ABSTRACT

Title of Dissertation: DETECTION AND SUPPRESSION OF AN INCIPIENT WILDFIRE USING A UAV

Patrick Collins
Masters of Science, 2025

Thesis Directed by: Professor Derek Paley
Department of Aerospace Engineering

This thesis investigates the development of an Uncrewed Aerial Vehicle (UAV) system designed to detect, localize, and suppress incipient wildfires. The system is designed with a commercial UAV in mind, but with an emphasis on the scalability and portability of its components. The key contributions of this research are the design and testing of a multi-spectral fire detection algorithm utilized in broad area surveys, a fire localization routine that utilizes the fire detection data to design an optimized flight path for revisiting potential fire sites, and the integration of a third-party drop system with the commercial UAV in order to deliver suppressant payloads to fire sites. These critical components are combined together to form an end-to-end fire process chain, which fuses autonomous and manual UAV functions into a procedure where one UAV can conduct fire detection, localization, and suppression activities utilizing two different mission profiles. Extensive testing of the fire process chain in totality and across its individual components is conducted, the results are thoroughly analyzed, and suggestions for future wildfire management UAV systems are provided.

DETECTION AND SUPPRESSION OF AN INCIPIENT FIRE USING A UAV

by

Patrick Collins

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Masters of Science
2025

Advisory Committee:
Professor Derek Paley, Chair/Advisor
Professor Fernando Raffan-Montoya
Professor Mumu Xu

Acknowledgments

While there is only one author whose name is published on a thesis, there are incalculable others whose support brings it to fruition. These pages are my attempt to acknowledge those who over the past two years have helped me grow the idea of the Crossfire system from a few notes on a piece of paper to an operational system. I can not include everyone on this list as it would rival the body of this thesis, however if you are not explicitly included please understand that my gratitude for your support is immeasurable.

I would first like to thank my advisor Dr. Derek Paley. Your guidance and mentorship has been generous and unwavering, and your commitment to the Crossfire project admirable. Your schedule may be busy, but you have *always* found the time the lead me through the nuances, difficulties, and successes of academic research. I am certain I would not have achieved as much as I have on this project without your sage advice. I wish you luck on your well-earned sabbatical next year.

To Dr. Mumu Xu and Dr. Fernando Raffan-Montoya, thank you for reading this tome and serving on my committee. You were both some of the first faculty relationships I developed at the University of Maryland, and through those connections I was able to access the resources and relationships needed to push this thesis to the finish line.

To my fellow researchers Eashwar Sathyamurthy, Lindon Luu, and Graham Buccheri, I have enjoyed immensely the long days, long nights, and exhausting car rides that we have shared over the course of this project. Your knowledge, dedication, and grit for simply getting things done is admirable, and I look forward to what is next in all three of your futures.

The Aerospace Engineering Department has played a critical part in both my education and in funding this project. I would like to extend my thanks to the entire faculty and staff of the department, with specific acknowledgment for Dr. Joseph Conroy whose mentorship extended beyond the classroom. Dr. Conroy was always willing to offer advice on UAVs, and his enthusiasm for the Crossfire project played no small part. The Maryland Robotics Center Lab manager Ivan Penskiy has also been stoic yet strong presence, always ensuring the labs are clean and professional. I would especially like to thank him for quickly ordering the random last-minute supplies I needed, and for never questioning *why* my project required eight reptile lamps.

I would like to thank the team at xFoundry who also in part funded this research. To the Director Amir Ansari, your vision for xFoundry and Crossfire is inspiring. Your belief in what Crossfire can become is felt across the team, and is a driving force in what we do. Special thanks are owed to Associate Director Dr. Phillip Alvarez. Dr. Alvarez plays a special role in Team Crossfire, whether it be in helping us form critical relationships, raising funds for our research, or serving as a technical advisor. If not for the team at xFoundry, much of this research would still only be an idea.

The Fire Protection Engineering Department at the University of Maryland has also been a critical partner in developing this thesis. Department Chair Dr. Arnaud Trouvé, Dr. Fernando Raffan-Montoya, and Dr. Stanislav Stoliarov have provided the resources, facilities, leadership, and mentorship necessary for the Crossfire Team in general. I'd like to think I've even learned a thing or two about fire dynamics— something I never expected to do prior to this project.

The team at the UAS Research and Operations Center have provided invaluable mentorship and operational support throughout this process. John Slaughter and Matt Scassero have been pillars on the Crossfire Team, providing critical guidance on UAS operations. Plus, Matt somehow found an incredible molé restaurant in Salt Lake City, an achievement unto itself. From the operational side, Rob Neuner, Grant Williams, and Alex Vrbka have been the linchpin to this whole operation. Their deep knowledge on UAS operations played a critical role in developing and testing the Crossfire system. Special thanks are owed to Grant, who has traveled hundreds of miles and spent many days breathing in fire fumes to help me bring my research forward bit by excruciating bit.

Speaking of fire fumes, the team at the Maryland Fire and Rescue Institute were the project's other linchpin. They granted us special access to their facilities, without which the Crossfire project would never have existed. Jeffrey Williams and the staff at the La Plata site were especially gracious in aiding us during long test days. They faced every situation with professionalism, an eye for safety, and a tinge of humor, even when our fires burnt their concrete (twice), melted their aluminum grates (also twice), or even started a small grass fire (thankfully this was extinguished within a few seconds and we didn't have to stress test our suppression drop system).

Last but not least, I would like to extend my gratitude to friends—old and new—and family. When I moved to the East coast for graduate school, I left behind many friends, yet their support for me while they themselves were raising families, planning weddings, attending graduate school themselves, or even living off-the-grid in a bus was unwavering. For that I am eternally grateful. Upon arriving to my new city I hardly knew anybody. Yet, I have met many others through shared loves of either motor sports, bar trivia, or simply exploring my new home. I am happy to call

them my friends as well.

In regards to my family, my sister Morgan has always been an unfaltering cheerleader, inspiring through the toughest of times and always reminding me to take a moment to appreciate the present. My sister Ali and my brother-in-law Richie have made DC feel like a new home. They were the only people in this city that I knew prior to moving, and their strenuous efforts to lend a sense of belonging and love have been tireless. To my parents, from a young age you inspired me to explore my love of science and math. It was because of this nurturing environment that all of this was possible. Perhaps just as importantly, you encouraged me to pursue my dreams, even if it meant moving 1,000 miles away. To the four-legged creature who will likely never read this, Vega, your patience with me over these last two years has been truly Herculean. Even as I write this you sit at my feet patiently waiting for me to finish so we can play fetch. Everyday you've enthusiastically greeted me as I've walked in the door, whether I've been gone for two hours or twelve. Its those moments that have kept me going. And last but most importantly—to my partner Angie. When I asked if you were okay with me quitting a steady job so that we could move across the country and I could attend grad school, you answered with an emphatic "Yes!". This meant leaving behind friends and family, downsizing our apartment, and taking on the extra tasks that make a home—all while starting your own career. Words can not describe my gratitude for the sacrifices you have made so that I could pursue my dreams. Your love and support through the stressful days and long nights of the past two years have been steady, relentless, unhesitating, unfaltering, and any other synonym I could helplessly use in attempting to describe it. This achievement is as much yours as it is mine. I love you.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

AGC	Automatic Gain Control
AGL	Above Ground Level
DBSCAN	Density Based Spatial Clustering of Applications with Noise
DCNN	Deep Convolution Neural Networks
ENU	East-North-Up
FAA	Federal Aviation Administration
FOV	Field of View
FPE	Fire Protection Engineering
FPS	Frames Per Second
GCS	Ground Control System
GPU	Graphics Processing Unit
KML	Keyhole Markup Language
LLA	Longitude-Latitude-Altitude
MPH	Miles per Hour
NWCG	National Wildfire Coordinating Group
OCT	Optical Character Recognition
POI	Point of Interest
POV	Point of View
RGB	Red-Green-Blue
RTK	Real-Time Kinematic
SDR	Scene Dynamic Range
TFR	Temporary Flight Restriction
TSP	Traveling Salesperson Problem
UAS	Uncrewed Aerial Systems
UAV	Uncrewed Aerial Vehicle
UROC	UAS Research and Operations Center
VRP	Vehicle Routing Problem
WUI	Wildland Urban Interface
YOLO	You Only Look Once

Chapter 1: Introduction

1.1 Statement of Problem and Motivation

As the effects of climate change become more tangible year after year, droughts and fire weather seasons have generally grown longer. This has led to a global increase in the potential for and occurrence of wildfires [5]. The damage and destruction these infernos can impart need not be speculated. In 2023 along the Wildland Urban Interface (WUI) on the Hawaiian Island of Maui, a destructive blaze grew from what is speculated to be a downed power line. In the ensuing hours, over 100 lives were lost. The material damage was accounted to be about 2,200 structures and $5.5 billion in property damage [6]. Maui is just one of countless examples. Another wildfire on American soil occurred around Los Angeles in January of 2025. At the time of this writing, preliminary estimates show that 28 people died due to this blaze, and that there were upwards of $250 billion in damages. Over 150,000 were temporarily or permanently displaced [7]. This is to say nothing of the intangible environmental effects such wildfires have.

Typical methods of fighting wildfires have remained somewhat fixed over the past few decades. These methods often employ some mix of satellite imagery, ground sensors, manned aerial and ground systems, and battalions of fire fighters. Looking through the Wildland Fire Incident Management Field Guide, published by the US government organization National Wildfire Coordinating Group (NWCG) in 2014, one will find a variety of tools and equipment listed as essential to managing a blaze [8]. Yet there is not a single mention of Uncrewed Aerial Vehicles (UAV)—sometimes referred to as Uncrewed Aerial Systems (UAS)—in the entirety of the 160 page guide. When originally written, this was for a good reason. The Federal Aviation Administration (FAA) often issues Temporary Flight Restrictions (TFR) for areas in which wildfire fighting operations are active. This is to clear the air space so that pilots of manned aircraft can safely perform their duties. At the time, reliable, low-cost UAVs were only just emerging in the marketplace. Their capabilities and cost-effectiveness have generally improved since then. This has even been noted by some government studies—for example in a case applied to military surveillance, it is more cost effective to operate unmanned systems than manned systems [9]. These findings have generated interest in expanding the domain of UAVs.

The application of UAVs and their sensors—oftentimes those being thermal and RGB cameras—in the domain of wildfire management has recently picked up steam. There have been many published papers on algorithms that utilize computer vision techniques to detect fires with ordinary cameras [10], as well as data sets collected to specifically aid in the development of hybrid fire detection techniques, i.e. those that utilize both thermal and RGB cameras [11]. Groups have derived methods to optimize UAV observation flight paths [12] as well as designed controllers that guide UAVs along fire boundaries [13]. Work has been done to simulate multi-UAV systems that can both detect and suppress wildfires [14]. Yet, there appears to be a lack a published literature on UAV systems that have demonstrated their abilities to search for, detect, and suppress wildfires.

The motivation of this thesis is to develop a UAV system that can perform the end-to-end tasks of searching for, detecting, and ultimately suppressing incipient wildfires. For the purposes of this thesis, incipient fires are defined as those that can be extinguished with a handheld fire extinguisher. By designing fire detection algorithms, fire localization methods, path-planning routines, and other critical software and hardware elements, these components are integrated together into an end-to-end system pipeline. This research aims to analyze and improve the capabilities of such a pipeline, ultimately contributing crucial knowledge to the nascent field of UAV fire management.

1.2 Relation to State of the Art Work

The application of UAV systems to wildfire management has been a known possibility for decades. However, due to previously high costs and difficulties in acquiring permissions to conduct testing, thorough research on UAVs as applied to wildfire detection and suppression only recently emerged. This thesis aims to contribute to this growing body of work.

Much of the research that will be analyzed here falls into three constituent categories: designing multi-agent systems for fire detection and suppression, developing detection algorithms that utilize computer vision techniques, and finally designing UAVs capable of suppressing fires.

Some of the earliest work on forest fire monitoring using small UAVs was conducted by Luis Merino in 2010 [15]. Merino demonstrated how small single-rotor and fixed-wing UAVs could utilize their mounted cameras and apply computer vision algorithms to characterize the shape and spread of wildland fires. The paper concluded that it was feasible to design swarms of UAVs than can conduct wildfire monitoring activities, but Merino did not extend his work to the task of suppressing fires.

A number of authors took a deeper dive into the art of the possible for utilizing UAVs in fire management. Afghah et al. designed a leader-follower network that could be utilized to monitor wildfires in remote areas [16]. In their architecture, a fixed-wing UAV serving as a leader agent guides a swarm of follower quadrotor UAVs that utilize on-board sensors to detect wildfires. Their design was proven to be robust through simulated tests, however it was not implemented on a real-world system. Panahi et al. designed a similar system that simplified the detection architecture (employing a single UAV) but that allowed the UAV to communicate and send instructions to autonomous ground vehicles that would suppress the fire [17]. This research was also tested via simulation. While work in the area of multi-agent UAV path-planning as applied to fire management is important, the lack of real-world system testing speaks to the emerging nature of this field.

Research on fire detection has been a well-trodden topic as of late. Common methods often employ object detection algorithms via Deep Convolution Neural Networks (DCNN) or classical computer vision techniques [18, 19]. Typically, researchers have utilized single-camera UAVs for detection, but more recent research has begun to fuse RGB camera feeds with thermal camera feeds [20]. A major step in this direction occurred in 2022 when Chen et al. published a publicly available dataset named FLAME2 [11]. This was a significant contribution to the research space as it was the first widely available public dataset to include "dual-feed" videos of wildland fires taken by aligned thermal and RGB cameras mounted on a UAV. The UAV was manually piloted and observed a prescribed wildland fire in Northern Arizona. Due to the difficult nature of acquiring approval to observe proscribed burns (or wildfires in general) with UAVs, the publishing of FLAME2 opened up the floodgates on thermal and RGB data fusion research for UAV wildfire detection systems.

The branch of research on fire suppression systems deployed by UAVs is less developed than the previous categories. Much work has been dedicated to designing "plug-and-play" systems that can activate a suppressive system when a fire is detected, but few papers have been published where suppressant has actually been drop (or sprayed) from a UAV. Aydin et al. designed a drop system for a custom-built hexacopter that could deliver commercially available fire extinguishing balls to a fire site [21]. These balls were heat activated and would explode and distribute suppressant material. However, concerns about liability if testing such a system injured someone led the authors to decide to not test their drop system on a UAV. Jahan et al. also contributed to the literature with a design for a gas sensor detection system[22]. They did not conduct suppression testing, but contributed a small literary analysis on papers for UAV-mounted suppression systems. Only one such paper had conducted extensive testing. Wang et al. constructed a UAV mounted hose designed to extinguish fires in high-rises [23]. While it required a connection to a source of water (e.g., a fire truck, fire hydrant, etc.) and would have limited benefit for wildfire applications, Wang et al.'s work represented a step forward in the space of UAV-mounted suppression systems.

Deeper dives in the field of UAV wildfire management are available in literary surveys written by Haeri et al. and Keerthinathan et al. [18, 19]. These surveys paint a picture of an emerging field.

1.3 Technical Approach

This thesis presents the development of a semi-autonomous, end-to-end fire detection and suppression system utilizing a commercial quadrotor UAV. Emphasis is placed on developing fully autonomous detection algorithms that identify points of interest (i.e., potential fires), and integrating these algorithms into a process which aims to suppress all fires in a defined space.

The system employs pre-planned broad area surveys that ensure complete coverage of the test area. Two camera payloads—an RGB camera and a thermal camera—are broadcast to a ground station. A software program is ran on the ground station computer that employs a different fire detection algorithm for each of the two video streams. After video processing aligns the two video points of view (POVs), the visual data is fused together and a heuristic algorithm determines if a fire is detected or not.

Upon detection, GPS data estimated from a laser rangefinder aligned with the RGB camera is analyzed. As a single detection is likely to occur over multiple frames of video, a clustering algorithm is employed on the laser rangefinder data to identify points of interest (POIs). These POIs are crucial for designing revisit/suppression missions.

Once POIs are identified, a revisit/suppression mission is designed utilizing a path-planning algorithm. Having a separate revisit/suppression mission for a one UAV system is not necessarily the optimal approach (as opposed to just suppressing fires when they are seen). However, designing the revisit/suppression missions was an important aspect of this research as it more accurately reflects requirements of multi-UAV systems—for example, if one UAV focuses on detection and the second on suppression. A process was designed and is discussed regarding how a pilot interfaces with the commercial UAV employed in this research to carry out suppression tasks.

All of these components are strung together into a single process chain. This end-to-end process is then subjected to testing and analyzed for viability regarding in-field deployment.

1.4 Contributions of Thesis

The primary contributions of this thesis are:

1. **Multi-spectral Fire Detection Algorithm:** An algorithm that utilizes thermal and RGB

cameras in order to detect fires is developed and implemented in a system consisting of a commercial UAV, its peripheral accessories, and a ground station laptop. This algorithm is able to run in real-time and can replace the need for manually inspecting a testing space for fires so long as the UAV observes the entire testing space through a broad area survey.

2. **Fire Localization and Suppression Route-Planning:** A method to estimate fire positions utilizing an embedded laser rangefinder within an RGB camera is developed and tested. This method implements clustering algorithms with the laser rangefinder data as inputs in order to estimate the fire positions, and from those positions a revisit/suppression flight trajectory for points of interest is generated for a UAV that has conducted a broad area survey of a testing space.

3. **Proof of Concept of an Integrated Suppression System:** A third-party payload drop system is integrated with a commercial UAV in order to demonstrate a proof of concept fire suppressant payload delivery system. Testing that demonstrates the ability of this proof of concept system to deliver a suppressant payload to a target is conducted.

4. **Development of a Semi-Autonomous Mission Framework for Fire Detection, Localization, and Suppression:** A comprehensive mission framework that ties together all of the previous contributions is developed. This framework begins with a broad area survey of a testing space, continues with fire localization and revisit/suppression flight trajectory planning, and ends in suppressant delivery flights to any fires within the test space. This mission framework utilizing a commercial UAV is rigorously tested in multiple scenarios.

1.5 Outline of Thesis

Chapter 2 provides a comprehensive background of the requirements on the developed system as laid out by competition guidelines as well as a technical review of the experimental platforms and sensors used to develop the Crossfire system. Chapter 3 details the development of individual components critical to the system, such as fire detection algorithms, positions of interest localization, and fire suppression utilizing a third party drop system. This chapter explains these contributions in detail and presents critical test data generated in the development of these components. Chapter 4 discusses how the individual components of the Crossfire system fit together in a semi-autonomous mission framework referred to as the fire process chain. Data from comprehensive testing is presented and analyzed. Finally, Chapter 5 concludes this thesis with a summary of research contributions and suggestions for future work to improve both the Crossfire system and autonomous UAV systems in general when applied to fire detection and suppression activities.

Chapter 2: Experimental Requirements, Platform, and Sensors

This chapter begins by providing context regarding the engineering competition that this thesis research was completed in conjunction with. Next, the chapter provides essential background information regarding the technological platforms utilized for this thesis.

2.1 XPRIZE Wildfire Competition

XPRIZE is an organization that strives to "inspire and empower humanity to achieve breakthroughs that accelerate an abundant and equitable future for all." [24] XPRIZE fulfills this mission by establishing and operating prize competitions across various domains—often pertaining to using technology to solve ecological problems.

In 2023, XPRIZE establish their Wildfire competitions, aiming to discover new methods of identifying and responding to wildfires. There are two tracks to the XPRIZE Wildfire competition, and the focus of this thesis is specifically Track B: Autonomous Wildfire Response.

2.1.1 Competition Guidelines

The Autonomous Wildfire Response track of the XPRIZE Wildfire was announced in 2023, had an initial qualifying round in May 2024, and is planned to go through the Summer of 2026. The unofficial schedule as published in the XPRIZE Competition Guidelines can be seen in 2.1.

Figure 2.1: XPRIZE Autonomous Wildfire Response Competition Unofficial Schedule [1]

The following information regarding the competition rules and requirements are lifted directly from the Competition Guidelines [1]. The explicit description of the competition is

> In the Autonomous Wildfire Response track, teams have 10 minutes to autonomously detect and suppress a high-risk fire in a 1,000 km^2, environmentally challenging area, leaving any decoy fires untouched.

The judges for the semi-finals and finals stages of the competition will emphasize a few key characteristics that any winning system must have.

- Quick, accurate, and precise detection of any fires.
- A rapid response to and full suppression of those fires.
- Fully autonomous, integrated solutions. i.e., “Human-on-the-loop” autonomy.
- Smart detection—detection algorithms can differentiate wildfires from false positives (such as water vapor) and non-threatening, non-moving fires (such as a camp fire).
- Functionality in high winds and complex terrain.
- Varied connectivity architectures—at least two different types of communication protocols for robustness.
- Appropriate safety checks embedded within the system.

- Scalability—the system is designed to be cost-effective.

XPRIZE designed these requirements to be strenuous. They are aware that any one team is unlikely to fulfill all of the requirements for the competition. Thus, collaboration among competitors is encouraged, and most teams have taken the approach to specialize in sub-areas of the competition requirements.

Lastly, the Competition Guidelines often make references to "incipient stage wildfires"—the types of wildfires any autonomous solution designed for the competition should be equipped to suppress. In the Guidelines, this is explicitly defined as

> **Incipient Stage Wildfire** - A fire which is in the initial or beginning stage and which can be controlled or extinguished by portable fire extinguishers, Class II standpipe or small hose systems without the need for protective clothing or breathing apparatus.

2.1.2 Team Structure

The University of Maryland has entered the XPRIZE Wildfire Competition with a team denominated as Crossfire. Much of the work in this thesis may refer to or be stated as part of the "Crossfire System", which simply indicates that specified systems are part of Crossfire's submission to the XPRIZE Wildfire Competition.

The Crossfire team is a collaboration between multiple departments and organizations within the University of Maryland system. There are a number of collaborators, but there are four primary parties: the Fire Protection Engineering (FPE) department, the Aerospace Engineering department, XFoundry, and the UAS Research and Operations Center (UROC). The expertise of the FPE and Aerospace Engineering departments are self-evident. XFoundry is an organization within the University of Maryland that aims to fund research, encourage entrepreneurship, and aid teams whose ultimate aim is to commercialize their products and launch ventures funded in part by XFoundry. Finally, UROC is an organization the specializes in UAS operations, provides pilots for testing UAS systems, and provides expertise in designing and integrating UAS systems.

2.2 Quadrotor UAV Platform

2.2.1 DJI M300 RTK

During the course of this research, one commercial UAV was utilized. This was the DJI M300 RTK. The M300 RTK is a quadrotor UAV platform with a built-in flight controller system, a 6-directional sensing and positioning system that is complimented with Real-Time Kinematic (RTK) Positioning, an FPV camera, and a gimbal system built to allow for up to two gimballed cameras [25]. The M300 RTK (which for brevity, will now be referred to as the M300) is a high-end industrial quadrotor UAV that is well-equipped for use in difficult tasks. It was selected for the Crossfire project as it is a UAS with enough modularity to test and demonstrate key aspects of the Crossfire system, it is capable of integrating with camera payloads well-suited to the task of fire detection, and—perhaps most importantly—the M300 was already on-hand and the Crossfire team could save significant money by using it in the first iteration of the Crossfire system.

The M300's positioning system leverages RTK information that improves position estimates. The RTK system compares the measured GPS position of grids of base stations that have defined absolute positions. Such a construction allows for an RTK receiver to have knowledge of real-time, systematic GPS errors given its position to RTK base stations and can adjust GPS estimates accordingly. RTK systems have been shown to achieve centimeter-level accuracy [26], which is well-above the requirements needed for the system in this research.

The M300 was designed with commercial operators in mind, so there are additional bells and whistles worth mentioning. The M300 has an integrated collision avoidance system that utilizes an array of cameras pointed radially outwards from the center of the UAV. It has built-in LED lights, including a strobe light system which is of importance regarding the third-party payload drop system that will be discussed later. It is also equipped with redundant Inertial Measurement Units (IMUs) and barometers, ensuring that it can produce reliable flight data.

The M300 utilizes a proprietary flight computer. This flight computer mostly prevents modifications (short of minor tasks integrated with DJI's software development kit), however it is user-friendly in regards to mission planning. Missions are usually designed by identifying waypoints and storing them in a KMZ file, which is a zipped version of a Keyhole Markup Language (KML) file [27].

