning model to enable faster response times and analysis from a team. This will bypass the need for any further analysis done by a human and can immediately notify any responders nearby of possible emergencies. With this feature scouting areas for endangered human lives will significantly be easier and take a larger load off of working humans.

Increasing the data set for a wider range of detections in various environments and locations will improve the model to the point where human detection can be effective no matter the conditions or terrain instead of limiting to forests or common locations. This will increase the analysis portion of the project and could lead to utilizing this solution in various locations across the world.

Incorporating a portable and adaptable solution which will improve this system to the point where it can be effective in farther more remote locations. As of right now the telecommunication only spans a few kilometers. Having a longer distance communication will allow for exploration in hard to reach areas ultimately increasing the surveyed area in more territory that is likely to lead to a dangerous situation if an individual finds themselves there.

References

[1] C. O. F. Ethics, “GUIDE TO THE,” *Egbc.ca*. [Online]. Available: https://www.egbc.ca/getmedia/33d03861-5d04-43e9-b76b-ff57ba8b9bdb/EGBC-Guide-to-the-Code-of-Ethics-V2-0.pdf.aspx. [Accessed: 05-Aug-2024].

[2] “First Time Setup | OpenHD,” *Gitbook.io*, Jan. 09, 2024. https://openhd.gitbook.io/open-hd/hardware/first-time-setup [Accessed: 05-Aug-2024].

[3] A. Bajaj, “Performance metrics in machine learning [complete guide],” neptune.ai, 21-Jul-2022. [Online]. Available: https://neptune.ai/blog/performance-metrics-in-machine-learning-complete-guide. [Accessed: 07-Aug-2024].

[4] “FLIR ONE Pro Thermal Imaging Camera for Smartphones | Teledyne FLIR,” www.flir.ca. https://www.flir.ca/products/flir-one-pro/?model=435-0007-03&vertical=condition+monitoring&segment=solutions [Accessed: 23-June-2024].

[5] Thermometry camera T2-search series manufacturer, https://www.infirayoutdoor.com/thermometry-camera-t2-search-series [Accessed: 05-Aug-2024].

[6] “HT 301 Mobile Phone Thermal Imager（384×288）,” Hti. https://hti-instrument.com/products/ht-301-mobile-phone-thermal-imager [Accessed: 23-June-2024].

[7] “Introduction | OpenHD,” Gitbook.io, Oct. 27, 2023. https://openhd.gitbook.io/open-hd [Accessed: 23-June-2024].

[8] "Testing AI-enabled drones for search and rescue," CU Boulder Today, 14-Jun-2024. [Online]. Available: https://www.colorado.edu/today/2024/06/14/testing-ai-enabled-drones-search-and-rescue. [Accessed: 05-Aug-2024].

[9] Weforum.org. [Online]. Available: https://www.weforum.org/agenda/2021/09/autonomous-drones-resuce-hurricanes-floods-disasters/. [Accessed: 05-Aug-2024].

Appendix

Appendix A: EGBC Code of Ethics

1. hold paramount the safety, health, and welfare of the public, including the protection of the environment and the promotion of health and safety in the workplace;

2. practice only in those fields where training and ability make the registrant professionally competent;

3. have regard for the common law and any applicable enactments, federal enactments or enactments of another province;

4. have regard for applicable standards, policies, plans and practices established by the government or EGBC;

5. maintain competence in relevant specializations, including advances in the regulated practice and relevant science;

6. provide accurate information in respect of qualifications and experience;

7. provide professional opinions that distinguish between facts, assumptions and opinions;

8. avoid situations and circumstances in which there is a real or perceived conflict of interest and ensure conflicts of interest, including perceived conflicts of interest, are properly disclosed and necessary measures are taken so a conflict of interest does not bias decisions or recommendations;

9. report to EGBC and, if applicable, any other appropriate authority, if the registrant, on reasonable and probable grounds, believes that:
a. the continued practice of a regulated practice by another registrant or other person, including firms and employers, might pose a risk of significant harm to the environment or to the health or safety of the public or a group of people; or

b. a registrant or another individual has made decisions or engaged in practices which may be illegal or unethical;

10. present clearly to employers and clients the possible consequences if professional decisions or judgments are overruled or disregarded;

11. clearly identify each registrant who has contributed professional work, including recommendations, reports, statements or opinions;

12. undertake work and documentation with due diligence and in accordance with any guidance developed to standardize professional documentation for the applicable profession; and

13. conduct themselves with fairness, courtesy and good faith towards clients, colleagues and others, give credit where it is due and accept, as well as give, honest and fair professional comment.

