EmberEye

Subcanopy Wildfire Monitoring Solution

Team B: Critical Design Review Report

Kavin Ravie
Ishir Roongta
Shashwat Chawla
Jaskaran Singh Sodhi
Sai Gangadhar Nageswar

Sponsors: Sebastian Scherer, Andrew Jong

May 2, 2024

Carnegie Mellon University

Abstract

With a surge in wild-land fires across the United States, rising from approximately 50,477 fires in 2019 to as high as 68,899 fires in 2022, understanding and managing these disasters are imperative [8]. Front-line firefighters confront a hostile and chaotic environment characterized by dense hot air, obscured vision, and dynamic fire behavior. To ensure personnel safety and effectively mitigate fire spread, spatial information about the fires is essential for predicting their trajectory and planning safe entry and exit routes. However, existing high-altitude solutions cannot provide detailed insights into the ever-changing scene beneath the tree line. Addressing this gap, EmberEye offers an autonomous aerial solution equipped with a sensor suite designed to navigate through sub-canopy environments and provide crucial information to firefighters. By generating fire maps and serving as aerial scouts, EmberEye enhances situational awareness, enabling firefighters to make informed decisions and safely manage wildfire incidents.

Contents

1 Project Description

With the increase in wild-land fires and acres demolished over the years from approximately 50,477 fires in 2019 to up to 68,899 fires in 2022 in the United States, [8] it is vital to understand these catastrophes in order to control the devastation. The firefighters tasked with this control face a hostile and chaotic environment in which they have to safely maneuver and perform their duties. Thick hot air, towering trees with no light, and a dynamic blazing fire are what awaits them every time.

In such an environment to ensure the safety of personnel and efficiently control the spread the front-line responders need spatial information about the fires to predict the spread and plan entry-exit routes. With entrapment being a leading cause of fatalities, there is a dire need for granular situational awareness in such a chaotic environment. Existing high-altitude solutions provide some degree of information about the wildfires but cannot provide a view of the internal ever-changing scene beneath the tree line.

EmberEye is an autonomous aerial solution equipped with a sensor suite to navigate through this cluttered scene and provide meaningful information to firefighters. Using this information the firefighters can plan their routes, avoid entrapment, and safely control the spread. EmberEye can assist by providing a heat map of the area, and act as eyes in the sky by performing the dangerous task of monitoring and scouting.

Figure 1: Depiction of a Wildfire Scene

2 Use Case

It is a bright summer day. A man is driving out to the countryside, with lush green forests around him. Suddenly he notices some smoke coming from the canopy. He isn't sure about the source of the smoke, but it seems dense enough to be a fire. He rings the fire department to tip this potential fire.

On the other side, using the location of the caller, the fire department can extract recent (but temporally inaccurate) satellite imagery and get a coarse estimate of the GPS location of the fire source. This is something we expect from the results achieved by the MRSD Team Firefly [4] where they deployed a over-canopy solution.

Using this estimate, they can deploy their firefighting unit, equipped with a state-of-the-art sub-canopy wildfire monitoring solution "Phoenix". This drone is meant to act as the eye in the sky for the firefighters, giving them real-time feedback about their environment, and helping them plan their routes and action plans more efficiently.

The drone flies in the vicinity and transmits real-time Thermal, RGB, 3D Scene Structure, etc. back to the Ground Control Station (GCS), which helps the firefighters make informed decisions about the current fire situation as well as secondary hotspots. These real-time actionable insights from the UAS act as a game-changer for the firefighting unit.

The drone finally lands, and along with the GCS, is able to help mitigate the wildfire efficiently.

Figure 2: Depiction of the Use Case

3 System Level Requirements

The team derived their system-level requirements from the use case and objective tree, illustrated in Figure 29. Specifically, performance requirements were deduced from functional requirements, each categorized into mandatory and desirable segments. Stakeholder feedback and potential customer input were also integral in refining these performance requirements, which remain subject to evolution throughout the project's advancement.

The entirety of the system-level requirements has been documented within the subsequent tables. Table 1 provides an overview of the mandatory functional requirements of our system, complemented by detailed descriptions for each requirement. Correspondingly, Table 2 elaborates on the mandatory nonfunctional requirements. Additionally, a set of desirable requirements has been formulated, outlined in Table 3 for functional aspects and Table 4 for non-functional aspects.

Table 1: Mandatory Functional and Performance Requirements

Functional	Performance	Description
M.F.1 Sense Environment	**M.P.1** Localize itself at least 10Hz.	The aerial platform should be able to perceive its surrounding environment with its equipped sensors.
M.F.2 Localize itself	**M.P.2.1** Localize itself at least 10Hz. **M.P.2.2** Localize itself within a drift of 4%	
M.F.3 Map Environment	**M.P.3** Localise itself within a drift of 4%	Aerial platform should be able to map the environment of its operation.
M.F.4 Plan safe trajectories towards goal	**M.P.4** Navigate trees with separation >=5m	Aerial platform should be able to navigate through subcanopy environment
M.F.5 Be capable of completing the mission	**M.P.5** Have a flight time of more than 5 minutes	The aerial platform should be able to complete the mission under its flight time
M.F.6 Detect fire	**M.P.6** Detect fires within the accuracy of >=70%(fire vs non-fire frames)	The fire perception module should be able to accurately detect the fire during operation
M.F.7 Localize fire	**M.P.7** Localize fire positions up to 3m of distance in front of the drone, with a 2.5m error threshold	During the operation, the aerial platform should accurately localize the location of the fire
M.F.8 Communicate with user	**M.P.8** Have a communication range up to 150m	The user should be able to visualize the fire map and telemetry information on the GCS up to the set range.
M.F.9 Be teleoperable	**M.P.9** Have a communication range up to 150m	The user should be able to teleoperate the aerial platform when required.
M.F.10 Operate in GPS degraded forest environment	**M.P.10** Localize itself at 10 Hz	The aerial platform is capable of operating in an environment where we are not relying on the GPS.

Table 2: Mandatory Non Functional Requirements

Requirements	
M.N.1	Have appropriate dimensions/size
M.N.2	Have failsafe and redundancies
M.N.3	Be of Rugged Design
M.N.4	Rely on Passive Sensors Only
M.N.5	Be Easy to Use

Table 3: Desirable Functional Requirements

Functional	Performance
D.F.1 Plan Safe Trajectory Towards Goal	**D.P.1** Navigate autonomously between trees at minimum 1m/s
D.F.2 Operate in GPS degraded forest environment.	**D.P.2** Localize itself in GPS degraded environment with less than 0.05Hz of GPS connectivity

Table 4: Desirable Non-Functional Requirements

Requirements	
D.N.1	Be FAA Compliant

4 Functional Architecture

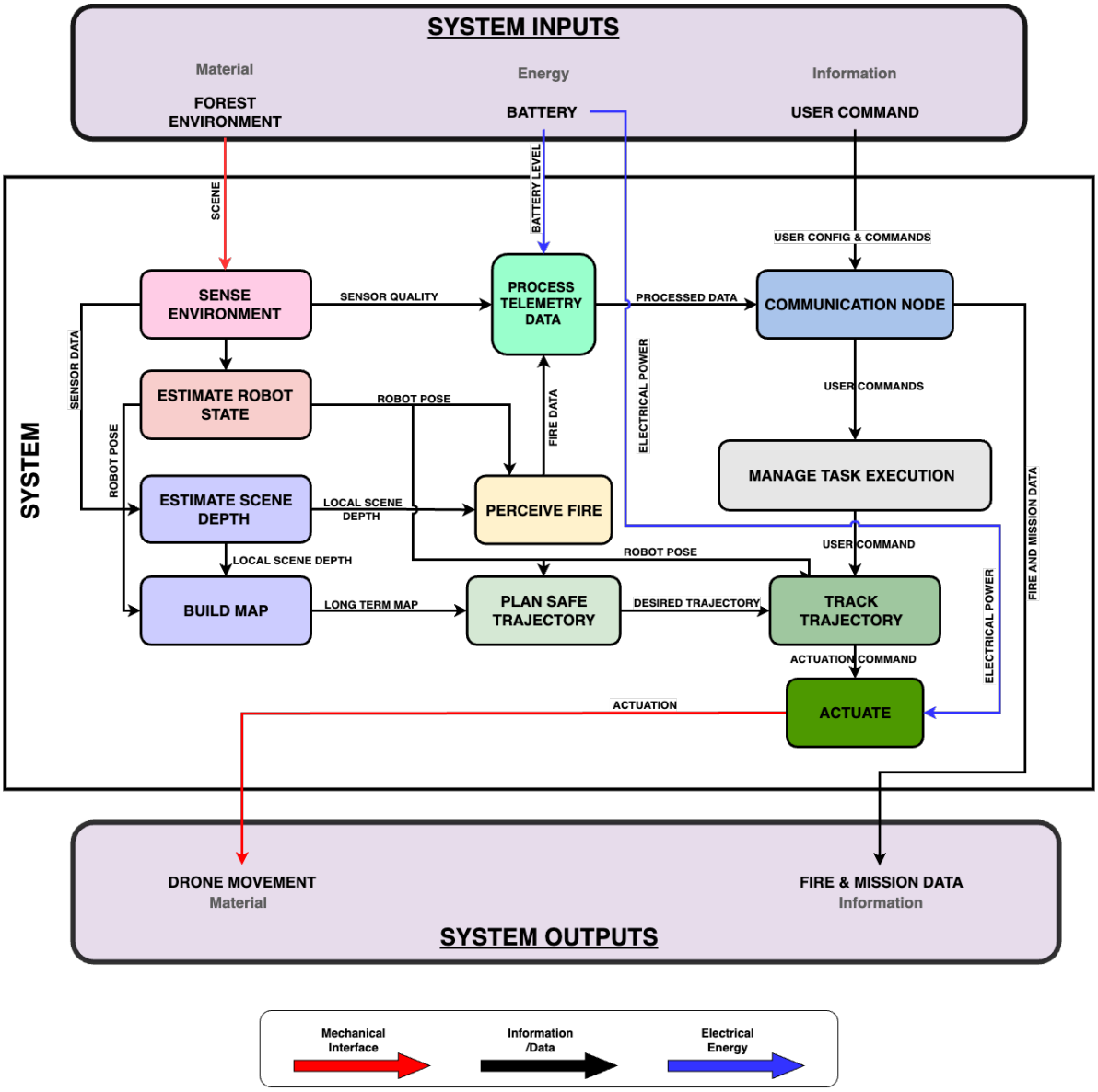


Figure 3: Functional Architecture

Figure 3 shows the functional architecture of our system. The inputs to the system are categorized broadly into Material/Mechanical input, energy into the system, and information input. The material input for our system is the wildfire environment the drone is traversing which is perceived by the on-board sensor suite. The electrical energy from the battery is utilized by the computation, communication, and actuation modules mounted on the drone's airframe. The only information input to the system is the GPS location of the estimated wildfire location. The system transmits mainly three data outputs, fire heat map, wildfire primary hotspot location, and the drone's telemetry data. Summing up, the system takes the GPS coordinates of the rough estimate of wildfire location and transmits the fire heatmap, primary hotspot location, and telemetry data.