A user can interface with the M300 via another DJI product: the Smart Controller. The Smart Controller can run DJI's Pilot 2 application, where a user can pilot the M300, load and plan missions, interface with camera payloads, and a plethora of other tasks [28]. There are two important capabilities the Smart Controller has that will be seen to play important parts in this research. Firstly, up to two Smart Controllers can be connected to a single M300 UAV and up to three different video streams can be transmitted to both controllers at once. Secondly, the Smart controllers can output a specific video transmission or a duplication of their screens via HDMI. This architecture allows for both RGB and thermal video payload streams to be inputted to the ground station computer in real-time.

Other important components of the M300 system used in the Crossfire system are a FLIR Vue TZ20 thermal camera, a Zenmuse H20 RGB camera (with a built-in laser rangefinder), and

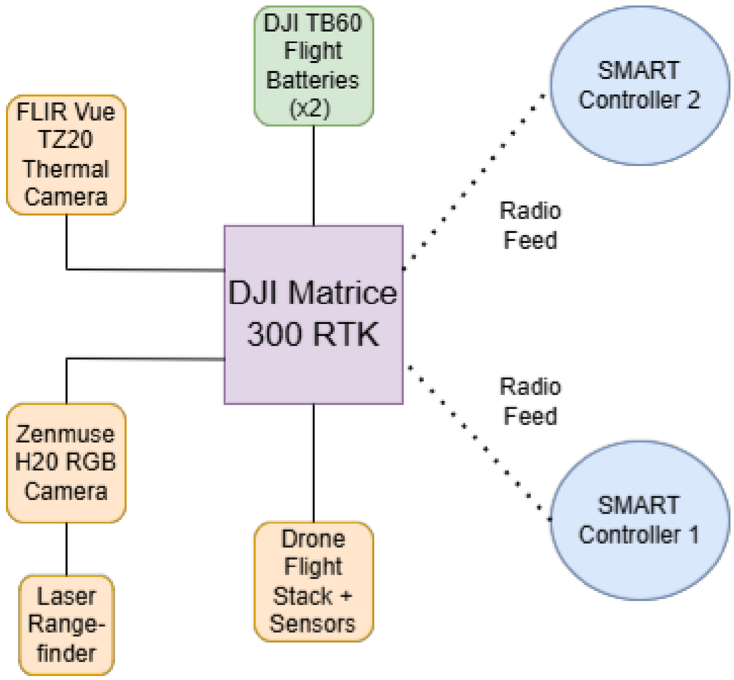


Figure 2.2: M300 hardware architecture with Smart controllers.

the DJI TB60 Flight Batteries. The system hardware architecture as it comes "off-the-shelf" can be seen in the block diagram in 2.2.

A table with important technical details regarding the M300 is presented in 2.1. The data was taken directly from the M300 User Manual [25].

Lastly, front and back images of the M300 with critical components labeled can be seen in 2.3. The labeled components are those that do not come "built-in" with the M300, meaning they must be installed prior to every flight.

2.2.2 Payload Systems

For the purposes of this research, the M300 is equipped with two gimballed camera payloads: a Zenmuse H20 and a FLIR Vue TZ20—the H20 being an RGB camera, and the TZ20 being a thermal camera.

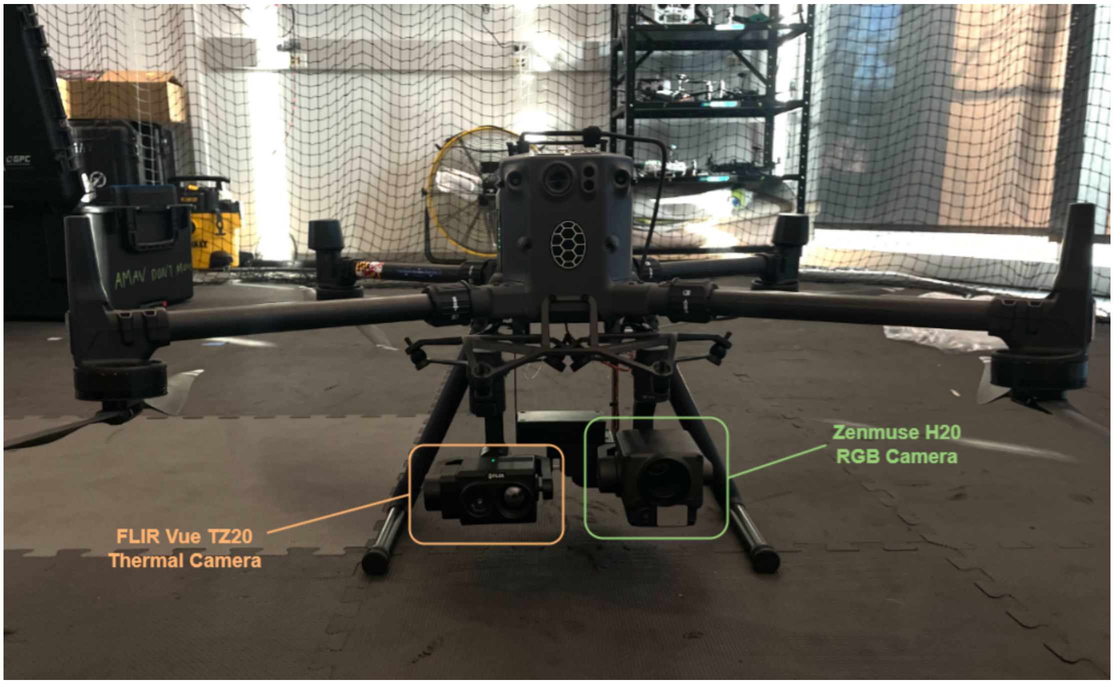
Characteristic	Value
Dimensions (unfolded, propellers excluded)	810 x 670 x 430 mm
Weight (batteries excluded)	3600 g
Max Payload	2700 g
Max Takeoff Weight	9000 g
Max Ascent Speed	6 m/s
Max Descent Speed	5 m/s
Max Horizontal Speed	23 m/s
Max Service Ceiling	5000 m ASL
Endurance (no payload)	55 min
Operating Frequency	2.400 - 2.4835 GHz; 5.725 - 5.850 GHz
Operating Temperature	-20° to $50^{\circ}C$

Table 2.1: M300 Technical Specification

The Zenmuse H20 is a specially built camera designed to integrate with DJI's gimbal connector. This means that the H20 can be interfaced with through the Smart Controllers, and the yaw and pitch angles of the H20 can be tightly controlled by the user. From a forward facing position, it has a pitch range from -120° to 30°, and a yaw range of $\pm 320^{\circ}$. The H20 has a variable zoom, but is best suited for 1x zoom for the wide angle lens and a 5x zoom for the zoomed lens. The video is captured at 30 frames per second (FPS), either at resolution of 1920 x 1080 or 3840 x 2160 (commonly referred to as 4K) [29].

The Zenmuse H20 also has a built-in laser rangefinder. This laser rangefinder operates at a wavelength of 905nm, and has a range of 3-1200 m. The measurement accuracy is recorded as $\pm 0.2 + 0.0015D\, m$, with D being the distance to the surface. As an example, at a range of 100m, the rangefinder would have an accuracy of $\pm 0.35m$. The rangefinder reports the position of what it is pointing at in a Longitude-Latitude-Altitude (LLA) frame [29].

The FLIR Vue TZ20 is a third party thermal camera designed by Teledyne FLIR. The Vue TZ20 contains a long-wave infrared Boson thermal camera mounted in a chassis designed to interface with DJI's camera gimbal system. Therefore the Vue TZ20 easily integrates with the



(a) M300 front view with cameras labeled.

(b) M300 back view with cameras labeled.

Figure 2.3: M300 Labeled Front 2.3a and Back 2.3b Views

M300. It measures the relative temperature of objects (i.e., it is not a radiometric sensor) and is capable of a 95° field of view, or as narrow as an 18° field of view when zoomed. The Vue TZ20's

Characteristic	Zenmuse H20	FLIR Vue TZ20
Type of Camera	Visual light RGB camera	Longwave infrared thermal camera
Resolution	**1920 x 1080**	640 x 512
Frame Rate	30 fps	30 fps
Field of View	82.9°	95°
Zoom	**1x**	**1x**
Color Pallette	N/A	**White Hot**
Temperature Range	N/A	**Hot range:**-40° to 550° C

Table 2.2: Payload Camera Characteristics, including selected settings for the Crossfire system in **bold**.

video streams at 30 FPS with a resolution of 640 x 512. The Vue TZ20 has a number of settings worth mentioning. For one, a number of color pallettes are available; for this project the White Hot pallette was selected. Secondly, the Scene Dynamic Range (SDR) setting allows the choice of one of two temperature ranges: Normal (-25° to 135° C), and Hot (-40° to 550° C). This research employed the Hot range, which sacrificed some temperature resolution for a much larger temperature range. Lastly, the Vue TZ20 employs Automatic Gain Control (AGC), which remaps the color range of pixels to different temperature ranges as the scene dynamically changes. This has both benefits and drawbacks, but it can be stopped by initiating an AGC lock. When AGC lock is initiated, the color-to-temperature mapping is kept static at whatever the mapping was for the frame at the time the lock was initialized [30]. This necessitated a calibration procedure to be integrated into the fire detection process chain.

The characteristics of both the Zenmuse H20 and the FLIR Vue TZ20 as well as some selected settings for the Crossfire Systems are summarized in Table 2.2.

The M300 has the ability to align the gimbal angles of both cameras so that the RGB and the thermal camera are pointed in the same direction at any time. This functionality is important to this research as it allows for data fusion techniques to be used for the purposes of fire detection, much like Chen et al. achieved when creating the FLAME2 dataset [11].

2.2.3 Suppressant Drop System

The DroMight Talon V1 is an integrated drop system for the M300. It allows the user of an M300 to attach a droppable payload via a pin mechanism to the underside of the UAV. The payload can be released by activating the M300's strobe lights via a Smart Controller. This is made possible due to a photo-resistor circuit that is placed around the underside strobe light of the M300. The Talon V1 itself weighs about 350 grams. Its maximum payload weight is limited by the maximum takeoff weight of the M300 [31]. Accounting for the camera payloads, the M300 system as configured in this research can support suppressant payloads of roughly 1 kg. While this weight may not be enough suppressant to extinguish fires of any significant size, it is certainly large enough for creating a demonstrable system.

2.2.4 Ground Control System Components

The Ground Control System (GCS) is defined to be made up of the following components: two Smart Controllers, the two HDMI cables, the two HDMI capture cards, and the ground station laptop that runs the fire detection software. The laptop used for this research is a Dell Precision 3580. The laptop runs of a Windows 11 operating system. The software for this research is mostly written in Python 3.X. A number of libraries were used, including OpenCV, Ultralytics' YOLOv8, and others. A more detailed description of the Ground Control System follows in Chapter 4.

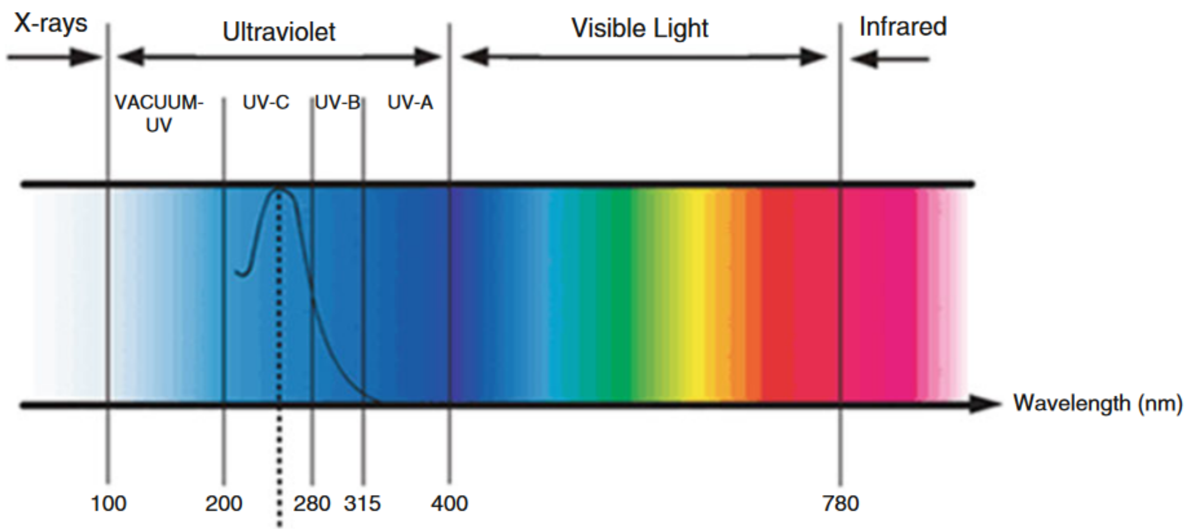


Figure 2.4: The Electromagnetic Spectrum near visible wavelengths. Figure originally from [2]. License Number: 5975680151600

2.3 Onboard Sensory Perception

Three components of the onboard sensory package—the RGB camera, the thermal camera, and the laser rangefinder—are critical to the fire detection system designed for this thesis. Therefore, this section will dive a bit deeper into the technical aspects of each type of sensor. The information presented here is critical to garnering a deeper understanding of the algorithms designed for this research.

2.3.1 RGB Camera

An RGB camera is a camera that collects and records electromagnetic radiation at the wavelengths typical for human vision (read: visible light). This visible spectrum is defined as those wavelengths between roughly 400 and 780 nanometers [2]. A figure graphically depicting the electromagnetic spectrum near visible wavelengths can be seen in 2.4.

All cameras have a defined resolution. This resolution defines the number of pixels along the width and length of the image that the camera captures. For example, a resolution of 1920 x 1080 would indicate that the camera frame is 1920 pixels wide and 1080 pixels in height. Each pixel is a solid block of a given color. When enough pixels are accumulated in a frame (a resolution of 1920 x 1080 has over 2 million pixels per frame), a clear image can be formed.

Red-Green-Blue (RGB) cameras are referred to as such because each pixel they produce has a color defined by an RGB tuple. In typical applications, each pixel has an associated intensity for each of these three primary colors that ranges from integer values of 0 to 255. Assuming R, G, and B are the relative Red, Green, and Blue intensities, these values are normalized as

$$R' = \frac{R}{255} \qquad G' = \frac{G}{255} \qquad B' = \frac{B}{255} \qquad R', G', B' \in [0, 1] \tag{2.1}$$

The resulting tuple (R', G', B')—where each value corresponds to an amount of relative red, green, or blue light—corresponds to some color that the pixel emits. The math behind converting this RGB tuple into a visible color wavelength can be complicated and is beyond the scope of this thesis [32]. The important takeaway is that the color of a pixel from an RGB camera is mathematically represent via an RGB tuple.

2.3.2 Thermal Camera

Thermal cameras—a subset of infrared cameras that specialize in detecting temperature gradients—differ from RGB cameras in a few major ways. For one, thermal cameras measure electromagnetic radiation in the infrared range. Second, the values of a pixel outputted from a thermal camera is mapped to the temperature of the object that the pixel spatially represents.

Thermal cameras detect infrared emissions from multiple sources—the object a given pixel is capturing infrared light from, radiation from the surroundings reflected off that object (often a function of the object's emissivity), and emissions from the atmosphere itself between the object and the camera [33]. The relationship between temperature and these three emission sources is very non-linear and beyond the scope of this thesis. All thermal cameras have built-in software that can differentiate these three sources of infrared emissions to determine the relative temperature of the object being measured relative to its surroundings.

The mapping of temperature to pixel intensity for a given thermal camera is determined by its calibration that is defined by the relationship

$$[temp_{min}, temp_{max}] \mapsto [0, 255] \tag{2.2}$$

where $temp_{min}$ and $temp_{max}$ are the minimum and maximum temperatures in the thermal cameras range, and $[0, 255]$ is the 8-bit range of values for the corresponding pixel intensity mapping (some expensive thermal cameras have 16-bit intensity values, but those cameras are often prohibitively expensive). Unlike RGB cameras—which utilize an RGB tuple to represent color—thermal cameras use a singular intensity value per pixel to represent temperature [34].

Thermal cameras must necessarily represent their image using an artificial color mapping for the pixel intensities. Common choices are to utilize blue for "cold" pixels and red for "hot" pixels, or—as in the case of this research—to utilize a "white-hot" gray scale mapping.

2.3.3 Laser Rangefinder

Laser rangefinders are sensors that measure the range to an object from the sensor. They operate off a simple principle: a laser pulse is emitted by the sensor, the pulse travels to an object and is reflected back to the sensor. The time difference (T) between sending and receiving the pulse is measured and compared to the speed of light (c) and the path-averaged index of refraction (n_{path}) to deduce the range of the object (R) [35]

$$R = \frac{cT}{2n_{path}} \tag{2.3}$$

The maximum range of a laser rangefinder can vary, however most have maximum ranges of a few kilometers or more. If one has knowledge of the position and orientation of the laser rangefinder, the position of the object it is pointed at can be estimated. Sensor packages have been developed that do so internally, such as the Zenmuse H20 RGB camera with a built-in laser rangefinder that is used for this thesis.

Chapter 3: Fire Detection, Localization, and Suppression using a UAV

This chapter discusses the three major independent components that make up the Crossfire system. It begins with a break down and deep dive on the fire detection algorithms in Section 3.1. This includes algorithms applied to data from both the thermal and RGB cameras, as well as how that data is fused together. Next, the process for estimating the location of the fire as well as how the necessary telemetry for that process is extracted will be discussed in Section 3.2. Finally, the fire suppression chain will be detailed in Section 3.3, where path-planning for revisit/suppression flights and the suppression drop procedure will be touched on. For all sections, testing results will be embedded within the discussion.

3.1 Fire Detection using RGB and Thermal Imagery

The Crossfire system takes advantage of the aligned thermal and RGB cameras of the M300 payloads for its fire detection algorithms. Specifically, a blob detection routine is performed on the output of a carefully calibrated thermal camera. In parallel, an object detection model tuned to fires and smoke is applied to the RGB camera output. These routines separately determine if a fire may be present. Their corresponding outputs and image frames are fused together utilize computer vision techniques and a heuristic algorithm is applied to determine if a fire is present in the fused image or not.

3.1.1 Thermal Blob Detection

In computer vision and image processing applications, blob detection is the name generally given to the process of identifying groups of pixels that share some unifying characteristic [36]. In RGB imagery, blob detection can be applied to identify groups of pixels of similar colors. In thermal imagery—where color definition is lost and pixels instead have scalar intensity values—blob detection is used to identify clusters of pixels with similar brightness/intensity.

Blob detection can be tuned to detect different types of shapes, sizes, relative orientation, and other characteristics. However, the simplest case is implementing blob detection to identify clusters of pixels of some minimal size within which all of the pixels meet some criteria. This simplest case was applied to this research.

For the Crossfire system, one of the video inputs for the Ground Control System and its relevant processing software is the live video stream from the FLIR Vue TZ20 thermal camera payload. Information on the initialization process for this specific camera and UAV configuration is given in Chapter 4. However, the thermal blob detection algorithm that will be discussed can be generalized. All that is required is a thermal video input stream where the thermal video's temperature-to-intensity mapping for its pixels remains constant.

The algorithmic logic for the thermal blob detection algorithm designed in this research is laid out in 3.1. There are two inputs: a frame from the thermal video and some pixel intensity threshold ($P_{threshold} \in [0, 255]$). In brief, thermal blob detection works as follows.

1. Pre-processing is applied to the thermal frame to reduce noise.

2. Each pixel in the processed frame is compared to the intensity threshold. If a pixel has an

intensity higher than the threshold, it is assigned a 1. All other pixels are assigned a 0. This produces a binary frame.

3. The contours are drawn on the processed frame and outputted as an annotated frame.

4. Bounding box information for each contour in pixel coordinates is derived.

The two outputs from the thermal blob detection algorithm are the annotated frame and the data structure that contains bounding box information for each contour. Within this bounding box lays a contour within which all pixels meet the minimum intensity threshold. For a properly calibrated thermal camera, this intensity threshold can be set to align with the lower bound temperatures of incipient wildfires.

The pre-processing applied to the thermal frame first consists of a conversion of the frame to gray-scale. A gray-scale image is one where each pixel ranges from black to white in color along a "gray-scale", with the explicit color mapping being $[Black, White] \mapsto [0, 255]$. By default the Crossfire system utilizes a white-hot thermal image pallete, which aligns with this color-to-intensity mapping by default. However, if one were to utilize a rainbow thermal image pallette (with red mapping to hot and blue mapping to cold), then this pre-processing to gray-scale is necessary.

The second step of pre-processing is to apply a bilateral filter to the gray-scale image. A bilateral filter is a non-linear filtering technique that can reduce image noise via blurring techniques that retain strong edges within the image [37]. Bilateral filtering is similar to Gaussian convolution however bilateral filtering does a better job of retaining edge information. As defining the contours of fires as seen by the thermal camera is of immense importance to the thermal blob detection algorithm, implementing a bilateral filter that will better handle the non-uniform edge

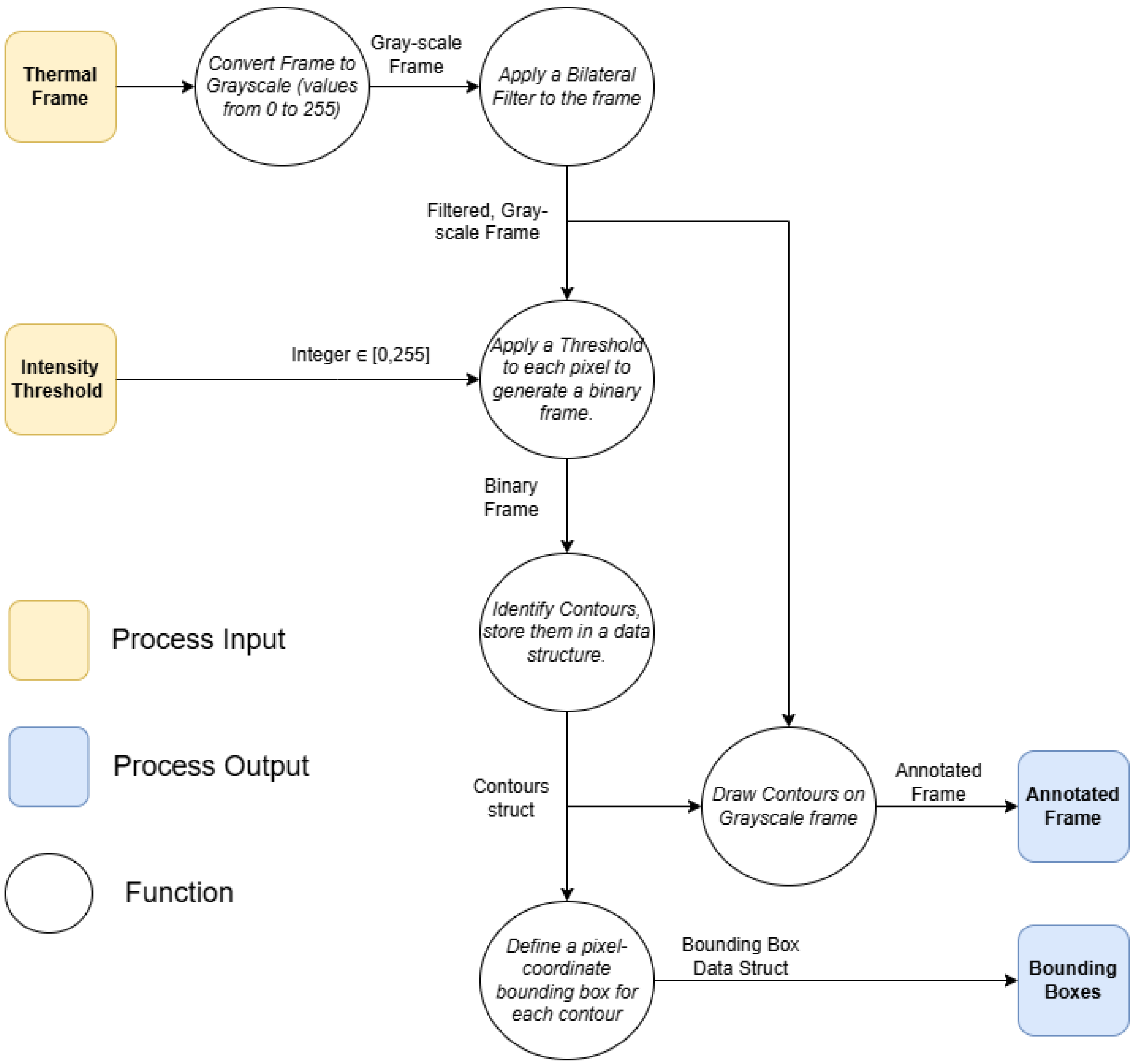


Figure 3.1: Thermal Blob Detection Algorithm Logic

shapes of fires was important—even if it introduced a bit more computational load.

