Vision-Driven Autonomous UAS for Search and Rescue

Ian Higgins

CMU-RI-TR-25-63

July 2025

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee:
Sebastian Scherer, *chair*
Maxim Likhachev
Samuel Triest

Submitted in partial fulfillment of the requirements for the degree of Master of Science in Robotics.

To my family.

Abstract

Search and rescue missions demand rapid, reliable, and intelligent action in uncertain, often degraded environments. Autonomous unmanned aerial vehicles (UAVs) are increasingly well suited for these tasks, offering scalable coverage, high mobility, and the ability to reach otherwise inaccessible areas. However, to be effective in real-world search and rescue operations, UAVs must navigate safely near obstacles, detect and avoid flying vehicles and animals, localize robustly under sensor and environmental degradation, and plan informative paths in vast, uncertain spaces.

This thesis presents a narrative centered on enabling search and rescue capabilities across several robotics domains and arguing for further development and use of vision-based autonomous UAVs in search and rescue contexts. The discussion focuses on the advantages of these systems. These include vision-based detect-and-avoid (DAA) capability, sensor-aware path planning based on locations of interest, vision-based localization when GPS is unreliable, and human interpretability of camera data. Also included is ongoing work on an informative path planning (IPP) system for maritime search, motivated by aerial search for lost vessels. Collectively, these efforts highlight the practicality and utility of robotics systems in the search and rescue domain.

Acknowledgments

Thanks to my friends and family for giving me a great life. Thanks to the AirLab for working with me on this research. Thanks to Sebastian Scherer for giving me this opportunity.

Funding

This work was supported by robot fans.

Contents

When this dissertation is viewed as a PDF, the page header is a link to this Table of Contents.

List of Figures

List of Tables

Chapter 1

Introduction

Introduction.

1.1 Thesis statement

The thesis of this document is as follows: robotics techniques – especially autonomous, camera-reliant, small unmanned aerial vehicles – offer immense practical utility in the domain of search and rescue, and the works presented demonstrate various capabilities from which such a system would benefit.

To bolster this claim, I will detail various search and rescue scenarios and hypothetically apply various robotics systems and techniques from the literature, focusing on contributions of myself and my colleagues, but also contributions from across the field of robotics.

1.2 Search and rescue

1.2.1 What is search and rescue?

Search and rescue is the process of locating a missing person and responding to their needs, which generally include bringing them supplies, administering first aid, and transporting them back to a safe place or emergency care. In this document, we will focus on the locating of a missing person by a team that is within a few miles. In the

Future Work section, we briefly mention work being done in the use of UAS to rescue as well.

1.2.2 Examples of search and rescue

Search and rescue comes in many different forms, some of which benefit from the use of UAS more than others.

Often, people go missing out in the wilderness, and their location may not be known to within many square miles. In this case, UAS show their utility especially well. Getting a bird's eye view of a large area allows for a much greater coverage of an area, and UAS do not have to deal with rough terrain where humans and ground vehicles struggle. The National Park Service alone responds to around 4500 search and rescue incidents costing around 50,000-100,000 personnel hours every year [47]. They often use helicopters to search for people due to terrain constraints and greater visual field of view. These cases in particular demonstrate the need for greater utilization of UAS in search and rescue. UAS can be deployed many at a time for much less money than helicopter flights.

Not all search and rescue efforts will benefit from UAS – for instance, the case of the boys soccer team that went missing in 2018 in a cave in Thailand [50]. Here, the group was 2.5 miles deep inside a cave system when it flooded with monsoon rain. They found refuge on high ground, but they were trapped behind several long spans of water that completely filled the exit path. Since UAS cannot fly through water, they would have been of no use in this case.

Search and rescue in caves, mines, and buildings are not out of the realm of UAS use however. For example, the DARPA SubT Challenge furthered the state of the art of UAS in such environments. The methods described in [45] show the amazing ability of sUAS to work in tight quarters to precisely locate objects including a heated manikin. Here, a combination of Lidar, RGB cameras, and IR cameras was used by a team of ground robots and sUAS to go deep into caves, mines, and buildings; such approaches can be used in real-world situations of search and rescue, especially where it may be dangerous to send in a team of humans, such as in case of gas leak, flood, cave-ins, active shooters, etc.

For the sake of this document, I will specifically refer to the case of a lost hiker in

Yosemite National Park. This is a large area, roughly the same size as Rhode Island, filled with cliffs and trees. These natural features mean that there are occlusions everywhere in the environment, and a drone that is searching for a person will benefit from low-altitude flight. There are over 11 deaths and 240 search and rescue missions that take place in this park every year. Not all of these situations will benefit from search UAS, but many will. There are no organizations that are legally allowed to fly UAS beyond visual line of sight in this area; even a police chief would need to call the superintendent of the Yosemite National Park before calling the Administrator of the FAA for immediate approval of such a flight in what is a time-sensitive emergency situation. This will be discussed more below.

1.3 Autonomous, camera-reliant sUAS

1.3.1 Full- and semi-autonomous systems

Autonomy comes in varying degrees. All robotic systems require human input at some point in their operation, but all robots have some level of their own technological decision. In the case of sUAS, they will almost always compute their controls themselves so that the operator does not need to command each motor individually. They generally have some kind of ground control station (GCS) and a pilot with a remote control who can send commands or missions to the UAS. The ideal level of autonomy versus human input depends on use case, and usually systems can accommodate a range of autonomy to satisfy various situations.

1.3.2 Small unmanned aircraft systems

Small unmanned aircraft systems (sUAS) come in many shapes and sizes. In this document, we will focus on two main types: multirotor and fixed-wing. Multirotor UAS have a number of vertical thrust propellers (usually 4, 6, or 8) and are highly maneuverable and agile. Fixed-wing UAS may have vertical thrusters for takeoff and landing, but will always cruise with at least one forward-facing propeller and usually take a shape similar to an airplane. Fixed-wing UAS are less maneuverable as they require constant forward movement to maintain lift, but they are much more energy

efficient and can therefore stay airborne for longer periods of time; in cases of search and rescue over large areas, fixed-wing UAS are to be preferred for this reason.

1.3.3 Types of cameras

There are many types of cameras that are used on UAS, but for this document we will highlight the distinction between RGB and IR cameras. RGB (red, green, blue) cameras are similar to the human eye and require an external light source such as sunlight in order to sense their environment. IR (infrared) cameras sense infrared radiation and do not require an external light source, but they have less clear resolution at long distances. They can be vital at night time search efforts, especially since the human body radiates infrared light, which is often clearly distinguishable against the terrain background in spite of the lower resolution. Both of these cameras are considered useful in case of search and rescue.

1.4 BVLOS and Regulation

1.4.1 What is BVLOS

Beyond Visual Line of Sight (BVLOS) operations are UAS flights where the pilot-in-command of the UAS cannot maintain visual contact with the UAS. This type of flight has certain challenges for safety and utility since the pilot-in-command cannot know the precise location or situation of the UAS. This also makes radio communication harder since radio communications benefit from, and often require, line-of-sight. The robotics methods described in this document illustrate a system designed to tackle these challenges.

1.4.2 Regulation

BVLOS operations of UAS are forbidden by the FAA except with special permission. One special permission is called a BVLOS waiver; this type of waiver only applies to one flight. Waivers can be granted to private applicants usually in 1-6 months or government applicants in 1-3 weeks if expedited. These waivers always require aircraft detection and avoidance systems, and they often require constant radio

communication as well as "spotters" who are people on the ground maintaining visual contact with the aircraft (other than the pilot-in-command). Another type of BVLOS permission is a BVLOS COA (Certificate of Authorization). A BVLOS COA grants authority for BVLOS flight over a specific area for use at any time. Only fewer than 20 police jurisdictions have a BVLOS COA, as well as a few state Departments of Transportation. The last type of permission we will talk about is the Special Governmental Interest (SGI). This type can be granted in just a matter of hours, and it does not require detection and avoidance systems, but they only apply within 1500 feet of the pilot-in-command. Other than these three permissions, BVLOS flight are generally banned by the FAA unless given explicit permission by the Administrator of the FAA.

In National Parks like Yosemite, drone flight is outright banned. For instance, Yosemite's compendium says, "Launching, landing, or operating an unmanned aircraft from or on lands and waters administered by the NPS within the boundaries of Yosemite National Park is prohibited." There is a process by which NPS can formally allow drone use "for search and rescue, resource management, and research" called a Special use Permit (SUP), but in practice, these permits are almost impossible to obtain. Grand Canyon National Park is one of the few cases of a National Park with a fleet of drones that the Park Service uses for search and rescue missions.

Chapter 2

Aircraft Detection and Avoidance

Much of this chapter is adapted from ViSafe paper [23].

2.1 Motivation

Search and rescue can require search over vast areas of land and sea. These types of environments require that the drone flies far away from the launch point, and therefore beyond the safety pilot who holds the remote control. This type of operation is called beyond visual line of sight, or BVLOS for short. BVLOS flight requires special considerations for safety and regulation. One of these is the ability to detect and avoid other aircraft. This ability ensures vehicle retention, data retention, and safety of aircraft and people on the ground. It also allows for the potential to have many drones flying in the same airspace without the need for constant communication with each other, which is technically challenging and can hinder search efforts. To mitigate this obstacle, we have developed ViSafe – a full system that uses cameras to detect and avoid airborne obstacles.