5 Cyberphysical Architecture

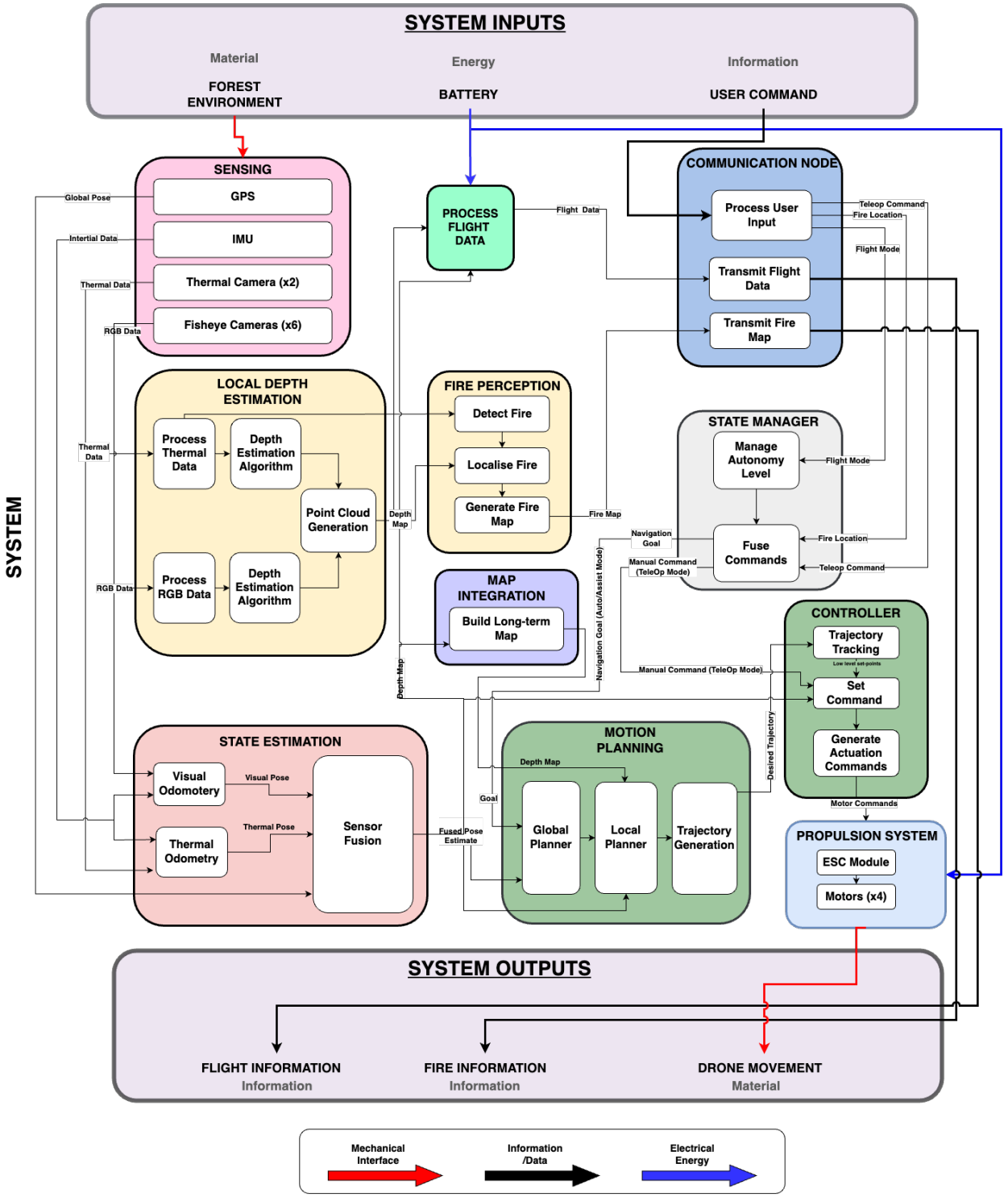


Figure 4: Cyberphysical architecture

The inputs to the system are the environment (material), electrical energy from the battery, and the user command (information). The desired output of the system is drone movement and fire information. The major subsystems present are Sensing, Depth Estimation, Fire Perception, Map Integration, State Estimation, Motion Planning, State Manager, Controller, Propulsion System, and Communication Node.

The flow of information between the modules is as follows. The sensor suite present on the aerial platform senses the environment and transmits the RGB and thermal image data to Depth Estimation and State Estimation modules. The State Estimation module takes in GPS location (degraded), thermal images, and RGB images and provides an accurate position of the robot. The local depth map is transmitted to the fire perception module for generating a heat map of fire locations, and to the Map Integration module. The output of the Map Integration module is a long-term global map that is used for motion planning. The planned trajectory is tracked using the Control and Propulsion subsystems. These are described in more detail in the subsystem descriptions.

One of the key inputs to the system is the **user command**. This command essentially consists of the rough GPS location of the fire (goal for navigation) and the autonomy mode command. The two modes of operation are assistive teleop mode and fully autonomous mode. State-manager module defines the state of modules, i.e. whether the particular subsystem is On/Off, and the flow of information.

We also have an external subsystem, namely the GCS (Ground Control Station), which is the means for information exchange between the Aerial System and the operator. As such a typical GCS consists of a field-rated touch screen for visualizing the flight data and telemetry, joysticks and switches for vehicle control, and a unified high bandwidth radio link for seamless low-latency wireless communication with the aerial system.

6 Current System Status

6.1 Targeted Requirements

During the Spring Semester, our team fulfilled the system-level requirements associated with the Aerial Platform, Ground Control Station (GCS), and Fire Perception. Moreover, we successfully met the performance requirement for localization. The remaining system-level requirements are centered around the Autonomy stack, which will be the primary focus for the upcoming Fall Semester. The complete list of targeted requirements is shown in Table 5.

Table 5: Targeted Performance Requirements for Spring Semester

Requirement	Status	Subsystem
M.P.1 Localize itself at at least 10Hz	Passed	State-Estimation
M.P.2 Localize itself within a drift of 4%	Passed	State-Estimation
M.P.3 Have a flight time of more that 5 minutes	Passed	Aerial Platform
M.P.4 Detect fires with accuracy ≥ 70% (fire vs non-fire frames)	Passed	Fire Perception
M.P.5 Localize fire positions up to 5 m of distance in front of the drone.	Passed	Fire Perception
M.P.6 Have a Communications Range up to 150m	Passed	Ground Control Station

6.2 Overall System Depiction

Figure 5: Overall System Depiction

The overall system comprises of the aerial platform: **Phoenix** and the Ground Control Station, Fire Perception, and State Estimation subsystem.

The drone has an extensive sensor suite including 6 RGB fisheye cameras and 2 forward-facing thermal cameras all powered by the onboard compute AGX Orin [2]. **The ground station** consists of mainly two systems: a laptop connected to the OTS GCS Herelink [6] which is connected to the on-board Mission computer via a wireless ethernet link to launch the relevant scripts and visualizers, and another laptop which monitors the QGC application to check for telemetry data and battery levels.

Coming to the software subsystems we have the **Fire Perception** module with the dual role of localizing the fire hotspots (in our case this is simulated using space heaters kept across a field), and then appending these predictions to a map which is then relayed back via the ground station in real-time. The other subsystem is the **visual-inertial odometry** (MSO) which utilizes the downward-facing RGB fisheye camera to track features as the drone moves. This using the IMU data is extrapolated to provide a stable pose of the drone at a rate of near about 200Hz.

A brief summary of a typical mission that we tested and demonstrated for our SVD is as follows:

The drone is kept on the launchpad, and a docker container is launched, which sequentially launches all the relevant launch files to start the state estimation, fire perception and mapping module, and data transfer scripts. The drone is first initialized at the launch pad to check for any initialization drift errors. Once this has been confirmed from the ground station, the pilot arms the drone and takes off. The drone traverses the field moving from one hotspot to another while sending back the fire map with localization predictions and a risk radius around these predictions. All this is visualized real time at the ground station on the big TV monitor. Battery status is continuously updated to the pilot to avoid unexpected crashes and battery sags. Once the drone finishes the run and is landed, the scripts are stopped, rosbag saved, and visualization closed.

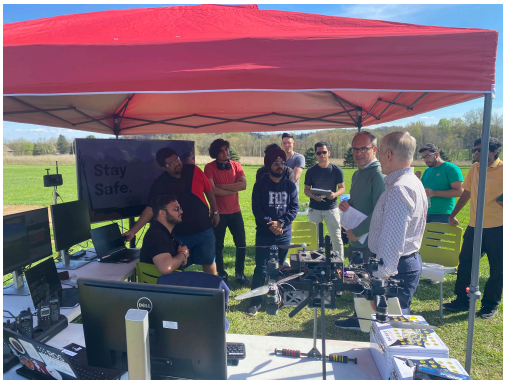


Figure 6: Ground Station on SVD

6.3 Subsystem Descriptions

6.3.1 Aerial Robotic Platform

As the name suggests this sub-system encapsulates the design, development, integration and testing of the fully-integrated aerial robotic platform. This platform is at the heart of our overall system as all the other software subsystems are deployed on the platform.

The platform's development started in November 2023, as dictated by our first external milestone, the CMU WUI UAS Research Symposium. Owing to the constrained timeline, we decided to utilize a pre-developed base airframe (the ORDv1) available at AirLab and build our system on top of it.

We conducted the first flight tests of the base airframe in November 2023 and iteratively started integrating mounts, sensors, compute and other components essential to meet our system's operational requirements and capabilities.

Figure 8: Base and Final AirFrame

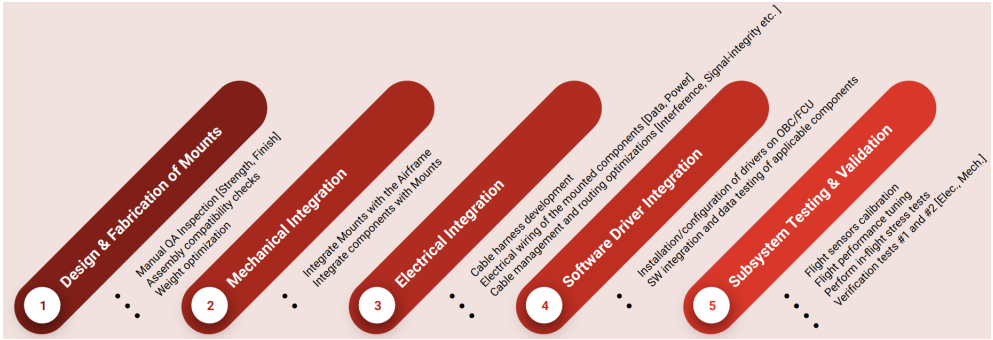


Figure 9: HW Development Workflow

Table 6: Comparison of Base and Final Airframe Configurations

Component	Base Airframe	Final Airframe
Airframe	AirLab Provided	AirLab Provided
Flight Control Unit (FCU)	Included	Included
Low Voltage (LV) Power System	5V	5V
High Voltage (HV) Power System	16.8V	16.8V
Propulsion System	ESCs, Motors, Battery	ESCs, Motors, Battery
Medium Voltage (MV) Power System	Not Included	12V
Thermal Cameras	Not Included	2x Included
RGB Fisheye Cameras	Not Included	6x Included
Inertial Measurement Unit (IMU)	Not Included	Included
On-board Mission Computer	Not Included	Included
GPS	Not Included	Included
Communications	Not Included	RC, WiFi Included
3D LiDAR	Not Included	Included (Only for GT DAQ)
Ground Control Station (GCS) Airside Unit	Not Included	Included
1D Range Finder	Not Included	Included
Other Modifications	CoG initial setup	CoG adjustment, Improved landing strut stability

Please note that, for all the new additional sensing components listed in Table 6, software-side driver integration was also undertaken, enabling a seamless development experience for downstream application-specific modules.

The current airframe is under-sized, implying that it is not sized appropriately for the high All-Up Weight (AUW), resulting in a low flight time. We are considering a redesign of the airframe in the Summer/early fall in collaboration with AirLab, who are also looking to integrate a new and improved unified research development platform for the lab.

6.3.2 State Estimation

As our system is required to operate in GPS-denied/degraded settings, we need a robust state estimation pipeline which uses an alternate modality. Furthermore, our mandatory non-function requirement **MN4** requires us to rely only on passive sensing modalities which led us to converge to vision/thermal-based state estimation.

To this end, AirLab had a suitable Multi-Spectral Visual Inertial Odometry system which was perfect for our requirements and hence we decided to employ the same for our system. However as this was a research undertaking, the estimator needed changes and tuning before integration with our physical robotic system.

The modular estimator enables us to choose any combination of the 8 onboard cameras (considering the performance penalties). For our indoor runs, we chose the Upper-Front-Left camera (RGB, ID:0)

and for our outdoor runs we chose the Bottom-Front-Middle (RGB, ID:2). We limited the camera count to one to save computational resources as even with a single camera we were meeting our design requirements. Furthermore, the choice of these specific cameras resulted from exhaustive trials and experimentation with different configurations.