Once the thermal image has been pre-processed, each pixel is sorted into a binary image according to the following logic.

$$P_{binary} = \begin{cases} P_{filtered}, & P_{filtered} \geq P_{threshold} \\ \\ 0, & P_{filtered} < P_{threshold} \end{cases} \tag{3.1}$$

where $P_{filtered}$ is the intensity of a given pixel, $P_{threshold}$ is the intensity threshold value set by the user, and P_{binary} is the intensity of a given pixel after it has been sorted into a binary frame.

The rest of the algorithm is applied to the binary frame. The most notable data to be pulled from the frame are the bounding boxes that tightly surround the high-intensity contours. For human readability, the contours are drawn on the gray-scale frame and displayed to the user.

An example frame with a variety of bit intensity thresholds applied to it can be seen in 3.2. The original raw image is shown in 3.2a. Notably, the thermal camera was uncalibrated when this image was taken, meaning that the pixel intensity-to-temperature mapping dynamically adjusted to fit the temperatures within frame. This state will generally be referred to as an "uncalibrated" thermal camera. The thermal blob detection algorithm was applied to it with different pixel intensity thresholds $P_{threshold}$. These intensities were 150 (3.2b), 180 (3.2c), and 210 (3.2d).

It is apparent that the success of the thermal blob detection algorithm is heavily dependent upon this pixel intensity threshold when the the thermal camera is uncalibrated. When the threshold was set to 150, many artifacts of the environment are identified as blobs (such as pavement from roads, the tree canopy in the background, and the smoke plume from the fire). Bumping this threshold up to 180 manages to filter out many of these artifacts. The blob detection

(a) Sample, uncalibrated thermal image from FLIR Vue TZ20 with a pool fire in the middle of the frame.

(b) Image from 3.2a annotated with a bit threshold of 150.

(c) Image from 3.2a annotated with a bit threshold of 180.

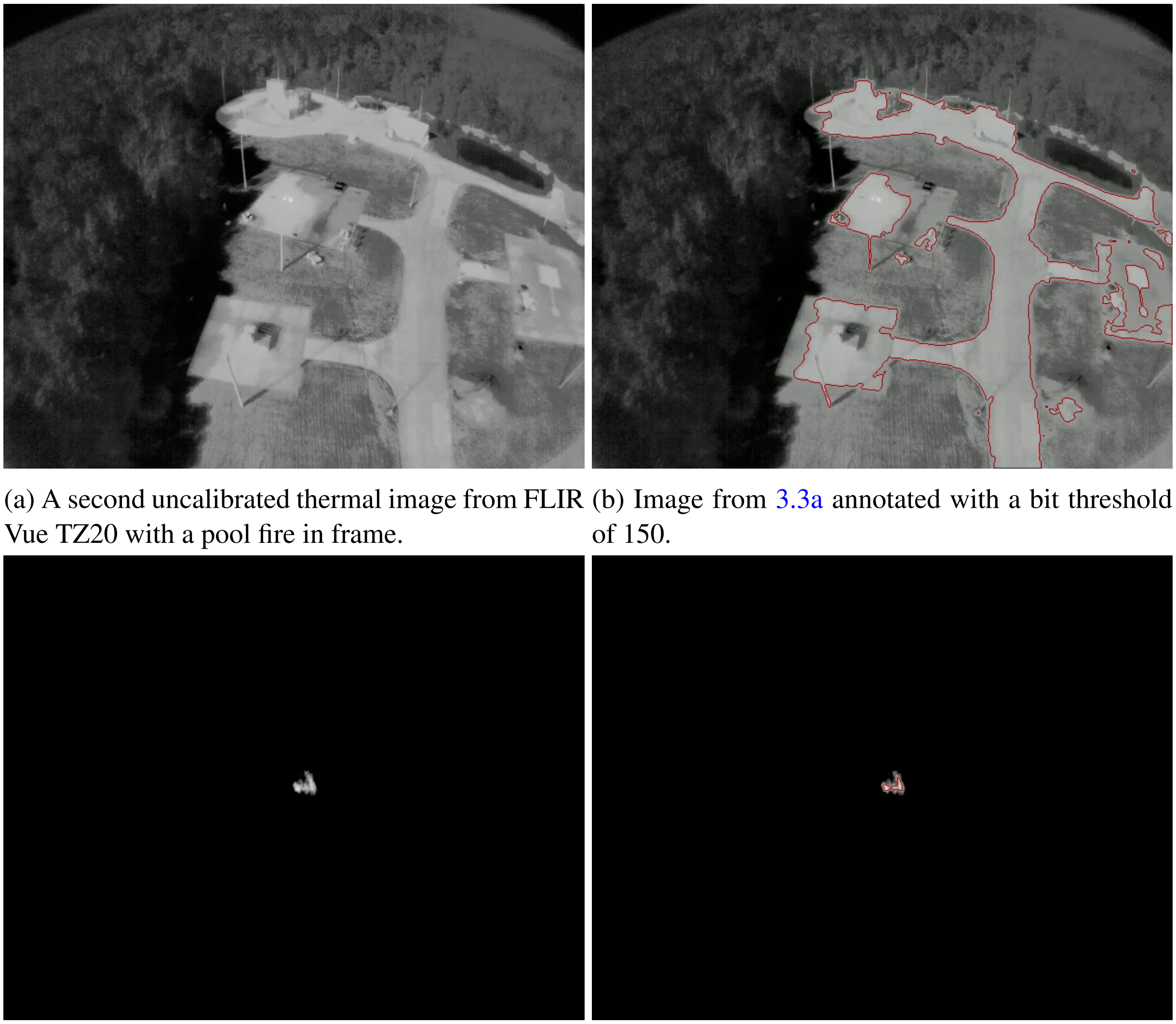
(d) Image from 3.2a annotated with a bit threshold of 210.

Figure 3.2: Thermal Blob Detection on a raw thermal image 3.2a, annotated at bit thresholds 150 3.2b, 180 3.2c, and 210 3.2d. The red lines indicate the contours of high-intensity blobs. Imagery taken at MFRI La Plata, October 8, 2024.

algorithm performs well, however there are still non-fire blobs detected—like the metallic roof of a building to the right side of the image. Finally, when the threshold is set to 210, only the fire is identified as a blob. This may seem to be the best case scenario, however the detected blob is small and only encompasses the very heart of the fire. Such a tightly constrained filter may miss incipient fires in the wild. While it is possible that the thermal blob detection algorithm can be applied to uncalibrated thermal images, domain knowledge is required of fire temperatures, atmospheric temperatures, and other parameters to tightly set the pixel intensity threshold. This may be an intractable approach.

A demonstration for how much calibrating the thermal camera can improve the thermal blob detection outputs is present in 3.3. Two frames just seconds apart from a video taken with the FLIR Vue TZ20 were extracted. The frame in 3.3a was extracted while the camera was uncalibrated. Then, the AGC lock function was initiated with the fire in the frame and the pixel intensity-to-temperature mapping was set to a static state. A frame extracted from the video shortly after initiating the AGC lock is displayed in 3.3c. Both frames had the thermal blob detection algorithm applied to them with pixel instensity thresholds at the bit value of 150. The annotated output of each can be seen in 3.3b and 3.3d.

From the annotated frames, there are a few artifacts of note. For one, the annotated uncalibrated thermal image is identifying many environmental objects as thermal blobs, much like we saw in 3.2b. This is not ideal, and it has been demonstrated that the best approach to fixing thermal blob detection on these uncalibrated images is by carefully selecting the pixel intensity threshold used in filtering the binary frames. This approach requires domain knowledge which may not always be available. Instead, the thermal camera can be calibrated so that the pixel intensity-to-temperature mapping is static with its highest pixel bit value (255) mapped to



(a) A second uncalibrated thermal image from FLIR Vue TZ20 with a pool fire in frame.

(b) Image from 3.3a annotated with a bit threshold of 150.

(c) A thermal image taken seconds after 3.3a with the FLIR Vue TZ20 now calibrated.

(d) Image from 3.3c annotated with a bit threshold of 150.

Figure 3.3: Thermal Blob Detection on an uncalibrated raw thermal image 3.3a and a calibrated raw thermal image 3.3c, both annotated at bit threshold of 150 in 3.3b and 3.3d. The red lines indicate the contours of high-intensity blobs. Imagery taken at MFRI La Plata, October 8, 2024.

the hottest object in the image. As mentioned, this calibration occurred with the pool fire itself within the frame, and a calibrated annotated frame can be seen in 3.3d.

In this calibrated image, the fire is the only object visible on a field of black. This is because its temperature is relatively much higher than the surrounding environment, and therefore any pixels near the fire have intensities around the maximum (255), while pixels depicting the ambient environment are "cold" and therefore have intensities near the minimum (0). For a White Hot color palette, the fire (or other hot objects) will be colored white and the background and nearly every other object will be colored black. The detected contours closely align with the shape of the fire, and therefore the resulting bounding box data will accurately reflect the fire's location on the thermal image. This is ideal behavior for the thermal blob detection algorithm.

When the thermal camera is calibrated, the pixel intensity threshold can be relaxed. Generally it is kept around 150 so that most of the fire body will be captured within the contours. It could potentially be relaxed even further as environmental artifacts will have very low pixel intensity values when the thermal camera is properly calibrated. The most imporant takeaway from this discussion is that proper thermal camera calibration enhances the robustness of the thermal blob detection algorithm. More information on how the Crossfire system calibrates its cameras will be presented in Chapter 4.

3.1.2 Object Detection using YOLOv8

The You Only Look Once (YOLO) series of object detection algorithms have become increasingly prevalent in object detection literature and tools over the past decade. One of the more popular iterations of YOLO is YOLOv8, released by Ultralytics [38]. Every version has built upon the last and multiple organizations have contributed to the series in independent ways. However, YOLOv8 is notable due to how many publicly available object detection models are trained with it.

The second aspect of the Crossfire system's fire detection algorithm is to apply an object detection model to the RGB camera stream. The model utilized for the Crossfire system was trained using YOLOv8. This model is publicly available on Github and is entitled YOLOv8-Fire-and-Smoke-Detection [3]. Eventually, the Crossfire system will utilize a custom-trained fire detection model using YOLOv8 or some other version of YOLO as a base. However, the Crossfire project began with little to no data on fires, so it was simpler to develop a system that references a pre-trained model. Since these models are stored in files referenced by a YOLO object in the software, integrating a custom-trained model into the Crossfire system software is as simple as changing the filename input for the model when initiating the fire detection software.

The process for applying the object detection algorithms in the Crossfire system is simpler than thermal blob detection. This is because Ultralytics has released a YOLO software library that has greatly simplified the process of inputting images to be analyzed by an object detection model. Under the hood, this software is actually using the image and its individual pixel data as an input layer for a convolutional neural network (CNN).

A CNN is a sub-class of feed-forward neural networks. The explicit math and theory behind neural networks is beyond the scope of this thesis, however a very brief explanation is in order. Feed forward neural networks consist of multiple layers of nodes. The first layer is the input layer, the last layer is the output layer, and there are many hidden layers between the input and output layers. For the CNN used for YOLOv8, the first layer consists of raw image data, and the last layer consists of object detection information such as confidence in detection, bounding box information, and class identification (classes being the types of objects that are detected) [38].

Each layer consists of neurons. Each neuron takes inputs—usually the output of every neuron (x_i) from the previous layer—applies a weighting schema (w_i) and a bias (ν_0) to those inputs, and takes that result as an input for an activation function ($f(.)$), typically a sigmoid function for CNNs. Mathematically, where y is the output of a given node that is receiving (N) inputs from the previous layer, this equation may look like [39]:

$$y = f(\nu_0 + \sum_{i=1}^{N} w_i x_i) \tag{3.2}$$

As the number of nodes in each layer and the number of layers themselves increases the overall mathematical picture for a feed-forward neural network quickly becomes complicated. With YOLO models, there are usually 20 or more layers applied in just processing the image. When classifying objects, the number of layers increases substantially.

The process for applying object detection in the Crossfire system for an RGB video frame input can be seen in 3.4. Since the YOLO software library operates at a high-level, the object detection process from the perspective of a Crossfire system user is simpler than the thermal blob detection algorithm. It starts with defining the YOLOv8 object detection model that will be used for detecting wildfires. With the model file, a model object is created in software which has a *model()* function. This function takes as input an RGB image and outputs a data struct containing results such as bounding box coordinates, the types of objects detected, etc.

The next steps are fairly simply. Bounding box information for detected fire objects are passed out of this process as an output. Additionally, the "results" data struct is referenced for detected fire information and is used to label the raw RBG frame input. This labeling produces

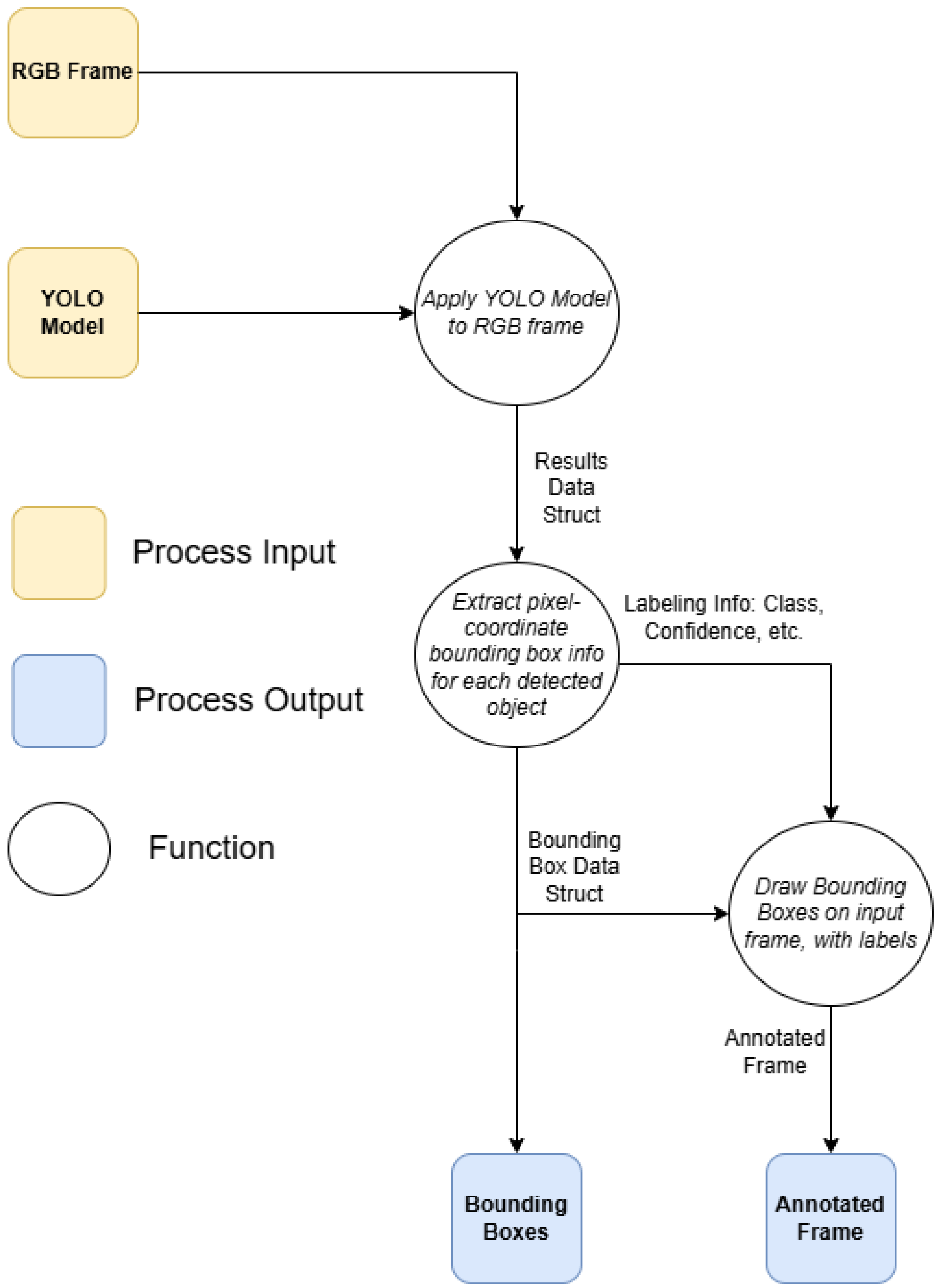


Figure 3.4: Object Detection Processing Logic

an annotated frame useful for human readability and debugging.

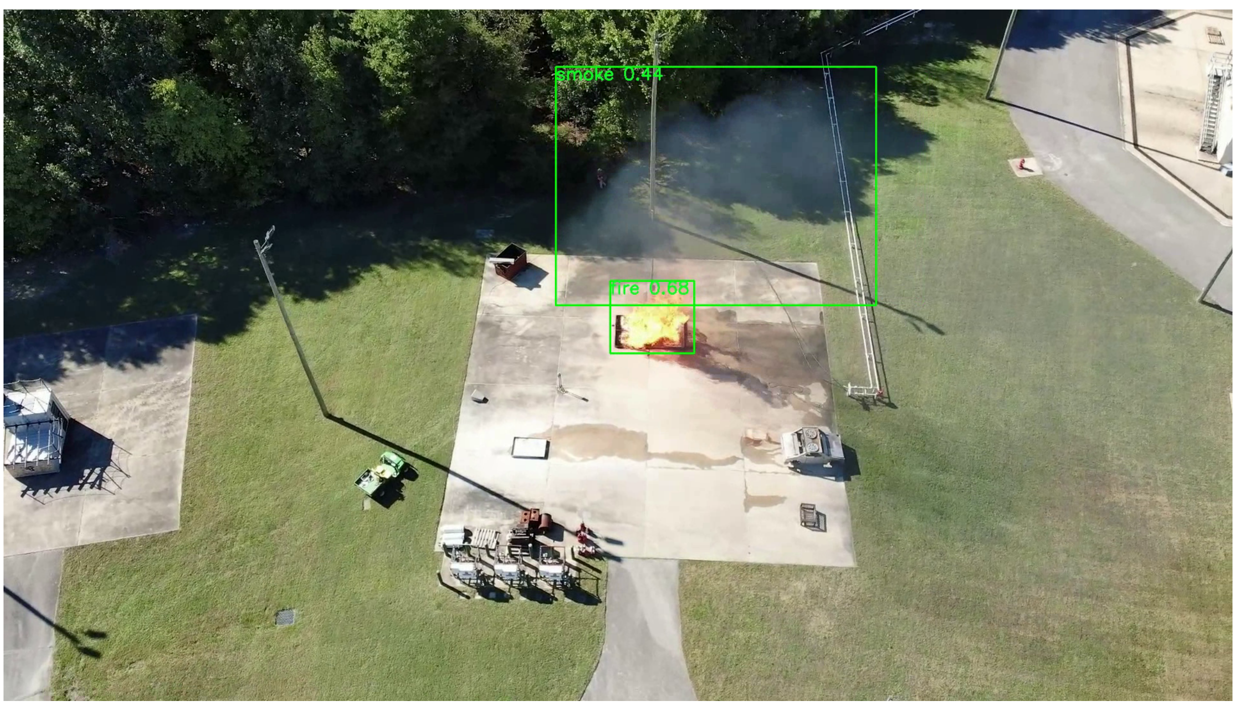
The model used for demonstrative purposes in this thesis detects both fire and smoke [3]. There are four versions of the model: nano, small, medium, and large. As models "grow" in size, they generally become more accurate at detections but require more processing time. For the work in this thesis, the medium version of the model was used. From a qualitative perspective, the model's fire class usually produces detections with higher confidence than its smoke class. These confidence levels are reflective of testing results, as the smoke detection class occasionally produces false positive detections of surfaces such as white pavement. Due to these characteristics, it was decided to only use the fire detection class for this phase of the Crossfire system's fire detection algorithm.

Two sets of sample images of the fire and smoke object detection model as applied to imagery taken from the M300 UAV's Zenmuse H20 RGB payload camera are present with image set 1 in 3.5 and image set 2 in 3.6. These images were taken from an altitude of roughly 150 feet. They demonstrate characteristics about the fire and smoke detection model worth noting.

To begin, fire detection seems to work best when the background behind the fire is not similar in color to the fire. We can see in 3.5a and 3.5b that the model picks up on the fire when, relative to the camera, the pavement behind the fire is stained with ash. When the pavement is whiter, the fire detection model struggles to identify the fire as in 3.6a and 3.6b. In wildland scenarios, it is likely that any fires will have backdrops of darker colors such as brown and green. When the Crossfire system eventually implements a custom fire detection model, the effects on the fire backdrop's color should be noted when selecting training data.

Also worth considering, smoke was detected at roughly 25% the rate that fire was detected. The images in 3.5 and 3.6 were selected for their variety, but there were roughly 1,100 annotated

(a) An annotated Zenmuse H20 image with a detected fire.



(b) An annotated Zenmuse H20 image with a detected fire and smoke.

Figure 3.5: YOLOv8 Fire Object Detection model applied to raw RGB images from a Zenmuse H20, taken at MFRI La Plata campus from a DJI M300 RTK UAV. YOLOv8 fire and smoke detection model sourced from [3]. Image Set 1. Imagery taken at MFRI La Plata, October 8, 2024.

(a) An annotated Zenmuse H20 image with detected smoke.

(b) An annotated Zenmuse H20 image with nothing detected.

Figure 3.6: YOLOv8 Fire Object Detection model [3] applied to raw RGB images. Image Set 2. Imagery taken at MFRI La Plata, October 8, 2024.

frames in the video that the sample images were drawn from. Of those frames, roughly 620 had the fire visible in them. 345 of those 620 annotated frames had a successful fire detection, but only 87 had a successful smoke detection. In terms of percentages, those are roughly 56% and 14% successful detection rates for fire and smoke detection, relatively. We see smoke detected in images 3.5b and 3.6a. In these images, the backdrop behind the smoke is notably darker than in other images. Other factors such as the thickness of the smoke plume from the viewing angle and the amount of smoke actually present at the moment the image was taken affect the success of smoke detection.

In short, there are a few key ideas to note. For one, fire detection is seemingly more reliable than smoke detection. This isn't to say that only one or the other can be used, but for this iteration of the Crossfire system it has been decided to just implement fire detection as it usually produces better results with the utilized object detection model. Additionally, the YOLOv8 fire detection data can easily be fused with thermal blob detection data. A second key idea is that the color of the backdrop of either fire or smoke has a drastic effect on the success of their relative object detection models. A robust survey of potential fires is one that views the ground at a given point from multiple angles. By varying the point of view of potential fires, one can give the object detection models a greater chance at success.

3.1.3 Homography

A critical aspect of the Crossfire system is the fusion of data from both the thermal and the RGB cameras. This requires knowledge of how the pixel coordinates from one camera—say the thermal camera—maps to the pixel coordinates of the other camera—in this case the RGB camera. There are multiple factors that influence this mapping. For one, although the cameras are aligned (i.e., pointed in the same direction), there is a physical distance between their lenses that affects their fields of view (FOV). Secondly, characteristics of each camera such as their resolutions, fields of view, focal lengths, etc., affect the representation of physical space that each camera can produce. Therefore, it is important to create a homography that can map the image produced by one camera to the image produced by the other.

A homography utilizes a 3x3 transformation matrix—typically referred to as a homography matrix—to represent the transformation of a given feature point between two corresponding image frames. A schematic representing such a scenario is visible in 3.7. In the schematic, both camera images contain the real-world polygon on the right, however camera perspective 1 utilizes coordinates $[x, y, 1]$ and camera perspective 2 utilizes coordinates $[x', y', 1]$. Each set of coordinates has a third element always equal to 1. In image processing, camera coordinate systems usually start at the top left of the image and the x-coordinate increases as pixels tend rightwards while the y-coordinates increases as pixels tend downwards.

