Development of a Peripheral-Central Vision System to Detect and Characterize Airborne Threats

Changkoo Kang

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Aerospace Engineering

Craig A. Woolsey, Chair
Kevin B. Kochersberger
Cornel Sultan
Mazen H. Farhood

October 5, 2020
Blacksburg, Virginia

Keywords: Counter-UAS, Computer vision, Aircraft dynamics, Optimization

Development of a Peripheral-Central Vision System to Detect and Characterize Airborne Threats

Changkoo Kang

(ABSTRACT)

With the rapid proliferation of small unmanned aircraft systems (UAS), the risk of mid-air collisions is growing, as is the risk associated with the malicious use of these systems. The airborne detect-and-avoid (ABDAA) problem and the counter-UAS problem have similar sensing requirements for detecting and tracking airborne threats. In this dissertation, two image-based sensing methods are merged to mimic human vision in support of counter-UAS applications. In the proposed sensing system architecture, a "peripheral vision" camera (with a fisheye lens) provides a large field-of-view while a "central vision" camera (with a perspective lens) provides high resolution imagery of a specific object. This pair form a heterogeneous stereo vision system that can support range resolution. A novel peripheral-central vision (PCV) system to detect, localize, and classify an airborne threat is first introduced. To improve the developed PCV system's capability, three novel algorithms for the PCV system are devised: a model-based path prediction algorithm for fixed-wing unmanned aircraft, a multiple threat scheduling algorithm considering not only the risk of threats but also the time required for observation, and the heterogeneous stereo-vision optimal placement (HSOP) algorithm providing optimal locations for multiple PCV systems to minimize the localization error of threat aircraft. The performance of algorithms is assessed using an experimental data set and simulations.

Development of a Peripheral-Central Vision System to Detect and Characterize Airborne Threats

Changkoo Kang

(GENERAL AUDIENCE ABSTRACT)

With the rapid proliferation of small unmanned aircraft systems (UAS), the risk of mid-air collisions is growing, as is the risk associated with the malicious use of these systems. The sensing technologies for detecting and tracking airborne threats have been developed to solve these UAS-related problems. In this dissertation, two image-based sensing methods are merged to mimic human vision in support of counter-UAS applications. In the proposed sensing system architecture, a "peripheral vision" camera (with a fisheye lens) provides a large field-of-view while a "central vision" camera (with a perspective lens) provides high resolution imagery of a specific object. This pair enables estimation of an object location using the different viewpoints of the different cameras (denoted as "heterogeneous stereo vision.") A novel peripheral-central vision (PCV) system to detect an airborne threat, estimate the location of the threat, and determine the threat class (e.g. aircraft, bird) is first introduced. To improve the developed PCV system's capability, three novel algorithms for the PCV system are devised: an algorithm to predict the future path of an fixed-wing unmanned aircraft, an algorithm to decide an efficient observation schedule for multiple threats, and an algorithm that provides optimal locations for multiple PCV systems to estimate the threat position better. The performance of algorithms is assessed using an experimental data set and simulations.

Acknowledgments

First, I would like to thank God, without whom nothing is possible. This dissertation summarizes my work and achievement during my PhD studies in Aerospace Engineering at Virginia Tech, but there are many people to whom I am indebted for their contributions to my research.

I would like to express my special appreciation and thanks to my advisor and mentor, Dr. Craig Woolsey, who has always supported, encouraged, and guided me to complete both my master's and PhD studies at Virginia Tech. I also would like to thank my committee members, Dr. Kevin Kochersberger, Dr. Cornel Sultan, and Dr. Mazen Farhood, for providing effective feedback and direction for my research. I owe much gratitude to my great friend and colleague, Haseeb Chaudhry. He has helped and taught me a lot for the MBZIRC project and the IVT project, which we worked on together. Also, I would like to acknowledge my awesome labmates at NSL, Seyong Jung, Javier Gonzalez-Rocha, Jean-Michel Fahmi, Nazmus Sakib, and James Gresham, for facilitating my research, and the Center for Unmanned Aircraft Systems for its sponsorship.

Finally, I cannot begin to express my gratitude and love to my parents and my sister for providing me with unwavering support and continuous encouragement throughout my years of study and through the process of writing this dissertation. This accomplishment would not have been possible without you. Thank you.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Unmanned aircraft systems (UAS) have been used for various purposes, such as package delivery services, hobby purposes, and even for organ delivery at the hospital [1]. UAS technologies have been developed quickly, and the Federal Aviation Administration (FAA) found the number of UAS is growing faster than their prediction [2]. Small-UAS (sUAS) are UAS that weigh less than 55lbs, and these are popularly used by the public these days because of the ease of control and low cost. Amazon has addressed their plans to use sUAS for drone delivery [3] and sUAS are being used for various other purposes such as search and rescue and infrastructure inspection. Already, sUAS have benefited people's lives and have become deeply integrated into modern society.

According to the FAA's report [2], more than 900,000 non-commercial drones have been registered, and they estimate 1.25 million drones are currently in operation. In this report, the FAA also reported that the commercial drone market is growing even faster than they expected, and they are predicting that the market will be tripled in 3 years (823,000 drones in operation). With this rapid growth of the sUAS market, concern is also rising about the possibility of malicious misuse and mid-air collisions between sUAS and manned aircraft. According to the FAA fact book [4], near mid-air collisions (NMACs), which is defined as when two or more aircraft fly within 500 feet of each other, has doubled from 2016 and keeps

increasing (145 in 2015, 302 in 2016 and 385 in 2017). The FAA especially mentioned that more than half of all NMACs reported currently are related to UAS. In 2017, a drone crashed into a Blackhawk helicopter [5], and a drone crashed into a fixed-wing manned aircraft in Canada in 2018. The Alliance for System Safety of UAS through Research Excellence (ASSURE) released reports [6, 7, 8] about the risk of the collision between a manned aircraft and an sUAS. The reports describe how sUAS (quadcopter and fixed-wing) can damage the body and the engine of commercial aircraft through realistic simulations and also show that the collision may result in a serious accident of the aircraft. These risks of sUAS on other manned aircraft caused major disruption at the Gatwick airport in the U.K. in December 2018 [9]. Likewise, unauthorized and non-cooperative sUAS can be a serious threat to people's safety. FAA recently stated the needs of the remote identification of all sUAS [10]. Considering that the remote ID is only for registered sUAS, this cannot be a solution for malicious privately made sUAS.

After the 'drone chaos' at the Gatwick airport, other airports started to invest in drone detection systems [11]. With this concern about non-cooperative sUAS, research on airborne detect-and-avoid (ABDAA) and counter-UAS (C-UAS) technologies has accelerated. These types of systems use a variety of sensors such as electro-optical/infrared (EO/IR) cameras, radar and acoustic sensors, and the data from these sensors are fused to detect and track a given threat. However, active sensors can be problematic for mobile C-UAS systems because of the high size, weight, and power and the cost (SWaP-C.) According to [12], 77 out of 123 commercial C-UAS systems cost more than $100,000. Gatwick airport spent several million dollars on installing a C-UAS system [13]. However, C-UAS systems are not only for the airport; a sUAS may be dangerous to audiences in a soccer stadium [14], or passengers in a train [15]. The C-UAS systems have become necessary in various places with the fast-growing number of sUAS, and the expensive C-UAS systems may be feasible only for limited

locations.