The estimates from the MSO (running on the NVidia Orin AGX onboard computer) are transformed into the correct frame and then passed to PX4 as an external state estimate using the **VISION_POSITION_ESTIMATE** MAVLINK message over a Serial communication link.

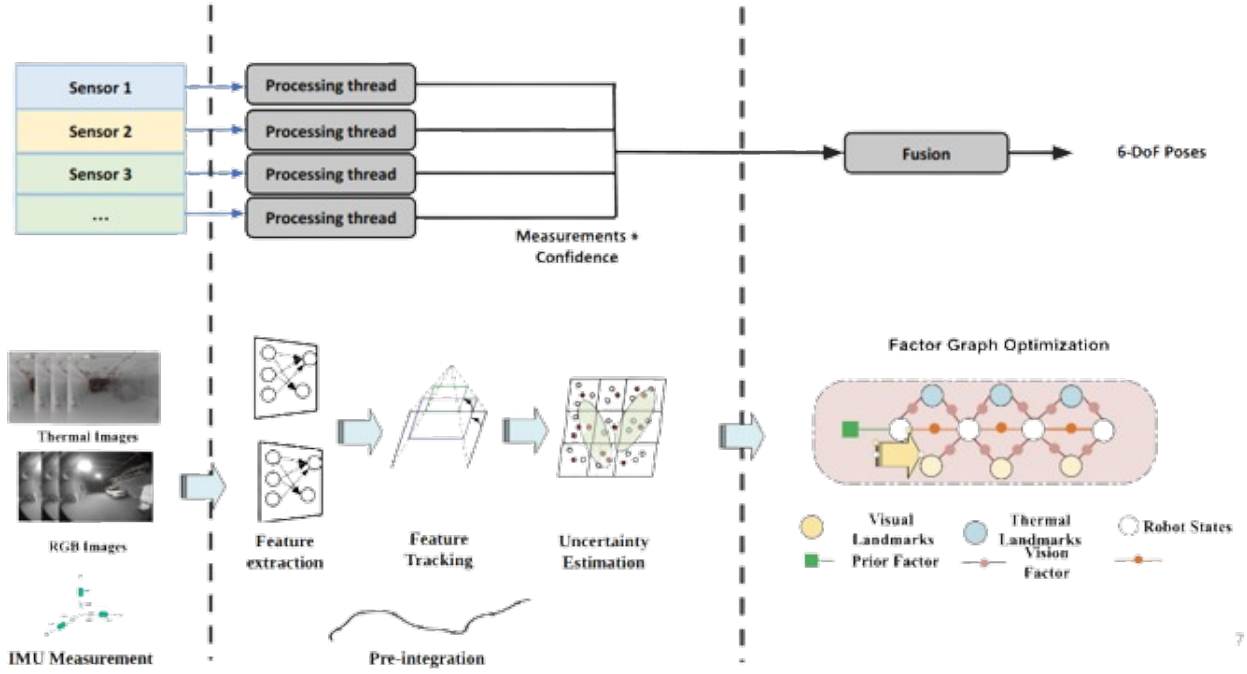


Figure 10: MSO Architecture

The final integrated system was able to output the state estimates at 200Hz (thanks to our use of IMU-interpolation b/w VIO estimates) and meet all the drift/variance requirements.

The estimates from the MSO which is running on the NVidia Orin AGX onboard computer are transformed into the correct frame and then passed to PX4 as an external state estimate using the **VISION_POSITION_ESTIMATE** MAVLINK message over a Serial communication link.

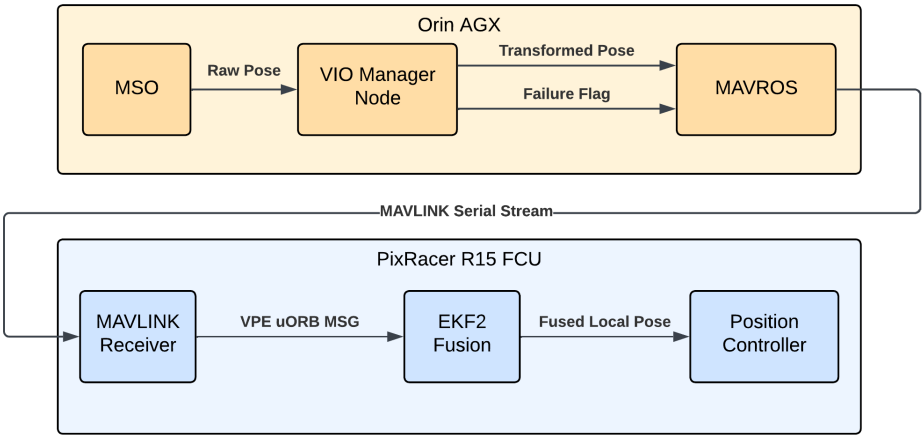


Figure 11: MSO-PX4 Interface

6.3.3 Fire Localization Pipeline

The fire localization module get the thermal images as input and outputs meaningful coordinates for the fire hotspot to the mapping module. To implement this, broadly two methods were pursued: learning based developed by Airlab, and the classical method independently developed by us.

Working Stable Pipeline The pipeline that was finally deployed on the drone was based on classical stereo matching of features around the heaters and using camera intrinsics and extrinsics to calculate the coordinate of the pixel in world frame.

Using a ROS Timer the information received through the thermal feeds is synced to allow for feature extraction at a single time-step. Once we pass down synced up feeds, we perform stereo rectification using cv2's inbuilt function *stereoRectify*. This rectifies the slight distortions and balances out the exposure. Now for the sake of detecting features only around the hotspots, we convert the rectified feeds to temperature mapped binary frames. Using a empirical formula which was calculated by extrapolation, we create a mask around the hotspots as shown in Fig 12

Now using this mask on the rectified images, we attempt to detect ORB features in the image. These features naturally are detected around the hotspot owing to the masking. The key-points and descriptors are computed using cv2's ORB *DetectandCompute* function and then feature matching is performed using cv2's *BFMatcher* as shown in Fig. 13

Now once we receive these matched coordinates in the left and right camera frame, we filter out only the good matched features using the epipolar constraint [1], [3]. The idea is to use a coordinate in left frame, and check if the right-matched coordinate is in the vicinity to the left's epipolar line. If that distance is under a certain threshold, we use that feature to calculate depth. The epipolar lines are shown in the Fig 14

Once we have our final list of matched feature coordinates, we calculate the mean of these coordinates and use it to calculate the disparity between the two camera frames. Using the baseline available to us we calculate the depth which is then used to calculate the 3D coordinate in the camera frame. Using the transformations available to us between the camera and IMU, and further IMU to the initial position of the drone, we can calculate the coordinate of the hotspot in the world frame.

Alternate Approaches Attempted In the development of the fire localization system, several meth-

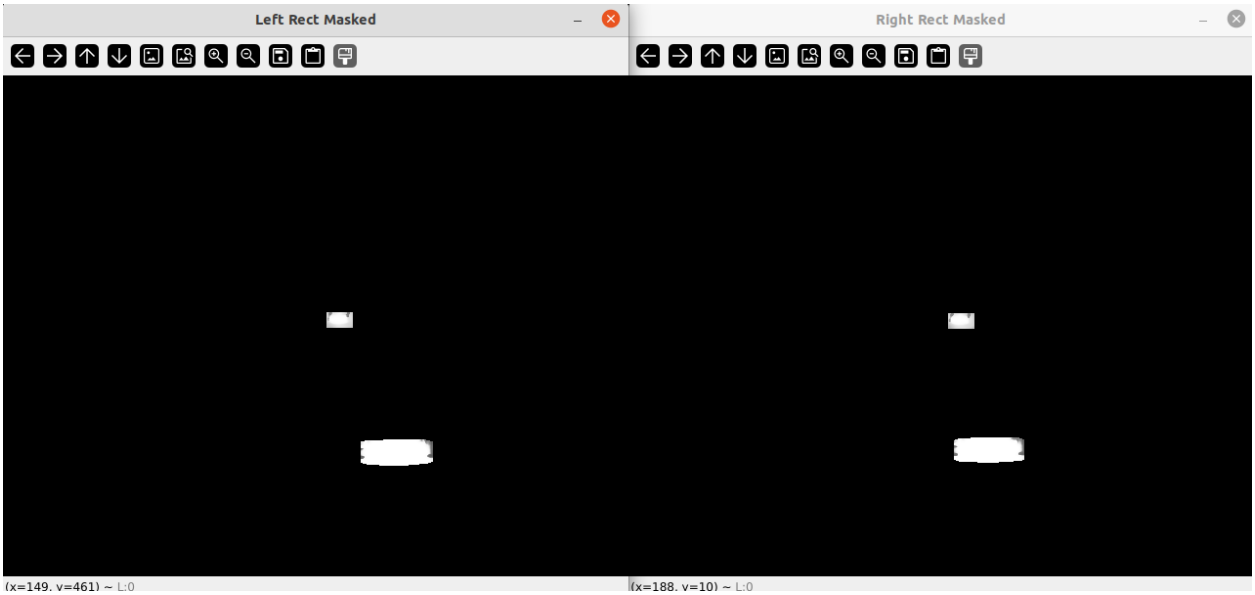


Figure 12: Masking around hotspots using temperature mapping

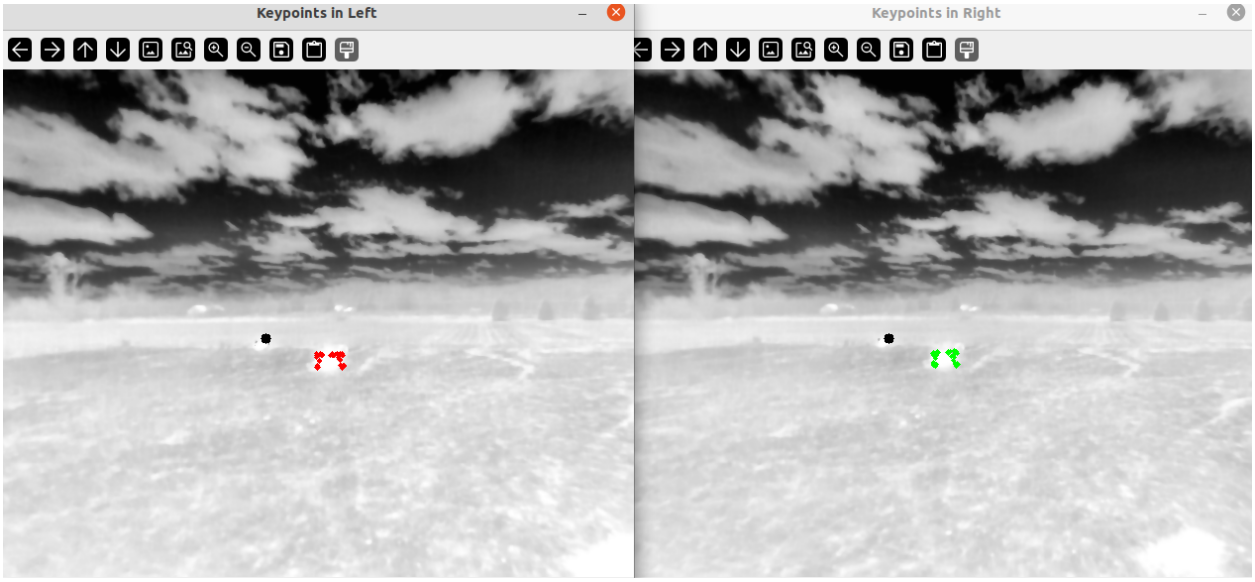


Figure 13: Keypoints of features in left and right frame

ods were explored but encountered various challenges. The first approach utilized the OpenCV StereoSGBM module to generate disparity maps from thermal stereo images. This method faced issues with accuracy due to the necessary conversion of 16-bit images to 8-bit, resulting in significant information loss (image smoothing and noisy disparity maps). An enhancement was attempted by performing image processing of the rectified thermal images; however, this improvement yielded minimal enhancement in the feature-matching process. Lastly, a learning-based method using a Fast-ACVnet architecture was trialled, which showed promising results with disparity maps achieving a performance of 13 frames per second. Despite some artefacts, the depth estimation near the ground was reliable, suggesting potential for practical application in fire mapping. Although the results from the learned model were good, this