Appendix B: Website and Home Image

https://joshruiz1414.github.io/ECE499_Capstone/

Appendix C: Code to train YOLOv8 model

```
import os

from datetime import datetime
from ultralytics import YOLO


# load a model
model = YOLO("yolov8n.yaml")  # build a new model from scratch

# set directory path
save_dir = '/content/gdrive/My Drive/ECE 499 Project/runs/detect'

#creates a new folder every time you run the model
os.makedirs(save_dir, exist_ok=True)

timestamp = datetime.now().strftime("%Y%m%d_%H%M%S")
unique_save_dir = os.path.join(save_dir, timestamp)
subdir_name = f"train{timestamp}"
```

```
# use the model
results = model.train(data=os.path.join(ROOT_DIR,
"google_colab_config.yaml"), epochs=150, imgsz=1280, plots=True,
project=save_dir, name = subdir_name, exist_ok=True)  # train the model
```

Appendix D: Code to test output of model on videos

```
import os
import cv2
from ultralytics import YOLO  # assuming you have this library installed

ROOT_DIR = '/content/gdrive/My Drive/ECE 499 Project'

VIDEOS_DIR = os.path.join(ROOT_DIR, 'testvids')

RESULTS_DIR = os.path.join(VIDEOS_DIR, 'results')  # assuming 'results'
directory exists

video_path = os.path.join(VIDEOS_DIR, 'testvid2.mp4')

video_name = os.path.splitext(os.path.basename(video_path))[0]  # get base
name of the video without extension

video_path_out = os.path.join(RESULTS_DIR, f'{video_name}_out14.mp4')  #
output video path with _out.mp4 appended

cap = cv2.VideoCapture(video_path)

if not cap.isOpened():
    print("Error: Could not open video.")
    exit()

# read the first frame to get dimensions
ret, frame = cap.read()

if not ret:
    print("Error: Failed to read the first frame.")
```

```
    exit()

H, W, _= frame.shape
out = cv2.VideoWriter(video_path_out, cv2.VideoWriter_fourcc(*'MP4V'),
int(cap.get(cv2.CAP_PROP_FPS)), (W, H))

# Path to the trained weights file (update this path as necessary)
weights_path = '/content/gdrive/My Drive/ECE 499
Project/runs/detect/train20240726_233538/weights/best.pt'

# Load the trained model
model_final = YOLO(weights_path)


threshold = 0.4

while ret:

    # Convert frame to grayscale
    gray_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    # Convert grayscale image back to 3 channels for YOLO input
    gray_frame_3ch = cv2.cvtColor(gray_frame, cv2.COLOR_GRAY2BGR)

    results = model_final(gray_frame_3ch)[0]
    #results = model_final(frame)[0]

    for result in results.boxes.data.tolist():
        x1, y1, x2, y2, score, class_id = result

        if score > threshold:
            cv2.rectangle(frame, (int(x1), int(y1)), (int(x2), int(y2)), (0,
255, 0), 4)
            cv2.putText(frame, results.names[int(class_id)].upper(),
(int(x1), int(y1 - 10)),
                        cv2.FONT_HERSHEY_SIMPLEX, 1.3, (0, 255, 0), 3,
cv2.LINE_AA)

    out.write(frame)

    ret, frame = cap.read()
```

```
cap.release()
out.release()
cv2.destroyAllWindows()

print(f"Output video saved to: {video_path_out}") # prints where output vid
was saved
```