Therefore, the motivation of this dissertation is developing a threat detection system that uses cheap sensors and that can be mounted on a sUAS. Cameras are the most common sensor for sUAS. Many drone manufacturers are mounting cameras on their products since cameras have low SWaP-C, which is necessary for sUAS. As the SWaP-C of cameras has continued to drop, while resolution and image quality has continued to improve, and as computer vision methods have continued to develop, it has become possible to extract more information (e.g., aircraft attitude) from an image than was possible in the past. Because of this usefulness, the camera is one of the most common sensors for mobile C-UAS. Considering the economical and functional characteristics of camera sensors, the camera will continue to be an important sensor for the development and the popularization of C-UAS systems.

1.2 Aim and Objectives

Camera imagery can be an effective means for threat detection as well as for pose estimation and threat classification, though one cannot resolve range to a threat from a single camera image without additional information, such as knowledge of the threat geometry. Also, a narrow field of view (FOV) of a perspective camera is limiting, so that covering a broad range of areas using a perspective camera is a difficult task without the help of other sensors. Fast and reliable initial threat detection is crucial for a C-UAS system, so the camera system must be able to see a large area at once. This observation suggests the use of an omnidirectional "peripheral" vision camera for the initial threat detection. To provide continuous visual coverage of the environment for threat detection requires a wide FOV ("peripheral vision") camera. To classify a threat aircraft and estimate its pose and better-predicted flight path, a higher resolution image is required, as might be obtained from a narrow FOV perspective

("central vision") camera. Incorporating each type of camera affords an opportunity to use stereo vision for ranging. Accordingly, one of the objectives of this dissertation is introducing a novel heterogeneous "peripheral-central" vision (PCV) system for a ground- or air-based C-UAS system that is capable of detecting flying objects within a wide FOV, confirming and classifying these threats, estimating their position (including range), velocity, and pose, and predicting their flight path using only two light and low-cost cameras. The entire threat tracking process using the PCV system will be depicted in detail from image processing to position and pose estimation of a single threat.

To consider an urban environment or more crowded areas in which multiple objects may exist in the air, the single threat scenario needs to be extended to a multiple threat scenario. Multiple threat scheduling and multiple sensor placement have become major research topics for C-UAS systems with the development of the UAS swarm technology. Another objective is thus developing a novel threat scheduling algorithm to decide the observation priority among detected threats and an optimal placement algorithm for multiple PCV systems to minimize the localization error of multiple threats. Once the threat data (position, pose, etc.) is obtained from the PCV system, the future path of the threat can be estimated; and this future path would be helpful to track the threat aircraft. Therefore, this dissertation also focuses on the path prediction algorithm for a small fixed-wing unmanned aircraft based on pose data.

1.3 Outline

This dissertation consists of eight chapters. The system algorithms and the proof-of-concept analysis through testing of the PCV system is described. The dissertation is organized as follows:

- Chapter 2 provides a literature review on C-UAS systems, sensors for C-UAS systems and heterogeneous stereo vision.
- Chapter 3 introduces the PCV system and algorithms for the PCV system.
- Chapter 4 proposes a heterogeneous stereo vision algorithm to estimate the threat aircraft position and analyzes the localization performance of the PCV system.
- Chapter 5 suggests a model-based path prediction algorithm for fixed-wing unmanned aircraft using pose estimates obtained from the PCV system.
- Chapter 6 introduces a multiple threat scheduling algorithm using a modified travelling salesman problem solution.
- Chapter 7 presents an optimal placement algorithm for multiple PCV systems to minimize the localization error of the threat aircraft.
- Chapter 8 concludes and summarizes the work in this dissertation.

1.4 Contributions

The contribution of this dissertation is summarized as below:

- **Development of a peripheral-central vision system [16, 17, 18]**

  - Integrated the hardware for the peripheral-central vision system with the assistance of my collaborator.

  - Generated the PCV system ground- and air-based dataset with the assistance of collaborators. Dataset comprises imagery of threat UAS from the ground- and air-

based PCV system, as well as vehicle state information, to support future algorithm developers.

- Developed the heterogeneous stereo vision algorithm that corrects for distortion and misalignment of disparate cameras (e.g., omnidirectional and perspective) to estimate range to an object.

- Devised the zoom optimization algorithm. This algorithm adapts the zoom setting of a pan/tilt/zoom (PTZ) camera to maximize pixels on target while minimizing the risk that the target is lost from the image.

- Designed visual-based aircraft pose estimation algorithm which estimates the aircraft attitude using target aircraft imagery.

- **Fixed-wing aircraft path prediction based on attitude data** [19]

  - Developed the acceleration estimation algorithm for unsteady flight based on an aircraft performance formulation in terms of specific power and energy.

  - Developed the wind disturbance estimation and path prediction correction algorithm

- **Multiple threat tracking algorithm using modified travelling salesman problem** [20]

  - Designed the multiple threat tracking algorithm using modified travelling salesman problem formulation and solution. This algorithm generates the threat schedule for the central vision camera to image threats and predict their paths when multiple threats are present. The formulation accounts for threat risk, angular distance between threats, etc.

- **Optimal placement algorithm for multiple PCV systems** [21]

- Devised the multiple PCV system placement algorithm which estimates optimal places for multiple host aircraft to minimize the localization error of multiple threat aircraft.

Chapter 2

Literature Review

In this chapter, a general architecture of the existing commercial C-UAS systems is described. The state-of-art of threat detection sensor technologies for C-UAS are summarized and the strengths and weaknesses of each sensor are described. Lastly, other research on heterogenous stereo vision systems reviewed.

2.1 Architecture of C-UAS Systems

As sUAS technology continues to be developed, the size is becoming smaller and the speed of the sUAS keeps increasing. One of the popular racing drones can fly up to 166mph (74.2 m/s) and nano (or micro) drones which are lighter than 0.55lbs (249g) are currently available [22], so detecting sUAS has become more difficult than in the past. The interest in C-UAS systems has started recently, and the incident at the Gatwick airport [9] attracted more attention to the world for the C-UAS systems. C-UAS systems are currently in the infancy level, so there is not a specific global standard [23]. Currently, according to the report from the Center for the Study of the Drone (CSD) [12] at Bard College, there are 537 C-UAS products from 277 manufacturers of 38 countries, and researchers from the government of many countries and industries are actively developing C-UAS technologies. Most of the existing commercial C-UAS systems generally consists of 3 steps to react to UAS [12, 22, 23, 24, 25]:

1) Threat detection

- This step is about the processes and technologies to detect, identify and track the threat using sensors, and analyze the threat
- The sensors for threat detection include radar, radio-frequency (RF), electro-optical (EO) camera, infrared (IR) camera, acoustic sensors, combined sensor system, etc.

2) Threat decision

- This step is a process to classify whether the target is a threat or not, and decide the reaction (defensive or offensive) to the identified threat.
- This step is the least developed among the C-UAS processes because this step highly depends on the policies and protocols, which are not fully established yet.
- The processing time should be minimized to allow sufficient time to react to the threat. A pre-programmed system can be involved in this process, but the human operator is normally included for this step given the current level of technology.