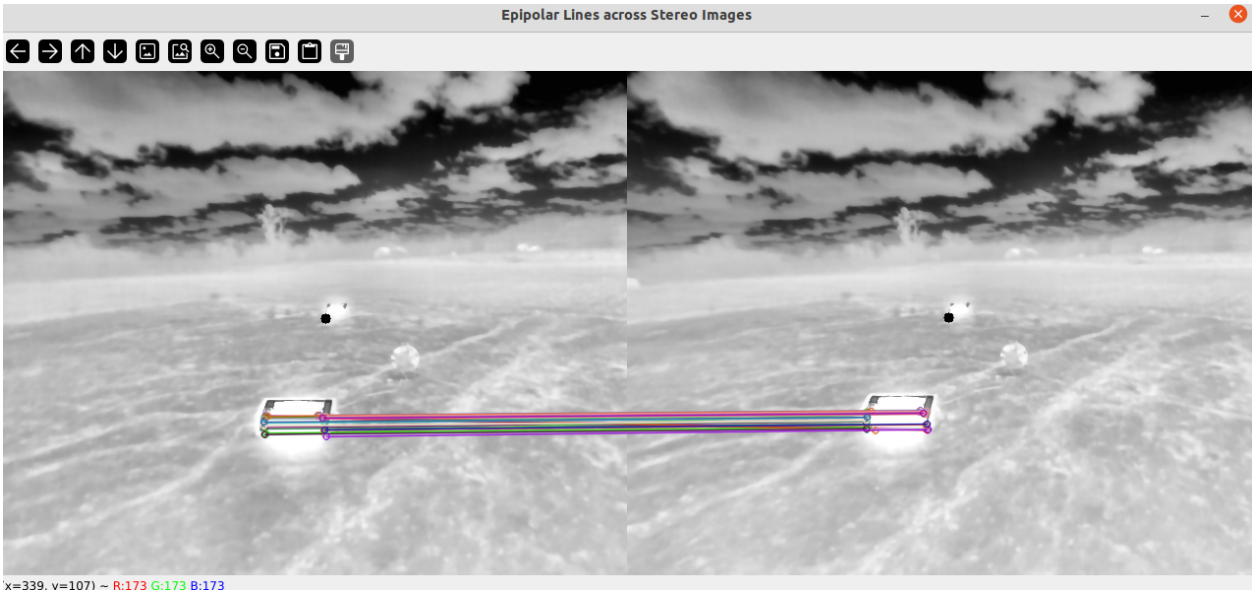


Figure 14: Epipolar lines joining the matched features

approach can not be used for our application since the space heaters that were used for simulating fire were out of the distribution of the model. Hence the learning-based model produced poor results on our test site. The results for these approaches are attached in the Fig. 15

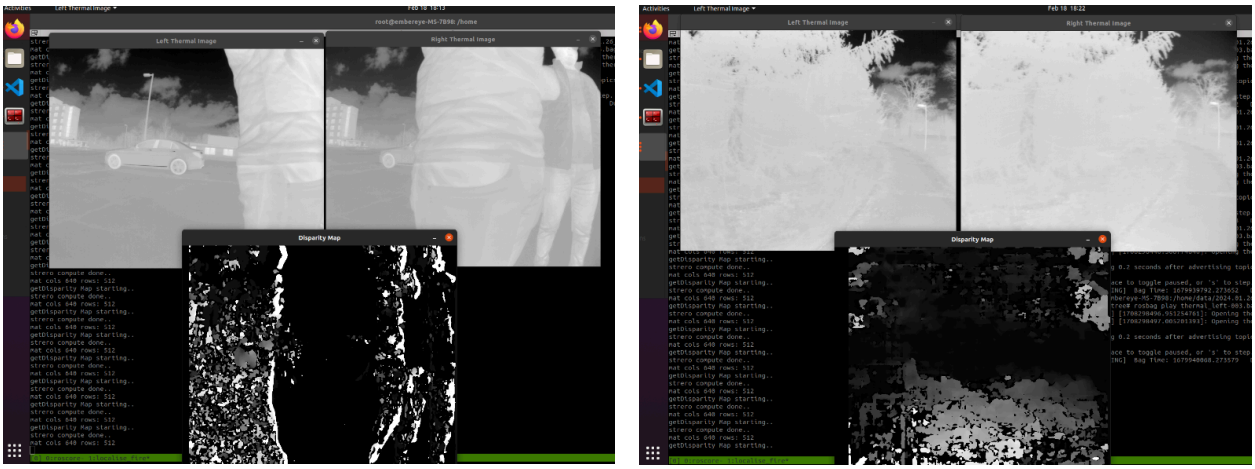


Figure 15: Disparity map from StereoSGBM algorithm

6.3.4 Global Fire Mapping

Our global mapping subsystem handles the temporal side of the fire perception subsystem. The fire perception module provides raw measurements of hotspot locations relative to the world frame. When the fire perception subsystem publishes a new set of measurements, this mapping subsystem is executed as described in Figure 17.

Filtering: Once we receive a hotspot measurement, we first need to filter out the good readings, because most readings are quite noisy. The following are the 2 main filters we employ. Figure 18

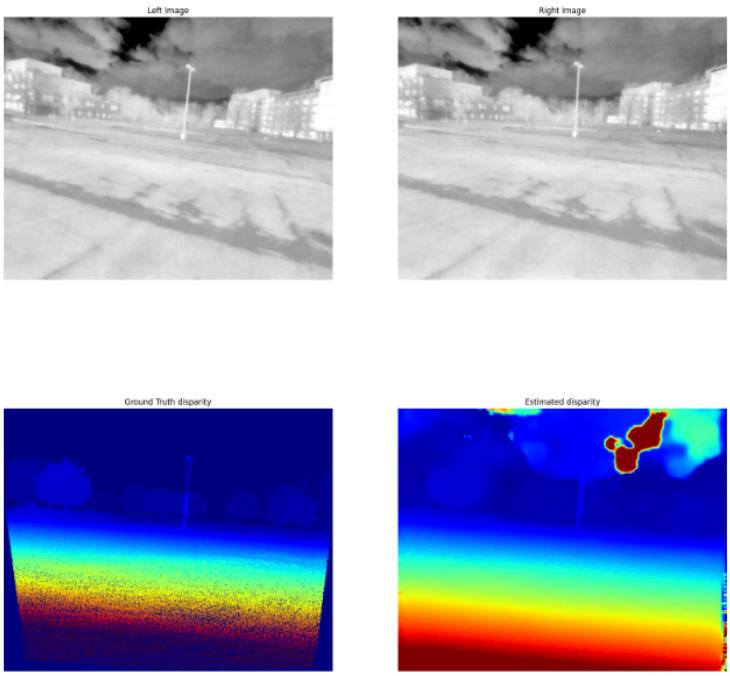


Figure 16: Disparity map from Learning-based Thermal Stereo

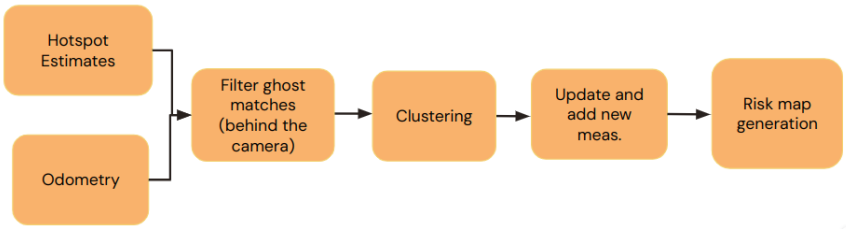


Figure 17: Overall architecture of Mapping Subsystem

shows the measurements from 2 successful flights (L) and the measurements after filtering (R). The red spots show the ground truth hotspot locations.

1. **Height Filter**: Filter out hotspot estimates above the UAS, since they are incorrect measurements.

2. **Distance Filter**: Filter out hotspot estimates too far from the UAS pose, since measurements which are too far are prone to parallax errors (as can be seen in Figure 18 (L)). The distance parameter of this filter can be tuned.

Adding New Hotspot: We then perform nearest neighbor clustering for each hotspot to determine whether it is a measurement for a new or existing one. A new hotspot is created if it exceeds distances from all previously seen hotspots.

Update Existing Hotspot: If we instead find any measurement corresponding to an existing hotspot, we add it to the existing nearest-neighbor map, and use a mean search to find the updated position of its parent hotspot.

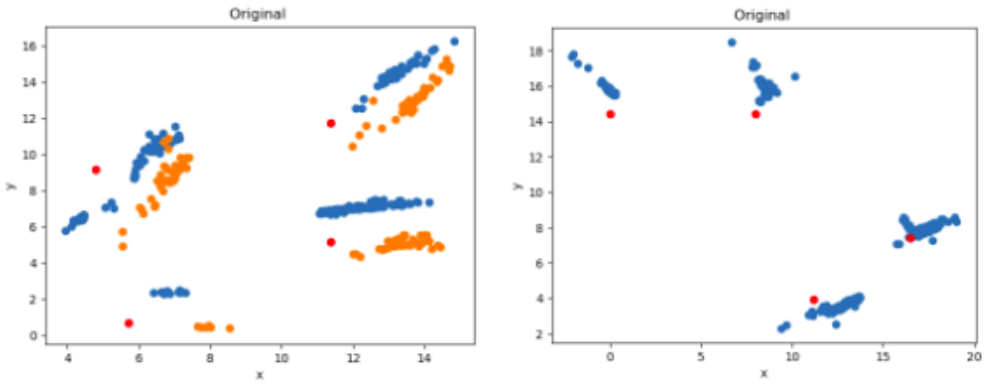


Figure 18: Measurements from 2 successful flights (L) and Filtered measurements from 1 flight (R)

6.3.5 Ground Control System

This is a key part of our overall system as it is central to relaying the fire analytics processed onboard back to the first responders.

After careful evaluation of different approaches to architect our GCS system, we ended up going for a development time optimal approach wherein most of the GCS hardware was outsourced as a fully integrated unit in the form of the HereLink v1.1 GCS Kit. We then proceeded to develop the required ROS wrappers and perform the necessary network configurations to enable a reliable and low-latency data stream link back to ground operations.

Figure 19: The HereLink GCS Kit

All non-image ROS topics were relayed as is over the Ethernet interface using distributed ROS over LAN. Image topics were processed using a custom ROS-GStreamer wrapper [7] to perform H.264 image encoding allowing for efficient low-latency image streams.

The same LAN was used to interface with the UAS via an SSH session to initiate the SW systems bring-up.