2.2 Introduction

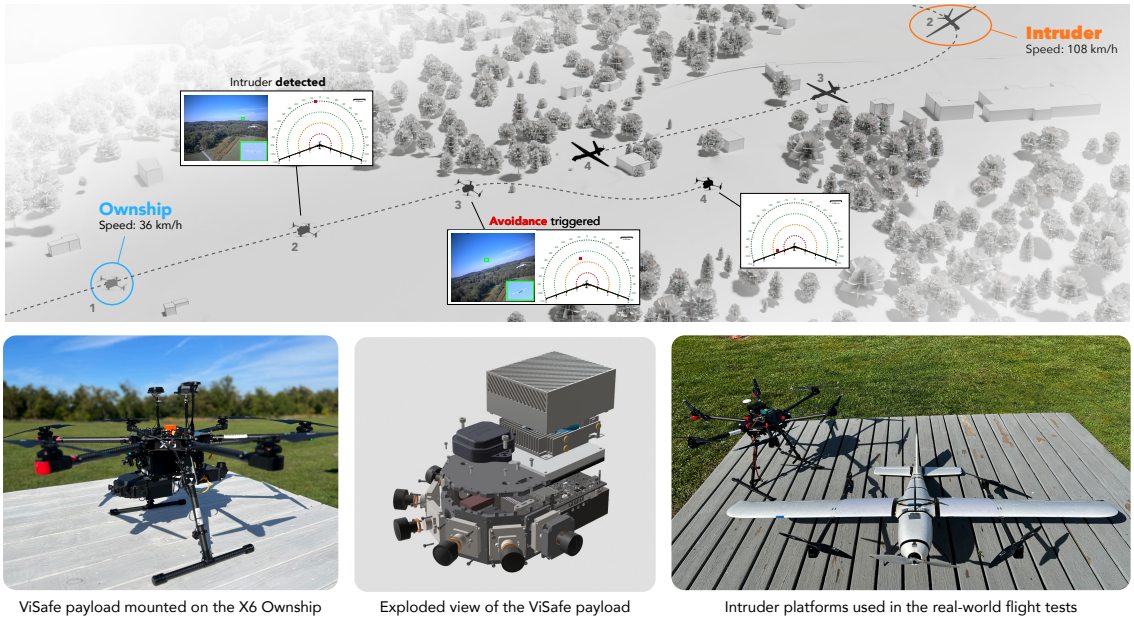

Figure 2.1: **We demonstrate *ViSafe*, a high-speed vision-only airborne collision avoidance system.** *Top:* Rendering of a real-world flight test log where the *ViSafe* system detects an incoming intruder with a 144 km/h closure rate and performs an avoidance maneuver to ensure safe separation. The annotations showcase the detections and multi-camera tracks from the vision-based aircraft detection and tracking system, while the trajectory shows the log of the performed real-world avoidance maneuver. The numbered annotations showcase the different stages of the flight test, from intruder detection to avoidance completion. *Bottom:* ViSafe payload, ownship, and intruder platforms used in the real-world flight tests.

In the USA national airspace, collision avoidance systems are critical to safe flight separation. Existing solutions, such as Autonomous Collision Avoidance Systems (ACAS) [30] and Unmanned Traffic Management (UTM) [16] frameworks work well for large manned aircraft, but they depend on ADS-B and radar systems, which are not practical for small UAS due to their size and weight. ADS-B is a system of transmitting and receiving GPS and pressure altitude data between aircraft, but this system is incomplete since not all aircraft are required to use it. There may always be aircraft (or wildlife) that have no reliable way to detect and avoid other aircraft other than the pilot's discretion. This is why UAS pilots are generally required to maintain line-of-sight with their aircraft.

With new advances in algorithms and computation, cameras offer a solution for UAS DAA. Cameras are small and lightweight, and mimic the most common and effective type of aircraft DAA – the human eye. Methods such as [15] demonstrate an effective low-error camera-based DAA method that can be mounted on a small UAS.

ViSafe is a vision-only airborne DAA system which extends AirTrack with multi-camera fusion for tracking as well as a Control Barrier Function (CBF) [7] as an avoidance algorithm, which provides computationally inexpensive provable guarantees of safety with certain assumptions about the other aircraft.

CBFs have been successfully applied in safe navigation [6, 17], robotic manipulation [36, 54], industrial automation systems [7], and research into aircraft in shared airspaces [14, 34, 39]. They are often used with global information availability, but *ViSafe* uses sensing with error bounds.

We show the advantages and challenges of CBF usage in real-world testing and in simulation with computational efficiency and effective safety margins.

We refine the AirTrack [15] inference system's uncertainty with a multi-view Kalman filter to create less error-prone, more reliable detections.

We propose a "digital twin" of our system with a simulation environment using Nvidia Isaac Sim [5]. We create a realistic model of our testing environment, render realistic camera streams in different weather conditions, use flight hardware for detection and avoidance computation, and conduct thousands of full flight tests in simulation, minimizing the sim-to-real gap for large-scale testing.

Finally, we deploy our system on real UAS in realistic situations. We conducted about 80 hours of flight testing across two test sites, and results are nearly identical to simulation experiments, lending credibility to our hardware-in-the-loop simulation realism.

The main contributions of this work are as follows:

1. Multi-view vision-only aircraft detection & tracking and CBF-based collision avoidance system that assumes no global availability of information or communication.
2. Custom-built SWaP-C hardware that simultaneously streams multiple camera inputs, provides state estimation, performs deep learning model edge inference, and computes avoidance maneuvers on board in real time.

3. Digital twin and hardware-in-the-loop simulation to perform DAA benchmarking and performance analysis under different agent types, closure rates, interaction geometries, and environmental conditions.
4. First-of-its-kind real-world flight tests demonstrating that *ViSafe* ensures safe aerial separation in encounter scenarios with a closure rate of up to 144 km/h.

2.3 Related Work

2.3.1 Collision Avoidance logics

Most air traffic collision avoidance systems are based on the Traffic Collision and Avoidance System (TCAS) [24]. This system uses Mode S ADS-B transponders, which must be installed on all participating aircraft. Different similar systems such as the ACAS family of avoidance algorithms have been developed which each use a different algorithm for different types of aircraft. These methods generally use large tables of different scenarios, making their calculations largely pre-computed, making the system very computationally efficient. When a near collision is anticipated, they find the corrective maneuver, called a Resolution Advisory (RA), and send this to the pilot or planning algorithm for suggested maneuvering. They are tested on large data sets and are deployed in the real world of airspace.
Since these methods generally assume the use of either radar sensors or position communication between agents, such as ADS-B, their use with camera sensors is relatively unexplored [38]. They generally are used with larger aircraft, but small UAS can be more maneuverable and may have many other UAS in the area, meaning cost tables would grow very large.
Our system uses camera inference detections, meaning we do not need heave radar sensors, nor do we rely on every aircraft we avoid to have an ADS-B transceiver. Our method has more fine-grained control for corrective actions, meaning we can create more efficient paths while maintaining safety.

2.3.2 Control Barrier Functions for Aerial Collision Avoidance

Recently, many CBFs have been used for UAS separation. Squires *et al.*uses CBFs for collision avoidance, but they assume constant availability of information and only test in simulation without sensor error. A follow-up work [51] uses a decentralized approach, but does not consider non-cooperative agents. Our proposed technique assumes all agents are non-cooperative and without communication.

2.3.3 Aircraft Detection and Tracking

Previous camera-based aircraft detection and tracking used computer vision techniques like frame stabilization, background-foreground separation, and flow-based object tracking [9, 11, 13, 27, 28, 32]. These modular approaches use optical flow or image registration for frame stabilization [31, 42, 46]. They use regression-based motion compensation and morphological operations to find optential intruders [43]. Background subtraction and SVMs were used to learn descriptors for aircraft identification [11, 41, 43]. Track-before-detect methods have been implemented using Hidden Markov Models, Kalman filters, and Viterbi-based filtering [26, 28, 33, 37]. Our system uses AirTrack [15], which uses a sequence of frame alignment, detection, tracking, and false positive filtering, motivated by previous approaches. Object detection has been done using convolutional neural networks (CNNs) [10, 19, 21, 52]. In aircraft detection, where objects can be small and far away, keypoint-based methods are generally more effective than anchor-based methods like YOLO or R-CNN [12]. The use of these methods has fueled the aggregation of aircraft video datasets [1, 40, 48, 49, 53]. CNNs that predict heatmaps have been shown to be effective for aircraft detection [21, 22]. Many effective approaches use algorithms like the Hungarian method to associate new detections with existing tracks [8, 11, 28]. Our detection algorithm AirTrack [15] employs an anchor-free detection and tracking system.