3) Threat neutralization (interdiction)

- This final step is to respond and influence the identified threat from the previous step.
- This process includes the kinetic response (projectile, airburst, net, sacrificial collision drone, laser, electromagnetic pulse, water projector, etc.) and the non-kinetic response (jamming, spoofing, signal control, blocking, etc.)

Herrera *et al.* analyzed the development trends of the technologies for the UAS and the C-UAS systems in their report [22]. They expect that the threat detection technology will steadily grow. They also expect that, however, there should be new capability gaps because

the sUAS technology will grow as well. Especially, they mentioned the major existing gap is threat detection in multiple threat scenarios.

The threat decision step can be difficult and complicated because of the definition of a threat. The criteria of the threat can be different depending on the situation, environment, policy, protocol, and so on. Herrera *et al.* predict that this process will be developed slowly because there are many factors to consider in developing the threat decision step.

The threat neutralization technologies have been developed quickly. Some interdiction technologies (i.e. jamming, missile), however, may harm other devices necessary for the aircraft communication, and these technologies are even dangerous to the public.

The main focus of this dissertation is threat detection and tracking. The existing detection technologies are reviewed in the next section.

2.2 Threat Detection Technologies

Developing sensor detection technologies is the most important part of building effective C-UAS systems, but this process requires ongoing attention because of the fast-growing sUAS technologies [23]. Threat detection can be conducted by various sensors such as radar, RF sensor, EO/IR cameras, and acoustic sensors. Some sensors emit signals and detect objects by analyzing the reflected energy. This type of sensor is called “active sensor”, and radar and lidar are good examples of the active sensors. Also, some sensors passively receive data such as light, sound, heat, radio signals, etc. These sensors are called “passive sensor”. EO/IR cameras, acoustic and RF sensor are common passive sensors. Because the information that each sensor gathers from objects is different, the data type generated and the environmental conditions for sensing are different for each sensor.

2.2.1 Active sensors

Radar is one of the most common sensor systems for airspace control [26]. Radar detects sUAS by emitting radio frequency pulses and receiving reflected radio signals from the object. The estimation accuracy of the target position and the range using radar is high compared to other sensors [27], and radar is capable of detecting and tracking a variety of targets with kilometers of detectable range even in adverse light and weather conditions. Moreover, the decreasing size of radar allows it to be mounted on a sUAS [28], and the accuracy of radar-based threat classification using deep neural network (DNN) is good [27]. However, radar has been typically used for detection of manned aircraft that have relatively a large radar cross section (RCS) and high speed [12, 27]; sUAS detection can be difficult using radar. Even with these weaknesses of radar, it is the most common element in the commercial C-UAS systems [12], and many researchers have been working on sUAS detection using various types of radars such as multiple-input multiple-output (MIMO) radar [29, 30] and continuous-wave radar [31, 32, 33]. Also, radar-based UAV classification has been actively studied [32, 34].

RF sensors detect signals that sUAS emit. It is also one of the most common sensors in C-UAS systems [12]. RF sensor is relatively cheap and effective, and is not affected by the weather or light condition; however, the signal easily drops with the range of the threat. Also, RF sensor can be distracted by other signal sources such as Wi-Fi and might not work for some quite sUAS (e.g. pre-programmed maneuvering sUAS) [35, 36]. Several studies have been conducted for sUAS detection using RF sensor. Ezuma *et al.* [37, 38] used RF signals to detect and classify sUAS using machine learning (ML) techniques, and Al-Sa'd *et al.* also used RF signals to detect sUAS using DNN [39].

2.2.2 Passive sensors

Acoustic sensors detect sUAS by finding a unique sound of the motors of sUAS. This sensor is comparatively small, lightweight and inexpensive since the system typically consists of microphones and processing units. The detection range is, however, short, and the sound library cannot cover all types of sUAS since the emitted sound from each sUAS is different [40, 41, 42]. Researchers have developed acoustic sensor-based UAS detection [40, 43, 44, 45]. Especially ML techniques such as the Gaussian Mixture Model (GMM) [46], support vector machine (SVM) [47, 48], and random forest [49] are the most popular approach to detect and identify sUAS using an acoustic sensor. Also, Casasanta *et al.* [50] used an acoustic radar (sodar), and Busset *et al.* [51] used an acoustic camera to detect sUAS in their work.

Electro-optical (EO) camera and infrared (IR) camera passively detect targets based on visual rays and heat rays, respectively. EO/IR camera is cheaper than other sensors and has a low SWaP-C. EO camera is thus a common sensor for Airborne Detect-and-Avoid (ABDAA) and mobile C-UAS systems. Classification performance of camera based on DNN is superior to other sensors [27], so many efforts have been made for detecting and classifying sUAS using camera imagery. Du *et al.*. [52], Saqib *et al.* [53] and Sommer *et al.* [54] used a convolutional neural network (CNN) to detect and classify sUAS. One of the major challenges in this area is to classify sUAS from birds. To improve classification accuracy, several studies have gathered datasets of birds and sUAS and developed effective DNN structures [55, 56, 57, 58]. Other than DNN approaches, several researchers have worked on detecting sUAS based on the motion of the object. Rozantsev *et al.* [59, 60] used temporal information from the sequence of image frames of a single camera to detect a moving sUAS. Hu *et al.* [61] detected the horizon and extracted the motion feature of sUAS. Some studies [62, 63, 64] focused on determining direction (azimuth and elevation) of a threat.

For a camera with a sufficient resolution, an unobstructed image can provide details that other sensors, such as radar, RF and acoustic sensor, cannot provide. As the SWaP-C of cameras has continued to drop, while resolution and image quality has continued to improve, and as computer vision methods have continued to develop, it has become possible to extract more information (e.g., aircraft attitude) from an image than was possible in the past [16]. However, EO camera is easily affected by the light and weather conditions and the background clutter. IR cameras can address this drawback of EO cameras by looking at the target using only the thermal signature. Birch *et al.* [65] compares the performance between visible (VIS) camera and three IR cameras (SWIR, MWIR, and LWIR) in their report, and the visible camera shows the best performance among four cameras for the uniform background, but it requires external light sources. Birch *et al.* finally concluded that LWIR may be the best suited for sUAS detection, although it can be confused with other thermal sources such as birds. Likewise, EO and IR cameras are capable of covering the drawbacks of each other, they are thus normally used in conjunction with each other for commercial C-UAS systems.

2.3 Heterogeneous Stereo Vision

Traditional stereo vision systems [63, 66] typically include two identical perspective cameras so that it can estimate the depth of an object, but the narrow field of view (FOV) of a perspective camera is limiting. To address this issue, Drulea *et al.* [67] and Kita *et al.* [68] proposed the use of a stereo fisheye vision system to relax the FOV limitation, but fisheye cameras provide lower pixel coverage in a given region, making it more difficult to classify a detected threat or to estimate its pose for trajectory prediction at greater distances. While the utility of cameras is limited to visual flight rules (VFR) conditions, these conditions are

typical for current small UAS operations.

Incorporating the strengths of a large FOV camera and a perspective camera, dual camera systems using an omnidirectional camera and a PTZ camera have been developed for surveillance purposes. Researchers mostly focused on person detection [69, 70, 71, 72] and human face tracking [73, 74] using dual-systems. Based on dual-camera system development, Baris *et al* [75] classified objects in a scene for better surveillance performance. The dual-camera systems in these studies have shown good performance for target tracking at a fixed location. The omnidirectional camera, however, was used mostly for initial detection cue, not for target ranging purposes, because the purposes of these studies were surveillance.