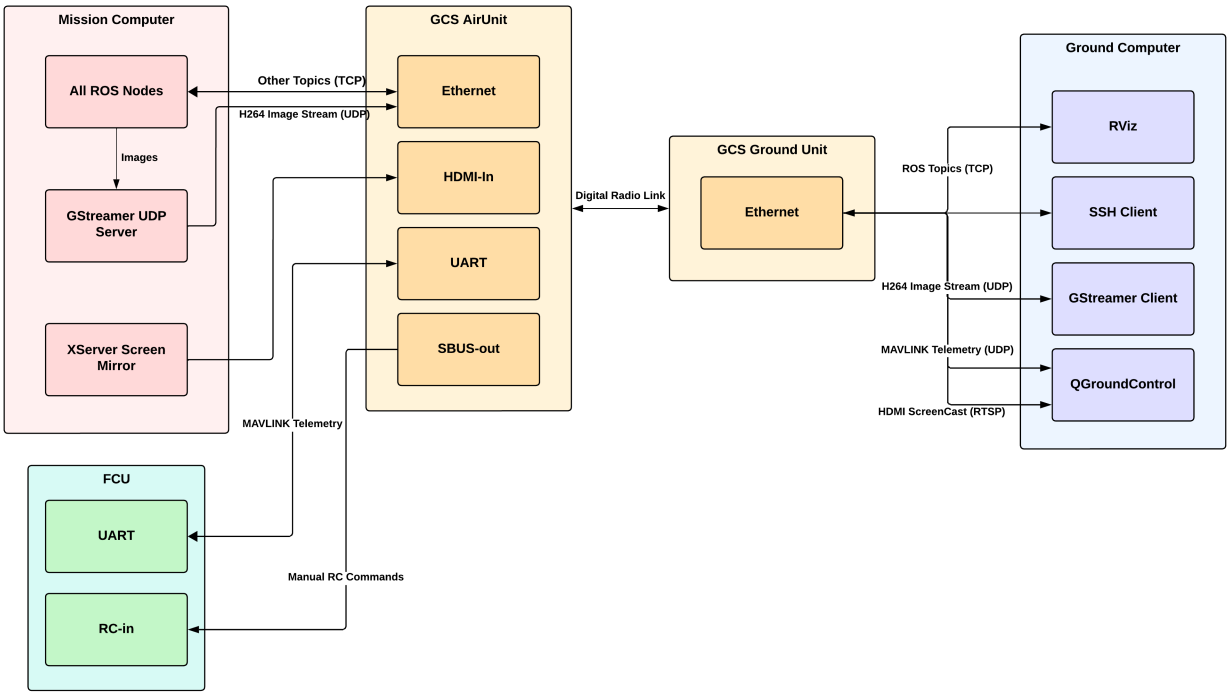


Figure 20: GCS Architecture

As seen in Fig. 20 we have a multitude of interfaces on the GCS AirUnit to pass a variety of data streams. All the streams are intercepted on the ground side via the Ethernet interface [5] and utilized by downstream processes for Visualization. Also, we should note that the link is bi-directional, enabling rich and dynamic user-centric workflows (like setting waypoints in RViz during our SVD Encore Demo).

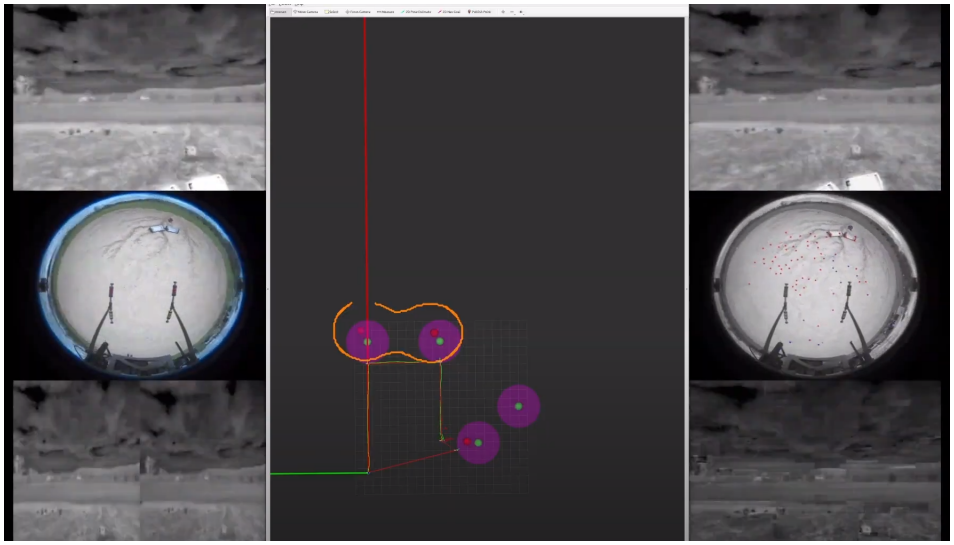


Figure 21: Result Visualization

6.3.6 Software Infrastructure for ORIN AGX

The software modules on the ORIN AGX are deployed utilizing the Docker containerization method, ensuring rapid deployment and scalability. The driver docker container currently contains the drivers and dependencies for launching the thermal and fisheye cameras. The `gst_ros_interface` module is responsible for streaming the image streams to the GCS handheld device. `Mavros` is required to bridge the communication between the FCU and the AGX Orin. The autonomy Docker container includes the dependencies necessary for launching the multi-spectral odometry and fire perception modules. This Docker container will undergo further expansion during the development of the Autonomy stack.

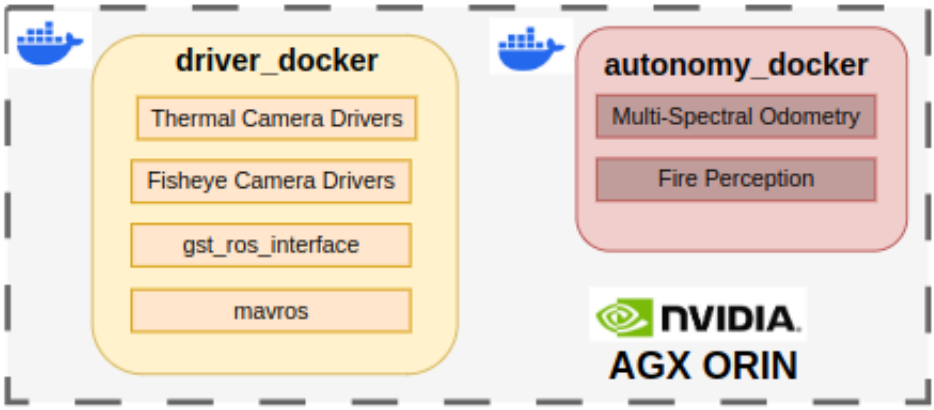


Figure 22: Software deployment on ORIN AGX

6.4 Modelling, Analysis and Testing

As detailed in our Spring Test Plan, we performed a continuous validation of our subsystems through several different tests, as shown in Table 7.

Table 7: Targeted Performance Requirements for Spring Semester

Test	Success Criteria	Result
Electrical and Power Delivery System Test	- Appropriate and stable regulated voltage is supplied to the FCU [5V]. - Appropriate and stable voltage supplied to the propulsion system [14.80-16.80V]. - Enough voltage supplied to on-board compute AGX Orin [12V]. - Stable sensor feeds streaming to verify proper wiring connections to AGX Orin.	- Appropriate voltage supply verified for FCU, propulsion and AGX. - Stable sensor feeds also verified.
Robotic Platform Structural Integrity Test	- Motor mounts rigidly affixed to the main airframe structure (no relative motion). - Sensor mounts (IMU, Thermal, and RGB Cameras) rigidly mounted to the airframe. - In-flight IMU vibration metrics should be lower than the prescribed limits (5 m/s/s). - No motion blur should be visible on all the camera image feeds.	- Motor and sensor mounts rigidly mounted and validated from in-flight sensor feeds. - Vibration metrics less than 5m/s^2 - Lack of motion blur verified through sensor streams.

Camera Calibrations	- The sensors should have a maximum relative latency of 100ms. - Calibration RMS reprojection error less than 2.5 pixels.	- Achieved relative latency of 10.3 $\pm$ 0.2 ms - Achieved mean calibration projection error 0.083 pixels.
Fire Segmentation on Dataset	- Detect fires with 70% accuracy. - The frames per second (fps) achieved from the segmentation model should be a minimum of 10 FPS.	- Detected fires with 100% accuracy - Achieved offline frequency of 30 Hz of segmentation system.
Fire Segmentation on Real-time Feed	- Detect fires with at least 70% accuracy. - The frames per second (FPS) achieved from the segmentation model should be a minimum of 10FPS.	- Detected fires with 100% accuracy - Achieved online frequency of 13 Hz of segmentation system.
Localize Fire on Static Aerial Platform	Accurately localize fire positions up to 3m* of distance in front of the drone.	Able to accurately localize fire up to 5m distance.
Visual-Inertial State Estimation Test	- Stable VIO estimates at the required update rate of 10Hz - Hovering positional RMSE of < 0.20 m w.r.t ground-fixed fiducial marker pose as ground-truth	- Achieved 200Hz state estimation frequency - Hovering positional RMSE of < 0.10 m
In-flight Fire Localization	Localize the fire within 2.5 m of the ground truth fire location.	- Fire localization demonstrated within 2.5 m of ground truth during SVD.
Ground Control Station Communications Test	- GCS unit connects to the Aerial Platform. - Able to configure drone mission parameters from the GCS unit. - Stable link connection with a communication range of at least 150 meters. - Stream processed visualization data at a minimum of 5 FPS.	- GCS unit successfully connects to system and can configure mission parameters. - Link achieves > 150m communications range. - Link achieves 30 FPS video streaming, 15-15 FPS map streaming, and 200 FPS VIO streaming.

6.5 SVD Performance Evaluation

We demonstrated our set of system capabilities (shown in Table 8) in SVD and SVD Encore. The system was able to meet all the given Performance Requirements, with a false positive hotspot localization in one of the configurations in SVD. This was rectified later in SVD Encore.

Table 8: Targeted Performance Requirements for Spring Semester

Procedure	Success Criteria	Requirements
UI displays updated fire map throughout duration of flight.	UI successfully displays real-time feed throughout duration of flight.	**M.P.8, M.P.9**
Communicate with drone up to 150 meters.	Successfully communicating with drone up to 150m.	**M.P.8, M.P.9**
Have a flight time of more than 5 minutes.	Runtine of mission was within endurance limits, so no endurance issues (complete endurance of > 6.5 mins demonstrated prior to SVD).	**M.P.5**
Detect all hotspots with at least 70% accuracy.	100% of hotspots detected in both configurations. False positive detected in Configuration 2	**M.P.1, M.P.2, M.P.3, M.P.6, M.P.7**

6.6 Strong/Weak Points

Our progress throughout the semester culminated in a successful demonstration during SVD and SVD Encore. While our system showcased several strengths, some areas require improvement. Below, we highlight both the strong and weak points of our system.

Strengths:

- Reliable Aerial Platform: Throughout extensive testing, our aerial platform experienced multiple crashes, yet remarkably, there was no significant damage to our system.
- Robust State-estimation: Our MultiSpectral odometry module functions effectively with just one fisheye camera, consistently maintaining the overall drift below the promised 4%. We also conducted tests with thermal cameras and found that the module performed well under these conditions too.

Weaknesses:

- Fire localization: The classical stereo matching algorithm struggles to accurately calculate the depth of heat sources due to the limited features present in the images captured by thermal cameras.
- Insufficient flight time: The integration of a rangefinder, GCS module, and its antenna has resulted in a decrease in the flight time of the aerial platform. Additionally, random voltage sags caused by poor batteries have further contributed to this reduction in flight time.

7 Project Management

7.1 Work Breakdown Structure explained

Figure 23 shows the Work Breakdown Structure (WBS) required to build the Aerial platform solution. These work packages help us in exactly defining the schedule and work dependencies to achieve our desired milestones as shown in Table 9. We have roughly six overarching technical work packages: Aerial robotic Platform, Autonomy Stack, Fire Perception, Ground Control Station, System Integration and Testing, and Project Management. Aerial Robotic platform includes the development of a structural airframe, propulsion drive, the sensor suite required for perceiving the surrounding environment during operation, low-level autopilot tuning and calibration, onboard mission computer integration, and power delivery system. The Autonomy Stack encompasses the development of the localization stack, depth estimation, mapping, motion planning, and trajectory tracking module. Within the Fire Perception subsystem, we have three distinct components: the fire detection model, fire localization module, and fire mapping pipeline. Within the Ground Control Station (GCS) module, we encompass hardware units, wireless network configuration, and user interface subsystems. The System Integration and Testing module involves 10 subsystems dedicated to integrating each of the aforementioned components onto the aerial platform and conducting separate and combined testing procedures. Throughout the spring semester, our primary focus was on developing the Aerial Robotics Platform, Fire Perception, and Ground Control Station modules, and subsequently conducting their integration and testing processes.