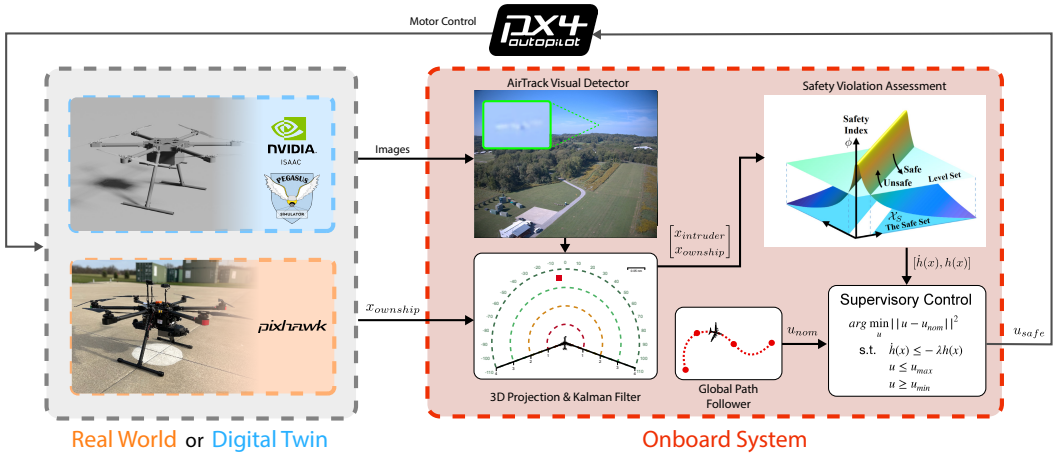

Figure 2.2: **Overview of our *ViSafe* framework for real-world testing and hardware-in-the-loop simulation:** Firstly, the onboard sensors or digital twin simulation stream the multi-cam videos to the AirTrack visual detection module, which detects the intruder across multiple views. Then, these detections, along with the ownship state information, are sent to the multi-view fusion and coordinate frame conversion module, which then tracks the intruder and sends the intruder state information along with the ownship state information to the CBF. The CBF uses the nominal global plan and the safety violation assessment to compute modifications to the nominal control input in case of violation. This safe control output is then sent to the drone autopilot system for execution. This loop continues to operate in real-time until the flight test is complete.

2.4 Problem Definition

We refer to our controlled UAS as the ownship agent and another nearby UAS as an intruder agent. We assume the intruder has no communication with us and is non-cooperative. Our goal is to maintain sufficient distance between these two UAS. We model the scenario in the horizontal plane and assume constant intruder velocity, inspired by [30].
The separation requirement can be formally defined as:

$$\forall t \in T_{operation} \bullet d_{ownship-intruder} \geq d_{thresh} \tag{2.1}$$

where d_{thresh} is the minimum distance threshold and $d_{ownship-intruder}$ is the relative horizontal distance between the agents. We will refer to this distance as d in our formulations. Additionally, $T_{operation}$ is the time of operation of the ownship agent, and eq. (2.1) formally states the problem of safe separation for the entire operation time.
As can be inferred from (2.1), this is equivalent to a forward invariance property. In order to encode and enforce this property, we make certain assumptions. First, we assume a simple unicycle dynamics model for our ownship agent:

$$\dot{x_{own}} = \begin{bmatrix} v_{own}\cos(\chi_{own}) \\ v_{own}\sin(\chi_{own}) \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} u \tag{2.2}$$

$$u^T = [\dot{v}_{own}\ \dot{\chi}_{own}] \tag{2.3}$$

Here, v_{own} denotes the speed of the ownship, and χ_{own} is the horizontal track angle. The control input $u \in \mathbb{R}^2$ consists of the rate of change of velocity, i.e., $\dot{v}_{\text{own}}$ and $\dot{\chi}_{\text{own}}$. We also consider control constraints to closely mimic real-world deployment scenarios.

$$u \geq u_{min}, u \leq u_{max}\ \forall t \in T_{operation}$$

Lastly, we define the dynamics of the intruder agent:

$$\dot{x_{int}} = \begin{bmatrix} v_{int} \cos(\chi_{int}) \\ v_{int} \sin(\chi_{int}) \end{bmatrix} \tag{2.4}$$

Here, v_{int} denotes the speed of the intruder, and χ_{int} is the horizontal track angle.

2.5 *ViSafe* Framework

An overview of our *ViSafe* framework (fig. 2.2) is as follows: The visual detection module (section 2.5.1) provides an image-frame intruder detection which is transformed to the North East Down (NED) coordinate system and broadcast to the avoidance module (section 2.5.2). The avoidance module (section 2.5.3) then makes modifications to nominal control if the minimum distance constraint will be violated.

2.5.1 Visual Detection Module

We modified the AirTrack [15] algorithm to incorporate multi-camera streams and multi-camera tracking. We also upgraded the hardware and implemented efficient image sharing with zero memory copies and optimized the deep learning model inference to operate at higher frequency. Our Detection Module is comprised of Frame Alignment, Detection, Secondary Classification, and Image-level Tracking. The module's inputs are two consecutive grayscale image frames, and its output is a list of aircraft tracks. The model is trained on the Amazon Airborne Object Tracking (AOT) dataset [1]. Its modules are as follows:

Frame Alignment: The two input video frames are cropped and downsampled to 1/32 size and input into ResNet-34 in order to predict the relative motion between the frames using optical flow. If the confidence is too low, the alignment prediction is rejected.

Detection: This module has two submodules: a primary detection module and a secondary detection module. The primary module takes in two full-resolution (1224×1024) grayscale image frames and their predicted relative alignment, while the secondary module takes in a $\approx 1/5$th sized crop of the image around the top k detector outputs (based on confidence) from the primary module. The network is fully-convolutional (HR-Net-W32), and it produces five outputs: a center heatmap

indicating the object's center, bounding box size, center offset from the grid to the object's center, track offset of the object's center from the prior frame, and object distance in log scale.

Secondary Classifier: The input for this module is a cropped image around the bounding box. ResNet-18 is used as a binary classifier for false-positive filtering, and if the object is not determined to be an aircraft, the detection is rejected

Image-level Tracking: We use an offset tracking vector-based approach [55]. The detection's relative motion is subtracted and its corrected position is compared to active detection tracks. If the position matches an active track within a time and location threshhold, the detection is associated with that track; otherise, a new track is initialized.

The sequential modules of AirTrack work together to provide precise image-frame aircraft tracks and distance from the ownship. Our system runs AirTrack on video streams from two cameras at 8 Hz on our Nvidia Orin AGX compute system (described in section 2.6.5). We encourage the readers to refer to the AirTrack [15] paper for further details.

2.5.2 Multi View Fusion & Coordinate Frame Conversion

An image-frame detection track is transformed into spherical coordinates in the ownship body frame (d, θ, ϕ) using the cameras' intrinsic and extrinsic matrices. This is then transformed into North-East-Down (NED) global Cartesian coordinates using the ownship pose, which comes from GPS and IMU.

We empirically observe that a detection's range (d) has more noise than its angles (θ & ϕ) since cameras are angular, not ranging, sensors. A single Kalman Filter could be used for tracking, but this could cause errors since the ownship coordinate frame is moving. Instead, we use two Kalman Filters, one for angles and one for range. Only the angular filter is used for multi-camera track correspondence. Active intruder tracks are propagated sent to the Avoidance Module at 24 Hz.

As shown in fig. 2.3, detection tracks are transformed into both Cartesian and spherical coordinates, and both are sent to the Avoidance Module.