Muñoz-Salinas *et al* [76] suggested a particle filtering-based multiple person localization algorithm using multiple heterogeneous cameras. The study concerns person tracking as well, but the suggested algorithm of this paper outputs the location of people with confidence data using particle filtering. Rathnayaka *et al* [77] suggested a camera calibration method using a large FOV camera lens and a perspective lens, but this study does not present the object ranging performance.

For a different application other than surveillance, Eynard *et al* [78, 79, 80] suggested an algorithm to estimate the altitude and motion of an unmanned aircraft using an onboard, heterogeneous stereo vision system that consists of a fisheye lens camera and a perspective camera. Their algorithm first finds the homography matrix relating the two camera views and then estimates the distance between the horizontal plane (i.e., the tangent plane to the earth's surface) and the first camera. The algorithm determines the altitude of own aircraft, but not the range or state of the other aircraft. As described above, most of the applications of heterogeneous camera systems are confined to area surveillance (<20 m range) on a fixed-platform. To the author's knowledge, no earlier studies discuss stereo ranging using a heterogeneous camera system for the C-UAS application.

2.4 Threat Aircraft Path Prediction

Conflict detection and resolution (CD&R) technologies, such as the traffic alert and collision avoidance system (TCAS), have played an important role in air traffic management (ATM) since at least the early 1990s [81, 82]. CD&R technologies for manned aircraft are supported by a large amount of aircraft performance model (APM) data, which has enabled extensive research and development of path prediction methods for manned aircraft. Recent emphasis on direct, real-time flight data transfer among aircraft has enabled even more accurate path prediction using flight safety communication systems such as automatic dependent surveillance-broadcast (ADS-B) [83, 84].

With the proliferation of sUAS, and the recognition that these aircraft operate very differently than manned aircraft, there has also been recent interest in developing path prediction methods for this class of aircraft. Many of the CD&R studies for sUAS focus on path prediction using only position data. In these studies, the sUAS is typically assumed as a particle, and the velocity and acceleration are computed as the first and second time derivatives of position, respectively. The future position of the sUAS is then propagated with time [85, 86]. In references [87], [88] and [89], the path is predicted under the simplifying assumption that the threat moves with constant velocity or with constant acceleration. Based on a similar particle dynamic model, references [90] and [91] predict the flight path using a Kalman filter, while references [92] and [93] instead use a particle filter. Reference [94] defines the reachable set of the sUAS in order to ensure collision avoidance. Small unmanned aircraft have the ability to maneuver much more aggressively than manned aircraft, however, and they are more susceptible to wind disturbances, so path prediction methods that are based solely on the position history of the threat aircraft may not provide sufficient warning to inform a CD&R system. These prediction methods assume a particle dynamic ("aircraft

performance") model for the threat aircraft, but accurate performance models may be unavailable [95].

In considering path prediction for sUAS, it is helpful to review methods that have been proposed for manned aircraft. ATM-related path prediction methods normally assume known flight data, such as thrust, mass, position and other variables associated with point mass models (PMMs) [96, 97, 98, 99, 100, 101]. Recent work concerning aircraft path prediction used radar and weather data from ground-based systems to infer the path, as in [102, 103, 104, 105, 106]. For model-based path prediction, it can be helpful to know, or to estimate, the mass or thrust of the threat aircraft. Alligier et al. [107] suggested a prediction algorithm based on an estimate of the mass of the threat aircraft developed using energy rate and learning algorithms. Thipphavong et al. [108] also estimated the mass of the threat aircraft to inform the path prediction. In [109] and [110], the aircraft thrust was estimated. These studies used mature manned aircraft performance models, such as the Base of Aircraft Data (BADA) model of EUROCONTROL [111], but such models may be inappropriate or impractical for a maneuvering sUAS.

2.5 Resource Management for Multiple Threat Aircraft

In an environment in which multiple threat aircraft exist in the air, the host sensor system needs a priority to observe threat aircraft based on particular parameters (e.g. class, speed, position, etc.) To solve this problem, several studies have been conducted to schedule multiple threats. Threat scheduling algorithms are commonly used with phased array radars. The phased array radar is capable of tracking multiple threats at the same time, but the

operating cost is expensive since the array of transmitters actively emits beams. Therefore, threat scheduling algorithms have been studied for efficient radar resource management to focus on those threats posing the greatest risk. Miranda *et al.* defined detailed criteria for threats in their papers [112, 113, 114]. They determined a priority of threats using fuzzy logic based on various parameters such as velocity, threat ID, heading, etc. Ding *et al.* [115] also used fuzzy logic to decide the priority of threats. Some studies [116, 117, 118, 119] proposed optimal threat scheduling algorithms for multiple PTZ cameras to minimize the uncertainty of threat data using information theory. Khosla *et al.* [120] proposed a novel threat scheduling algorithm for a radar-cued camera system. They considered time elapsed from the initial detection, FOV of the camera, and a slew angle of the PTZ camera for threat capture to determine the priority of the threats. The threat scheduling algorithms mentioned above prioritize the risk posed by each threat over other concerns. These scheduling algorithms are especially appropriate for ground-based sensing systems, which are less limited by size, weight, power, and cost (SWaP-C) and which generally have a long sensing range.

Another strategy to cover the multiple number of threats is to use multiple host sensor systems. The optimal drone placement problem has enjoyed considerable attention within the larger context of wireless sensor networks (WSNs). Many of these studies have addressed ground target coverage problems using multiple camera-mounted drones. Various approaches include the geometric algorithm [121], the brainstorm algorithm [122], the moth search algorithm [123], the bare bones fireworks algorithm [124], and so on. These camera-mounted drone placement algorithms address the ground target coverage problem using a perspective camera model. Other studies [125, 126, 127, 128] have considered optimal placement of camera-mounted drones with the aim of minimizing total energy consumption while maximizing target coverage. Again, these efforts considered a perspective camera model, rather than an omnidirectional or heterogeneous stereo vision setup.

The studies above focused on ensuring ground target coverage, rather than minimizing localization error, and adopted a relatively simple camera model. Other studies on optimal placement have emphasized target observation using a variety of visual sensors. References [129, 130, 131] propose algorithms that find optimal positions for ground-based mobile cameras to maximize the aggregate observability of objects' motion in a confined area. The investigators used a pinhole camera model and implemented the algorithms both in the simulation environment and in experiments. Other work has addressed the problem of placing multiple cameras for 3D reconstruction [132, 133, 134] and 3D motion capture [135]. The investigators analyzed camera models to minimize the depth estimation error. However, these studies involved indoor operation for objects at short distances. A number of papers have provided a detailed analysis of triangulation-based localization error [136, 137, 138], illustrating how this error is affected by both the location of sensors and the line-of-sight (LOS) pointing angles from the sensors to a target. The authors then used these results to develop optimal sensor placement algorithms to minimize the target localization error using ground-based robots. Reference [139] discusses triangulation-based localization accuracy using two omnidirectional cameras to estimate the optimal positions.