The $[x, y, 1]$ coordinates relate to the $[x', y', 1]$ coordinates via a homography matrix (H). This relationship is typically expressed as:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = H \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{3.3}$$

H is a 3x3 homogeneous matrix. Its final element, h_{33}, is always normalized to 1 so H always has eight degrees of freedom. These eight values are grouped into three sets partitioned as

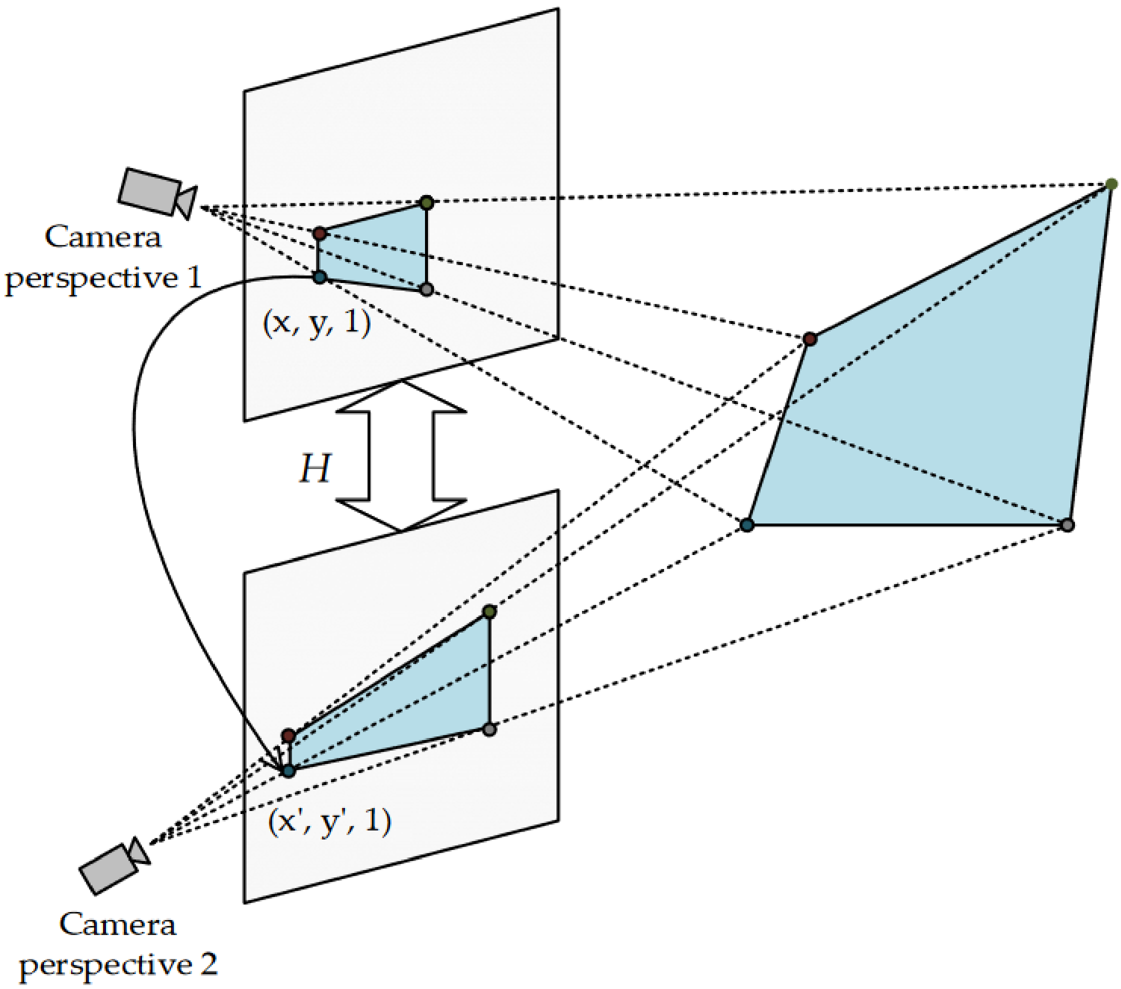


Figure 3.7: Schematic diagram of homography transformation relationship [4] (no license required).

$[h_{11}, h_{12}, h_{21}, h_{22}]$, $[h_{13}, h_{23}]$, and $[h_{31}, h_{32}]$, which respectively represent the affine transformation, the translation transformation, and the perspective transformation [4].

To derive H between two image frames, one has to explicitly define the coordinates of four feature points across both image frames. These four points have two coordinates each (x and y or x' and y'), so one can derive eight linear equations to solve for the eight unknowns in the homography matrix. Typically there is a degree of noise in deriving these coordinate points, so better methods for deriving H usually employ selecting more than four feature points. Across this extended set of coordinates, a linear least squares approximation is usually applied in estimating the values of the H matrix.

For the Crossfire system, it was deemed necessary to calibrate a homography matrix that could transform the thermal image frame coordinates into the RGB image frame coordinates. This was determined to be the best course of action because the FOV of the FLIR Vue TZ20 Thermal Camera was wider than that of the Zenmuse H20 RGB camera. By transforming a wider image into a narrower one, the resulting warped images are usually cropped as opposed to adding blank spaces to the transformed image. Additionally, the RGB camera's resolution is higher. Any sort of processing on a blended image resulting from a transformed image laid overtop its counterpart is better suited to be done on higher resolution images. Therefore, compressing the thermal image to fit the dimensions of the raw RGB image was determined to be the best course of action.

The figures in 3.8 demonstrate success with using a homography matrix to transform raw thermal imagery from a thermal camera into the perspective and resolution of imagery from an aligned RGB camera. The raw images are seen in 3.8a and 3.8b. A homography matrix was calibrated between these two images using features on the face of the building.

From there, a homography matrix was derived and applied to the raw thermal image. Utilizing OpenCV's *warpPerspective()* function, the H matrix was used to warp the raw thermal image into a transformed image (3.8c) whose perspective closely matches the original raw RGB image (3.8d). The explicit homography matrix derived for the demonstration was:

$$H = \begin{bmatrix} 3.672 & 1.066 & -292.544 \\ -0.159 & 4.749 & -505.853 \\ -0.0005 & 0.0012 & 1.000 \end{bmatrix}$$

Utilizing knowledge of what each value represents, it is apparent that the perspective transformation (denoted in the bottom row) was minimal while the translation transformation (denoted in the right-most column) was significant. Given that the cameras were aligned but that they had significant differences in resolution, this derived homography matrix is reasonable but not perfect. A homography is dependent upon the depth of the images with which it is calibrated, so there will be some small error in the homography implicit with images with a depth differing from the calibration images. Still, a homography with some error is a practical tool.

Homography transformations are useful for more than warping one image on to another. As will be discussed in the next section, the homography matrix plays a significant role in comparing the pixel coordinates of thermal blobs and detected fires.

3.1.4 Visual Data Fusion

A significant component of the contributions of this thesis is the fusion of thermal camera detection data with that of RGB camera detection data. As discussed in the pervious two subsections, thermal blob detection and object detection of fires are generally reliable at detecting fires,

(a) A raw thermal image as taken by the FLIR Vue TZ20.

(b) A raw RGB image as taken by the Zenmuse H20 at the same moment as 3.8a.

(c) A transformed thermal image that closely reflects the POV of 3.8d.

(d) The same raw RGB image in 3.8b. The POV from 3.8c now closely aligns with it.

Figure 3.8: A homographic transformation of a raw thermal image (3.8a) into the dimensions of a raw RGB image (3.8b). Imagery taken at MFRI College Park, September 19, 2024.

but they both have pitfalls. The thermal camera must be carefully calibrated for thermal blob detection to work accurately. Meanwhile, successful object detection of fires with RGB cameras is dependent upon the angle at which the fire is viewed.

The Crossfire system utilizes cameras that capture data at 30 frames per second (FPS), so by default these algorithms are being applied to a multitude of frames. If there is set of frames (S) where a fire may be present, there is likely a subset where the thermal blob detection algorithm detects a fire ($S_{thermal\_fire}$) and a subset where the object detection algorithm detects a fire (S_{RGB_fire}).

$$S_{thermal\_fire}, S_{RGB\_fire} \in S \tag{3.4}$$

The union of these two subsets represents the frames where a fire is detected by both algorithms ($S_{possible\_fire}$).

$$S_{possible\_fire} = S_{thermal\_fire} \cup S_{RGB\_fire} \tag{3.5}$$

The visual data fusion algorithm of the Crossfire system is designed so that the set of frames where thermal blob detection and RGB object detection both indicate a possible fire ($S_{possible\_fire}$) is truncated in such a way so that a resulting subset ($S_{detected\_fire}$) of frames each have a very high likelihood of actually having captured a wildfire.

$$S_{detected\_fire} \in S_{possible\_fire} \tag{3.6}$$

The process for how the set of possible fires ($S_{possible\_fire}$) is truncated into a set of likely fires ($S_{detected\_fire}$) is displayed via the diagram in 3.9. The process takes multiple inputs: the annotated frames and fire bounding boxes from the thermal blob detection and RGB object detection algorithms discussed previously, a homography matrix that maps the thermal image frames to the RGB image frames, and a pixel distance threshold value ($p_{threshold} \in [0, \sqrt{w^2 + h^2}]$, where w is the width of the RGB frame and h is the height of the RGB frame.

The visual data fusion algorithm consists of two parallel processes. One process loops through all combinations of thermal bounding boxes and RGB bounding boxes passed as inputs. The coordinates of the thermal bounding boxes are transformed into RGB image coordinates via the homography transformation equation given in (3.3). From there, the euclidean pixel distance between the centroids of the transformed thermal bounding box ($[x_{thermal}, y_{thermal}]$) and the RGB bounding box ($[x_{RGB}, y_{RGB}]$) is calculated as:

$$p_{dist} = \sqrt{(x_{RGB} - x_{thermal})^2 + (y_{RGB} - y_{thermal})^2} \tag{3.7}$$

This euclidean distance for each pair of thermal bounding box and RGB bounding box centroids is then compared to the pixel distance threshold. It is assumed that all of these distances are stored in a set for the given frame ($S_{distances}$). If any of the distances is within the threshold, the fire detection signal is set to true. Otherwise the signal is set to false.

$$FIRE\_SIGNAL = \begin{cases} 1, & \exists\, p_{dist} \leq p_{threshold} \qquad \forall\, p_{dist} \in S_{distances} \\ 0, & otherwise \end{cases} \tag{3.8}$$

The other process that occurs in parallel is a bit simpler. The annotated thermal frame is warped to align with the RGB frame utilizing the homography matrix. These two annotated frames are then blended together using OpenCV's *addWeighted()* function. This function effectively

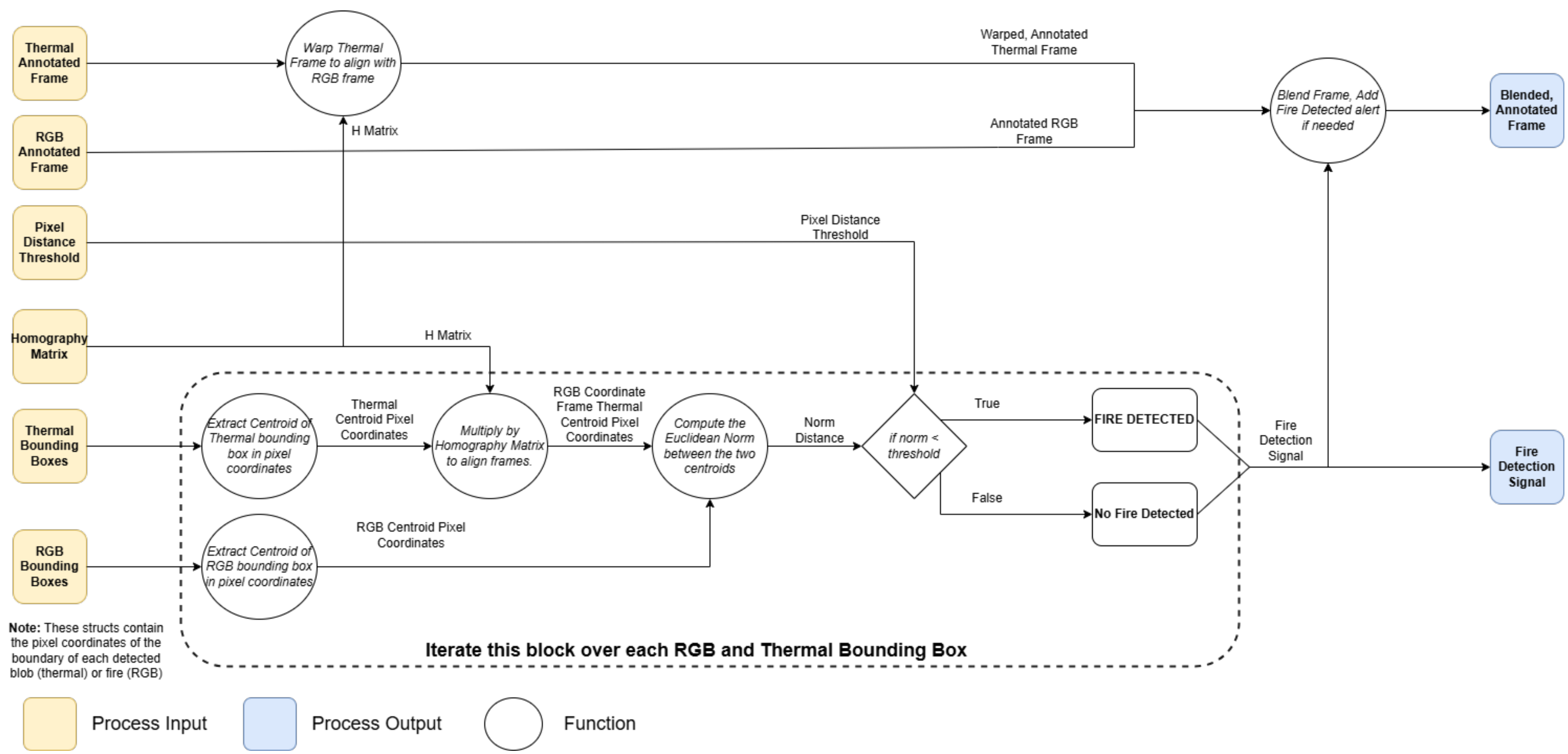


Figure 3.9: Fusion of Thermal Blob Detection and RGB Object Detection data.

blends two frames together with a weight $\alpha \in [0, 1]$ that sets the intensity of the first inputted frame as α and the second inputted frame as $1-\alpha$. If FIRE_SIGNAL is determined to be true, the blended frame is annotated further to mark where the fire has been detected and to add a "FIRE DETECTED" flag to the output frame.

Examples of fused frames where a fire has and has not been detected are presented in 3.10. Notably, the fire is present in both frames, but only one of them positively identifies the fire. Given the design of the Crossfire system's fire detection algorithms, the presence of some false negatives regarding fire identification is allowable.

Video data fusion in the Crossfire system takes a conservative approach to identifying potential fires. It requires affirmation from both the thermal blob detection and RGB object detection algorithms to even consider if a fire is in frame. From there, the heuristic algorithm described above must return a true condition for a fire to be detected. This is stringent, however it significantly reduces the amount of potential false positives. False negatives are allowable so long as each real fire is identified by some minimal number of image frames (usually 4 due to the implementation of a DBSCAN clustering algorithm that will be discussed shortly). For example, if a fire is in frame for five seconds, and the processing software is able to process 10 frames per second, then the successful detection rate for that fire need only be 8%. In cases above (when the UAV is flying at an altitude of 100ft.), we've shown that the thermal blob positive detection rate is very high (near 100%) for a well-calibrated camera, and the RGB Object Detection algorithm has a roughly 50% true positive detection rate. These probabilities are much higher than what is needed to positively identify the fire.

Fire Detection is a critical aspect of the Crossfire system. However, it must be combined with fire localization techniques in order to plan missions to suppress said fires. The localization

(a) A fused image output where a fire has been detected..

(b) A fused image output where no fire is detected. A fire is not detected because the object detection algorithm struggles to identify the fire at this angle.

Figure 3.10: Two fused images displaying an image with a detected fire (3.10a) and without a detected fire (3.10b). Imagery taken at MFRI La Plata, October 8, 2024.

technique used in the Crossfire system will now be discussed.

3.2 Positions of Interest Localization

Designing a response to a wildfire begins with knowledge of where that fire is. These positional estimates are critical in dispatching and optimizing the use of fire-fighting assets. Therefore, fire localization is a crucial aspect of the Crossfire system.

3.2.1 Target Specification with Available Telemetry

As mentioned previously, the Crossfire system currently employs a DJI M300 RTK quadrotor UAV. The M300 configuration in use is discussed in detail in Section 2.2. The M300 has sensors typical of quadrotor UAVs, such as IMUs, GPS receivers, altimeters, etc. However, the RGB payload camera—the Zenmuse H20—also has a built-in laser rangefinder. This laser rangefinder is aligned with the RGB camera's image frame so that it is aimed at the frame's center.

The Zenmuse H20's rangefinder has access to the M300's real-time position and orientation estimates. Therefore, by fusing together these estimates with the rangefinder's reported range, the pitch and yaw angles at which the RGB camera is pointed, and the UAV's altitude, a geospatial estimate of the point on the ground which the laser rangefinder is pointing at can be reported to the user. This information is displayed as a telemetry overlay with the latitude and longitude coordinates of the point as well as its estimated altitude reported as feet above sea level. A screenshot of this telemetry overlay with the rangefinder data annotated is displayed in 3.11.

This information can be extracted every single time that a fire is detected in the RGB camera image. Therefore, the Crossfire system will know the geospatial data of the exact points

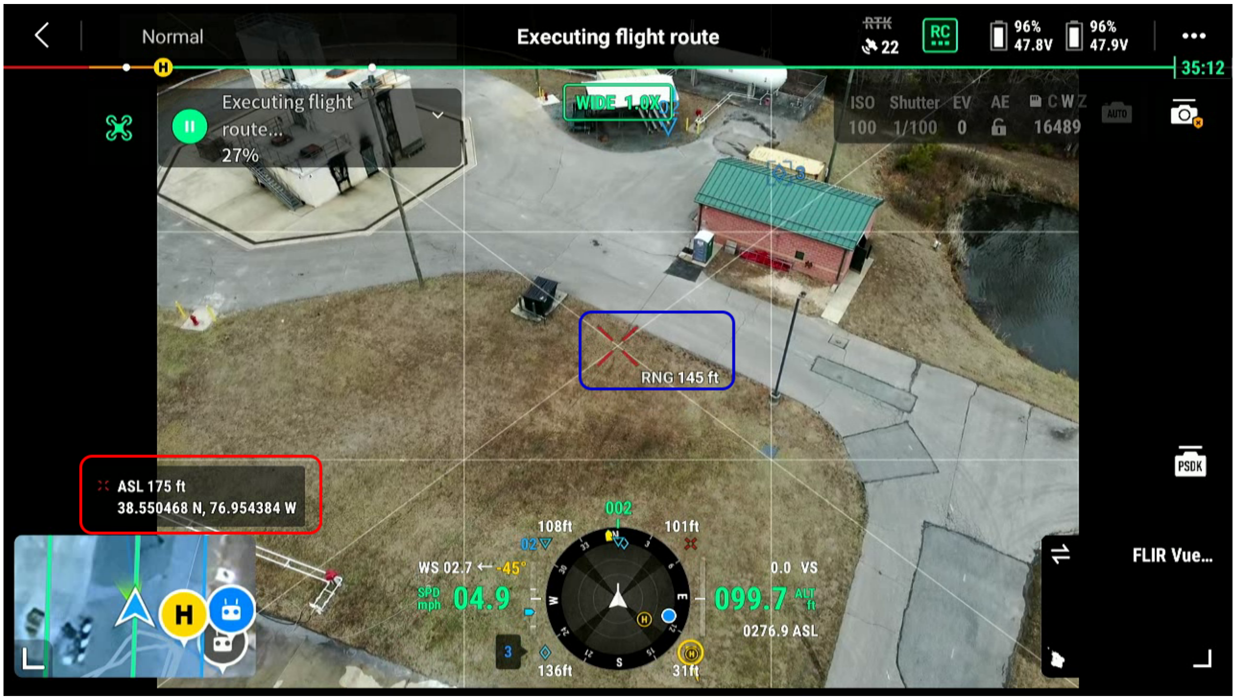


Figure 3.11: M300 RGB Camera stream with laser rangefinder position estimates circled in red and the rangefinder "target" and reported range circled in blue. Imagery taken at MFRI La Plata, February 5, 2025.

at the center of the RGB camera's images when fires are present. This is by no means an exact localization of each fire, but the data can be analyzed in such a way that the estimates produced are pretty close (on the scale of $\sim$ 10 meters positional error according to tests). These fire location estimates are then referred to as "Positions of Interest" (POIs).

The Smart Controller is capable of processing M300 imagery and combining that imagery with data from the flight controller and laser rangefinder. However, due to the limited storage capacity of the Smart Controller and the need to save test flight data for post-flight development activities, it was deemed logical to run the processing algorithms via a ground station computer. Unfortunately, the ground station laptop had no means of directly receiving the laser rangefinder data. This problem was solved by implementing an Optical Character Recognition (OCR) model to read the rangefinder position estimates into text.

The OCR specifically used in this application is Tesseract, an open-source model maintained by Google [40]. There is a publicly available Python library named *pytesseract* that is based on the Tessaract model that contains wrapper functions critical in image-to-text applications. The Crossfire system utilizes the following steps to extract the laser rangefinder data from the RGB image frame:

1. Copy and crop the RGB frame so that it is just the annotated red box from 3.11.

2. Process the cropped image by converting it to gray scale and applying a threshold to creating a binary frame that only keeps white pixels.

3. Extract the text utilizing *pytesseract's image_to_string()* function.

4. Verify that the text is valid, i.e. check for the proper number of digits after decimal points, directional indicators (N, E, S, W), etc.

5. If the text is valid save it. Otherwise, discard it.

There are a multitude of ways to improve OCR performance. In this research, basic optimizations such as image pre-processing and increasing the resolution of the input images were applied. With these implemented, the Crossfire system reached a high enough rate of successful image-to-text operations for the POI estimation pipeline to work properly.

3.2.2 Recording Relevant Flight Data

In order to carry out Positions of Interest estimation, the ground station laptop must record data extracted during a broad area survey flight that is useful in estimating the POIs. This data is composed of the time of detection in milliseconds since epoch; the latitude, longitude, and altitude estimates for the laser rangefinder's target; the fire detection signal; if the fire was detected: the pixel centroid coordinates in the RGB frame and the pixel area of the fire's YOLOv8 detection bounding box.

In real-time during the flight, the relevant data is packaged into an XML file. Entries are created for each analyzed frame and the file grows as the flight continues. A sample of two entries from such an XML file—one where a fire was detected and one where one wasn't—is presented in 3.12.

3.2.3 Positions of Interest Estimation

The geospatial data that the Crossfire system has access to is not a direct estimate of the fire's position derived from each image frame. It is in fact a positional estimate of the point on the ground at the center of said images. This is sufficient for the purposes of the Crossfire system. So long as an estimate of the fire's position is relatively close to the ground truth, when a pilot operates a revisit/suppression flight they can scan the ground nearby Point of Interest waypoints for the fire's actual position. Additionally, the geometry of the survey flights is usually helpful in improving the POI positional estimates. More on why these characteristics are allowable and/or take place will be discussed later.

After a survey flight, the packaged XML file is loaded into a script that utilizes a clustering algorithm named DBSCAN to differentiate geospatial clusters of laser rangefinder positions associated with a detected fire. DBSCAN, which stands for Density Based Spatial Clustering of Applications with Noise, is an algorithm which identifies sets of clusters according to two parameters: ϵ—a minimum distance between points in the same ϵ-neighborhood—and n_{min}—a

```xml
<Entry type="Valid">
    <UTC>1739552608435</UTC>
    <ASL>179</ASL>
    <Coordinates>
        <Coordinate1>38.550434 N</Coordinate1>
        <Coordinate2>76.954796 W</Coordinate2>
    </Coordinates>
    <FireDetection>False</FireDetection>
</Entry>
<Entry type="Valid">
    <UTC>1739552608748</UTC>
    <ASL>179</ASL>
    <Coordinates>
        <Coordinate1>38.550434 N</Coordinate1>
        <Coordinate2>76.954796 W</Coordinate2>
    </Coordinates>
    <FireDetection>True</FireDetection>
    <DetectionData>
        <CentroidX>194.5</CentroidX>
        <CentroidY>415.0</CentroidY>
        <Area>8970</Area>
    </DetectionData>
</Entry>
```


Figure 3.12: Sample XML file output with relevant flight data showing an entry instance where fire is detected and one where it is not.

minimum number of neighbor points required to form sets around a core point [41].

DBSCAN identifies clusters by checking the ϵ-neighborhood for every single point in the initial set (D). The number of points within the ϵ-neighborhood (N_ϵ) for a given point 'p' is defined as

$$N_\epsilon = [dist(p,q) < \epsilon \quad \forall q \in D] \tag{3.9}$$

If the ϵ-neighborhood of point p contains the minimal number of points

$$N_\epsilon(p) \geq n_{min} \tag{3.10}$$

then p is referred to as a core point. p's neighbors may themselves be core points, or they may not be. What matters is that p is a core point to a cluster and all of p' s neighbors, neighbor's neighbors, etc., are part of the defined cluster. If p happens to be part of the cluster of another point 'v', then p and v are simply considered to be part of the same clusters. Clusters can not overlap, and some points may not belong to any clusters and are generally considered to be noise.