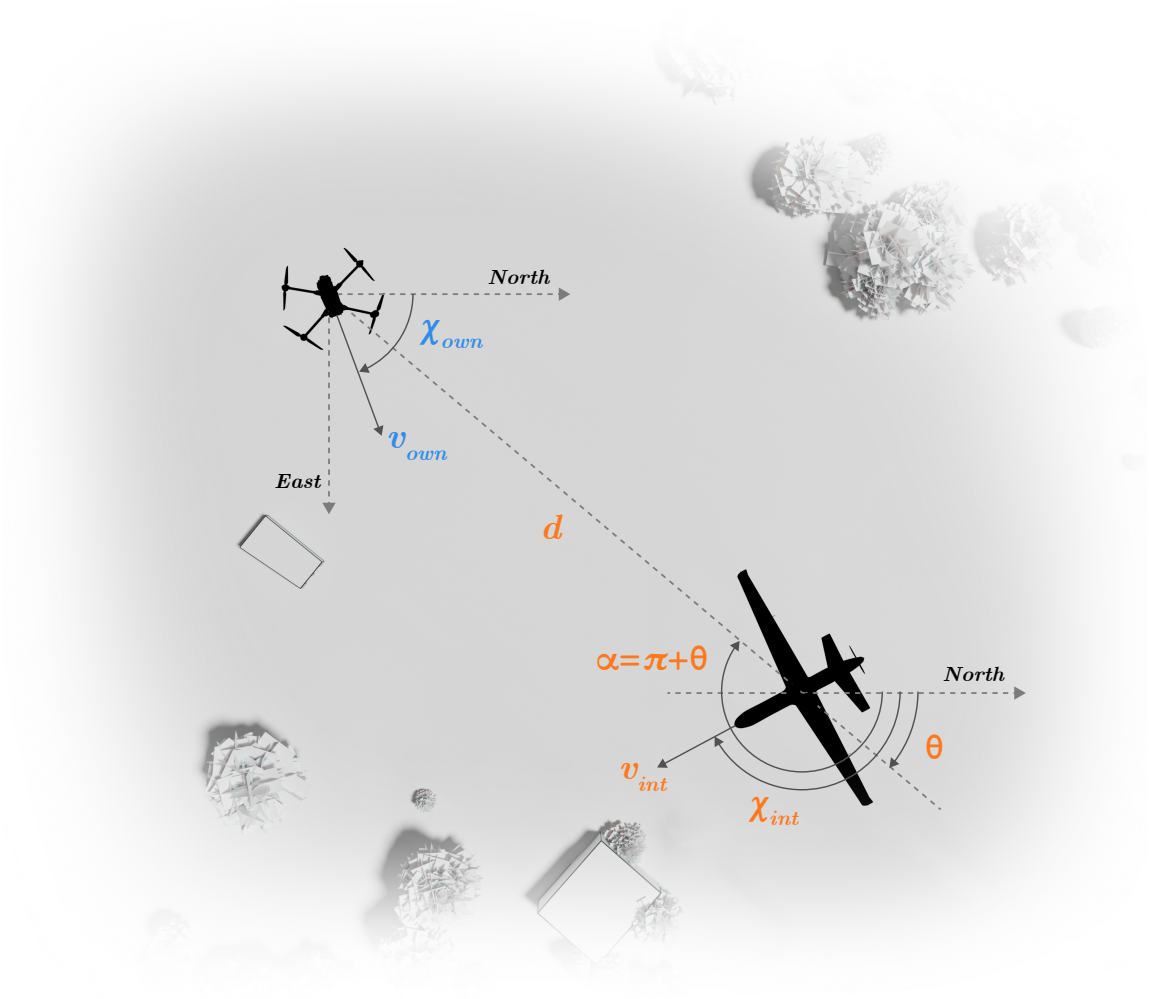

Figure 2.3: **Encounter geometry and information required for vision-enabled collision avoidance.** The velocity of the ownship v_{own} and heading with respect to North χ_{own} are obtained from the ownship odometry. The intruder's range d, azimuth θ, velocity v_{int}, and heading with respect to North χ_{int} are obtained using the visual detection module.

2.5.3 Supervisory Safety Controller

The three key requirements for this controller are:

- **Certified safety**: Enforcement of a strict notion of safety while working with vision inputs
- **Run time safety**: Reactive low latency solution that can make decisions in real-time
- **Performant safety**: Preserving original mission requirements and introducing a minimal modification to the original control input provided by a nominal controller

Considering these three key requirements, we used a Quadratic Program (QP) as our Control Barrier Function (CBF). CBFs were introduced in [7] to certify the safety of robotic control systems.
Inspired by [29], we propose the following CBF:

$$h(x) = c + d_{thresh}^{n} - d^{n} - k\dot{d} \tag{2.5}$$

where $c > 0, n > 0$ and $k > 0$ are hyperparameters to be tuned.
Our defined CBF takes negative values when the agents are safely separated and positive values when the requirement is violated. The CBF condition we enforce for forward invariance is this:

$$\dot{h}(x) \leq -\lambda h(x)$$

Our final QP formulation is defined as follows:

$$\begin{aligned} arg\min_{u} \quad & ||u - u_{nom}||^2 \\ \text{s.t.} \quad & \dot{h}(x) \leq -\lambda h(x) \\ & u \leq u_{max} \\ & u \geq u_{min} \end{aligned} \tag{2.6}$$

where u_{nom} is the control input provided by the nominal controller and u_{max} and u_{min} encode the actuation constraints.
First, we derive the values of $\dot{h}$ in terms of our control input u using our encounter

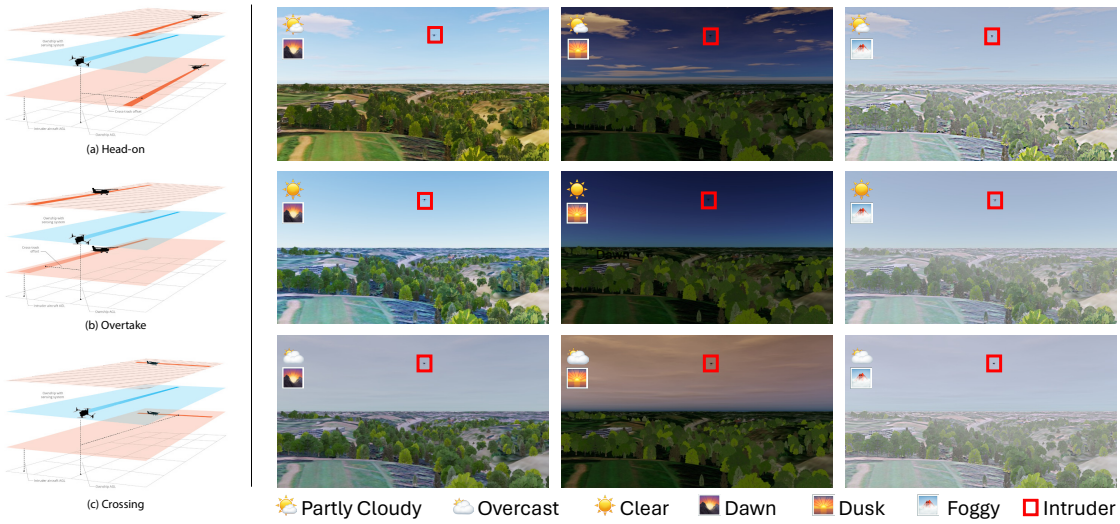

Figure 2.4: **Diversity of the airborne collision testing scenarios:** (a) The various encounter geometries used for real-world & simulation flight testing. (b) The diverse weather and lighting conditions that were used to evaluate *ViSafe*'s robustness in simulation. In the simulation, based on the chosen encounter geometry, the intruder position is sampled randomly on the flying corridor.

geometry as illustrated in fig. 2.3.

$$\dot{h}(x) = nd^{n-1}\dot{d} - k\ddot{d} \tag{2.7}$$

where

$$\dot{d} = v_{own}\cos\left(\alpha - \chi_{own}\right) + v_{int}\cos\left(\alpha - \chi_{int}\right) \tag{2.8}$$

We compute $\dot{d}$ by taking the projection of v_{own} and v_{int} along the line of sight (LOS). To compute $\ddot{d}$, we differentiate eq. (2.8) with our original assumption of constant intruder velocity vector (i.e. $\dot{v}_{int} = 0$ and $\dot{\chi}_{int} = 0$) to get:

$$\begin{aligned}\ddot{d} = \dot{v}_{own}\cos\left(\alpha - \chi_{own}\right) + \dot{\chi}_{own}v_{own}\sin\left(\alpha - \chi_{own}\right) \\ + \frac{\left(v_{own}\sin\left(\alpha - \chi_{own}\right) + v_{int}\sin\left(\alpha - \chi_{int}\right)\right)^2}{d}\end{aligned} \tag{2.9}$$

Table 2.1: Experimental Configurations

Scenario ID	Intruder	Ownship	Collision Geometry	Hor. Closure Rate	Location
E1	Multirotor (10 m/s)	Aurelia X6 (10 m/s)	Head-On	20 m/s (72 km/hr)	Nardo Airfield
E2	Multirotor (5 m/s)	Aurelia X6 (10 m/s)	Overtake	5 m/s (18 km/hr)	Nardo Airfield
E3	Multirotor (5 m/s)	Aurelia X6 (10 m/s)	Crossing	11.18 m/s (40 km/hr)	Nardo Airfield
E4	VTOL (30 m/s)	Aurelia X6 (10 m/s)	Head-On	40 m/s (144 km/hr)	Nardo Airfield
E5	Multirotor (10 m/s)	Aurelia X6 (10 m/s)	Head-On	20 m/s (72 km/hr)	Leesburg, VA

Table 2.2: Digital Twin & Hardware-in-the-Loop Benchmarking

	Separation Minima (m) ↑		P(NMAC) ↓		Risk Ratio ↓		Number of violations ↓	
Scenario	Nominal	*ViSafe*	Nominal	*ViSafe*	Nominal	*ViSafe*	Nominal	*ViSafe*
E1	19.9 ± 2.15	35.55 ± 14.00	1.0	0.55	1.0	0.55	4000	2213
1-9 E2	22.72 ± 2.63	27.635 ± 6.27	1.0	0.439	1.0	0.439	4000	1759
1-9 E3	49.14 ± 0.7	59.04 ± 11.37	1.0	0.5248	1.0	0.5248	4000	2099
Above Horizon - E1, E2, E3	30.58 ± 3.02	50.90 ± 4.72	1.0	0.1	1.0	0.1	6000	605
1-9 Below Horizon - E1, E2, E3	28.6 ± 2.73	30.46 ± 3.26	1.0	0.91	1.0	0.91	6000	5466

Now given $u = \begin{bmatrix} \dot{v}_{own} \\ \dot{\chi}_{own} \end{bmatrix}$, we can rewrite eq. (2.9) as

$$\ddot{d} = \begin{bmatrix} \cos(\alpha - \chi_{own}) \\ v_{own} \sin(\alpha - \chi_{own}) \end{bmatrix}^{\top} u + \frac{(v_{own} \sin(\alpha - \chi_{own}) + v_{int} \sin(\alpha - \chi_{int}))^2}{d} \tag{2.10}$$

As can be observed, the constraints on h are affine in u and fit the form of a QP, as defined in eq. (2.6), which can be solved to compute the desired safe control u_{safe}. We use the following tuned hyperparameters for the final formulation of eq. (2.5) and 2.6: k = 0.2, c = 0.01, n = 0.3, and $\lambda = 0.2$.