A few studies have considered optimal placement of cameras with different fields of view. Reference [140] defines various types of camera models to solve an area coverage problem. Similarly, reference [141] addresses a target coverage problem using perspective cameras with different FOVs. Reference [142] describes the use of omnidirectional and perspective cameras to cover an area with the minimum number of cameras.

While a number of studies have considered optimal drone/camera placement to cover an area, or to minimize the depth estimation error, none have addressed the localization of threat aircraft using multiple aircraft equipped with heterogeneous camera systems.

Chapter 3

Peripheral-Central Vision (PCV) System

3.1 Introduction

In this chapter, a peripheral-central vision (PCV) system that detects, localizes, classifies, tracks and predicts the motion of sUAS for C-UAS applications is introduced. Fast and reliable initial threat detection is crucial for a C-UAS system, so the camera system must be able to see a large area at once. This observation suggests the use of an omnidirectional "peripheral" vision camera for the initial threat detection. To provide continuous visual coverage of the environment for threat detection, however, requires a wide FOV ("peripheral vision") camera. To classify a threat aircraft and estimate its pose, and to better predict its flight path, a higher resolution image is required, as might be obtained from a narrow FOV perspective ("central vision") camera. Incorporating each type of camera affords an opportunity to use stereo vision for ranging. Accordingly, this chapter introduces a heterogeneous "peripheral-central" vision system for a ground- or air-based C-UAS system that is capable of detecting flying objects within a wide FOV, confirming and classifying these threats, estimating their position (including range), velocity, and pose, and predicting their flight path (Chapter 5). The PCV system hardware integration was conducted in collaboration with others. The focus here is on the development of the system algorithms and the software for

them. This work is presented in [16, 17, 18].

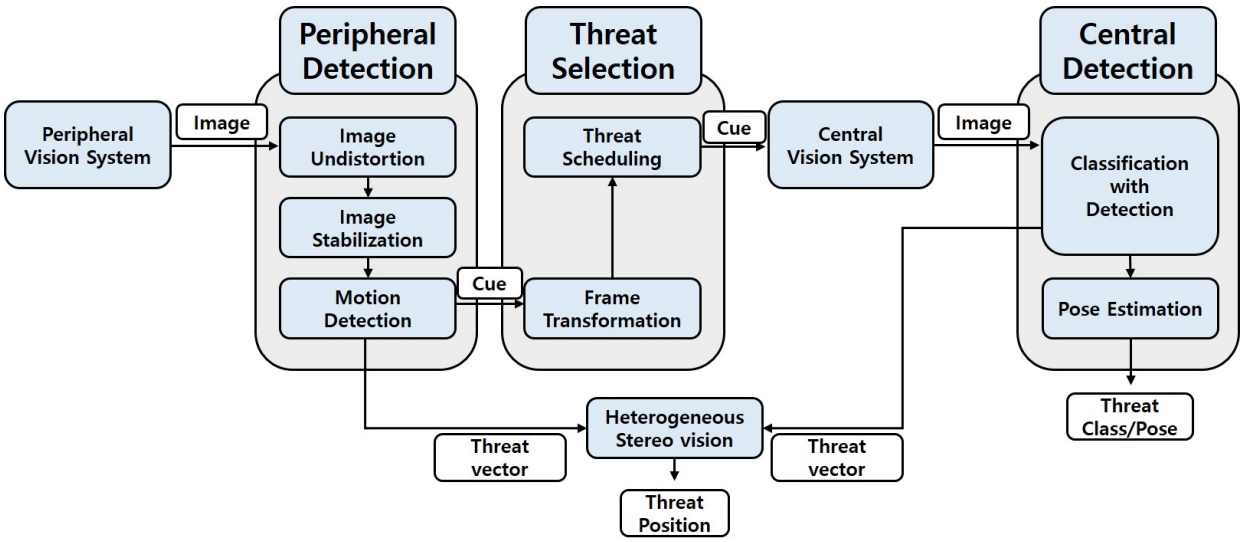


Figure 3.1: The flow chart of the PCV system

3.2 Peripheral-Central Vision System Setup

Referring to the left side of Figure 3.1, threat detection and characterization begins with the peripheral vision system. First, the lens distortion of the peripheral vision imagery is removed. The undistorted imagery is then stabilized using a homography matrix computed from the image sequence. Threats are then detected by their motion through the undistorted, stabilized imagery using optical flow. The bearing angle to the detected threat, measured relative to the boresight axis of the peripheral vision camera, is then converted to an approximate bearing angle in a frame fixed in the central vision camera. This angle provides a cue for the gimbal that controls the central vision camera's attitude. The central vision camera slews to the indicated direction and the threat is then detected in the central vision image. The higher resolution image from the perspective camera enables the classification of the threat using a deep neural network (DNN) and the estimation of its attitude using

the method presented in [16]. In addition, a heterogeneous stereo vision strategy uses the two distinct "threat vectors" obtained from the two camera images to estimate the threat position.

Figure 3.2: Peripheral-central vision (PCV) system setup.

The Insta360 Air was selected as the peripheral vision camera. This camera contains two fisheye lenses, providing 360° coverage in both azimuth and elevation, and is compatible with a variety of embedded hardware systems and with widely used software tools such as those available through OpenCV and the robot operating system (ROS). The GigE DFK Z12G445 color zoom camera from The Imaging Source serves as the central vision camera. This global shutter camera captures images at up to 41 frames per second (fps) and has a software-controllable, motorized 12x optical zoom lens. The GigE camera is mounted on an HDAir Infinity MR S2 gimbal, which enables the camera to be directed by the cue provided from the peripheral vision camera. These two cameras are accessed using ROS, installed on an NVIDIA TX2. This on-board computer runs all system algorithms, such as detection and cueing. The detailed camera specifications are shown in Table 3.1. Figure 3.2 shows the PCV system setup, as configured for air-based operation. For ground-based testing, the PCV system was wired directly to the ground station.

Table 3.1: System parameters

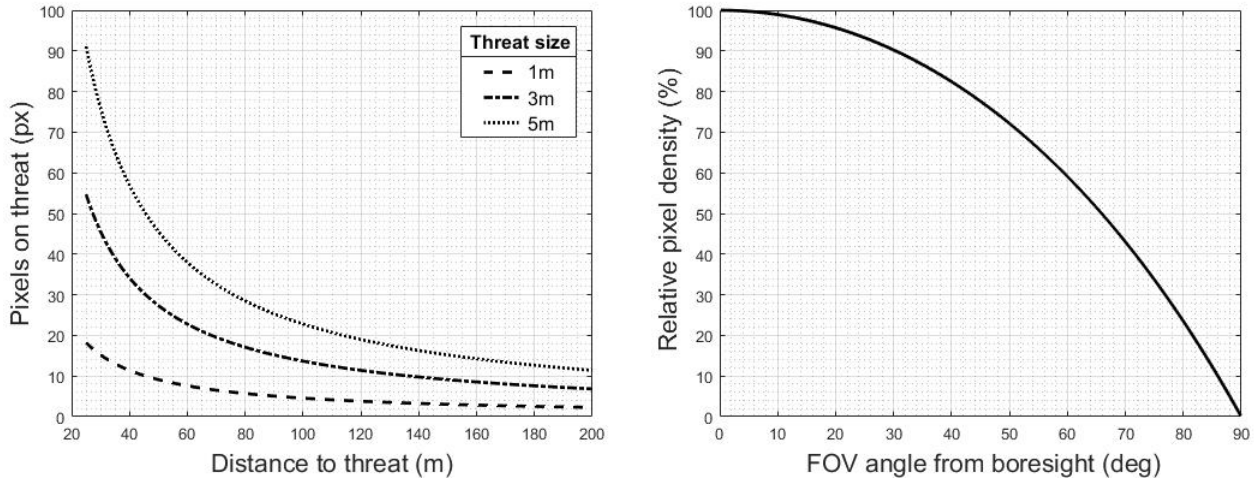
Parameter	Peripheral Vision Camera (Insta360 Air)	Central Vision Camera (GigE DFK Z12G445)
Focal length (mm)	1.0	4.8 - 57.6
Sensor size (mm)	3.3 × 3.3	4.8 × 3.6
Pixel size (μm)	2.19 × 2.19	3.75 × 3.75
Resolution (px)	1,504 × 1,504	1,280 × 960
Size (mm)	ϕ36.6 × 39.6	50 × 50 × 103
Weight (g)	26.5	330