DBSCAN is most simply applied in Euclidean space, in this case $\mathbb{R}^2$ or $\mathbb{R}^3$. The Crossfire system converts the coordinates in the Latitude-Longitude-Altitude (LLA) reference frame to an East-North-Up (ENU) reference frame. For DBSCAN, the origin of the ENU frame is arbitrary and in this case is taken to be the first valid coordinates of a potential detected fire (outlier coordinates are removed prior to applying DBSCAN). The Haversine formula [42] is used for this reference frame transformation of coordinates. The formula references the ENU coordinates' origin's latitude (ϕ_0) and longitude (λ_0), and takes as input the latitude (ϕ_1) and longitude (λ_1) of some point whose ENU coordinates are desired. It is presumed all latitude and longitude values are in radians. A delta is defined for each latitude and longitude.

$$\Delta\phi = \phi_1 - \phi_0 \tag{3.11}$$

$$\Delta\lambda = \lambda_1 - \lambda_0 \tag{3.12}$$

A couple of parameters are then defined:

$$a = \sin^2(\frac{\Delta\phi}{2}) + \cos(\phi_0)\cos(\phi_1)\sin^2(\Delta\lambda) \tag{3.13}$$

$$c = 2\arctan(\frac{\sqrt{a}}{\sqrt{1-a}}) \tag{3.14}$$

The distance between the coordinate inputs and the origin (l) is then calculated as

$$l = R_{earth}\ c \tag{3.15}$$

where R_{earth} is Earth's radius in whatever units of length are desired.

The x- and y- coordinates of the input coordinates are finally calculated in the ENU frame, with (x_1) being the coordinate along the East axis and (y_1) being the coordinate along the North axis. It is presumed that the origin coordinates (x_0, y_0) = $(0, 0)$.

$$x_1 = R_{earth}\ \Delta\lambda\ \cos(\frac{\phi_0 + \phi_1}{2}) \tag{3.16}$$

$$y_1 = R_{earth}\ \Delta\phi \tag{3.17}$$

The above formulae are applied to all LLA coordinate points relative to a defined origin.

The East-North-Up coordinates that are estimated using a flat-Earth model in $\mathbb{R}^3$ euclidean space are accurate enough for the DBSCAN algorithm to be applied to them.

DBSCAN as applied to an example set of data can be seen in 3.13. This data set was captured with the M300 in the configuration utilized for the Crossfire system. Each data point corresponds to a set of laser rangefinder LLA data transformed into the ENU reference frame and coordinates. For this specific example, all rangefinder data (not just those associated with detected fires) was analyzed by the DBSCAN algorithm. The laser rangefinder data was collected while the drone sat stationary on the ground and the gimballed cameras + laser rangefinder were "swept" in an arc with a zero pitch angle. At specific points in this arc the camera's angular velocity was stopped to collect multiple data points at specific angles.

When DBSCAN was ran in post-processing on the collected data set, the ϵ parameter value for cluster identification was set fairly low in order to differentiate multiple clusters. Four clusters and their associated centroids were identified and are marked in ENU coordinates relative to an origin point. There is one noise data point at the northwest corner of the grid.

3.13 demonstrates strengths of DBSCAN—for example, Cluster 0 and Cluster 3 are easily identified and differentiated. It also demonstrates potential weaknesses—the border between Cluster 1 and Cluster 2 appears to be somewhat arbitrary. It is therefore important to carefully select the algorithmic parameters, namely ϵ and n_{min} when applying DBSCAN to real data sets. A discussion on what the parameters were selected to be for the Crossfire system will come later.

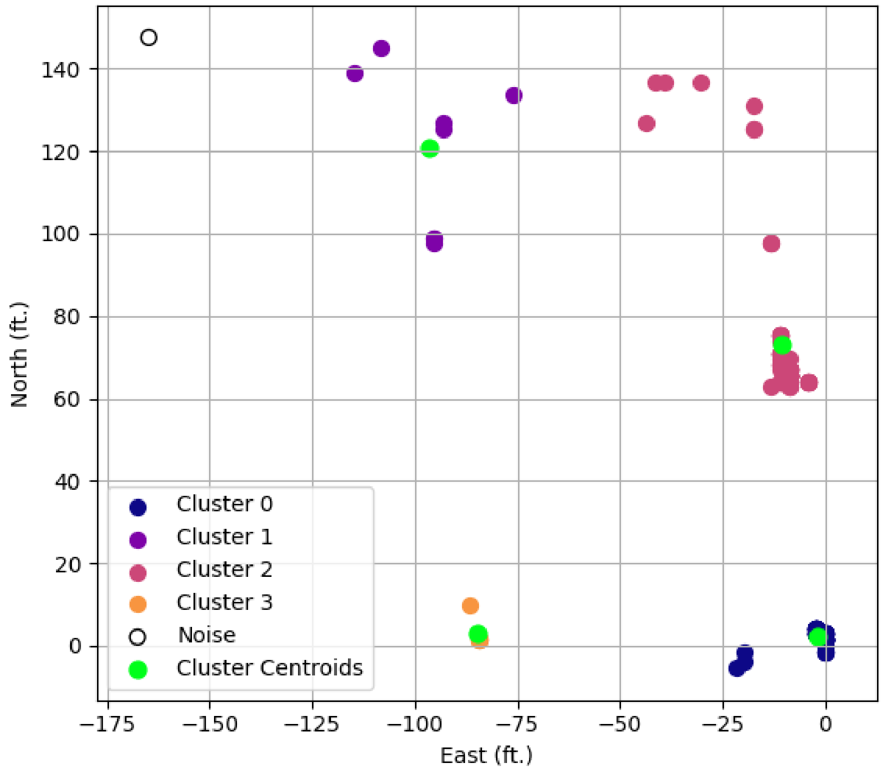


Figure 3.13: DBScan as Applied to an Example Set of Data. Parameters: $\epsilon = 30ft.$, $n_{min} = 4$

3.3 Fire Suppression using Drop Mechanism

Perhaps the most critical component of the Crossfire system is its ability to suppress the fires that it detects. This thesis places emphasis on the detection and localization tasks of the system, but it will still touch on suppression. This includes a discussion on revisit/suppression flight path-planning to potential fires, the suppression drop procedure as specifically applied to the M300, and reviewing some data from suppression drop testing.

3.3.1 Path-Planning using Positions of Interest

After an initial broad area survey flight where a UAV has identified potential fires, the previously discussed DBSCAN clustering algorithm is applied to laser rangefinder data associated with a detected fire. The important outputs from the application of DBSCAN are the identified cluster centroids. These centroids are locations relatively close to potentially detected fires, and are generally referred to as POIs in the Crossfire system. They can be used to plan a revisit/suppression flight for the Crossfire system that visits each of these POIs optimally according to some constraint. This class of problem is typically referred to as a routing problem.

Google has published an open source library of optimization tools that it refers to as OR-Tools. Within this tool suite is a sub-library dedicated to routing optimization that has published APIs and software libraries [43]. These solvers can be applied to problems such as the one in this thesis: finding a route that visits all POIs while minimizing the route's distance. The specific solver selected for the Crossfire system was the Traveling Salesperson Problem (TSP) solver.

The TSP solver is equipped to calculate the optimal (i.e., least cost or shortest) path for a single vehicle to visit all nodes in a given set. The more general case for the TSP solver is the Vehicle Routing Problem (VRP), which allows for more than one vehicle in the solution. This is currently out-of-scope for the Crossfire system and this thesis, however it may be introduced as the system grows in complexity.

The TSP solver generally works as follows:

1. A symmetrical nxn distance matrix that calculates the distance between n nodes is computed.

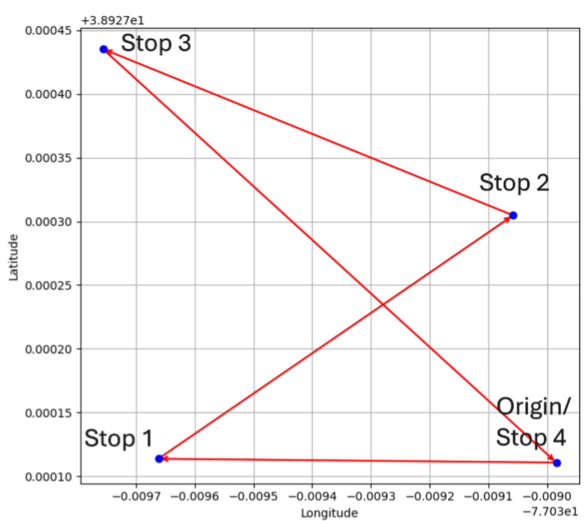
2. A cost evaluator function is defined that can reference the distance matrix and some function for cost and returns the cost to travel between two given nodes.

3. Search parameters are set, such as optimizing for the shortest path.

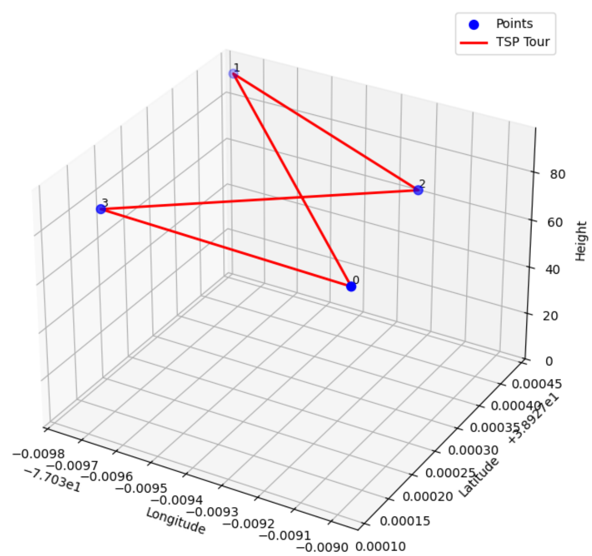
4. The TSP solver is called with the above inputs and returns an ordered array of the optimal path with node identifiers.

The TSP solver is an admittedly complex algorithm, and is more computationally intensive than needed for cases where only a few POIs are identified. That stated, an important aspect of the Crossfire system is its *scalability*. In test cases where there is a singular suppression UAV visiting a few POIs, computationally cheaper routing algorithms would work just as well. However, the decision was made to implement a TSP solver so that if the scope of the Crossfire system grows—as is likely—than the framework needed to solve for more complex routing problems will already be in place.

An example of the TSP solver applied to the output of the DBSCAN clustering algorithm in 3.13 can be seen in 3.14. It is displayed in both 2 dimensions (3.14a) and 3 dimensions (3.14b), while annotated to demonstrate the calculated route in the 2-dimensional version.



(a) The outputted TSP route projected in 2D.



(b) The outputted TSP route in 3D.

Figure 3.14: A TSP route solution projected in 2D 3.14a and 3D 3.14b with nodes taken to be cluster centroid from 3.13.

For the Crossfire system, the generated route is exported from the post-processing script in the form of a Keyhole Markup Language (KML) file. This is a file type that is used to format and store geographic data. Most of DJI's UAVs—such as the M300—can import KML files via their controllers and covert the data into an automatic waypoint flight. This is precisely what the generated KML files in the Crossfire system are used for. More on this process will be discussed in Chapter 4.

3.3.2 Suppressive Agent Drop Procedure

The third-party drop system integrated into the Crossfire system is the DroMight Talon V1. Information on this system is given in Section 2.2.

The process for dropping the suppressive agent payload on any fires is relatively simple but requires mostly manual inputs from the UAV pilot. The procedure is laid out in the diagram in 3.15.

The drop system and procedure have a few requirements. The first and most important is that the suppressive agent can interface with the drop system. This is relatively simple to do, the suppressive agent—i.e. water for all of the testing discussed in this thesis—is contained in a water balloon that is either attached via net or rubber band to the DroMight Talon V1. Second, after the water balloon containing the suppressant is attached to the drop system, the photo-resistor circuit within the Talon V1 must be calibrated. This occurs by turning on the M300's strobe lights, turning them off, and waiting five seconds. After this period, a green LED light on the Talon V1 will indicate that the photo-resistor circuit is calibrated. The next time the strobe light is turned on, the pin holding the suppressive payload will be pulled and the payload will drop.

For the scope of this thesis, all of the suppressive payloads are simply water balloons that burst upon impact with the ground. The Crossfire project team is designing and experimenting with "air-burst" mechanisms that rupture the balloon's membrane some specified time and height

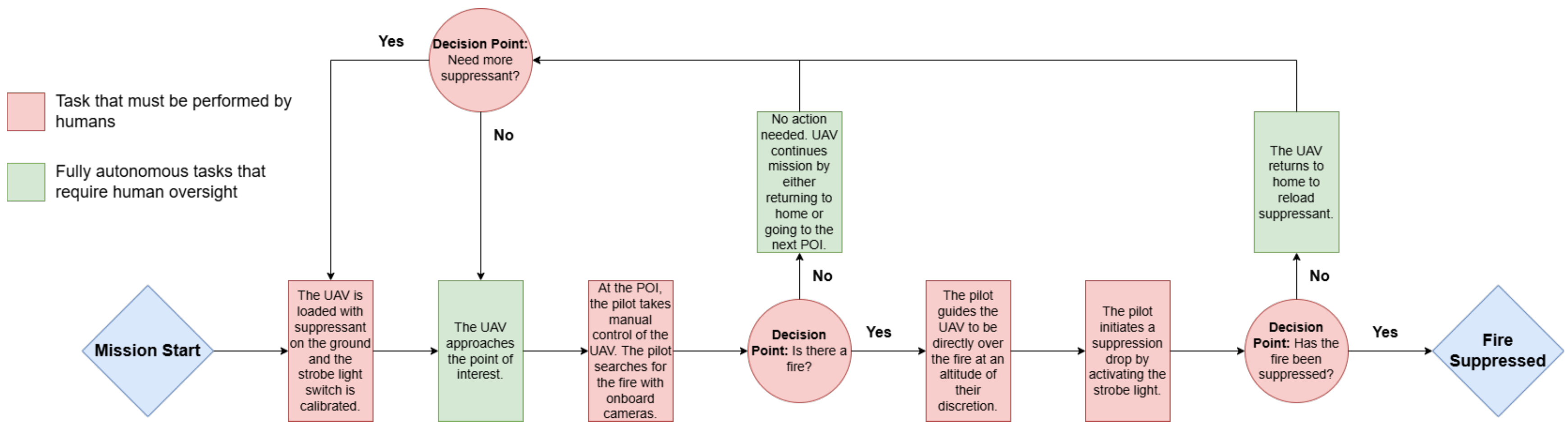


Figure 3.15: Fire drop procedure for a single fire. Test conducted at MFRI La Plata, February 5, 2025.

after being dropped. This will allow for a more dispersed application of suppressant, which will likely perform better at extinguishing fires. The Crossfire team is pursuing a purely-mechanical version and a electric-circuit version of the “air-burst” mechanisms. These improved drop systems will not be expounded upon any further in this thesis, but are worth noting.

3.3.3 Suppression Testing

The maximum payload capacity of the M300 when accounting for the weights of its camera payloads as well as the Talon V1 drop system is somewhere in the range of 1.2-1.5 kg. The Crossfire team has conducted experiments to estimate the required mass of a water ballon that bursts upon impact to suppress an incipient wildfire. Those estimates are in the range of 7-8kg—well above the maximal payload for the M300 platform. Therefore, suppression drop testing utilizing the M300 was conducted in order to *demonstrate* the capability of delivering suppressant payloads to potential fires, not to actually extinguish fires.

The procedure for drop testing is relatively simple:

1. A target on the ground is marked. A reference frame emanating from the target is defined.
2. A ballon is loaded on to the drop system, and the photo-resistor circuit is calibrated.
3. The UAV is piloted directly above the target at some desired altitude. Wind conditions are recorded.
4. The balloon payload is dropped. The point of impact is recorded.
5. The distance between the point of impact and the target is recorded in terms of coordinates in the defined target reference frame.

Drop Altitude (ft./m)	Impact Distance from Target (m)	Drop Altitude / Impact Distance Ratio	Attachment Method
50 / 15.24	0.050	0.0033	Net
60 / 18.29	0.639	0.0349	Net
70 / 21.34	0.573	0.0268	Net
80 / 24.38	1.332	0.0546	Rubber Band
100 / 30.48	1.202	0.0394	Rubber Band

Table 3.1: Drop Testing Data

The results from drop testing are tabulated in Table 3.1. During this test, all balloons were filled to a mass of 1kg and there was no measurable wind. Generally, as the drop altitude increased, the impact distance distance (error distance) from the target increased as well. There are deeper subtleties not necessarily captured by this data.

There are two different attachment methods for the M300 drop system: a net and a rubber band. They are displayed in flight configurations in 3.16. From the limited testing that was conducted, it is apparent that the net generally leads to a smaller impact distance (see the Drop Altitude/Impact Distance Ratio column). From a qualitative perspective, this is unsurprising. On-board video taken by the UAV during drop testing demonstrated that balloons attached via rubber band will oscillate like a pendulum when the UAV moves through space. Therefore, the ballon has some notable horizontal velocity component when it is dropped, leading to larger impact distances from the target. The net attachment also experienced this pendulum motion, however it was generally damped more and therefore the initial horizontal velocities imparted on the balloon at drop time were smaller. In order to confirm these hypotheses, more testing will need to be conducted.

The drop testing conducted for this thesis has been invaluable in understanding how any revisit/suppression missions for the Crossfire system must be designed. In the next Chapter, the

(a) The M300 with a balloon attached via a rectangular net.

(b) The M300 with a balloon attached via rubber band.

Figure 3.16: Images of the M300 in flight carrying a balloon payload attached by net (3.16a) and by rubber band (3.16b).

end-to-end system design will be discussed, and experimental results will be presented.

Chapter 4: Experimental Demonstration of Integrated System

This chapter will begin by detailing how the previously discussed components in Chapters 2 and 3 are integrated together to form the Crossfire system. Then, a short briefing on the testing facilities utilized throughout this research will be presented. Finally, experiment results and discussion thereof will complete the chapter.

4.1 Concept of Operations

The Concept of Operations (ConOps) for the Crossfire system consists of two different types of flight missions: a *Broad Area Survey*, and a *Revisit/Suppression Flight*. During the broad area survey, the UAV surveys the entirety of the test area and streams data down to the ground station, where potential fires are detected and localized. During the revisit/suppression flight, the UAV returns to Points of Interest identified during the broad area survey and follows the Suppressive Agent Drop Procedure if the pilot confirms the presence of a fire.

4.1.1 System Components

Most of the components of the Crossfire system have already been discussed in detail. To reiterate, the system components are

- **UAV Platform:** The DJI M300 RTK quadrotor UAV with its built-in flight computer, sensor suite, etc.

- **Payload Cameras:** The Zenmuse H20 RGB camera and the FLIR Vue TZ20 thermal camera, both of which interface with the M300.

- **UAV Contollers:** Two DJI Smart Controllers that interface with and stream video from the M300.

- **Ground Control System:** A laptop or PC that can accept video streams via HDMI capture card from the Smart Controllers. The laptop or PC must also run the real-time processing software.

The Ground Control System (GCS) has not yet been discussed in detail. A more thorough explanation of its operational responsibilities follows.

4.1.2 Ground Control System Design

The Ground Control System, in brief, is designed as follows: two Smart Controllers transmit video streams via HDMI and HDMI capture card to the ground station laptop. These video streams are used as inputs in the fire detection software. A primary operator can pilot the drone via a Smart Controller while its video simultaneously streams to the laptop. A second operator can initiate the fire detection software on the laptop and also has the ability to adjust the camera gimbal angles on the second Smart Controller. A diagram which demonstrates the roles and responsibilities as well as how they interface with the GCS components can be seen in 4.1.

The fire detection software—sometimes referred to as processing software—admits HDMI streams from the Smart Controllers. One stream is the RGB payload camera video with overlaid

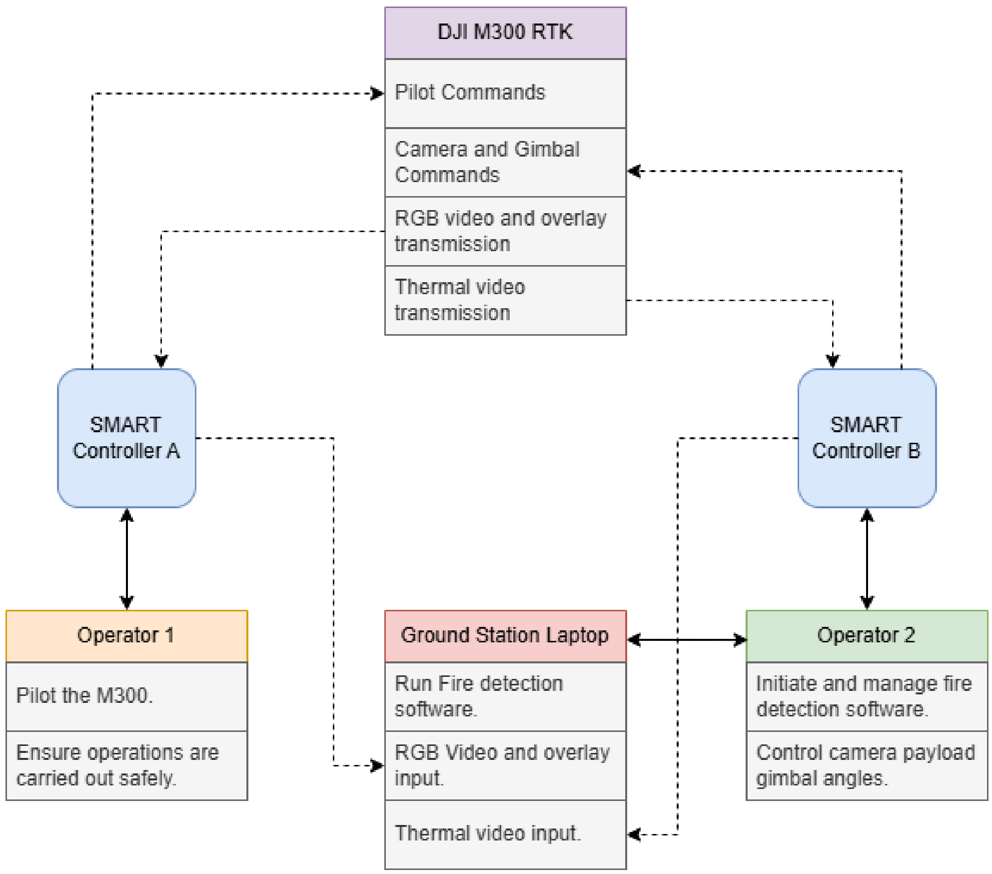


Figure 4.1: Operator roles and responsibilities. Dashed lines show wired or transmitted connections between system components. Solid lines indicate where humans are required to interface with components.

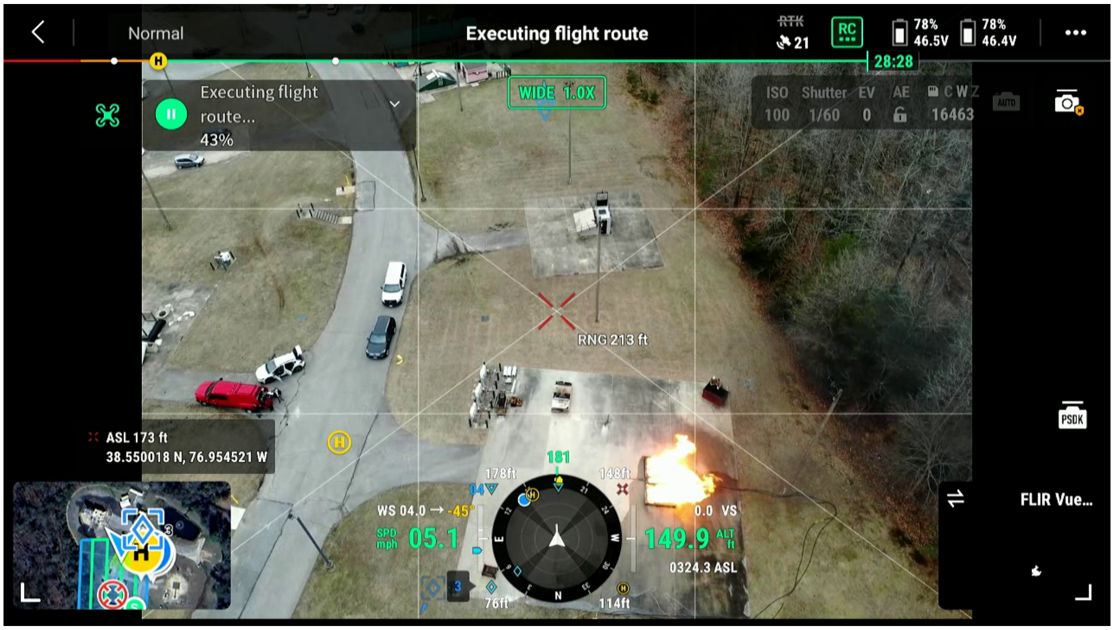
telemetry (from Smart Controller A operated by the pilot), and the second stream is the thermal camera video (from Smart Controller B operated by the second operator/ground station manager). Samples of the video inputs as they are inputted into the the fire detection software are shown in 4.2.