We use a simple PD controller as our nominal controller, where the computed desired safe control u_{safe} is then converted into low-level drone control actions in the form of velocity and yaw setpoints, which are executed on our ownship agent by the flight controller (PX4 in Isaac Sim and Ardupilot on the real-world drone hardware).

Table 2.3: Real world benchmarking

Scenario	Separation Minima (m) ↑		P(NMAC) ↓		Risk Ratio ↓		Number of violations ↓	
	Nominal	*ViSafe*	Nominal	*ViSafe*	Nominal	*ViSafe*	Nominal	*ViSafe*
E1	41.28 ± 0.75	51.58 ± 12.76	1.0	0.33	1.0	0.5	4	2
1-9 E2	44.25 ± 0.64	56.22 ± 2.88	1.0	0.25	1.0	0.25	4	1
1-9 E3	47.45 ± 0.22	71.69 ± 11.65	1.0	0.5	1.0	0.25	4	1
1-9 E4	30.54 ± 2.07	46.05 ± 8.29	1.0	0.0	1.0	0.0	2	0
1-9 E5	27.39 ± 5.03	38.94 ± 7.83	1.0	0.0	1.0	0.0	3	0
Above Horizon - E1, E2, E3	44.43 ± 3.02	59.43 ± 4.72	1.0	0.0	1.0	0.0	6	0
1-9 Below Horizon - E1, E2, E3	44.59 ± 2.73	60.23 ± 18.76	1.0	0.667	1.0	0.667	6	4

2.6 *ViSafe* Testing

2.6.1 Experiment Design

We consider single-intruder scenarios. We start by identifying three scenarios for our two-agent setup: Head-on, Overtake, and Lateral. A Head-On scenario has the intruder fly parallel to the ownship in the opposite direction. An Overtake scenario has the intruder fly parallel to the ownship in the same direction at a lower speed. A Lateral scenario has the intruder cross in front of the ownship perpendicular to the ownship's path. To test cases of detection both above and below the horizon (relevant for visual detection), each of the three scenarios listed is tested with the intruder both slightly above and slightly below the ownship altitude, leading to 6 total scenarios, pictured in fig. 2.4, (which also shows the different weather conditions for simulation tests). These six scenarios were performed in both a high-fidelity digital-twin simulation and real-world settings. Table 2.1 shows the various agents, collision geometries, commanded ground speeds, and test locations.

2.6.2 *ViSafe* Hardware Prototype

ViSafe uses custom hardware as shown in fig. 2.1. The hardware has the following major components:

1. Cameras: 6 x Sony IMX264 cameras providing a total of 220° FOV horizontally and 48° vertically.
2. Compute: NVIDIA AGX Orin Developer Kit

Table 2.4: ViSafe System Profiling

Component	Performance
AirTrack Detection & Tracking (GPU)	8 Hz
Multi-View Fusion & State Estimation (CPU)	24 Hz
CBF-based Avoidance Control (CPU)	50 Hz (solving a QP)
Mean CPU Utilization	55.21 %
Mean GPU Utilization	50.02 %
Peak Memory Consumption	15.96 GB

3. Camera Adaptor: Leopard Imaging LI-JXAV-MIPI-ADPT-6CAM-FP
4. State Estimation: 3DM-GQ7 GNS/INSS module with a dual-GPS antenna setup.
5. ADS-B In: uAvionix PING™ MAVLink based ADS-B receiver (not used)

The payload is mounted below the ownship on a custom mounting plate. The entire payload weighs approx. 1.6 kgs. The detection and avoidance algorithms described above run onboard the payload in real-time. The payload communicates with the ownship autopilot via MavLink in the form of velocity commands using DroneKit [2]. The payload mount and command communication are compatible with a wide array of different UAS.

2.6.3 Reaction Time Profile for Successful Detect & Avoid

Using real-world data from AirTrack [15], the detection algorithm can detect a Bell 407 helicopter (frontal diagonal length of 4.27 meters) with 95% probability of track with about 14 pixels. This means that we can detect with confidence at a distance of about 163.2 times the frontal diagonal length of an aircraft. For our tested drones, this equates to about 287 meters for our M600 intruder and about 525 meters for our fixed-wing VTOL intruder. This gives about 13-14 seconds of reaction time for each, which is quick enough that we need a fast, accurate, decisive maneuver from avoidance, but this is more than enough for our Avoidance system to handle; this statement is proven in both real world and simulation as shown in table 2.2, 2.3, and table 2.4.

2.6.4 Simulated Digital-Twin Experiment Setup

Simulation provides a low-cost, low-time method to test the robustness of our approach. In order to be useful, the simulation must be as realistic as possible. We use the Pegasus Simulator [20], which is a framework built on top of NVIDIA Omniverse and IsaacSim [4]. We use ArcGIS CityEngine [3], which takes satellite images, terrain maps, roads map, and structure information from OpenStreetMap to build a 3D model USD file of our real-world test site. We also added trees and a realistic sky with clouds and weather conditions as depicted in Fig. 2.4. The experiments were carried out on a desktop computer with an Nvidia 3080Ti and an Nvidia A6000 running Ubuntu 20.04.

To further test our system, our real payload is connected to the simulation computer over ethernet; video streams are rendered and sent to the payload which handles the streams as it would its physical camera streams. The payload's Avoidance commands are then sent back to the simulation over MavLink to be carried out by the simulated UAS.

We simulated 10 different weather conditions, including time-of-day and cloud covers, three representative collision configurations, and two altitude setups (Above/Below horizon), leading to a total of 60 distinct scenarios. For the above scenarios, we simulated 200 tests per scenario, leading to a total of 12000 tests.

2.6.5 Real World Experiment Setup

Our system was tested at two different test sites – the Nardo Flight Test Field [18] (Valentin Vassilev Memorial Field), USA `[FAA Identifier: 77PA]`, `[40-35-00.2420N 79-53-59.1890W]` as well as a secondary test-site in Leesburg, VA, USA. All tests are performed under the FAA 14 CFR Part 107. The Nardo flight test airfield has a grass runway with a total length of 1800ft (550m) and a width of 100ft (30.5m). The Leesburg field is an open test range. We performed test configurations shown in table 2.1. The field experiments use three airborne platforms, as shown in Fig. 2.1. Their details are as follows:

1. Aurelia X6 Standard: A hexa-rotor ownship.
2. Hexarotor: A multi-rotor low-speed intruder.

3. VTOL: A quad-plane vertical takeoff and landing fixed-wing aircraft that acts as our high-speed intruder.

2.6.6 Metrics

We compare our technique with a nominal planner without an avoidance strategy. By default, the nominal plans violated safe separation; we include it as a reference point to quantify ViSafe's performance via metrics. Our metrics are inspired by the operational standards proposed for collision avoidance systems by the Radio Technical Commission for Aeronautics (RTCA) [44]. These are the key metrics which we use:

1. *Probability of Near Mid Air Collision P(NMAC)*: This metric measures the probability that two agents come within a predefined unsafe distance. Lower values signify better collision avoidance.
2. *Separation Minima*: This metric captures the minimum distance between agents during an episode. Higher values indicate greater safety margins.
3. *Horizontal Rate of Closure (HROC)*: This metric measures the relative velocity at which two agents approach each other, with negative values indicating impending collisions. Higher values reduce collision risks.
4. *Risk Ratio*: This metric measures the ratio of collisions compared to a baseline (nominal planner). Lower values indicates higher safety improvements over the baseline.

2.6.7 Results and Discussion

In this section, we highlight the key findings of comparing *ViSafe* with a nominal planner without an explicit avoidance strategy. The results for simulated digital-twin and real-world experiments are summarized in Tables 2.2 and 2.3.

***ViSafe* enables high-speed collision avoidance [H1]**

Our results show that our system is effective at reducing collision risk and increasing separation distances over the nominal plan. The total number of losses of separation is significantly reduced by our system for all real-world tests.

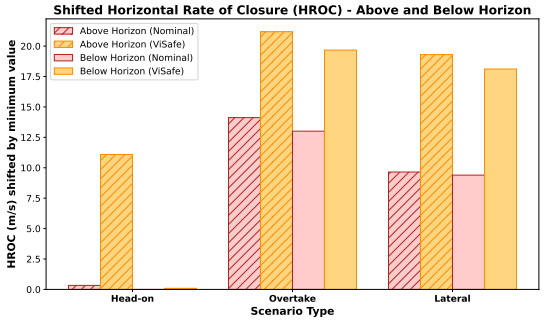

Figure 2.5: **Average horizontal rate of closure comparisons across different encounter geometries in real-world testing:** Higher values indicate that agents are moving apart, showcasing diverging & safe trajectories. Under different testing scenarios, it can be seen that *ViSafe* consistently shows a significant boost in average HROC over the nominal plan.