3.3 Peripheral Vision Subsystem

While the peripheral vision camera system provides complete coverage, the images have relatively low resolution and high distortion. The number of pixels associated with an object at a given range is low, compared with a perspective camera, and varies nonlinearly over the camera FOV. As an example, for the given camera, the pixel width of a 5 m object at 100 m distance is only 22 pixels, as shown in Figure 3.3a. When there are enough pixels on the threat, the peripheral camera can provide information that is useful in characterizing a threat, but the lens introduces a high level of image distortion, especially at the edges of the image. Distortion can be partially addressed through proper camera calibration [143], but the pixel density is unavoidably lower away from the camera boresight. As an example, Figure 3.3b shows the relative pixel density for an undistorted Insta360 Air image versus angle from boresight.

Image resolution is less of a concern in threat detection than in the other image processing tasks, however, because detection methods such as the optical flow algorithm used here [144] can detect moving objects with a small amount of pixels. A peripheral vision camera system can be quite useful for initial threat detection because of its large FOV. Having detected a threat, a central vision camera may be cued to investigate further. Any additional informa-

tion that becomes available, e.g., when there are more pixels on the threat in the peripheral image, can be combined with central vision imagery to improve overall awareness. The complementary nature of the peripheral vision camera's wide FOV and the steerable central vision camera's high resolution motivated the proposed architecture.



(a) Pixels on threat versus distances from camera

(b) Relative pixel density of the undistorted image

Figure 3.3: Capabilities of Insta360 Air

3.3.1 Image pre-processing

For a camera that is fixed in space, the pixels associated with static objects and with the unmoving background do not move within an image. For an airborne system, however, the camera translates and rotates as the aircraft moves, so the background and static objects appear to move within the image. This apparent motion of static elements (camera ego motion) must be eliminated in order to detect moving objects using optical flow.

The homography matrix indicates the relative rotation and translation between two images of a given scene [145, 146, 147]. The homography matrix between consecutive frames of a moving camera can therefore be used to remove the apparent motion in the image. This

process is called image stabilization in the computer vision community; it is a software variant of mechanical image stabilization. Computing a homography matrix requires the pixel locations of common feature points in the sequence of images. A corner detector [148] is applied to the peripheral camera imagery, and the optical flow algorithm is used to track detected corners (feature points) in the next sequence of images. From the sequence of pixel locations of selected feature points, the homography matrix is estimated using the RANSAC algorithm [149], which generates an optimal estimate while excluding outlier feature points. The homography matrix is estimated for each new image frame and is then used to remove the effect of camera motion by rotating and translating the image.

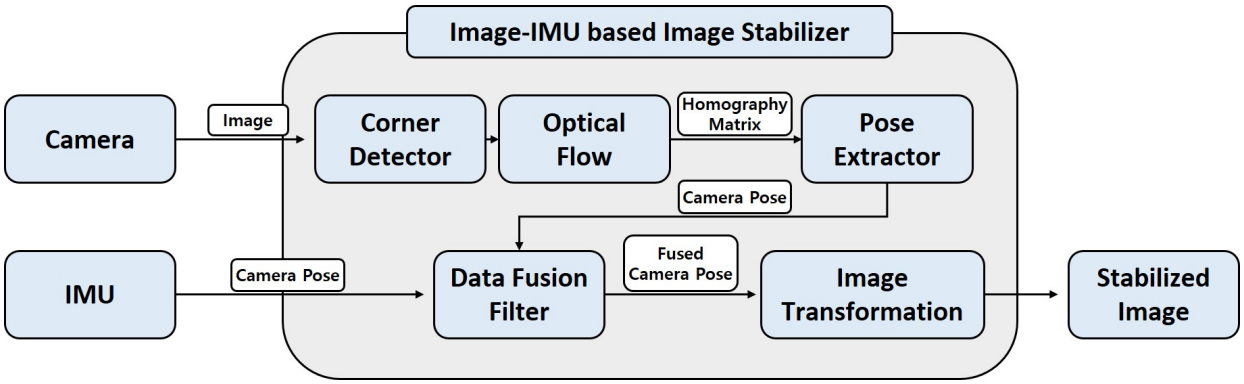


Figure 3.4: Homography-IMU data fusion for image stabilization.

Homography-based image stabilization performs quite well in nominal conditions, but the algorithm can be affected by lighting conditions and image noise. As an alternative, one may consider image stabilization based on direct measurements of camera motion, obtained using an inertial measurement unit (IMU). This approach compensates for camera motion using camera pose data from the IMU, rather than homography data extracted from images. This IMU-based approach requires that the image and inertial motion data be accurately synchronized for a good performance, however, and the IMU's accuracy is an important performance factor. Moreover, the IMU-based approach cannot accurately account for the camera's translational motion, which is an effect that is included implicitly in the homography-based

approach.

To leverage the strengths of both approaches, a data fusion Kalman filter was developed. The fused camera motion data are converted to a rotation matrix that is used to stabilize images for use in optical flow-based threat detection. The flow chart in Figure 3.4 illustrates the image stabilization process.

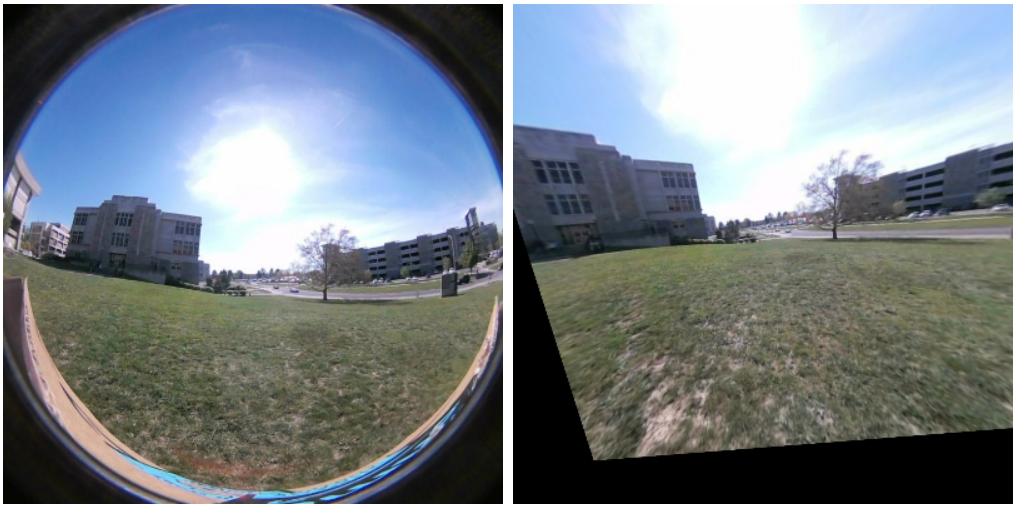


Figure 3.5: Peripheral vision image before (left) and after (right) undistortion and stabilization.