There are a few items of note about the sample images. For one, the thermal image streams with a significant black border. The actually boundaries of the thermal image are annotated in red (Figure 4.2b). Upon input in the fire detection software, every thermal image frame is cropped to the highlighted red boundaries. Secondly, a separate homography matrix is utilized to transform the cropped live-stream thermal image to the live-stream RGB image. Its values are different than the sample homography matrix discussed in Section 3.1.

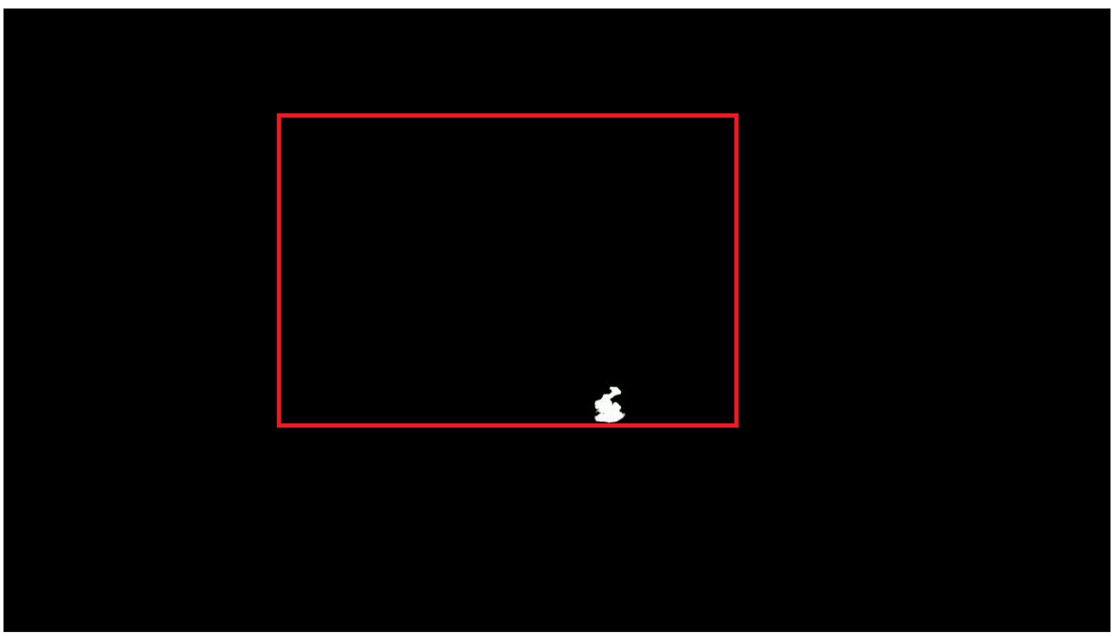
The fire detection software is ran as a Python program from a command line terminal. It is able to handle "streaming blackouts", i.e. whenever the HDMI streams cut out the software continues to run. User input is required to terminate the program, however. Depending on the ground station laptop/PC, it can take up to a minute for the fire detection software to initialize. Therefore, it is the best practice for the software to be initialized while the UAV is on the ground prior to its broad area survey flight.

While the fire detection software runs, the blended video output with annotated fire detections is displayed in real-time on the ground station laptop/PC. Therefore, the processing software can also be used to aid the system operators in fire detection tasks beyond the broad area survey. One such task may be to help determine fire detections during revisit/suppression flights.

Lastly, the fire detection software records and stores the blended video output and the fire detection XML file into defined file paths on-board the ground station laptop/PC.



(a) A sample image frame of the RGB video live-stream with telemetry overlay.



(b) A sample image frame from the thermal video live-stream with the boundary of the image annotated in red.

Figure 4.2: Sample live-stream images as they are inputted into the Crossfire GCS. Imagery taken at MFRI La Plata, February 5, 2024.

4.1.3 System Initialization

"System Initialization" refers to the processes required to properly power up and calibrate components of the Crossfire system. The steps for initializing the system are as follows:

1. Power on the UAV and associated Smart Controllers.

2. Connect the Smart Controllers via HDMI and HDMI Capture Card to the ground station laptop/PC. Ensure the proper video configurations are streamed by the Smart Controllers.

3. Begin running the fire detection software—configured to save fire detection information as well as a recording of the blended video output.

4. Calibrate the FLIR Vue TZ20 thermal camera. More information follows after this list.

5. Program the automated broad area survey or revisit/suppression flight profile.

6. Initiate the takeoff sequence.

In order to calibrate the FLIR Vue TZ20 thermal camera, the camera settings are set to a Hot Scene Dynamic Range (SDR). The camera is then calibrated so that its temperature-to-intensity mapping for each pixel is locked via the Automatic Gain Control (AGC) setting. This means that the said mapping will not dynamically change with the scene. When the AGC Lock is initiated, the highest intensity detected by the thermal camera (bit value 255) is the minimum of either the maximal temperature encountered in the scene or the hottest allowable temperature in the SDR. The lowest intensity (bit value 0) is mapped to the maximum of either the minimum temperature encountered in the scene or the lowest allowable temperature in the SDR.

Through experimentation, it has been found that there are two reliable methods to calibrate the FLIR Vue TZ20 with AGC Lock. These are to set the AGC lock with either a fire or a 150W ceramic reptile lamp within the field of view of the thermal camera. The reptile lamp is the preferred method, as after a few minutes of being powered on it will reach surface temperatures high above any other items in the testing environment besides the fire itself (the exact surface temperature is not stated by the manufacturer but technical specs place the air temperature 6 inches from the lamp at $113^{\circ}F$).

Programming an automated flight (either for a broad area survey or a revisit/suppression flight) is conducted on the pilot's Smart Controller. DJI has software on-board the Smart Controller that automatically generates broad area survey flight profiles that follow lawnmower patterns. The user must define the search area as well as the flight altitude. For the revisit/suppression flights, a KML file generated by Ground Station post-processing software is imported via a flash memory card (namely a microSD card) that is inserted into the Smart Controller. The flight planning software is capable of reading the KML file and generating an automated waypoint flight from it.

4.1.4 End-to-End System Design

The end-to-end system design for the Crossfire system details all the components and procedures necessary for the process spanning from an initial broad area survey to the suppression of all fires in a testing space. It is extensive, and many of the individual components to the design have been discussed in previous sections.

The end-to-end system design is generally referred to as the fire process chain in this thesis. A block diagram of the fire process chain is presented in 4.3. There are notable identifiers in the block diagram. Any processes in green blocks are fully autonomous, any in red are actions performed by humans, and any in yellow are "semi-autonomous"—meaning that those processes are partially automated. The diagram is also split up into three rows, each of which corresponds to actions performed on the Ground Station, the Smart Controllers, and by the UAV itself.

Beginning from MISSION START, the first part of the fire process chain consists of initializing the Crossfire system and conducting a broad area survey. This is the most automated component of the fire process chain and represents a significant share of contributions of this thesis. In summary, during the broad area survey:

1. The Crossfire system is initialized and a broad area survey flight is defined.
2. The UAV conducts the broad area survey while the ground station receives video streams from the payload cameras and runs fire detection algorithms in real-time.
3. At the end of the broad area survey, the UAV either loiters or lands.
4. During the broad area survey, a fire detection data file in an XML format is generated that will be used to define the revisit/suppression flights mission plan.

This fire detection data file is the critical output from the broad area survey. It is used to calculate the coordinates of Points of Interest that the UAV must revisit. At these POIs, there may be fires that must be suppressed.

At the end of the broad area survey, post-processing software is ran which calculates the POIS and determines the optimal route for re-visiting them. This route is loaded on to the UAV as described above. From there, an iterative loop occurs such that:

1. The UAV approaches a designated POI. At the POI the pilot takes manual control.

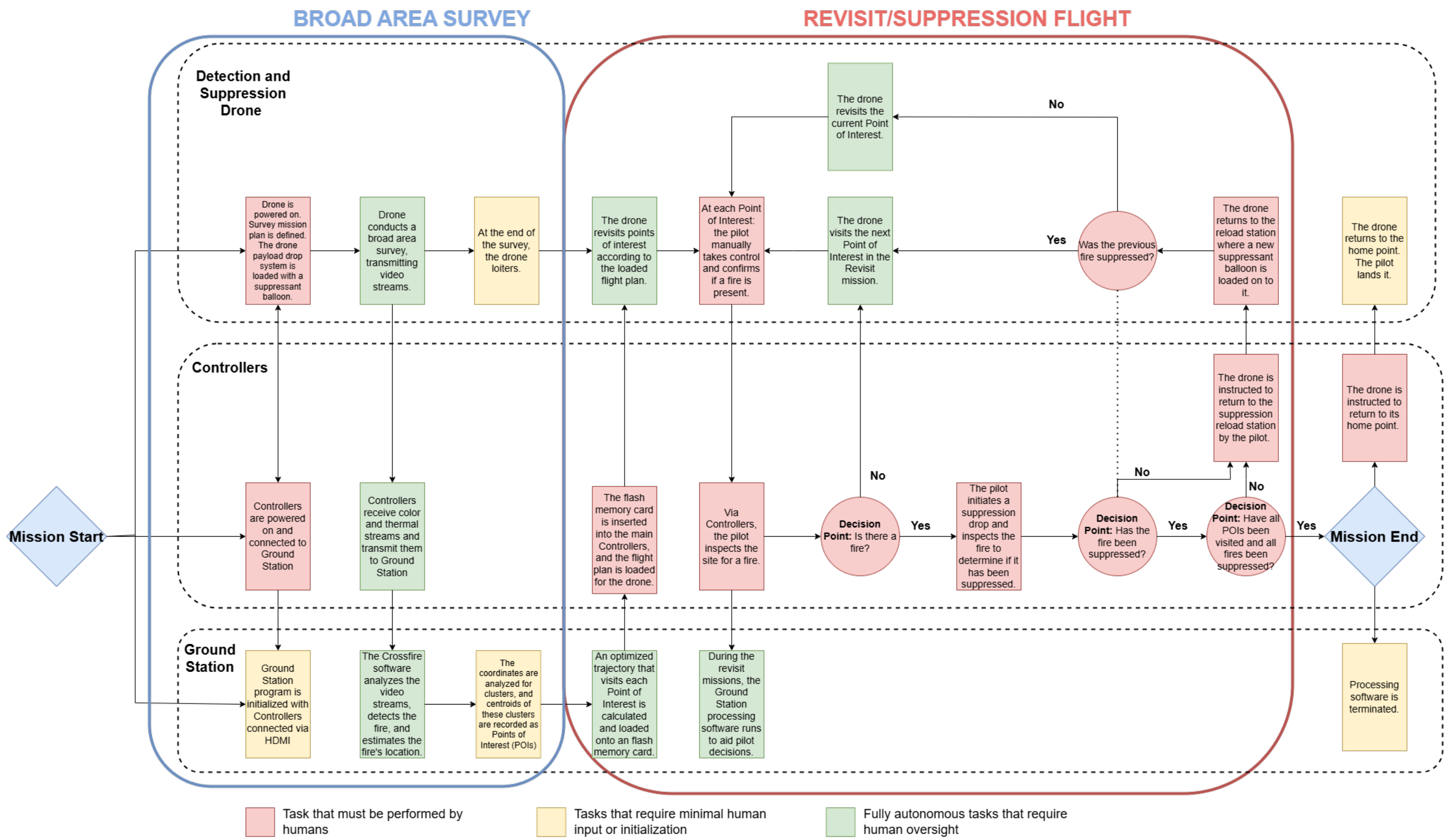


Figure 4.3: Fire Process Chain Procedure

2. The pilot determines if there is a fire present—both by manual inspection and/or with the aid of fire detection software.

3. If a fire is present the, the pilot positions the UAV above the fire and initiates a suppressant drop. If it is not present, then the next POI on the flight plan becomes the designated POI and the process returns to step 1.

4. After dropping suppressant on the fire, the pilot determines if it had been extinguished. If so, the next POI becomes the designated POI. If not, then the current designated POI remains so.

5. The UAV is returned to a reloading station, a new suppressant payload is loaded. The process returns to step 1.

This loop continues until all POIs have been investigated and all fires have been extinguished. Depending on the complexity of the test, this may not be possible with maximum suppressant payload masses of roughly 1kg. That stated, this iteration of the Crossfire system is intended to be *demonstrative* in regards to suppression tasks, not necessarily a be-all end-all answer.

4.2 Description of Testing Environments

There is one primary and one secondary testing environment that was used for data collection throughout this research. Both of these environments are run by the Maryland Fire and Rescue Institute (MFRI), with the primary based in La Plata, Maryland, and the secondary in College Park, Maryland. Satellite images of both sites can be seen in 4.4.

The MFRI La Plata site was where most testing was conducted. It lays outside the Capitol

(a) MFRI La Plata test site with testing space boundaries annotated in red.

(b) MFRI College Park test site with testing space boundaries annotated in red.

Figure 4.4: Testing sites used in testing the Crossfire system.

Exclusion Zone, meaning that no special waivers are needed to fly light-weight UAVs up to altitudes of 400 ft. The permitted air space for testing (outlined in red in 4.4a) has an area of roughly $20,000\ m^2$, or roughly $0.02\ km^2$. The facility is capable of lighting a pool fire (whose power are unrecorded by site staff) and pallet fires (power estimated at 500 kW). The pool fire is roughly sized at 8 by 10 feet, while the pallet fires are about 2 by 3 feet. Additionally, there are obstacles and potential false positives scattered throughout the testing space such as numerous aluminum car chassis that are capable of reaching temperatures higher than their external environments. Most figures and data presented in this thesis were collected the the La Plata facility.

The MFRI College Park facility was utilized for ground testing for this thesis. It lays within the Capitol Exclusion Zone so a special waiver is needed for flying UAVs at any altitude. It therefore remained an option but was not utilized once flight testing began. The College Park facility has multiple training towers and a large concrete pad where pallet fires can be placed. As an example, images from Figure 3.8 were taken at MFRI College Park.

4.3 Experimental Testing and Results

In this section, experimental results from tests of the fire process chain will be discussed. The analysis will first focus on broad area survey results, then revisit/suppression flights results, and finally a discussion on the performance of the end-to-end system.

4.3.1 Analysis of Broad Area Survey Testing

The procedure for testing fire detection and localization is relatively simple. A testing area is defined where some sort of fire (usually a pool fire managed by MFRI) could be lit within its boundaries. From there, the UAV embarks on a broad area survey and the fire detection software is ran on the ground station laptop/PC. The fire processing software is configured such that data can be recorded and analyzed in post-processing. All fire detection and localization flight tests occurred at MFRI La Plata.

For the first round of testing, one pool fire was lit in the testing area and the UAV conducted two broad area surveys at an altitude of 100ft. above ground level (AGL) and one broad area survey at 150ft. AGL. For one of the 100 ft. AGL tests, the thermal camera was calibrated with the pool fire in frame, and for the other two tests the thermal camera was calibrated with a reptile lamp in frame. For all tests, the cameras were pitched at an angle of -45° from the line radiating forwards of the UAV body.

The results from the first test—survey altitude of 100 ft. AGL and thermal calibration via the fire—are visible in 4.5. In the plot, the flight trajectory is depicted in green. The circles correspond to positions reported by the laser rangefinder, where white infills indicate no fire was detected in the given frame and orange infills indicate that a fire was detected in the frame. The ground truth position of the fire is marked as a red square. The cluster centroid(s) resulting from the DBSCAN algorithm being applied to the data set of detected fire image frames are marked as blue squares.

There is a notable pattern with the geometry of the flight path. Due to the camera gimbals being pitched at -45°, the geospatial positions of the reported laser rangefinder coordinates are

Figure 4.5: Detections and Flight Trajectory from broad area survey of a singular pool fire at 100 ft. AGL, thermal camera calibrated via fire. Test Date: February 5, 2025.

"ahead of" the flight trajectory. For example, if the UAV is traveling northwards, then the reported laser rangefinder coordinates would be expected to be a distance equal to the altitude of the UAV northwards of the UAV's current positions, assuming the gimbal pitch angle is -45°. For a flat-Earth model, this expected leading distance (d) would vary by a function of the altitude (h) and pitch angle (θ) as

$$d = \frac{h}{\tan(-\theta)} \tag{4.1}$$

This equation is important to note in order to sanity check the data. In 4.5, from a glance, "lines" of laser rangefinder data appears to be shifted by a distance of roughly 30m ($100ft.$) from the beginning of each column sweep of the lawnmower pattern. This tracks according to our equation for the expected distance shift (we'd expect $d = 100ft.$, the altitude of the flight), and therefore the curious geometry of the laser rangefinder positions relative to flight trajectory is indicative of our expected results.

For the application of the DBSCAN algorithm, the parameters were set such that $\epsilon = 300ft.$ and $n_{min} = 4$. This was a generously large cluster radius that effectively ensured that any detected fire image frames taken relatively close together would register as the same cluster. The centroid of this singular cluster was estimated to be the fire location—within only a few meters of the ground truth location. Granted, this approach heavily relies on utilizing the constraint that there was only one fire in the test space. For real applications, the ϵ parameter must be reduced to allow for multiple fire detections within small areas. From a brief post-processing analysis of the data from the tests depicted in 4.5, 4.6, and 4.7, it was evident that ϵ can be reduced to values as low as 100 ft. for only one cluster to be detected. Ultimately, this ϵ value is bounded by some combination of the expected leading distance, broad area survey geometry (distance

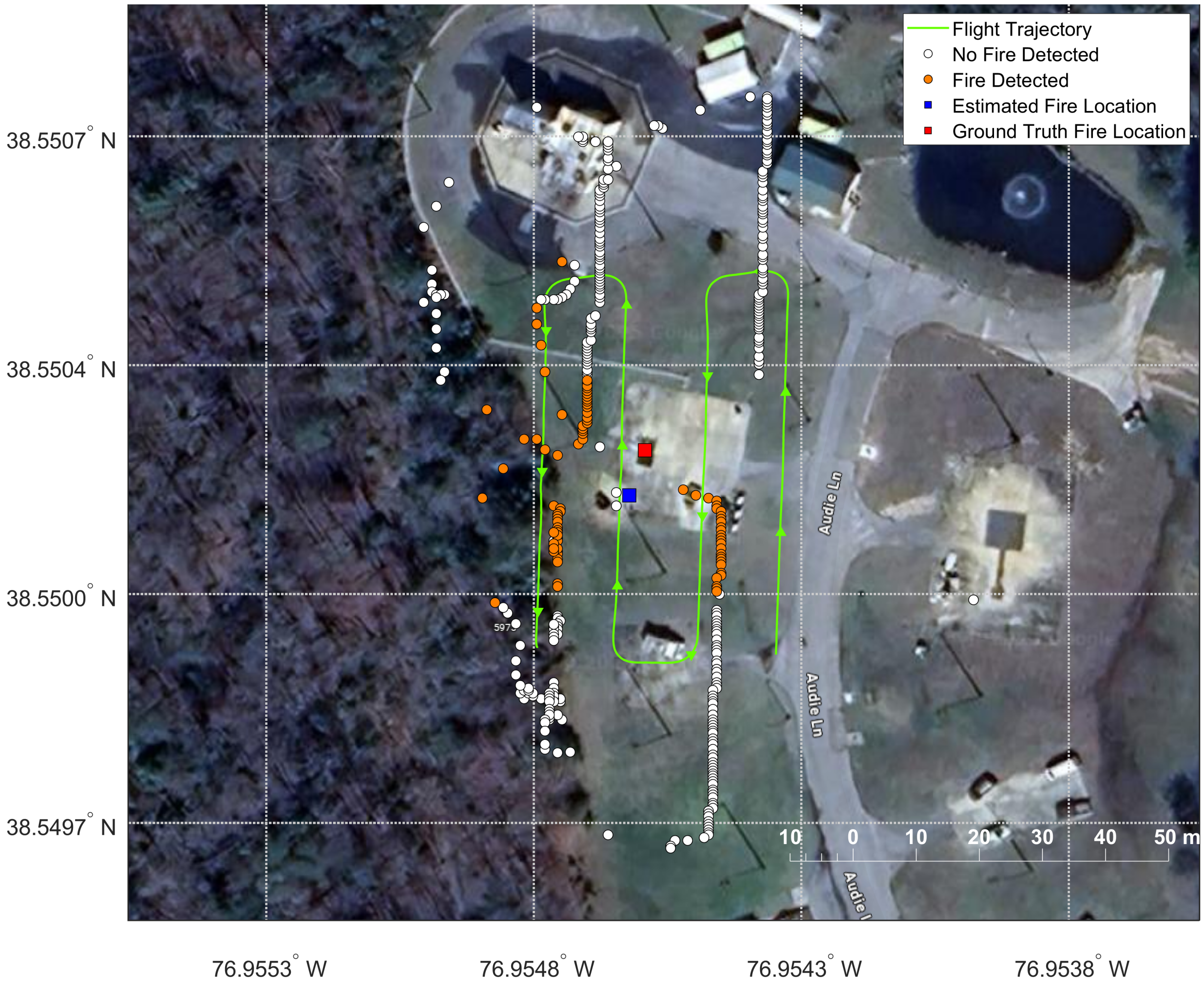


Figure 4.6: Detections and Flight Trajectory from broad area survey of a singular pool fire at 100 ft. AGL, thermal camera calibrated via lamp. Test Date: February 5, 2025.

between columns), and the flight altitude. The exact relationship is not trivial to solve for but would make for an interesting investigation.

The same broad area survey flight profile at an altitude of 100 ft. AGL was applied to a test where the thermal camera was calibrated with a reptile lamp. The results are visible in 4.6. The data is very similar to that discussed previously. This demonstrates that the reptile lamp is a viable calibration source. It also speaks to the repeatability of the lawnmower flight profile in producing valid results for broad area surveys.

Figure 4.7: Detections and Flight Trajectory from broad area survey of a singular pool fire at 150 ft. AGL, thermal camera calibrated via lamp. Test Date: February 5, 2025.

Another broad area survey was conducted at 150 ft. AGL with a reptile lamp calibration. The expected leading distance of the laser rangefinder data relative to the flight trajectory is now longer, however it is apparent that the detection data when clustered still produces an estimated fire location within 10m of the ground truth location. This is likely due to the symmetry of the flight trajectory around the ground truth fire location. The plotted results are presented in 4.7.

Another test date was planned where more broad area surveys were ran. Specifically, five were conducted: two at an altitude of 100ft., and one each at altitudes of 150ft., 200ft., and 300 ft. Both surveys at 100ft. had two fires lit in the test space. One of these ”two-fire” surveys was conducted as part of an end-to-end Crossfire system demonstration and will discussed in a later subsection. The most notable change in this second round of testing was that the flight trajectory was expanded so that the surveilled area was roughly four times larger. The thermal camera was calibrated with a reptile lamp for all of these tests.

The most unique results are presented in 4.8. During this test, two fires were lit in the test space while the UAV surveyed at an altitude of 100 ft. AGL. Laser rangefinder coordinates associated with detected fires were clustered in two groups, each of which was near either of the ground truth fire locations. These results demonstrate that the DBSCAN algorithm used to group these coordinates can be applied to appropriately differentiate data resulting from different fires. For this survey flight—and all others on March 26th—the DBSCAN parameters utilized were $\epsilon = 100ft.$ and $n_{min} = 4$.