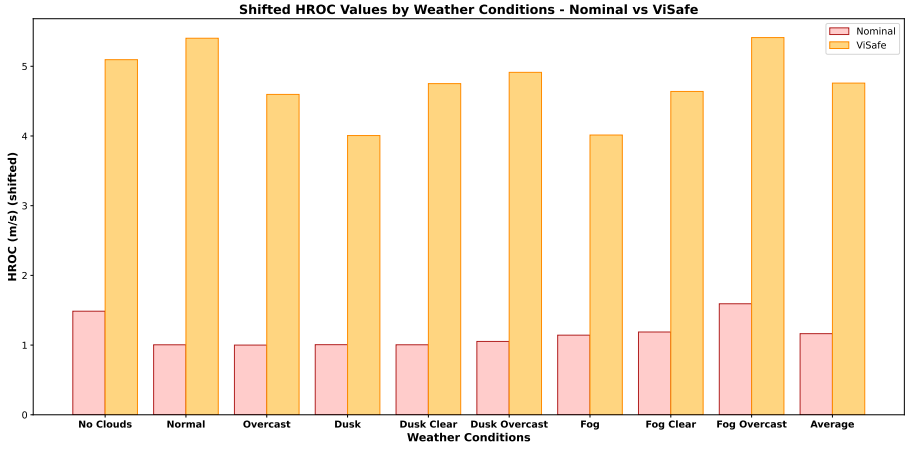

Figure 2.6: **Horizontal rate of closure comparisons across different weather conditions in the digital twin:** Higher values indicate that agents are moving apart, showcasing diverging & safe trajectories. Across the different weather scenarios, *ViSafe* showcases consistent boost in HROC over the nominal plan.

The results in fig. 2.5 indicate that *ViSafe* improves the HROC values compared to the nominal planner. In the head-on scenario, the HROC improved from $-19.25\,\mathrm{m/s}$ (nominal) to $-13.83\,\mathrm{m/s}$ (*ViSafe*). Similar trends are observed in lateral and overtaking cases, where *ViSafe* maintains a higher and safer HROC, effectively mitigating dangerous approaches.

ViSafe is robust to weather and lightning conditions [H2]

We also test the robustness of *ViSafe* to weather and lighting conditions in simulation. The results are highlighted in 2.2. Our risk ratios increase due to detection and tracking failures for some weather conditions; however, our overall risk ratios with respect to all trajectories are still considerably lower than the nominal values across all configurations. *ViSafe* also achieves higher HROC values than the nominal plan for all weather conditions as shown in fig. 2.6.

Digital-twin provides a reliable estimate of real-world performance [H3]

Our results indicate that the performance metrics observed in the digital-twin simulation environment closely align with those obtained in real-world testing. As summarized in Tables 2.2 and 2.3, *ViSafe* demonstrates higher separation minima and lower risk ratios in all configurations.

2.7 Conclusion

This work represents a first of its kind, state of the art approach to UAS collision avoidance using cameras only. This is an important feature of a search and rescue UAS system, since in most cases, they will be flying in airspace that is subject to the jurisdiction of the FAA and shared with other aircraft and/or wildlife. Ensuring that a drone is safe, especially in BVLOS flight, is vital to ensure a successful search effort and data gathering. Using cameras lets us save on money and weight, since cameras are generally smaller and less expensive than other options such as radar sensors. This allows for deployment of multiple UAS in the same area, even if the UAS and pilots cannot be in constant communication with each other, vastly increasing search efficiency.

Chapter 3

Localization

3.1 Motivation

UAS need to know where they are when performing search and rescue for multiple reasons: they must know where they are going to make sure they are searching in the right area, they need to know where they are in order to know where their target is once found, and they need to know how to get back to the launch point in order to relay the data to the search team.

3.2 Methods

3.2.1 GPS and magnetometer

GPS is the Global Positioning System that most of us use everyday. It listens to a system of satellites that transmit time and location data to the robot. The robot's GPS receiver must maintain line-of-sight with at least four satellites in order to accurately localize itself.

A magnetometer augments a GPS by sensing the Earth's magnetic field. GPS gives the robot its location, and an IMU can tell the robot which way is down, but with only these two sensors, the robot will not know which way it is facing in the horizontal plane. Since the magnetic field always points towards magnetic North, the robot can use this sensor in order to orient its yaw horizontally, just like an orienteer would use

a compass.

3.2.2 Camera localization

Sometimes GPS cannot be relied upon in search and rescue applications. This includes cases of flying in severe weather, flying low near mountainous terrain, flying low near buildings, electromagnetic interference, high levels of iron in the ground, or even hostile GPS-jammers in search and rescue near warzones. These methods can be solved by using cameras to do localization. FoundLoc [18] is a technology developed at Carnegie Mellon University that uses exclusively a downward-facing camera, an inertial measurement unit (IMU), and preloaded satellite images of terrain in order to accurately, absolutely localize a UAS. These methods are impervious to the drawback of GPS such as jamming attacks, multipath interference, electromagnetic interference, and lack of satellite line-of-sight. In the case of the lost hiker in Yosemite National Park, this is especially relevant since we want to be flying at low altitude in mountainous terrain, where GPS can be inaccurate and unreliable.
Another benefit of this method that is specific to search and rescue is that a very small error in UAS localization, magnetometer yaw, camera intrinsics, etc. can equate to a large error in distance on the ground when a missing person is spotted with the camera. The ability to directly map the camera feed with actual satellite images allows for absolute precision of missing person localization since they can be localized against the ground terrain itself rather than the camera-relative frame coordinate transformation.

3.3 Conclusion

Search and rescue occurs in many different types of environments and scenarios. Robust localization is vital to the search process, to know where to search, to know where a person is located, and to know how to return that data to the launch point. A combination of GPS/magnetometer and camera-based localization like the one discussed here can ensure with high certainty that UAS localization will not be lost. GPS will usually be the best option for UAS localization, but in cases of search and rescue, robustness can be the difference between life and death. Camera-based

methods allow for less expensive solutions that can greatly improve the certainty of robustness of localization. These methods are especially important in search and rescue applications, since a small error in GPS or magnetometer yaw can equate to a great distance of error on the ground. By locating a missing person with respect to images of the ground itself, the result can be much more precise.

Chapter 4

Search Planning

4.1 Introduction

On long flights over a large search area, the UAS must be able to fly autonomously since often constant communication links cannot be guaranteed. A ground control station (GCS) is used to create a flight plan for the UAS. This generally involves a polygonal boundary for total potential search area as well as Gaussian distributions of most likely areas of finding the missing person, such as a last known position. The UAS is then launched; it generates a trajectory that maximized information gain given its limited flight time budget – this formulation is an example of an Informative Path Planning (IPP) problem. The IPP algorithm is aware of the field of view (FOV) and range of its sensor (camera in this case), the total flight time (called budget), and a map of the area of interest with each point on the map given a likelihood of information gain (in this case likelihood of finding missing person). Below in the Methods section of this Chapter is detailed one specific approach designed by researchers at Carnegie Mellon University led by Brady Moon called TIGRIS.
In some scenarios, the search area may be very large and the search and rescue team may have very little information about the whereabouts of the missing person. A team may just want to make sure they check every spot in a large area, and they are willing to wait for the UAS to look over the whole area. These types of cases lend themselves better a different framing of the problem. Coverage Path Planning (CPP) is another problem formulation in the field of path planning which seeks to gather

sensor data for every point on a map rather than focusing on key locations. Below is also a brief summary of "Complete, Decomposition-Free Coverage Path Planning" by Tushar Kusnur and Maxim Kikhachev from Carnegie Mellon University.

4.2 Methods

TIGRIS [35] is a sampling-based approach to the IPP problem. It involves random sampling of the state space over the total search area for different camera positions, weighting them based on their predicted information gain, and then performing a tree search on those positions in order to form a full trajectory. It does this by starting at the entry point or root node, finding nearby nodes, computing the path between nodes, calculating the predicted information gain and flight time at the new node, and repeating this process at the new node until the flight time budget is exhausted. Once the allotted planning time is reached, the node with the highest total information gain is returned, along with its parent nodes, representing a complete trajectory. With unlimited computation and time, it is guaranteed to converge to an optimal trajectory for information gain based on the probability map it is given. This method is especially good for camera-based UAS since it deals well with high-dimensional spaces, such as a camera sensor on a UAS that can point in any direction.
Kusnur and Likhachev demonstrated an efficient method of covering an arbitrary area fully in [25]. Often, CPP approaches decompose an irregularly-shaped area into subdivisions that can each be more easily covered with a predefined pattern, use a Traveling Salesman Problem (TSP) solver to determine paths between subdivisions, then filling each region with a simple space-filling path. The approach referenced here instead couples the subdividing and TSP steps into one step. This makes the algorithm more general to irregularly-shaped areas and more computationally simple. The decomposition-free solution here can solve CPP problems 2-10 times faster than some state-of-the-art decomposition-based methods.

4.3 Discussion

In many search and rescue situations, one must search over a large area with a small amount of information about potential locations of interest. The area is generally constrained, and some areas are more likely than others. When using UAS, there are constraints on flight time, and even in line-of-sight operations, autonomy is to be favored since it is impossible for a human to optimally search a space when needing to factor in sensor limits, flight time, speed, etc. Humans are much better at filling out a map with areas of more and less likely locations where missing people may be found. The planning can then use this information as an input and efficiently find the best path given this information from the search team. TIGRIS is a state-of-the-art option for the IPP problem, and its optimality and efficiency lend itself to be a great fit for search and rescue applications.