3.3.2 Threat detection using optical flow

Various computer vision algorithms have been proposed to detect a moving threat using visual imagery. Attributes that are unique to a particular scenario can pose special challenges or opportunities for visual detection. For a C-UAS system, for example, one may find that flying objects appear with reasonable contrast against a static background (e.g., a clear blue or overcast sky). Moreover, non-antagonistic aircraft, including many sUAS, include lighting to make them more visible.

In the approach described here, an optical flow algorithm computes the translational dis-

placement of pixels in consecutive images. Because the pixel coordinates change for an object moving through an image against a static background, such pixel displacements suggest possible threats. In order to detect a threat, several feature points within the image are extracted using a corner detector. The optical flow algorithm is then applied to track these feature points in consecutive images. Pixel velocity vectors whose magnitude exceeds a threshold indicate candidate threats. A high threshold value may not detect anything, while a low threshold value may misjudge every pixel as a moving object. To explore the sensitivity of threat detection to conditions, representative "receiver operating characteristic (ROC)" curves for optical flow detection are generated. Two experiments were performed. In experiment 1, the threat was located relatively close (20 - 30 m) to the PCV system. In experiment 2, the threat was farther away (50 - 60 m). Figure 3.6 shows a ROC curve of each of these two experiments with detection threshold values varying from 0 px to 5.5 px. Note the "knees" in the two ROC curves which indicate appropriate threshold values for detection. For the closer threat, the knee occurs at a threshold value of 3.5 px. For the more distant threat, the indicated threshold was 1.5 px. The threshold values for each range of the threat were tuned in this way. For detection, a low threshold value (e.g. a threshold value for threats at the maximum detectable range) is used to find threat candidates when the threat range is unknown, and an object that has the fastest pixel speed among the candidates is determined as the threat. Once the range of the threat using the ranging algorithm is estimated (which will be described in the next chapter,) the threshold value is switched for the corresponding threat range.

Special challenges arise in vision-based threat detection as proposed, and these are the subject of continuing study. For example, a threat coming straight toward the camera may not be detected since the optical flow algorithm detects relative motion in the image frame. Also, a dynamic or variegated background will increase the number of false detections. This

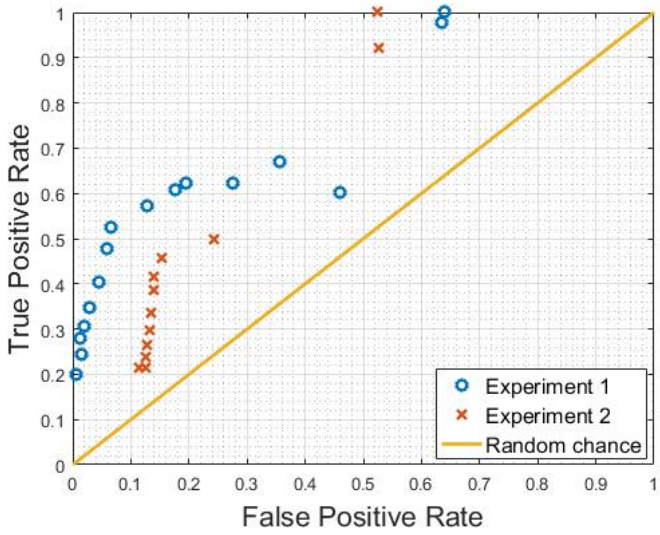


Figure 3.6: Receiver operating characteristic (ROC) curve to use to find an appropriate threshold value for optical flow detection (data points: detection performance using different threshold values)

problem may be addressed in part by adaptively tuning the detection threshold values and by incorporating complementary feature-based detection algorithms.

After a threat is detected in the peripheral camera imagery, a Kalman filter estimates its bearing in the central vision camera frame and cues the gimbal toward the threat. For multiple threat detections, a scheduling algorithm can provide an optimal cueing sequence [20]. When the optical flow algorithm loses track of a threat, the Kalman filter predicts the threat's pixel location based on its last known pixel velocity. If and when a direct measurement of the threat location becomes available once again, the Kalman filter corrects the threat location estimate that serves as a cue to the central vision camera. Figure 3.7 shows example results for two (undistorted) images obtained using the peripheral vision camera with Kalman filtering. Optical flow appears to be effective at detecting candidate threats against a static background, even for threats of small pixel size. Given a manageably low number of false detections, one may use the threat location and velocity in pixel coordinates,

Figure 3.7: Example results of the optical flow detection with Kalman filter (red circle on the left indicates a moving object detected by optical flow. The Kalman filter corrects its estimation (a green dot) based on the detection. There is no optical flow detection on the right, the Kalman filter predicts its estimation based on the estimated pixel position and velocity.)

as described in the next sections, to cue the central vision camera system, which can then classify the threat and estimate its position and attitude.

3.4 Cueing Algorithm

The threat detected by the peripheral vision camera must then be observed by the central vision camera to obtain more detailed information such as position, velocity, attitude, and classification. For the gimballed central vision camera to slew to the threat, a cue is required from the peripheral vision system. This section describes a cueing algorithm that computes the azimuth and elevation angle of the threat in the central vision camera-fixed reference frame. The section also discusses the error in this threat cue and its effect on threat acquisition by the central vision camera system.

3.4.1 Threat bearing angle

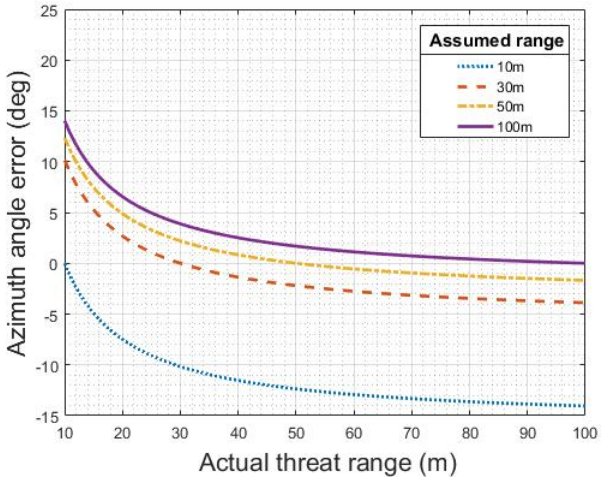
Having detected a threat and extracted its pixel coordinates from a peripheral vision image, one may estimate the azimuth and elevation angles to the threat in a reference frame fixed in the peripheral vision camera. These angles are then transformed to a frame fixed in the central vision camera. Given the relative pose between the peripheral ("p") and central ("c") vision cameras, as defined by the proper rotation matrix $R_{\mathrm{c/p}}$ and the translation vector $T_{\mathrm{c/p}}$, the threat vector in the central vision camera-fixed reference frame is

$$\vec{r}_{\mathrm{t/c}} = \begin{bmatrix} x^{\mathrm{c}} \\ y^{\mathrm{c}} \\ z^{\mathrm{c}} \end{bmatrix} = \begin{bmatrix} R_{\mathrm{c/p}} \mid T_{\mathrm{c/p}} \end{bmatrix} \begin{bmatrix} \frac{u^{\mathrm{p}} r_{\mathrm{a}}}{f^{\mathrm{p}}} \\ \frac{v^{\mathrm{p}} r_{\mathrm{a}}}{f^{\mathrm{p}}} \\ r_{\mathrm{a}} \\ 1 \end{bmatrix} \tag{3.1}$$