Further broad area surveys and their detection results for a one-fire test space are presented, with a survey altitude of 150 ft. seen in 4.9, 200 ft. in 4.10, and 300 ft. in 4.11. Much of the same behavior as seen in previous results are present in these three plots. The surveilled area was significantly larger than in previous tests, so the velocity at which the UAV traveled was increased

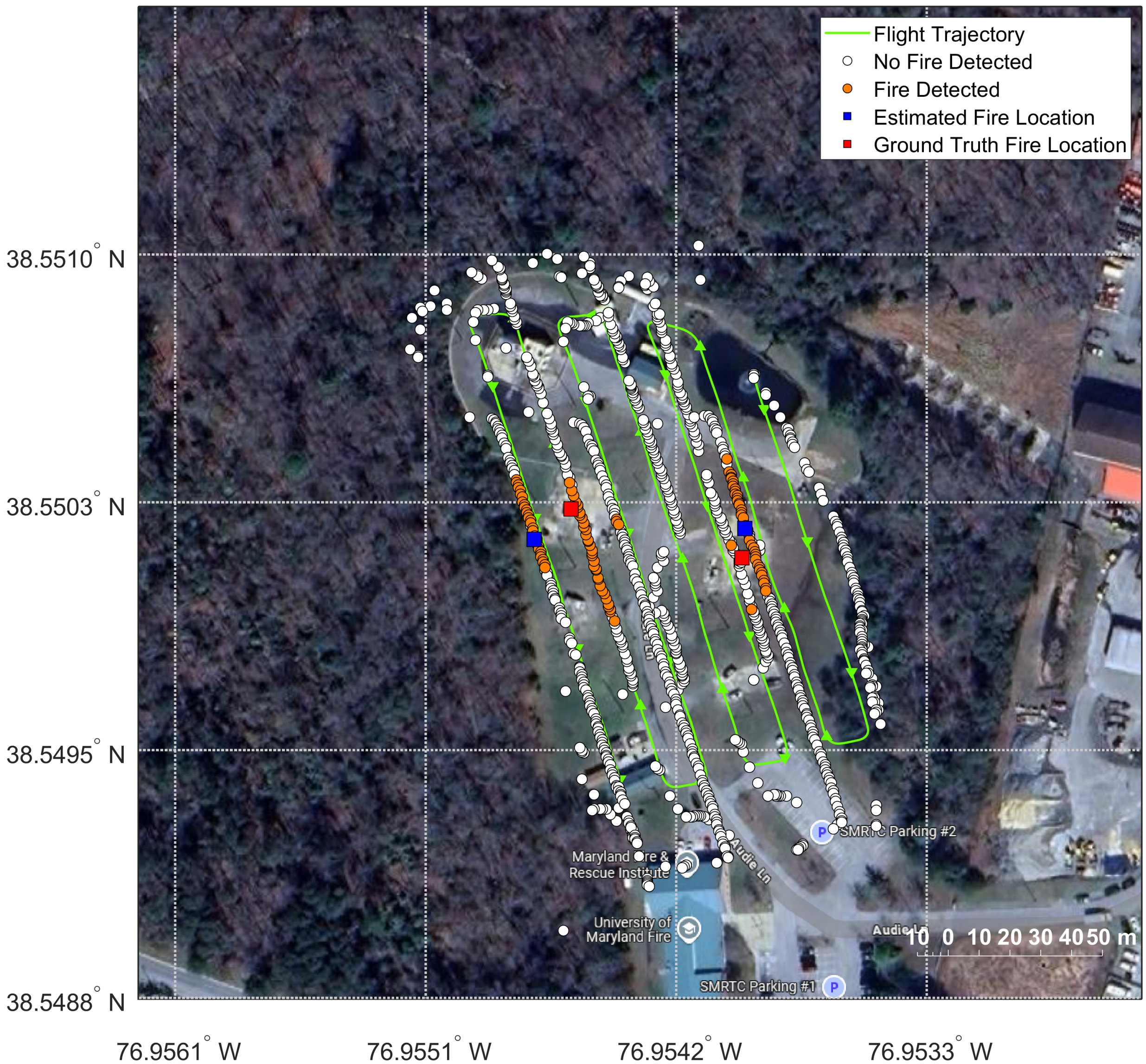


Figure 4.8: Detections and Flight Trajectory from broad area survey of two fires at 100 ft. AGL. Test Date: March 26, 2025.

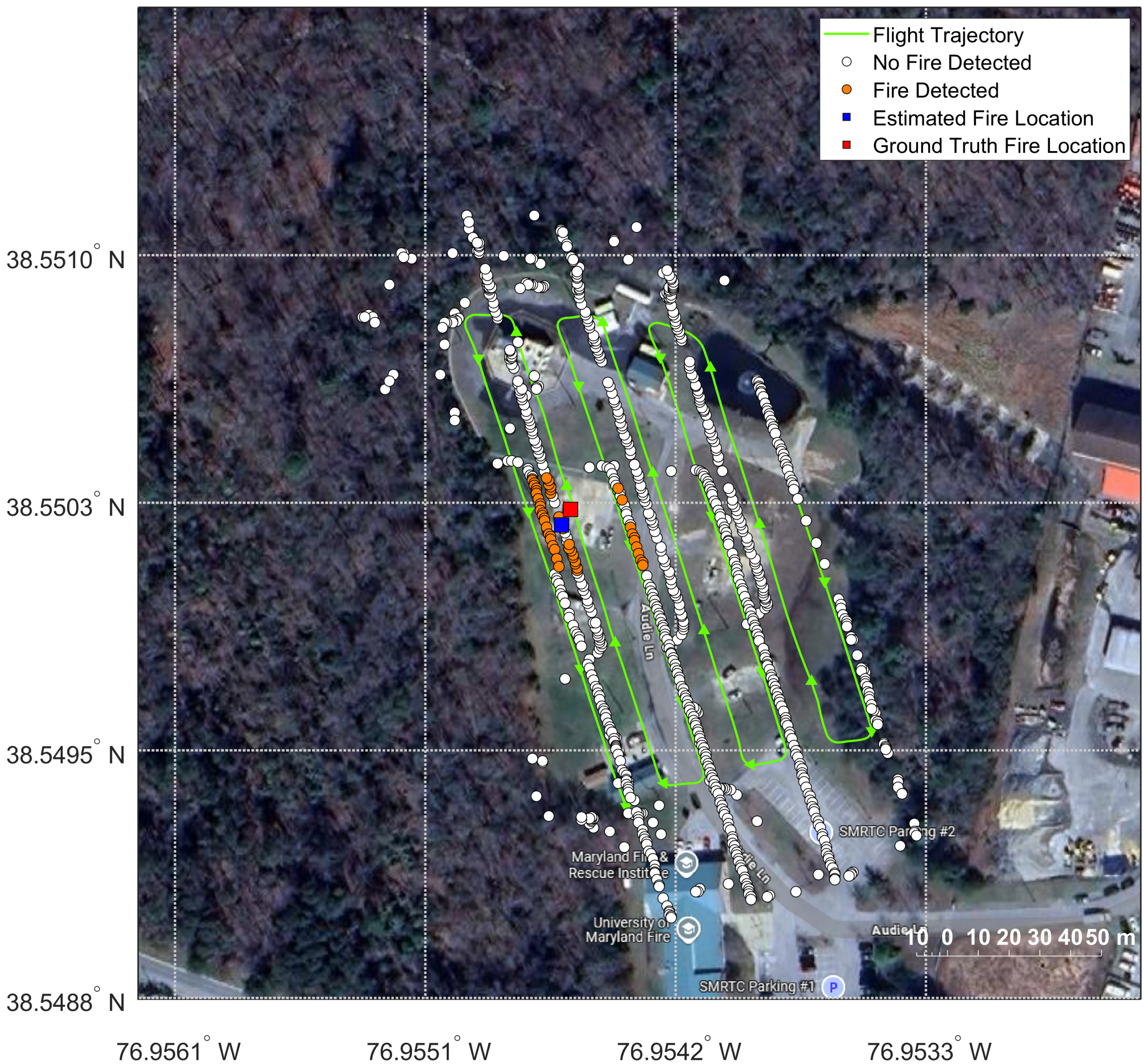


Figure 4.9: Detections and Flight Trajectory from broad area survey of a pool fire at 150 ft, UAV Velocity = 10 MPH. AGL. Test Date: March 26, 2025.

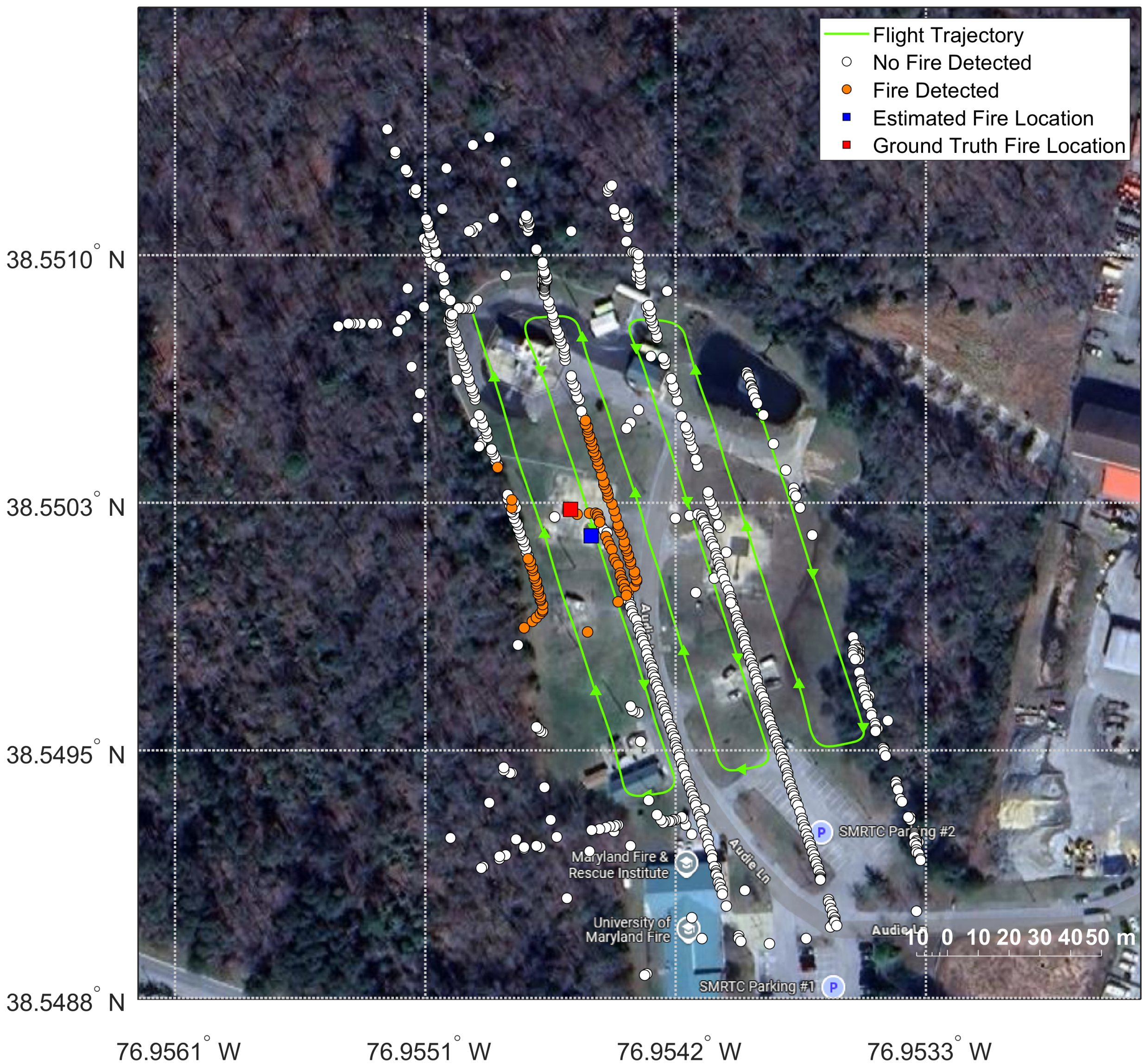


Figure 4.10: Detections and Flight Trajectory from broad area survey of a pool fire at 200 ft. AGL, UAV Velocity = 10 MPH. Test Date: March 26, 2025.

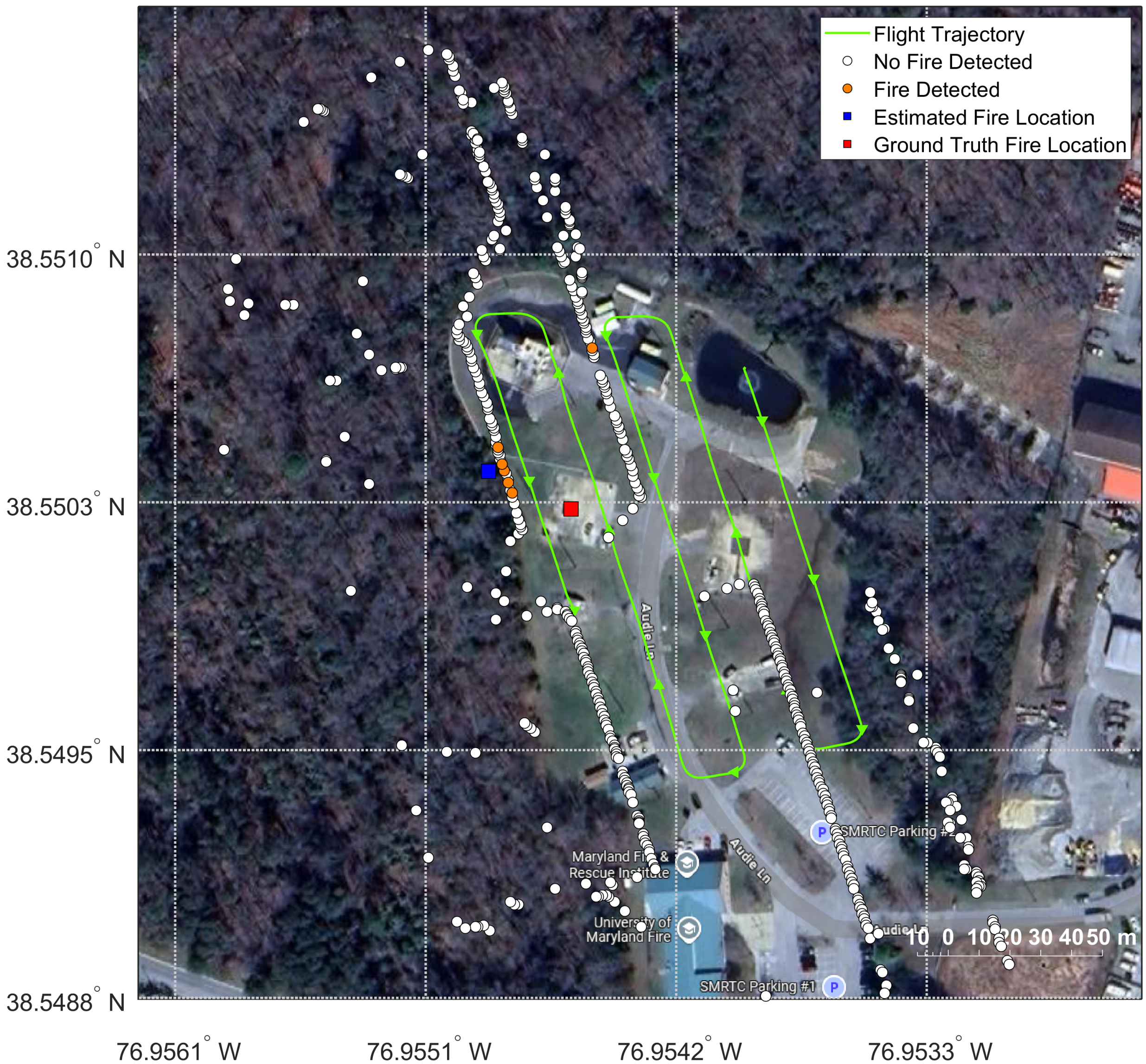


Figure 4.11: Detections and Flight Trajectory from broad area survey of a pool fire at 300 ft. AGL, UAV Velocity = 10 MPH. Test Date: March 26, 2025.

Flight Survey Altitude (ft. AGL)	“Medium” Model No. of Detections	“Large” Model No. of Detections
150	60	68
200	112	124
300	2	6

Table 4.1: Number of Detections for Model Type, Test Date: March 26, 2025

from 5 MPH to 10 or 12 MPH. In each case, the surveillance flights were completed in roughly 5 to 6 minutes. There do not appear to be drawbacks from increasing the UAV velocity, however the numbers of detections drops drastically as some non-linear function of altitude.

As altitude increases, the fire is comfortably detected by the detection algorithm up to at least 200 ft. AGL. At some point between 200 and 300ft. AGL, the relative size of the fire to the UAV from the ground is too small for the RGB object detection model to reliably detect the fire. There appears to be some limit to how high this version of the Crossfire system can fly and still produce a reliable number of detections. To further this investigation, experiments were conducted in post-processing to compare the “medium” and the “large” fire detection models in the hope that the “large” model would be better at detecting fire at higher altitudes. As a reminder, the “medium” model has been used throughout this thesis in order to optimize accuracy and processing speed.

From these experiments, the absolute number of fire detections from post-processing runs of the detection software on the flight data were tabulated. The results can be seen in Table 4.1. In essence, the “large” model usually induces slightly more true positive fire detections than the “medium” model. That stated, the “large” model has a processing time of roughly double that of the “medium” model, so in most scenarios the “medium” model is the optimal choice. At higher altitudes, the number of detections produced by both types of models drops drastically, so the

Test Date	Broad Area Survey Altitude (ft.)	Thermal Camera Calibration Method	Fire Localization Radial Error (m)
Feb. 5, 2025	100	Fire	5.45 ± 0.27
Feb. 5, 2025	100	Lamp	7.51 ± 0.27
March 26, 2025	100	Lamp	12.66 ± 0.27
March 26, 2025	100	Lamp	9.77 ± 0.27
March 26, 2025	100	Lamp	15.53 ± 0.27
March 26, 2025	100	Lamp	9.59 ± 0.27
Feb. 5, 2025	150	Lamp	9.31 ± 0.30
March 26, 2025	150	Lamp	5.84 ± 0.30
March 26, 2025	200	Lamp	10.81 ± 0.33
March 26, 2025	300	Lamp	29.47 ± 0.39

Table 4.2: Fire Localization Error

altitude limit on reliable detections can not simply be solved by running a more complex RGB fire detection model. In all broad area survey flights but one, the "medium" detection model was utilized. For the survey presented in 4.11, the "large" model was used as the "medium" model did not produce enough detections for the minimal cluster size needed in order to estimate the fire's position.

The error of the estimated fire location relative to to ground truth location across the all previously discussed tests is presented in Table 4.2. There are limited tests to reference data from due to the involved nature of carrying out one such test, however there are trends evident. For one, as the broad area survey altitude grows, it is likely the fire localization error grows as well. This is due to the fact that the fire will be in frame for laser rangefinder coordinates that are increasingly far away from the fire ground truth location. The variance of these coordinates will grow with altitude, and therefore the centroid of those clustered points will likely drift away from the ground truth position as surveillance altitude grows.

This data is assembled into a plot seen in 4.12. There are only 10 data points in the plot, however some trends are evident. As the surveillance flight altitude increases, the fire localization

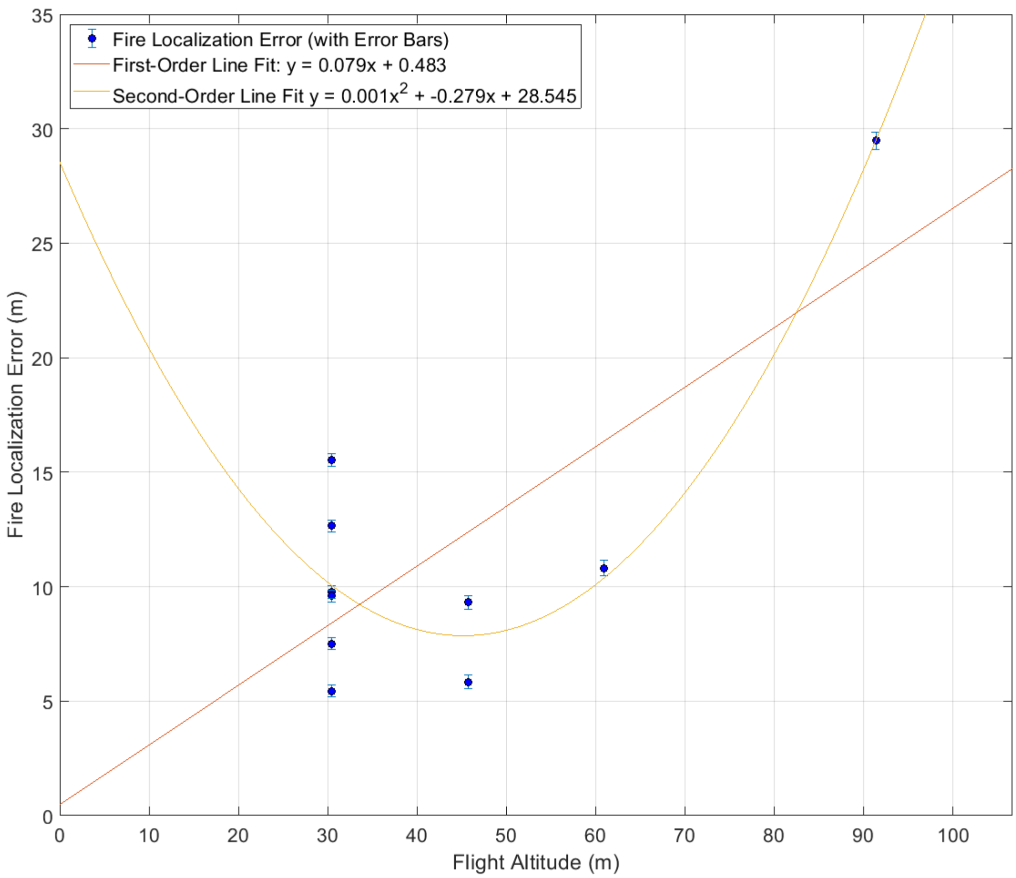


Figure 4.12: Fire Localization Error as Compared to Altitude. Test Dates: February 5 and March 26, 2025.

error as compared to the ground truth positions generally grows. This is not surprising as there are less detections at higher altitudes, meaning there is less laser rangefinder data to use in estimating the fire position. The error bars are set according to an error formula from the laser rangefinder manufacturer[29], where the measurement accuracy σ is

$$\sigma = \pm(0.2 + 0.0015D)m \tag{4.2}$$

where D is the range to the measured object.

A first-order and second-order polynomial fit was generated in attempts to quantify the relationship between the localization error and the flight altitude. More data needs to be collected in order to make more certain claims about this relationship, however a few trends are evident. For one, as the flight altitude nears 100m ($\sim 330ft.$), the localization error increased drastically. This is likely due to the fact that the number of detections at higher altitudes is small, so the geometry of laser rangefinder coordinate clusters around fires is unlikely to be symmetrical. Therefore, the centroid of said clusters will be further from the ground truth location of the fire.

A second consideration is that as the flight altitude decreases, it is possible the fire localization error increases (as seen in the second-order polynomial fit). An explanation may be that at lower altitudes, the number of image frames where a fire can possibly be detected is relatively small. Thus, the position estimate of the fire is more influenced by the geometry of the survey flight than for surveys at higher altitudes. Ultimately, more data would need to be collected in order to verify or disprove this hypothesis.

A final investigation bore from the results of this section regarded the coverage rate achievable by this configuration of the Crossfire system. From the above results, in order to get a relatively accurate estimate of the fire's position, a lawnmower surveillance pattern should be designed where the fire is visible in at least two passes. This constraint allows for symmetry about the fire's ground truth location in the set of laser rangefinder coordinate associated with that fire's detection.

From this information, an equation can be derived for an upper bound on the coverage rate achievable for this iteration of the Crossfire system. Defining and setting a couple of parameters regarding the camera configurations—the smallest FOV is 82.9° and the gimbal angle $\theta = -45^{\circ}$—we can derive the width (w) of the area observed by the UAV sensors via a simplified triangle model of the camera's field of view. Skipping the detailed derivation, this width is

$$w = 2\frac{h}{\cos\theta}\tan(FOV/2) = 2.498h \simeq 2.5h \tag{4.3}$$

where h is the surveillance altitude.

The coverage rate (CR) is simply this width times the UAV velocity divided by 2 (2 passes are required of each point). Therefore

$$CR = wv = 1.249hv \simeq 1.25hv \tag{4.4}$$

where v is the UAV velocity and coverage rate is in units of area per time.

Utilizing this equation, a plot was created that displays the coverage rate of the Crossfire system at different UAV velocities and altitudes, seen in 4.13. These slopes are compared to the coverage rates required for 2, 4, and 8 UAVs of the Crossfire system configuration to fully surveil a $1km^2$ area in a 10 minute period.

From flight test data, the maximum altitude desired for surveillance is likely around or slightly above $200ft.$ AGL, or roughly $60m.$ We can observe the plot and determine that, to cover a $1km^2$ area in a 10 minute period, we would need 4 UAVs flying at slightly faster than 10 MPH. This would ensure that the UAV's can operate within a reasonable flight envelope (in terms of altitude and speed) while minimizing the number of UAVs required. A 2 UAV system may not be achievable for these constraints unless the detection algorithms/camera hardware is updated such that it can reliable detect fires at higher altitudes. Information such as this will help inform the next iteration of the Crossfire system.