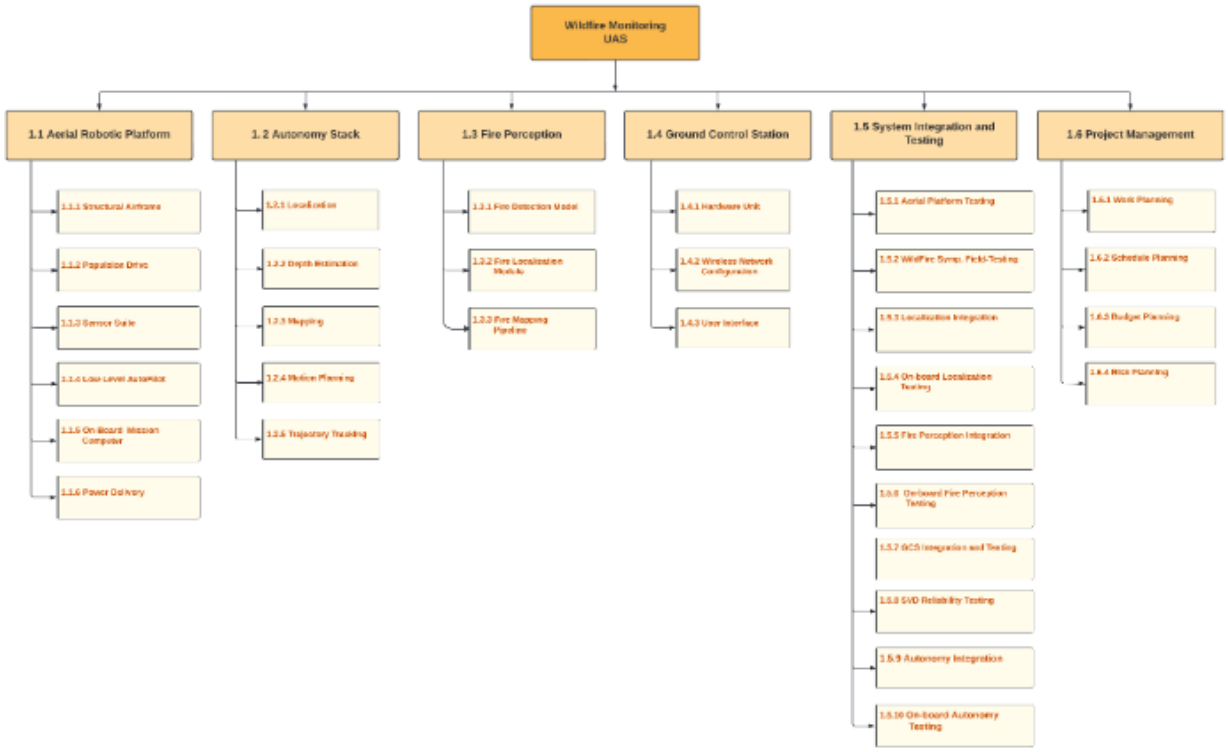


Figure 23: Work Breakdown Structure

7.2 Schedule

7.2.1 Biweeklies, Milestones, and Demos

Table 9 outlines the key milestones for our project, with bi-weekly reviews thoughtfully set to enhance our tracking of progress and attainment of significant external milestones. Initially, during the early fall semester, we develop a fundamental framework to address the needs of all subsystems, and then iteratively refine this to meet our essential requirements. The system is segmented into four main components:

1) Depth Estimation
2) Mapping
3) Motion Planning
4) Autonomy Integration

Our tentative objectives for some of the major bi-weekly schedule during the project are as follows:

Bi-weekly 1 - Reconsider new platform due to SVD challenges and Demonstrate dense depth calculation from the RGB. We explore various methods available for fisheye depth estimation and perform a trade study between the methods.

Milestone 1 - Have the new airframe (if any) completely ready. Achieve sparse depth estimation from the fisheye cameras

Bi-weekly 2 - After a detailed analysis, if we consider a new platform to be essential, then complete the new airframe design. For mapping, use VIO and local depth information to create map of environment.

Milestone 2 - Complete mapping and motion planning subsystems

Bi-weekly 3 - Complete autonomy stack integration; test and validate the system

Fall Validation Demonstration (FVD) - A working demo of the entire integrated fully autonomous drone with the fire perception module integrated.

Bi-weekly 4 - An aerial platform autonomously navigates to a specified location while simultaneously mapping the surrounding environment, using the on-board fisheye and thermal sensors to accurately rely the fire information to the ground control station. All the reports and demos done.

Table 9: Major System Development Milestones (bold are external, rest are internal)

Date	Milestones
15 September	Bi-weekly 1 (B1)
7 October	Milestone 1 (M1)
8 October	Bi-weekly 2 (B2)
4 November	Milestone 2 (M2)
15 November	Bi-weekly 3 (B3)
22 November	**Fall Validation Demonstration (FVD)**
4 December	Bi-weekly 4 (B4)

7.2.2 Schedule

Our Fall schedule can be seen represented in a Gantt chart format in Figure 24. The subsystems delivered by the SVD are Aerial Platform, State Estimation, Fire Perception, and Ground Control Station. For the fall, we plan to deliver the Depth Estimation, Mapping, Motion Planning, and the Integrated Autonomy stack. We have parallelised the work of the subsystems such that there are no blockers. As of now, Spring sem, we are on track with the schedule.

7.3 Test Plan

For Fall Validation Demo, we plan to demonstrate the autonomy capability of our drone. We will be incorporating mapping using fisheye and thermal cameras, motion planning, and obstacle avoidance capabilities. We will be integrating our current fire perception and state estimation modules in the autonomy stack, providing seamless transmission of fire information from a fully autonomous drone.

7.3.1 Testing Activities

- **Testing logistics**: The testing location, Nardo, can not be used for the FVD because of the harsh weather conditions in the fall. Additionally Nardo does not have smoke permissions. Therefore we plan to use other testing locations, potentially indoor locations.
- **Fire Perception**: The drone will autonomously detect all the fire hotspots with 70% accuracy and localise them within 2.5 meters radius of ground truth. To show the system robustness, we will test the system in various configurations with edge cases.

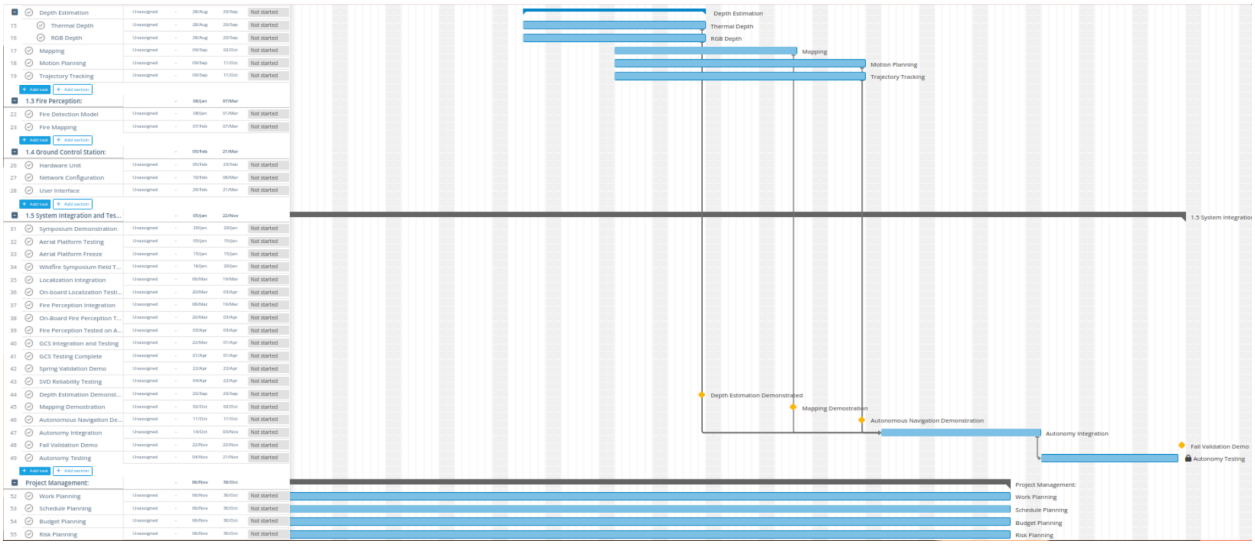


Figure 24: FVD Schedule

- **Mapping**: For mapping, we require both the drone state estimation and the depth estimates of the surrounding environment. For SVD, we have demonstrated state estimation which will integrated with the depth estimation module to obtain a map of the environment.
- **Depth Estimation**: For depth estimation, we use the passive sensors on-board, i.e. the six fisheye-cameras for predicting the depth of the obstacles in the environment.
- **Motion Planning**: Given a target area and the drone home location, the motion planner algorithm will provide the plan (both global and local). This is followed by trajectory planning which is sent to the control systems module.
- **Autonomy Stack Integration**: The final integration of all the systems, fire perception with the autonomy stack and further with the ground control station will be done.

7.3.2 Fall Semester Capability Milestones

The detailed structure of the Fall Semester Capability Milestones is shown in Table 10.

7.3.3 Fall Validation Demonstration

The test conditions are as follows:

- **Location**: Nardo (tentative), final testing location to be decided
- **Elements**: Aerial Robotic Platform, State Estimation, Thermal Perception, GCS and Autonomy
- **Key Equipments**: Fully integrated Aerial Robotic Platform, and space heaters, Router; Stable Internet Connection, Router, RC Transmitter, Ground station Laptop with QGC, Telemetry Link, Fire Extinguishers
- **Operating area**: Open environment of approximately 15x15m, with space heaters located at various locations and obstacles such as trees present in the arena

Table 10: Fall Capability Milestone

Progress Review	Subsystem	Objectives	Capability Milestone
PR 7	Aerial Platform, Thermal, and RGB Depth	Reconsider new platform due to SVD challenges, Demonstrate dense depth calculation from RGB	FVD aerial platform decision Evaluate the accuracy of the RGB and Thermal depth maps.
PR 8	Aerial Platform, Thermal, and RGB Depth Mapping	(If needed) Complete new airframe design Use VIO and local depth information to create a map of the environment.	Stable platform with accurate map to be used for autonomous navigation.
PR 9	Mapping, Motion Planning, and Trajectory Tracking	Improve mapping Complete basic obstacle avoidance to rough goal location	UAS is capable of autonomously navigating to rough goal location
PR 10	Drone Platform, Autonomy Stack, GCS	Aerial platform autonomously navigates to given location and maps the environment.	UAS is capable of autonomously navigating to rough goal location while mapping hotspot locations
PR 11	Drone Platform, Autonomy Stack, GCS	Full system integration and testing	UAS is capable of autonomous mission completion, and relaying real-time hotspot information to GCS.
PR 12	Drone Platform, Autonomy Stack, GCS	Full system integration and testing	UAS is capable of autonomous mission completion, and relaying real-time hotspot information to GCS.

The Fall Demonstration procedure and the verification criteria can be found in Table 11 and in Table 12 respectively. A rough schematic of the final run is depicted in the figure 25. The user will input a rough large area where the fire locations can be present. The drone will autonomously take off from the home position, search the area, detect and locate all the fire hotspots, and relay back the information to the ground control station. For FVD, we plan to place various trees in the test area which act as obstacles. The drone will autonomously search for the fire hotspots while doing obstacle avoidance.

7.4 Budget

We have spent a total of **USD 2993.30** so far for our project expenses. The majority of the spending was for the GCS Unit (USD 900), closely followed by field testing expenses (USD 700). Other expenses also correspond to planned system upgrades and testing logistics like space heaters and so on.

This leaves us with **USD 2006.70** remaining from our allocated budget of **USD 5000**.