For the case of a lost hiker in Yosemite, it would be beneficial to get views from multiple angles since rocks and trees provide occlusions that the camera cannot see through. Future work on TIGRIS in this vein includes increasing the dimensionality of the map from a 2D grid to a 4D grid. The first two dimensions will still be the horizontal plane, but the third and fourth dimensions are the camera's azimuth and elevation angles relative to the point on the ground. This simple change would make the planner occlusion-aware, encouraging it to view points on the ground from all angles, meaning any occlusions will be seen past if possible.

The CPP solver described above is specifically designed for indoor environments, but in a place like Yosemite with large cliffs and in SaR where we want to fly at low altitude, the large amount of obstacles creates a very irregularly-shaped search region that would benefit from this indoor-focused approach. In the case where a SaR team wants to have complete coverage over a large area, this approach would likely fare better than the IPP solver described above.

Chapter 5

Conclusion

5.1 Future Work

The bulk of work in UAS search and rescue applications is really just search as far as UAS are concerned. Small UAS are great at flying cameras around at high speed over large distances, which is an indispensable tool, but they lack the size and power to actually rescue people (although they can carry supplies). Rescue is usually reserved for people on the ground, but work is being done to improve the ability of robots to participate in rescue efforts specifically. One such effort is the upcoming GoAero challenge. This is an upcoming robotics competition focused on the transportation of an injured human. A team at Carnegie Mellon University is going to participate in this challenge, with plans to create a car-sized multirotor autonomous aircraft. Such an aircraft may be the most maneuverable vehicle for casualty transportation ever devised, but we will have to wait to see what happens.

Another glaring omission from this document is the lack of any object detection of the objects on the ground. I believe that this may help the system be able to reason about its planning decisions while airborne if used wisely, but I really believe that the focus should be on recording all the camera data and getting it in front of the eyes of human beings. The information gain prediction of the search and rescue teams should be trusted above the reasoning of the drone while airborne even though the drone has newer information. If humans on the ground want more video of a given area, they can command the drone to that location once communication is regained

or for a future flight.

5.2 Discussion

Search and rescue is a life-saving pursuit that we all have the potential to benefit from; any person can be caught in an emergency situation where they need to be rescued. It is not easy work, and those who join search and rescue teams deserve to have the best tools at their disposal, as do the missing people. Small UAS have the potential to save a great deal of time and money and potentially lives in this field. Locating missing people is a time-sensitive task with lives on the line, and there is simply no system more fit for the job. UAS can be inexpensively launched in five minutes to cover hundreds of square miles. Every search and rescue team should dedicate the meager money needed to add this tool into their toolbox.

Regulation is a large obstacle in this area, and both the FAA and the National Parks Service need to cut out exceptions for search and rescue missions when lives are on the line. The technology is already at the point where such flights can be carried out safely. In defense of regulation, it is impossible to create laws that account for all edge cases. I am no fan of drones flying around in my daily life, and I am not advocating for delivery drones everywhere, but there are clear cut cases where lives are in danger and BVLOS UAS could make a big difference. Search and rescue teams sometimes put their own lives in danger, and these regulatory bodies owe it to the brave men and women who search and rescue to allow the use of UAS.

That said, red tape is no reason to give up on finding people in danger. Frankly, I would encourage pilots to use their good judgment and put the safety of their fellow man first; only second should they consider the details of going by the book and the fear of retribution from law enforcement. Such decisions should not be taken lightly, but neither should the utility of UAS and the potential good they can do. I hope that the rules change for these emergency edge cases, but rules change slowly sometimes. I hesitate to state the following because it may encourage doofuses to get into deep trouble or impede official search and rescue operations or crash a drone that starts a fire and/or hurts people, but I'll say it anyway. If in your good judgment, you believe that flying a drone may help save a life and the safety risks are low, but the law forbids it, break the law.

Bibliography

[1] Airborne object tracking dataset. URL https://registry.opendata.aws/airborne-object-tracking. 2.3.3, 2.5.1

[2] Dronekit-python: Developer sdk for python to control drones using the mavlink protocol. URL https://github.com/dronekit/dronekit-python. 2.6.2

[3] Procedural City Generator — 3D City Maker — ArcGIS CityEngine — esri.com. https://www.esri.com/en-us/arcgis/products/arcgis-cityengine/overview. [Accessed 11-12-2024]. 2.6.4

[4] Isaac Sim — developer.nvidia.com. https://developer.nvidia.com/isaac/sim. [Accessed 11-12-2024]. 2.6.4

[5] Nvidia isaac sim, 2022. URL https://developer.nvidia.com/isaac-sim. 2.1

[6] Ayush Agrawal and Koushil Sreenath. Discrete control barrier functions for safety-critical control of discrete systems with application to bipedal robot navigation. In *Robotics: Science and Systems*, 2017. URL https://api.semanticscholar.org/CorpusID:1780280. 2.1

[7] Aaron D. Ames, Samuel Coogan, Magnus Egerstedt, Gennaro Notomista, Koushil Sreenath, and Paulo Tabuada. Control barrier functions: Theory and applications. In *2019 18th European Control Conference (ECC)*, pages 3420–3431, 2019. doi: 10.23919/ECC.2019.8796030. 2.1, 2.5.3

[8] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *2016 IEEE international conference on image processing (ICIP)*, pages 3464–3468. IEEE, 2016. 2.3.3

[9] Ryan Carnie, Rodney Walker, and Peter Corke. Image processing algorithms for uav" sense and avoid". In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 2848–2853. IEEE, 2006. 2.3.3

[10] Xueyun Chen, Shiming Xiang, Cheng-Lin Liu, and Chun-Hong Pan. Aircraft detection by deep convolutional neural networks. *IPSJ Transactions on Computer Vision and Applications*, 7:10–17, 2014. 2.3.3

[11] Debadeepta Dey, Christopher Geyer, Sanjiv Singh, and Matt Digioia. Passive, long-range detection of aircraft: towards a field deployable sense and avoid system. In *Field and Service Robotics*, pages 113–123. Springer, 2010. 2.3.3

[12] Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian. Centernet: Keypoint triplets for object detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6569–6578, 2019. 2.3.3

[13] Giancarmine Fasano, Domenico Accardo, Anna Elena Tirri, Antonio Moccia, and Ettore De Lellis. Morphological filtering and target tracking for vision-based uas sense and avoid. In *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 430–440. IEEE, 2014. 2.3.3

[14] Junjie Fu, Guanghui Wen, and Jinde Cao. High-order control barrier function based collision avoidance formation tracking of constrained fixed-wing aircraft. In *2023 5th International Conference on Industrial Artificial Intelligence (IAI)*, pages 1–6, 2023. doi: 10.1109/IAI59504.2023.10327509. 2.1

[15] Sourish Ghosh, Jay Patrikar, Brady Moon, Milad Moghassem Hamidi, and Sebastian Scherer. Airtrack: Onboard deep learning framework for long-range aircraft detection and tracking. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1277–1283, 2023. doi: 10.1109/ICRA48891.2023.10160627. 2.1, 2.3.3, 2.5.1, 2.6.3

[16] Asma Hamissi, Amine Dhraief, and Layth Sliman. A comprehensive survey on conflict detection and resolution in unmanned aircraft system traffic management. *IEEE Transactions on Intelligent Transportation Systems*, 2024. 2.1

[17] Marvin Harms, Mihir Kulkarni, Nikhil Khedekar, Martin Jacquet, and Kostas Alexis. Neural control barrier functions for safe navigation, 2024. URL https://arxiv.org/abs/2407.19907. 2.1

[18] Yao He, Ivan Cisneros, Nikhil Keetha, Jay Patrikar, Zelin Ye, Ian Higgins, Yaoyu Hu, Parv Kapoor, and Sebastian Scherer. Foundloc: Vision-based onboard aerial localization in the wild. *arXiv preprint arXiv:2310.16299*, 2023. 2.6.5, 3.2.2

[19] Sunyou Hwang, Jaehyun Lee, Heemin Shin, Sungwook Cho, and David Hyunchul Shim. Aircraft detection using deep convolutional neural network in small unmanned aircraft systems. In *2018 AIAA Information Systems-AIAA Infotech@ Aerospace*, page 2137, 2018. 2.3.3

[20] Marcelo Jacinto, João Pinto, Jay Patrikar, John Keller, Rita Cunha, Sebastian Scherer, and António Pascoal. Pegasus simulator: An isaac sim framework for multiple aerial vehicles simulation. In *2024 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 917–922, 2024. doi: 10.1109/ICUAS60882.2024.10556959. 2.6.4

[21] Jasmin James, Jason J Ford, and Timothy L Molloy. Learning to detect aircraft for long-range vision-based sense-and-avoid systems. *IEEE Robotics and Automation Letters*, 3(4):4383–4390, 2018. 2.3.3

[22] Jasmin James, Jason J Ford, and Timothy L Molloy. Below horizon aircraft detection using deep learning for vision-based sense and avoid. In *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 965–970. IEEE, 2019. 2.3.3

[23] Parv Kapoor, Ian Higgins, Nikhil Keetha, Jay Patrikar, Brady Moon, Zelin Ye, Yao He, Ivan Cisneros, Yaoyu Hu, Changliu Liu, Eunsuk Kang, and Sebastian Scherer. Demonstrating visafe: Vision-enabled safety for high-speed detect and avoid, 2025. URL https://arxiv.org/abs/2505.03694. 2