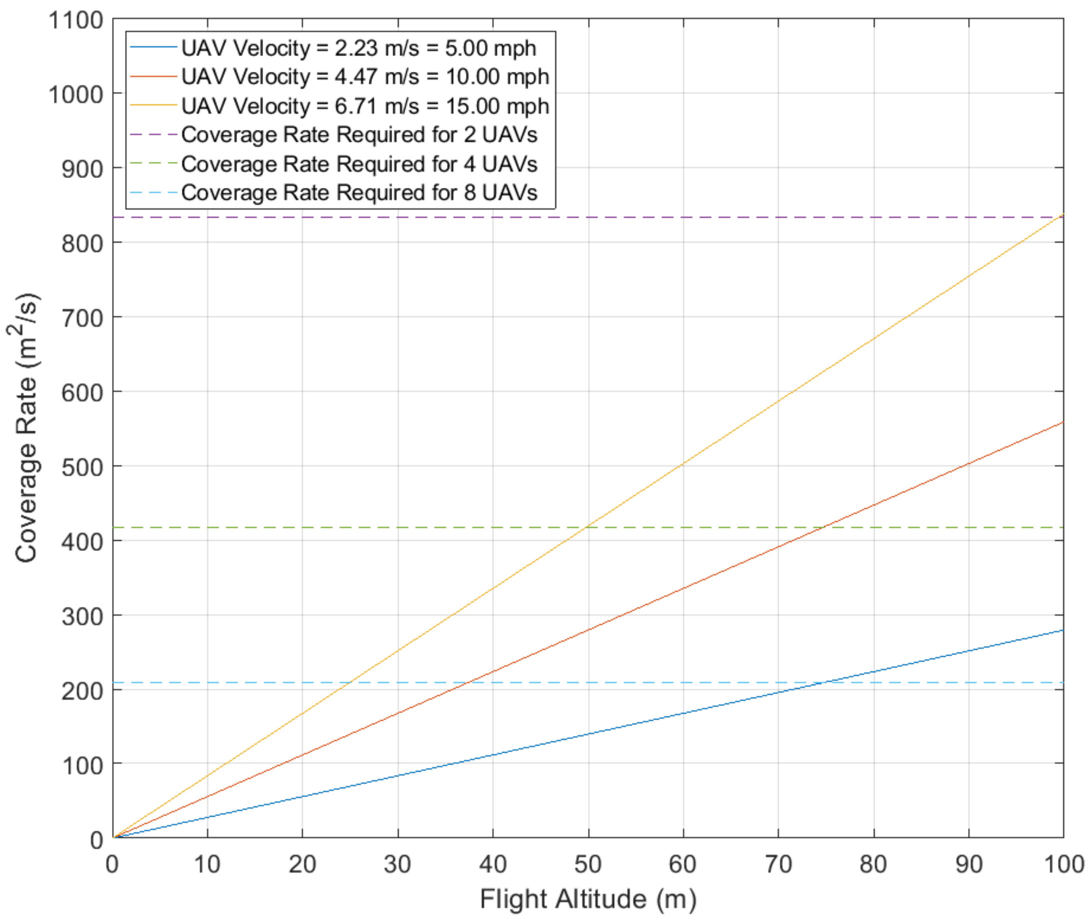


Figure 4.13: Coverage Rate Analysis According to a Simplified Model of the Crossfire system.

4.3.2 Analysis of Revisit/Suppression Flight Testing

Revisit/suppression flight testing is *demonstrative* in nature as balloons of the maximum payload size for the M300 are not large enough in mass to suppress fires. This section will focus on revisit/suppression flights generated from specific broad area survey flights as well as reference the drop testing details in Section 3.3.

During the first round of testing (February 5, 2025), a revisit/suppression flight was generated from the broad area survey depicted in 4.5. The revisit/suppression flight utilizes all estimated fire locations as waypoints, and adds a final waypoint that is near the takeoff point of the UAV during the broad area survey. In the referenced test, there was only one detected fire, so in total the revisit/suppression flights consists of two waypoints.

The broad area survey results and trajectory data from the revisit/suppression flight are presented in 4.14. Although no explicit suppression payloads were dropped during this test flight, it was important to confirm that the automatically generated waypoint flight began at a home point, visited the estimated fire location, and then returned to near that home point. From the flight telemetry plotted in 4.14b, it is apparent that that is what the M300 did.

There is a slight delta between the estimated fire location in 4.14a and the corresponding waypoint in 4.14b. This is due to the fact that the broad area survey plot was generated from a post-processing run of the fire detection software after changes were made to the XML output routine in order to publish more information. The revisit/suppression flight KML file used for the revisit flight was generated in the field from a real-time run of an earlier version of the fire detection software. Therefore, the expectation that the first waypoint in 4.14b matches exactly with the fire location estimate in 4.14a is violated. Ultimately, the small difference ($\sim 5m$) is

(a) Broad Area Survey Results, 100ft. AGL survey altitude.

(b) Revisit/Suppression Flight generated and conducted from broad area survey in 4.14a

Figure 4.14: Revisit/Suppression flight generated from broad area survey at 100ft. AGL

due to slight changes in the frame rate at which the image data was processed as well as small changes to when when the data was truncated in post-processing. There is nothing concerning about this delta, but it is worth noting that the frame rate at which fire detection processing occurs can have slight but random effects on the final estimated fire positions.

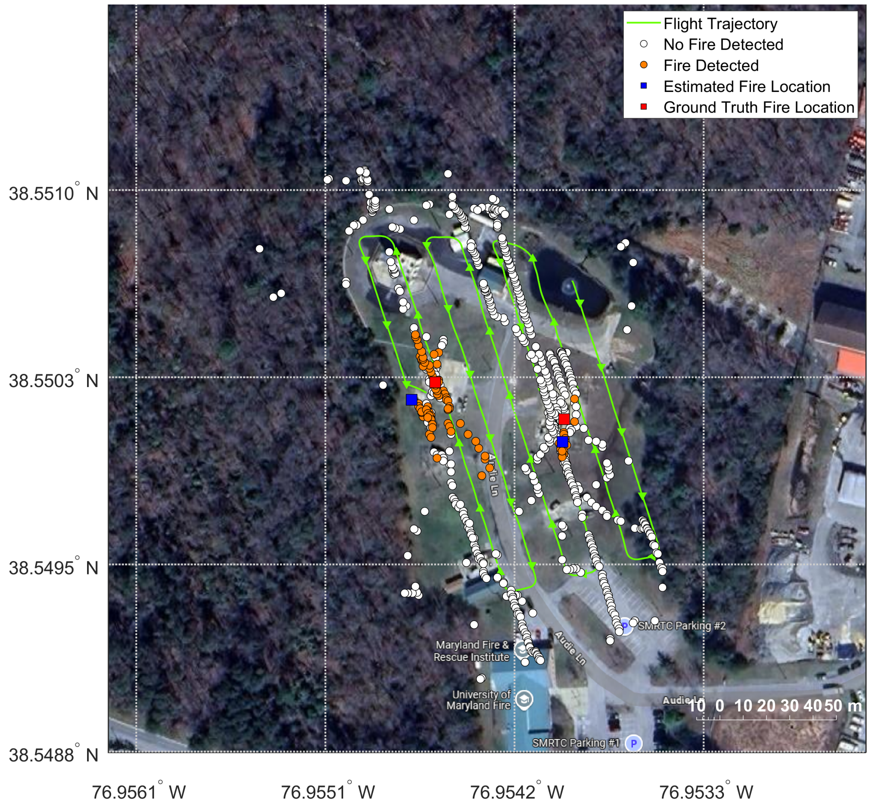
Ultimately, this revisit/suppression flight trajectory test demonstrated that the M300 is capable of interfacing with the Crossfire system pipeline in order to generate and then carry out the trajectories needed for the revisit/suppression flights. A full end-to-end demonstration which included a revisit/suppression flight for two detected fires will be discussed next.

4.3.3 Performance of End-to-End System

On the last testing day (March 26, 2025), there was a full end-to-end demonstration of the fire process chain performed by the Crossfire system. This included the processing software running in real-time from video data transmitted from the UAV, a revisit/suppression flight whose route was generated from the data, and attempts to suppress a fire with a 1kg balloon payload.

The broad area survey was conducted at $100ft$. AGL, and the camera gimbal was operated manually to point directly at the fire (this was part of an investigation to see if manually pointing the camera at the fire would improve the localization estimates, which is thus far inconclusive). The results are presented in 4.15.

For one, the estimates of the fire positions were both within about 12m of the ground truth location. From these estimates, a revisit/suppression Flight was generated whose trajectory is visible in 4.15b. A water balloon payload was attached to the M300 during the revisit flight. A suppression drop was attempted on the eastern fire. The first drop missed and the M300 had to be



(a) Broad Area Survey Results, 100ft. AGL survey altitude, two fires.

(b) Revisit/Suppression Flight generated and conducted from broad area survey in 4.15a

Figure 4.15: Revisit/Suppression flight generated from broad area survey at 100ft. AGL, with two fires in the test space.

piloted back to the ground station so that it could be reloaded with suppressant. It then returned to the fire and managed to successfully hit the target. That is why a looping trajectory is present in the eastern half of the flight path.

Ultimately, 1 kg water balloon dropped from roughly 50 ft. was not expected to suppress a fire lit with three stacked 2' x 3' wooden pallets. The water balloon did make a notable difference in suppression efforts, however. Images taken from the on-board payload RGB camera for the UAV pre- and post-impact are presented in 4.16. During the drop, the balloon struck the metal support bar slightly above the fire—fortuitously simulating an air burst effect. From the imagery, it is evident that most of the top layer of the fire was suppressed. These results are promising and indicate that a larger water balloon payload with a built-in air burst mechanism is potentially capable of suppressing an incipient wildfire.

Overall, the end-to-end system demonstration went reasonably well. The fire detection software which was ran in real-time performed adequately, and in post-processing routines applied to its generated XML data file, the two fires were easily distinguishable. Therefore, the revisit/suppression flight trajectory was generated in such a way that at each waypoint the pilot was easily able to identify if a fire did or did not exist at each point of interest. From there, when suppressant drops were initiated, the UAV system did well in delivering the suppressant to the fire. In cases where the suppressant struck the fire, the conflagration was notably affected. In cases where it wasn't, the UAV was quickly instructed to return to the ground station so that a new suppressant payload could be loaded on to it.

A timeline of the broad area survey and the revisit/suppression flight for this integrated test is presented in 4.17. Generally, both flights in the mission took about 6 minutes to execute. There was 20 minutes of downtime in between those flights due to in-field debugging (resolved

(a) Suppressant Drop, Pre-impact.

(b) Suppressant Drop, Post-impact.

Figure 4.16: Onboard UAV images for a drop test on a real fire, pre- and post-impact.

by disconnecting the FLIR Vue TZ20 thermal camera). The flight velocity could be doubled from the ~ 6 MPH it was during this testing, cutting the broad area survey time in half. The downtime between the broad area survey and the revisit/suppression flight could also be cut down to roughly 5 minutes. With these improvements, the timeline of the end-to-end mission starting at the beginning of the broad area survey as conducted in this scenario could be reduced to the following, with pain points residing in the downtime between the broad area survey and the revisit/suppression flight.:

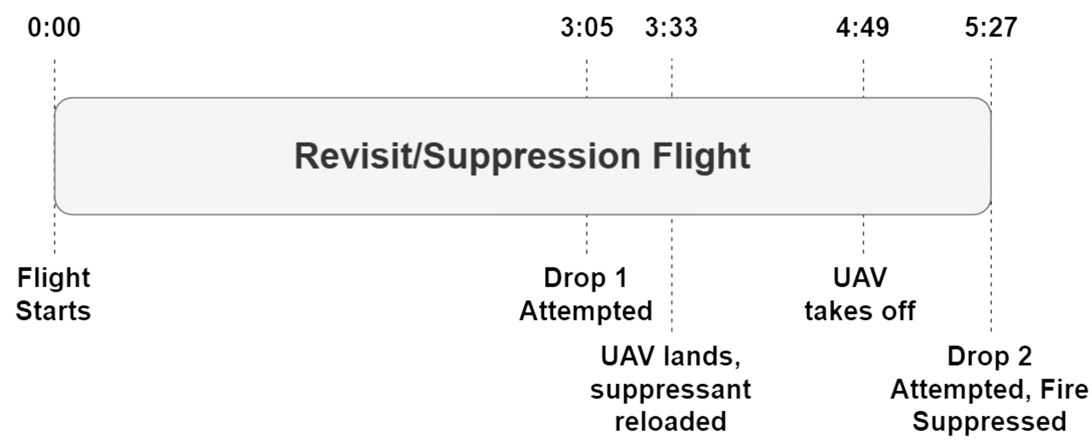
1. **0-3 minutes:** Conduct broad area survey.

2. **3-8 minutes:** Return to home, load UAV with suppressant and new flight profile.

3. **8-14 minutes:** Conduct revisit/suppression flight, attempting two suppressant drops on one fire.

The steps in the fire process chain were conducted relatively smoothly and easily. There is room for improvement—such as in further refining and automating the localization routines, reducing the processing time for routines between the broad area survey and the revisit/suppression flight, or designing methods to allow for multiple UAVs to be used in the system. That stated, this iteration of the Crossfire system is promising in its ability to detect, localize, and suppress wildfires. Critical information was learned throughout designing this initial system that will be key in the development of future multi-agent iterations.

(a) Broad Area Survey Timeline



(b) Revisit/Suppression Flight Timeline

Figure 4.17: Broad Area Survey and Revisit/Suppression Flight Timelines for the Integrated System Test.

Chapter 5: Conclusion

5.1 Summary of Contributions

This thesis explores the development of a UAV system that is capable of detecting, localizing, and suppressing incipient wildfires. First, the thesis details the extensive development and testing of a multi-spectral fire detection algorithm. The algorithm leverages neural network object detection routines as well as classical computer vision techniques in order analyze and fuse together data spanning across the infrared and visual spectrum. This capability is critical in detecting and localizing fires. Second, the thesis reviews how the integration of a laser rangefinder with multi-spectral sensors allows for localizing fires while they are detected. This technique relies on clustering algorithms, and its output can be used with Traveling Salesman Problem solvers in order to optimize flight trajectories for revisiting potential fires. Third, a third-party drop system is integrated with a commercial UAV and drop testing is conducted in order to demonstrate the capability of delivering suppressant to a target. Finally, the three contributions are bundled together into a semi-autonomous mission framework referred to as the fire process chain. The Crossfire system—the UAV platform, sensors, and peripherals used for this thesis—is rigorously tested across multiple scenarios of the fire process chain in order to validate its capabilities. Ultimately the Crossfire system demonstrates its ability to detect and localize fires while ignoring potential false positives, route for revisit/suppression missions, and to deliver suppressant payloads to identified targets.

5.2 Future Work

There are several key areas in which future work should aim to improve the Crossfire system and in general for autonomous UAV systems applied to wildfire detection, localization, and suppression.

- **Implementing Detection and Localization Algorithms On-board a UAV:** In the future, the Crossfire system will need to utilize UAVs that are capable of processing and analyzing image data using their on-board computers. This means that the detection and localization algorithms must be re-written into a language such as C and implemented to run in real-time on the UAV's on-board computer. Work will soon begin to integrate the detection and localization algorithms into a Chimera-D—a quadrotor UAV with RGB and thermal camera payloads, a laser rangefinder, and a flight computer with a built-in GPU.
- **Training a Custom Fire and Smoke Detection Model:** An important aspect of the future Crossfire system will be utilizing a custom-trained fire and smoke YOLO model. This is important as the training data can be fit to environments such as wildlands, forests, and deserts—the likely locations of any deployed systems. Having a more thorough understanding on what types of smoke are detected would also allow for integrating smoke detection routines into surveillance UAVs. For example, if during a broad area survey a UAV spots smoke on the horizon, its flight trajectory could be adjusted to immediately head towards the source of the smoke.
- **Improving Fire Localization Routines:** There are multiple ways that the fire localization

routines could be improved. Immediately, one could estimate the fire's location by taking a weighted average of cluster laser rangefinder coordinate data such that the coordinates associated with image frames where the images are closer to the center of the image are weighted more heavily. Another method could be to automate gimbal angle routines such that when a fire is detected in an image frame, the gimbal angles adjust such that the fire is centered in the RGB image frame. From there, a more precise fire location estimate could be extracted.

- **Splitting the fire process chain across Multiple UAV Types:** In this thesis, the M300 was used as both a surveillance UAV and a suppressant UAV. While it is equipped to carry out both tasks, it does not do so optimally. The next iteration of the Crossfire system should split surveillance and suppressant tasks among UAVs designed for each purpose. This will add many layers of complexity and will require edge processing, redundant communication systems, and a ground control station capable of communicating with multiple UAVs. However, it will allow for the Crossfire system to be implemented in larger and more complex domains.

Bibliography

[1] XPRIZE. *XPRIZE Wildfire Competition Guidelines*, 2.2 edition, 2024.

[2] Joanne Zwinkels. *Light, Electromagnetic Spectrum*, pages 1–8. 09 2015.

[3] Abonia1. Yolov8-fire-and-smoke-detection.

[4] Yinhui Luo, Xingyi Wang, Yanhao Liao, Qiang Fu, Chang Shu, Yuezhou Wu, and Yuanqing He. A review of homography estimation: advances and challenges. *Electronics*, 12(24):4977, 2023.

[5] Doug Richardson, Amanda S. Black, Damien Irving, Richard J. Matear, Didier P. Monselesan, James S. Risbey, Dougal T. Squire, and Carly R. Tozer. Global increase in wildfire potential from compound fire weather and drought. *npj Climate and Atmospheric Science*, 5(1):1–12. Publisher: Nature Publishing Group.

[6] Preliminary after-action report: 2023 maui wildfire.

[7] Roger Vincent. Estimated cost of fire damage balloons to more than $250 billion. Section: Business.

[8] Wildland fire incident management field guide.

[9] Usage patterns and costs of unmanned aerial systems | congressional budget office.

[10] Taha Zaman, Muhammad Hasan, Saneeha Ahmed, and Shumaila Ashfaq. Fire detection using computer vision. In *2018 IEEE 61st International Midwest Symposium on Circuits and Systems (MWSCAS)*, pages 356–359, 2018.

[11] Xiwen Chen, Bryce Hopkins, Hao Wang, Leo O'Neill, Fatemeh Afghah, Abolfazl Razi, Peter Fulé, Janice Coen, Eric Rowell, and Adam Watts. Wildland fire detection and monitoring using a drone-collected rgb/ir image dataset. *IEEE Access*, 10:121301–121317, 2022.

[12] Alberto Viseras, Michael Meissner, and Juan Marchal. Wildfire front monitoring with multiple uavs using deep q-learning. *IEEE Access*, pages 1–1, 2021.

[13] Licheng Feng and Jay Katupitiya. Vector field based control of quadrotor uavs for wildfire boundary monitoring. *Journal of Intelligent Robotic Systems*, 106, 09 2022.

[14] Fereidoun H. Panahi, Farzad H. Panahi, and Tomoaki Ohtsuki. A reinforcement learning-based fire warning and suppression system using unmanned aerial vehicles. *IEEE Transactions on Instrumentation and Measurement*, 72:1–16, 2023.

[15] Luis Merino, Fernando Caballero, J Ramiro Martinez de Dios, Ivan Maza, and Aníbal Ollero. Automatic forest fire monitoring and measurement using unmanned aerial vehicles. In *Proceedings of the 6th International Congress on Forest Fire Research*. Citeseer, 2010.

[16] Fatemeh Afghah, Abolfazl Razi, Jacob Chakareski, and Jonathan Ashdown. Wildfire monitoring in remote areas using autonomous unmanned aerial vehicles. In *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 835–840.

[17] Fereidoun H. Panahi, Farzad H. Panahi, and Tomoaki Ohtsuki. A reinforcement learning-based fire warning and suppression system using unmanned aerial vehicles. *IEEE Transactions on Instrumentation and Measurement*, 72:1–16. Conference Name: IEEE Transactions on Instrumentation and Measurement.

[18] Sayed Pedram Haeri, Abolfazl Razi, Sahand Khoshdel, Fatemeh Afghah, Janice Coen, Peter Fule, Adam Watts, Nick-Marios Kokolakis, and Kyriakos G. Vamvoudakis. *A comprehensive survey of research towards AI-enabled unmanned aerial systems in pre-, active-, and post-wildfire management*.

[19] Pirunthan Keerthinathan, Narmilan Amarasingam, Grant Hamilton, and Felipe Gonzalez. Exploring unmanned aerial systems operations in wildfire management: data types, processing algorithms and navigation. *International Journal of Remote Sensing*, 44(18):5628–5685. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/01431161.2023.2249604.

[20] Lingxia Mu, Yichi Yang, Ban Wang, Youmin Zhang, Nan Feng, and Xuesong Xie. Edge computing-based real-time forest fire detection using UAV thermal and color images. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, PP:1–12.

[21] Burchan Aydin, Emre Selvi, Jian Tao, and Michael Starek. Use of fire-extinguishing balls for a conceptual system of drone-assisted wildfire fighting. *Drones*, 3.

[22] Nusrat Jahan, Tawab Niloy, Jannatul Silvi, Mahdi Hasan, Ishrat Nashia, and Riasat Khan. Development of an IoT-based firefighting drone for enhanced safety and efficiency in fire suppression. *Measurement and Control*, 57:1464–1479.

[23] Kai Wang, Yingfeng Yuan, Mengmeng Chen, Zhen Lou, Zheng Zhu, and Ruikun Li. A study of fire drone extinguishing system in high-rise buildings. *Fire*, 5:75.

[24] Mission — xprize foundation.

[25] DJI. *M300 RTK User Manual*, 4th edition, 2023.

[26] Yanming Feng and Jinling Wang. GPS RTK performance characteristics and analysis. *Journal of Global Positioning Systems*, 7.

[27] KMZ files | keyhole markup language.

[28] DJI. *DJI SMART Controller User Manual*, 1st edition, 2019.

[29] Specs - zenmuse h20 series - DJI enterprise.

[30] Teledyne FLIR. *Vue TZ20 User Manual*, 1st edition.

[31] The best drone payload drop system - DroMight talon frequently asked questions (FAQ).

[32] Richard Szeliski. *Computer Vision: Algorithms and Applications*. Springer-Verlag, Berlin, Heidelberg, 1st edition, 2010.

[33] Vladimir Pašagić, Marijan Muževič, and Dubravko Kelenc. Infrared thermography in marine applications. *Brodogradnja: An International Journal of Naval Architecture and Ocean Engineering for Research and Development*, 59(2):123–130, 2008.

[34] Rikke Gade and Thomas B. Moeslund. Thermal cameras and applications: a survey. *Machine Vision and Applications*, 25(1):245–262, Jan 2014.

[35] Robert W. Byren. *Laser Rangefinders*. SPIE, July 1993.

[36] Sathya S, Madhumathi R, Manjula D, and MeenaKowshalya A. Performance evaluation of blob detection techniques using image processing. In *2024 4th International Conference on Sustainable Expert Systems (ICSES)*, pages 1101–1106, 2024.

[37] Pierre Kornprobst, Jack Tumblin, and Frédo Durand. Bilateral filtering: Theory and applications. *Foundations and Trends in Computer Graphics and Vision*, 4:1–74, 01 2009.

[38] Rejin Varghese and Sambath M. Yolov8: A novel object detection algorithm with enhanced performance and robustness. In *2024 International Conference on Advances in Data Engineering and Intelligent Computing Systems (ADICS)*, pages 1–6, 2024.

[39] Gerard Dreyfus and G Dreyfus. Neural networks: an overview. *Neural Networks: Methodology and Applications*, pages 1–83, 2005.

[40] Ray Smith. An overview of the tesseract ocr engine. In *ICDAR '07: Proceedings of the Ninth International Conference on Document Analysis and Recognition*, pages 629–633, Washington, DC, USA, 2007. IEEE Computer Society.

[41] Kamran Khan, Saif Ur Rehman, Kamran Aziz, Simon Fong, and S. Sarasvady. Dbscan: Past, present and future. In *The Fifth International Conference on the Applications of Digital Information and Web Technologies (ICADIWT 2014)*, pages 232–238, 2014.

[42] C Carl Robusto. The cosine-haversine formula. *The American Mathematical Monthly*, 64(1):38–40, 1957.

[43] Vincent Furnon and Laurent Perron. Or-tools routing library.