7.5 Risk Management

The importance of risk management was realized by us nearing the last 2 weeks of SVD as our mitigation strategies were actually helping us resolve our issues fairly quickly. We identified 5 major risks at the start of the semester which encompassed nearly all the technical aspect of our project planned for the spring semester. These risks were kept track of in our team's notion made for project

Table 11: Fall Demonstration Procedure

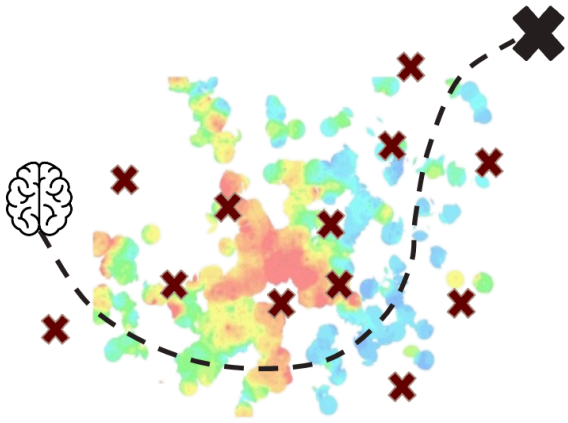
Steps	Procedure
1	Place the heat sources, and measure & record the fire pit locations from take-off (origin).
2	Switch on all heat sources and input fire pit locations into the ground station laptop.
3	Power on the RC Transmitter; make sure the kill switch is engaged on the RC transmitter during the on-ground setup phase to avoid unforeseen accidents.
4	Connect the batteries, and establish a wireless telemetry link between FCU and GCS.
5	Connect the portable display, keyboard, and mouse; start the sensor script on the mission, visualize the data on RViz, and enable ROS bag recording with necessary information.
6	Launch the autonomy stack, state-estimation, and fire perception station and disconnect all the I/O accessories from the Mission computer.
7	Begin a timer, perform safety checks, and arm the UAS.
8	Give a desired goal location and press start for UAS exploration from the UI
9	UAS autonomously navigates to the given location while avoiding obstacles.
10	Mission progress is displayed on the UI, press Stop once the mission is complete.
11	Make sure the kill switch is engaged on the RC transmitter and turn down heat sources.
12	Reconnect I/O accessories to visualize and evaluate the fire map in a world frame.
13	Analyze the accuracy of hotspots displayed in UI compared to measured ground truth.

Table 12: Verification Criteria

S.No.	Criteria
V1	Detect all hotspots with at least 70% accuracy.
V2	Localize fire positions up to 5m of distance in front of the drone.
V3	Localize itself and fire with at least 10 Hz.
V4	Localize itself within a drift of 4%.
V5	Navigate obstacles with a separation of atleast 5 meters.
V6	Have a flight time of more than 5 minutes.
V7	Communicate with the drone up to 150 meters.
V8	UI displays an updated fire map throughout the duration of the flight.

management including a section for their occurrence, and mitigation action. Out of the risks shown in table 13, two risks are still marked as active. These are:

- **R4: Unavailability of testing space**: This risk has been mitigated throughout our semester. We took the bold decision of doing our SVd outdoors to show realism. We changed 2 locations



* Refer to Risk 4 for further information

Figure 25: Schematic of the FVD run

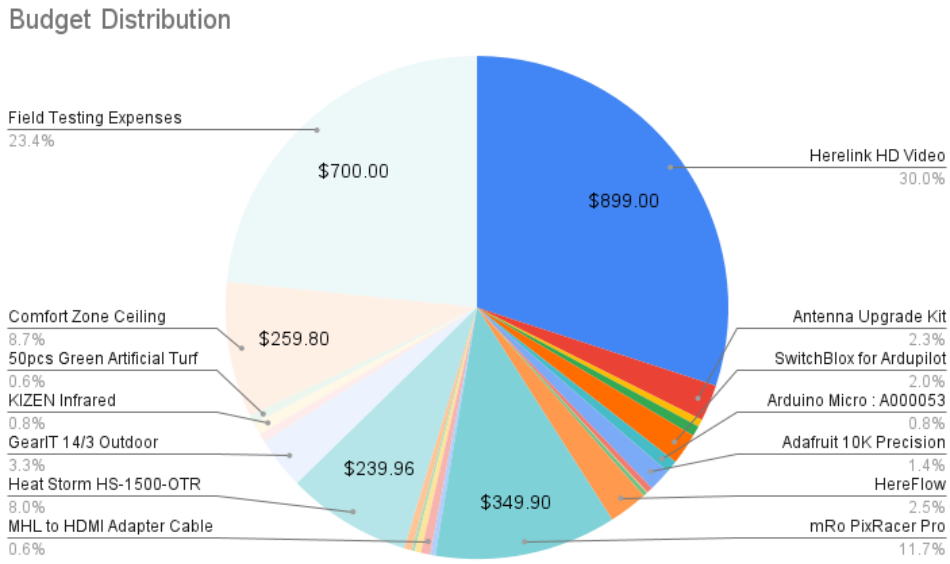


Figure 26: Spend Distribution

since the beginning of this semester. First we landed on Hawkins, which looked like the perfect site with smoke permissions, empty parking lots, and nearby shelter. However a change in city administration forced us to look somewhere else. After visits to Gascola and Nardo, we converged on Nardo due to its open fields, and power generator. Now as we move towards our VD next semester, we would need a location to showcase our system in smoke. The risk has been hence marked active one again as certain mitigation actions are in play in an effort to find a suitable location on and off campus with smoke permission being the top focus right now.

- **R2: AirLab stacks not Meeting PR**: This risk was also mitigated this semester allowing us to

Date Requested	Quantity	Part Name	Unit Price	Total Price
04/02/2024	1	Herelink HD Video Transmission System (V1.1)	$899.00	$899.00
04/02/2024	2	Antenna Upgrade Kit	$35.00	$70
04/02/2024	2	Herelink Air Unit (V1.1) Ethernet Cable	$7.00	$14
04/02/2024	2	Herelink HDMI Cable	$10.00	$20
04/03/2024	1	SwitchBlox for Ardupilot	$60.00	$60.00
04/05/2024	1	Arduino Micro : A000053	$23.93	$23.93
04/05/2024	6	Adafruit 10K Precision Epoxy Thermistor [ADA372]	$7.19	$43.14
04/05/2024	1	MPM3610 5V Buck Converter	$11.64	$11.64
04/05/2024	7	Digikey JST-XH 2 Pin Male Vertical	$0.19	$1.33
04/05/2024	7	2-Pin Female JST XH-Style Cable	$1.15	$8.05
04/07/2024	1	HereFlow	$75.00	$75.00
04/07/2024	1	mRo PixRacer Pro	$349.90	$349.90
04/09/2024	1	Seadream 4K Micro HDMI to HDMI Cable 1FT 2Pack	$9.88	$9.88
04/09/2024	1	MHL to HDMI Adapter Cable	$17.99	$17.99
04/09/2024	1	MHL to HDMI Adapter Dongle	$11.99	$11.99
04/09/2024	1	2-in-1 Micro USB to USB Adapter (OTG Cable + Power Cable)	$5.97	$5.97
04/09/2024	1	UGREEN USB 3.0 to Gigabit Ethernet Adapter	$14.99	$14.99
04/11/2024	4	Heat Storm HS-1500-OTR Infrared Heater, 1500-watt	$59.99	$239.96
04/11/2024	2	GearIT 14/3 Outdoor Extension Cord (100 Feet)	$49.98	$99.96
04/11/2024	1	Amazon Basics Open Reel Fiberglass Tape Measure	$16.79	$16.79
04/11/2024	1	KIZEN Infrared Thermometer Gun	$22.99	$22.99
04/11/2024	1	50pcs Green Artificial Turf Stakes Staples	$16.99	$16.99
04/11/2024	4	Comfort Zone Ceiling Mounted Space Heater	$64.95	$259.80
-	-	Field Testing Expenses	$700.00	$700.00
			TOTAL:	$2,993.30

Figure 27: Parts List

comfortably switch to a stable stereo matching pipeline for localization of the fire hotspots, when the learning based approach developed by airlab reserchers didnt work for us. This risk has been again marked *Active* as we need to look for stable pipelines again for our planned modules for Fall semester.

Table 13: Risk Management

Risk ID : Requirement	Aa Name	Risk Type	Status	Conseq...	Likeliho...	Occurre...	Mitigation Action
R5 :: MNF4, MF1, MF6, MF9	Supply Chain Snags and Issu	Prog, Schedule, Cost, Techn	Mitigated	Medium	Low	No	Order parts beforehanc
R3 :: MNF2, MNF3	Hardware Failure	Schedule, Cost, Technical	Mitigated	High	Low	Yes	Keep spare parts and p
R1 :: MNF1, MNF3	Delay in Hardware Developm	Schedule, Cost	Mitigated	Medium	Medium	Yes	Secure 3d printing at M
R6	CPU OVerload on FCU: Drone		Mitigated	Medium	Low	Yes	Order better FCU
R4 :: MF9, MF10	Unavailability of Testing spac	Schedule, Prog.	Active	High	Medium	Yes	Keep 2-3 options as ba
R2 :: MF2, MF3, MF9	AirLab stacks not meeting PR	Schedule, Technical	Active	Low	Medium	Yes	Integrate dependent st
	Tent Flies Off					Yes	
	Dog Eats Drone					No	

Talking about the risks that were well mitigated: to avoid supply chain issues and delay in hardware development due to shared space in airlab, we acted fast and completed the hardware development early in Jan working throughout the winter break. Moving forward we kept spare parts, and stress tested components of the airframe. This knowledge about the limits of our system helped Phoenix survive 8-10 crashes till-date with some happening on the day of Encore as well. The major crash which happened inside airlab's flight cage had left Phoenix with a broken battery mount, landing gear and broken antenna mount. We were able to perform a full repair overnight thanks to the mitigations we undertook early as mentioned in Fig 28.

Some risks that were unexpected to us came in the form of a friendly dog coming to meet us every day at 4:30PM at Nardo. Half the team members were elated, and half worried about the drone. Also owing to the strong winds on some days, the tent, equipemtns, and even laptops nearly flew off.

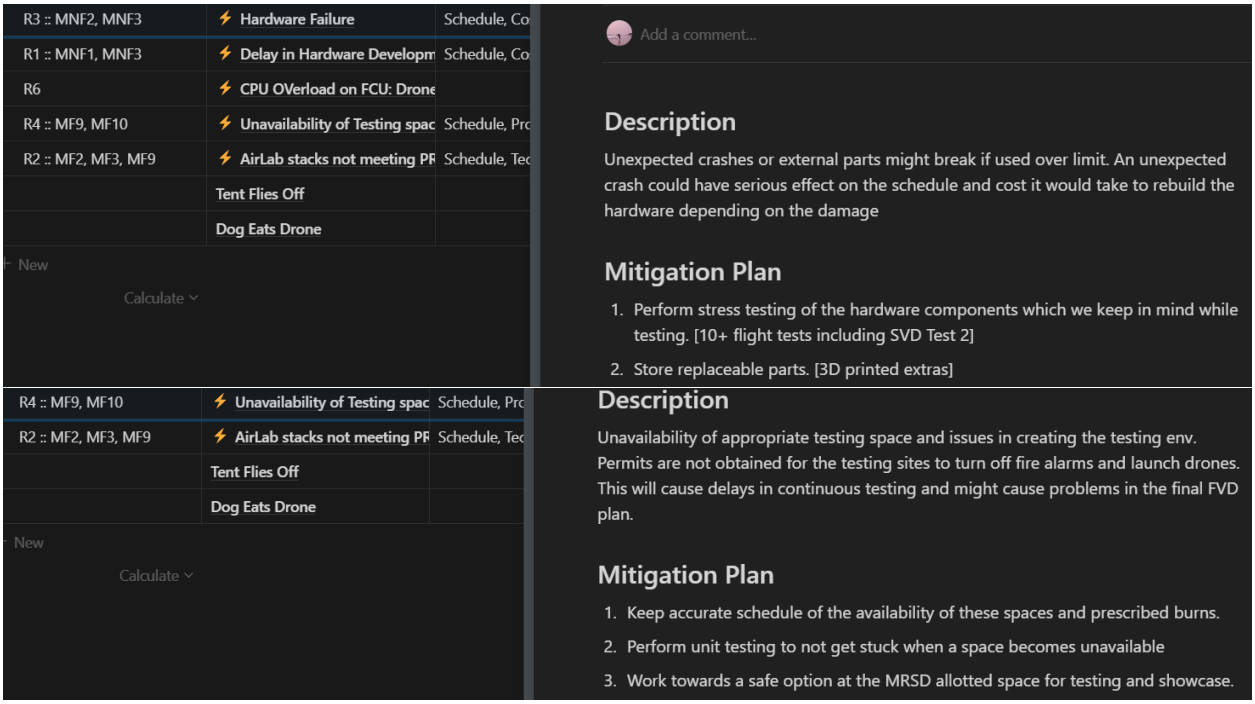


Figure 28: Risk Descriptions and Mitigations

8 Conclusions

8.1 Lessons Learned

Fail Fast

We realized through our experience with the fire localization module that as important as it is to dig deep into a subsystem and keep pushing, it is also important for engineers to know when to look for alternatives. In our case, we tried many approaches to localize the hotspots using the thermal cameras - dense depth estimation, learned disparity estimation, etc. - and what ultimately worked was an implementation of sparse depth estimation from scratch.