[24] JE Kuchar and Ann C Drumm. The traffic alert and collision avoidance system. *Lincoln laboratory journal*, 16(2):277, 2007. 2.3.1

[25] Tushar Kusnur and Maxim Likhachev. Complete, decomposition-free coverage path planning. In *2022 IEEE 18th International Conference on Automation Science and Engineering (CASE)*, pages 1431–1437, 2022. doi: 10.1109/CASE49997.2022.9926483. 4.2

[26] John Lai, Jason J Ford, Peter O'Shea, and Rodney Walker. Hidden markov model filter banks for dim target detection from image sequences. In *2008 Digital Image Computing: Techniques and Applications*, pages 312–319. IEEE, 2008. 2.3.3

[27] John Lai, Luis Mejias, and Jason J Ford. Airborne vision-based collision-detection system. *Journal of Field Robotics*, 28(2):137–157, 2011. 2.3.3

[28] John Lai, Jason J Ford, Luis Mejias, and Peter O'Shea. Characterization of sky-region morphological-temporal airborne collision detection. *Journal of Field Robotics*, 30(2):171–193, 2013. 2.3.3

[29] Changliu Liu and Masayoshi Tomizuka. Control in a safe set: Addressing safety in human-robot interactions. In *IEEE/ACM International Conference on Human-Robot Interaction*, 2014. 2.5.3

[30] Guido Manfredi and Yannick Jestin. An introduction to acas xu and the challenges ahead. In *2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC)*, pages 1–9, 2016. doi: 10.1109/DASC.2016.7778055. 2.1, 2.4

[31] Jeffrey W McCandless. Detection of aircraft in video sequences using a predictive optical flow algorithm. *Optical Engineering*, 38(3):523–530, 1999. 2.3.3

[32] Luis Mejias, Scott McNamara, John Lai, and Jason Ford. Vision-based detection and tracking of aerial targets for uav collision avoidance. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 87–92. IEEE,

2010. 2.3.3

[33] Timothy L Molloy, Jason J Ford, and Luis Mejias. Detection of aircraft below the horizon for vision-based detect and avoid in unmanned aircraft systems. *Journal of Field Robotics*, 34(7):1378–1391, 2017. 2.3.3

[34] Tamas G. Molnar, Suresh K. Kannan, James Cunningham, Kyle Dunlap, Kerianne L. Hobbs, and Aaron D. Ames. Collision avoidance and geofencing for fixed-wing aircraft with control barrier functions, 2024. URL https://arxiv.org/abs/2403.02508. 2.1

[35] Brady Moon, Satrajit Chatterjee, and Sebastian Scherer. Tigris: An informed sampling-based algorithm for informative path planning. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5760–5766, 2022. doi: 10.1109/IROS47612.2022.9981992. 4.2

[36] Muhammad Ali Murtaza, Sergio Aguilera, Vahid Azimi, and Seth Hutchinson. Real-time safety and control of robotic manipulators with torque saturation in operational space. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 702–708, 2021. doi: 10.1109/IROS51168.2021.9636794. 2.1

[37] Andreas Nussberger, Helmut Grabner, and Luc Van Gool. Aerial object tracking from an airborne platform. In *2014 international conference on unmanned aircraft systems (ICUAS)*, pages 1284–1293. IEEE, 2014. 2.3.3

[38] Michael P. Owen, Sean M. Duffy, and Matthew W. M. Edwards. Unmanned aircraft sense and avoid radar: Surrogate flight testing performance evaluation. In *2014 IEEE Radar Conference*, pages 0548–0551, 2014. doi: 10.1109/RADAR.2014.6875652. 2.3.1

[39] Jay Patrikar, Joao Dantas, Sourish Ghosh, Parv Kapoor, Ian Higgins, Jasmine J Aloor, Ingrid Navarro, Jimin Sun, Ben Stoler, Milad Hamidi, et al. Challenges in close-proximity safe and seamless operation of manned and unmanned aircraft in shared airspace. *arXiv preprint arXiv:2211.06932*, 2022. 2.1

[40] Jay Patrikar, Joao Dantas, Brady Moon, Milad Hamidi, Sourish Ghosh, Nikhil Keetha, Ian Higgins, Atharva Chandak, Takashi Yoneyama, and Sebastian Scherer. Image, speech, and ads-b trajectory datasets for terminal airspace operations. *Scientific Data*, 12(1):468, 2025. 2.3.3

[41] Stavros Petridis, Christopher Geyer, and Sanjiv Singh. Learning to detect aircraft at low resolutions. In *International Conference on Computer Vision Systems*, pages 474–483. Springer, 2008. 2.3.3

[42] Vladimir Reilly, Haroon Idrees, and Mubarak Shah. Detection and tracking of large number of targets in wide area surveillance. In *European conference on computer vision*, pages 186–199. Springer, 2010. 2.3.3

[43] Artem Rozantsev, Vincent Lepetit, and Pascal Fua. Flying objects detection from a single moving camera. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4128–4136, 2015. 2.3.3

[44] RTCA. *Minimum Operational Performance Standards for Airborne Collision Avoidance System X (ACAS X) (ACAS Xa and ACAS Xo), Volume I and Volume II*. DO 385A. RTCA, 2023. Approved by the RTCA Program Management Committee (PMC) as documented in June 2023. 2.6.6

[45] Sebastian Scherer, Vasu Agrawal, Graeme Best, Chao Cao, Katarina Cujic, Ryan Darnley, Robert DeBortoli, Eric Dexheimer, Bill Drozd, Rohit Garg, Ian Higgins, John Keller, David Kohanbash, Lucas Nogueira, Roshan Pradhan, Michael Tatum, Vaibhav K. Viswanathan, Steven Willits, Shibo Zhao, Hongbiao Zhu, Dan Abad, Tim Angert, Greg Armstrong, Ralph Boirum, Adwait Dongare, Matthew Dworman, Shengjie Hu, Joshua Jaekel, Ran Ji, Alice Lai, Yu Hsuan Lee, Anh Luong, Joshua Mangelson, Jay Maier, James Picard, Kevin Pluckter, Andrew Saba, Manish Saroya, Emily Scheide, Nathaniel Shoemaker-Trejo, Joshua Spisak, Jim Teza, Fan Yang, Andrew Wilson, Henry Zhang, Howie Choset, Michael Kaess, Anthony Rowe, Sanjiv Singh, Ji Zhang, Geoffrey A. Hollinger, and Matthew Travers. Resilient and modular subterranean exploration with a team of roving and flying robots. *Field Robotics*, 2:678–734, 2022. doi: 10.55417/fr.2022023. 1.2.2

[46] Falk Schubert and Krystian Mikolajczyk. Robust registration and filtering for moving object detection in aerial videos. In *2014 22nd International Conference on Pattern Recognition*, pages 2808–2813. IEEE, 2014. 2.3.3

[47] National Park Service, Apr 2023. URL `https://www.nps.gov/orgs/aviationprogram/visitor-protection.htm`. 1.2.2

[48] Tianjun Shi, Jinnan Gong, Shikai Jiang, Xiyang Zhi, Guangzhen Bao, Yu Sun, and Wei Zhang. Complex optical remote-sensing aircraft detection dataset and benchmark. *IEEE Transactions on Geoscience and Remote Sensing*, 61:1–9, 2023. 2.3.3

[49] Elysia Smyers, Sydney Katz, Anthony Corso, and Mykel J Kochenderfer. Avoidds: Aircraft vision-based intruder detection dataset and simulator. *Advances in Neural Information Processing Systems*, 36:8076–8091, 2023. 2.3.3

[50] Christina Soontornvat. *All Thirteen: The incredible cave rescue of the thai boys' soccer team*. Candlewick Press,U.S, 2021. 1.2.2

[51] Eric Squires, Pietro Pierpaoli, Rohit Konda, Samuel Coogan, and Magnus Egerstedt. Composition of safety constraints with applications to decentralized fixed-wing collision avoidance. *CoRR*, abs/1906.03771, 2019. URL `http://arxiv.org/abs/1906.03771`. 2.3.2

[52] Vladan Stojnić, Vladimir Risojević, Mario Muštra, Vedran Jovanović, Janja Filipi, Nikola Kezić, and Zdenka Babić. A method for detection of small moving objects in uav videos. *Remote Sensing*, 13(4):653, 2021. 2.3.3

[53] Bo Yang, Dongjian Tian, Songliang Zhao, Wei Wang, Jun Luo, Huayan Pu, Mingliang Zhou, and Yangjun Pi. Robust aircraft detection in imbalanced and similar classes with a multi-perspectives aircraft dataset. *IEEE Transactions on Intelligent Transportation Systems*, 2024. 2.3.3

[54] Mingxin Yu, Chenning Yu, M-Mahdi Naddaf-Sh, Devesh Upadhyay, Sicun Gao, and Chuchu Fan. Efficient motion planning for manipulators with control barrier function-induced neural controller, 2024. URL https://arxiv.org/abs/2404.01184. 2.1

[55] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Tracking objects as points. In *European Conference on Computer Vision*, pages 474–490. Springer, 2020. 2.5.1