Parallelize

Specific to our team, two members were staying back in Pittsburgh during the winter break. This gave us the opportunity to get a headstart on the aerial robotic platform design and fabrication. Also, since our first external milestone - the AirLab Symposium - lied in the last week of January, that did not give us much time to develop and demonstrate other subsystems. So, we decided that the team would split up into 2 subteams, where the folks working on the hardware would continue with that, while the other 3 get a headstart on the perception stack. Similarly, later in the semester, when the hardware

prototype was ready, the former 2 team members shifted to the state estimation stack. This format of parallelizing work is what resonated well with our team.

Meetings can be messages

Although our team started with the traditional weekly meetings method, with 2-3 meetings with different agendas, we soon realized that what worked for everyone didn't exactly work for us, especially with our parallelization structure. So as we broke down into subteams (which we kept shuffling), what was more efficient for us were less often meetings that would help bring everyone on the same page, but more intra-subteam meetings, which helped progress significantly.

Classical is king, DL has to catch up

Another important lesson that we learned from our experience with the fire perception subsystem was the systems engineering process and the practicality of deep learning approaches versus classical methods. We realized the hard way the significant advantages classical methods provide wherever they can be deployed, over learning-based methods (learned disparity in our case).

Test, Test, Test

As stressed immensely in the Systems Engineering course, as well as through our Progress Reviews, we learned the importance of subsystem integration and testing. This was especially true with the complete system testing which we did quite regularly at Nardo Flight Testing Facility, and that is what proved to be immensely helpful in shaping our SVD, and figuring out the edge cases.

8.2 Key Fall Activities

Explore Better Airframes

As detailed during our Progress Reviews and the SVD, one of the critical weaknesses of our system is the low flight time as a result of the degradation of batteries, as well as the increased payload of the system. Thus, our first point of concern going into the Fall semester would be to consider the development of a new airframe. This would ideally streamline testing and deployment, owing to higher flight time.

Testing Site Logistics

As rightly pointed out by Professors and TAs, one of the most crucial logistical components of our testing is the testing site. This would immensely enhance the realism of the system, and enable us to test our system in a more subcanopy and wildfire-like environment. To do this, we wish to explore better alternatives (with preferably better proximity) for testing sites.

Autonomy

Finally, the main fall activity we're looking forward to is our key fall deliverable, which is the autonomy stack for the system. This would include mainly the dense depth estimation module (for obstacle avoidance, etc.) the navigation stack, and ultimately the complete system integration.

References

[1] 720 cv course lectures. http://16720.courses.cs.cmu.edu/lec.html.

[2] Agx orin datasheet. https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-orin/.

[3] Epipolar geometry. https://learnopencv.com/introduction-to-epipolar-geometry-and-stereo-vision/.

[4] Firefly mrsd team. https://mrsdprojects.ri.cmu.edu/2022teamd/team/.

[5] Gcs ethernet data transfer. https://discuss.cubepilot.org/t/ethernet-on-the-herelink-1-1/9766.

[6] Gcs herelink setup. https://docs.cubepilot.org/user-guides/herelink/herelink-overview.

[7] Gstreamer setup. https://gstreamer.freedesktop.org/documentation/tools/gst-launch.html?gi-language=c.

[8] Wildfire statistics. https://www.nifc.gov/fire-information/statistics/wildfires.

A Appendix

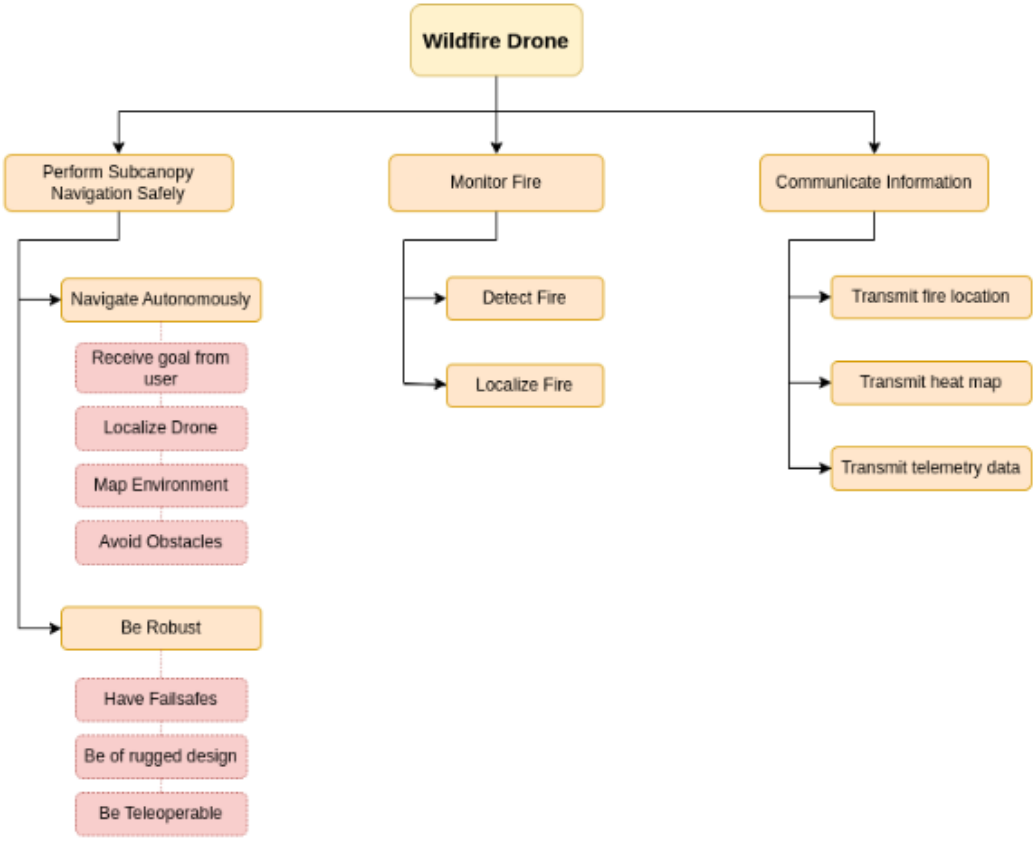


Figure 29: Objectives tree

Table 14: Risk ID 1: Delay in Hardware Development

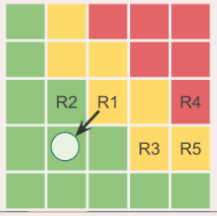
Risk Type	**Risk Owner**	**Date Submitted**	**Date Updated**
Schedule, Cost	Owner Name	12/01/2023	12/01/2023
Description	**Cause**	**Consequence vs Likelihood**	
Delay in Hardware Development at any stage in the testing phase	Can be due to shared inventory and delays in the outsourced manufacturing of certain parts		
Consequence			
If the hardware is not functional, there would be delay in the systems integration phase and in field testing.			
Mitigation Plan		**Expected Outcome**	
1. Parallely start working on setting up simulation to perform simulation testing 2. Use maximum off-the-shelf tested components to avoid manufacturing delays.		1. Simulation testing reduces the consequence of halt in software subsystems integration 2. OTS components reduce the likelihood of manufacturing delays.	

Table 15: Risk ID 2: AirLab stacks not meeting PR

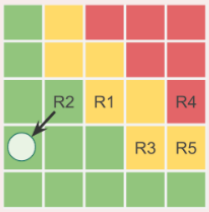
Risk Type	**Risk Owner**	**Date Submitted**	**Date Updated**
Schedule, Technical	Owner Name	12/01/2023	12/01/2023
Description	**Cause**	**Consequence vs Likelihood**	
AirLab stacks fail to meet the PRs which we expected.	While integrating the dependent stacks, their PRs get affected.		
Consequence			
This directly contradicts our promised PRs as part of the MRSD requirements. It also adds a schedule delay to find alternate options or work around negotiating PRs			
Mitigation Plan		**Expected Outcome**	
1. Parallely explore and document SOTA and possible open source solutions to estimate PR cushions. 2. Parallely integrate the dependent stacks as soon as individual stacks are tested.		Alternate options beforehand would helps us estimate the cushion our PRs would need and parallely integrating the stacks would let us know of any possible issue ahead in time.	

Table 16: Risk ID 3: Hardware Failures

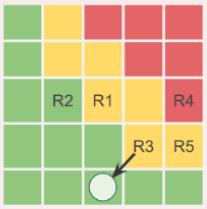
Risk Type	**Risk Owner**	**Date Submitted**	**Date Updated**
Schedule, Cost, Technical	Owner Name	12/01/2023	12/01/2023
Description	**Cause**	**Consequence vs Likelihood**	
Unexpected crashes or external parts might break if used over limit.	Replaceable parts like propellers, motors, prop guards get broken during random testing.		
Consequence			
An unexpected crash could have serious effect on the schedule and cost it would take to rebuild the hardware depending on the damage			
Mitigation Plan		**Expected Outcome**	
1. Perform stress testing of the hardware components which we keep in mind while testing. 2. Store replaceable parts.		Having an estimate of the hardware limits to reduce the likelihood and having spare parts can reduce the consequence.	

Table 17: Risk ID 4: Unavailability of Testing Space

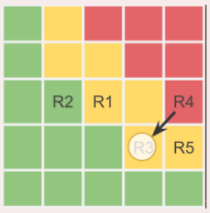
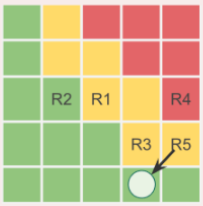
Risk Type	**Risk Owner**	**Date Submitted**	**Date Updated**
Schedule, Programmatic	Owner Name	12/01/2023	12/01/2023
Description	**Cause**	**Consequence vs Likelihood**	
Unavailability of appropriate testing space and issues in creating the testing env.	Permits are not obtained for the testing sites to turn off fire alarms and launch drones		
Consequence			
This will cause delays in continuous testing and might cause problems in the final FVD plan.			
Mitigation Plan		**Expected Outcome**	
1. Keep accurate schedule of the availability of these spaces and prescribed burns. 2. Perform unit testing to not get stuck when a space becomes unavailable 3. Work towards a safe option at the MRSD allotted space for testing and showcase.		Unit testing reinforces our system integration process and accurate scheduling keeps us on track to perform tests. A safer MRSD space option is to reduce the consequence.	

Table 18: Risk ID 5: Supply Chain Issues/Lags

Risk Type	**Risk Owner**	**Date Submitted**	**Date Updated**
Schedule, Cost, Technical, Programmatic	Owner Name	12/01/2023	12/01/2023
Description	**Cause**	**Consequence vs Likelihood**	
Supply chain issues while ordering parts.	Parts which need to be shipped from overseas but cannot due to permissions/ or some other reason		
Consequence			
If it's a integral part, it could cause us to not meeting our PRs and add more delay in system testing.			
Mitigation Plan		**Expected Outcome**	
1. Freeze hardware design ahead of time as hardware is an essential component of our system 2. Explore alternate hardware options beforehand 3. Order essential parts beforehand		Gives us an estimate of what all hardware components we need if the designed is frozen early, and what parts can be replaced with other options.	