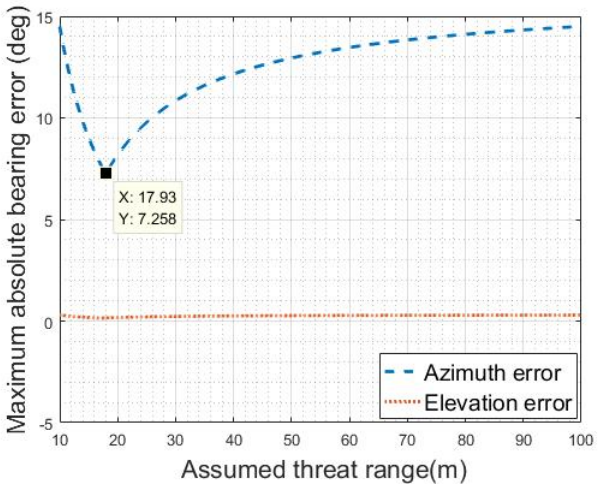
where u^{p} and v^{p} are the pixel coordinates of the threat in the peripheral vision image, f^{p} is the focal length of the peripheral vision camera, and r_{a} is an "assumed range" to the threat. The actual range is unknown, at this point, because the threat has only been imaged by the single, peripheral vision camera. However, a range to the threat is needed in order to compute the bearing angle in the central vision camera-fixed reference frame. There are some methods for estimating range using a monocular camera, such as depth-from-focus/defocus [150, 151, 152]. Because these methods are not robust for distant objects, however, an *assumed* range is adopted instead to obtain the initial cue for the central vision camera, and this assumed range is replaced by a more accurate estimate once the threat has been acquired by both cameras.

3.4.2 Optimal assumed range

Because an assumed range was used, the error between actual range and the assumed range may lead the central vision camera to miss the object to which it has been cued. For example, if the assumed range is 30 m, but the actual range is 90 m, the azimuth angle error would be -3.5°. If the actual range is 10 m, then the azimuth angle error would be 9.7°. If this cueing error is large, the threat may lie outside the central vision camera's FOV and fail to be acquired. To resolve this issue, an assumed range value which minimizes the cueing error is chosen by computing the maximum absolute bearing error for each assumed threat range and finding the assumed range which outputs the minimum absolute bearing error.



(a) Bearing error with example assumed ranges



(b) Maximum absolute bearing error versus assumed range

Figure 3.8: Optimal assumed range estimation

Figure 3.8a shows the azimuth error versus actual range to a threat for a number of assumed ranges varying from 10 m to 100 m. (The maximum range at which the given peripheral vision system can detect a sUAS is roughly 100 m.) When the assumed range is 10 m (blue dotted curve) or 100 m (purple curve), the maximum absolute bearing error is nearly 15°. On the other hand, when the assumed range is 30 m (red dashed curve), the maximum

absolute bearing error is 10°, and less than 4° for threats beyond 20 m. An optimal assumed range that minimizes the maximum absolute bearing angle error is indicated in Figure 3.8b which shows the maximum absolute bearing angle error versus assumed range. As shown in this figure, when the assumed range is 17.93 m, the maximum absolute azimuth angle error is minimum, at roughly 7.3°. The maximum elevation angle error is much smaller than the maximum azimuth error and it is relatively insensitive to the assumed threat range, although larger elevation angle errors may result for threats that are high above the horizon. In any case, an assumed range of 17.93 m is used for the given system.

3.5 Central Vision Subsystem

While the peripheral vision camera system has a large FOV and low resolution, the pan-tilt-zoom (PTZ) central vision camera has a narrower FOV, but high resolution. The central vision camera is intended to obtain detailed imagery of a threat that has been detected by the lower resolution peripheral vision camera. Once the central vision camera has slewed toward the cue provided by the peripheral vision system and acquired the threat within an image, the central vision camera begins to visually servo on the threat, classify the threat, and estimate its position, attitude, and velocity.

3.5.1 Gimbal control for pan and tilt

The reference azimuth and elevation angles computed from peripheral vision imagery are sent to the gimbal, which uses a PID controller to rotate the camera to the desired attitude. The gimbal controller uses Kalman filtered orientation measurements that incorporate data from an IMU fixed to the rotating camera frame and from the gimbal's servo axis encoders.

Because the gimbal's IMU lacks a magnetometer, the gimbal's yaw axis encoder angle is used as the primary azimuthal orientation reference. The readings from the gimbal's IMU are recorded synchronously. The quality of the IMU filtered measurements should be considered in determining the system's ranging accuracy. The gimbal manufacturer claims an angular precision of 0.02° in all axes, which has implications for using the central camera's zoom capability; a narrower FOV (e.g., fully zoomed) requires a smaller error in gimbal orientation to maintain the view of a threat during tracking. For the current system setting, the minimum horizontal FOV of the central vision camera is about ±2.5°. The gimbal precision is fine enough to support tracking at full zoom, though accurate tracking also depends on the disturbance environment and the servo-controller performance. In the implementation described here, the central vision camera adopts its widest FOV when tracking the reference cue to increase the likelihood of acquiring the threat.

3.5.2 Threat classification using deep learning

Object classification can reveal whether a detected object poses a threat and can aid in motion prediction by indicating the performance capabilities of a given threat. A threat taxonomy might include coarse categories of aircraft such as fixed-wing, helicopter, or multirotor, as well as other objects that might be mistaken for threats such as birds, kites, or floating debris. Finer classifications might include specific models of aircraft. In general, a finer taxonomy requires more development and implementation effort and more capable sensing hardware. Here, a simpler, proof-of-concept classification strategy is used.

Several approaches to classification are described in the literature; the ones explored here employ machine learning and computer vision frameworks. Machine learning methods, specifically those involving neural networks (NNs), have been extensively developed for problems

involving the detection and classification of objects in images. Existing deep NNs such as MobileNet [153], YOLOv2 [154] and YOLOv3 [155] have been shown to exhibit high "true positive" rates when trained on sufficiently large and diverse datasets. Both networks can be accelerated to run in real-time using GPU resources, however performance depends strongly on the datasets used to train the networks.

As a baseline test, an implementation of YOLOv3 that was trained on the Common Objects in Context (COCO) dataset was used to generate bounding boxes around aircraft in images obtained using the central vision camera. Some examples are shown in Figure 3.9.

Figure 3.9: Example results using COCO trained YOLOv3

Even with a generically trained NN, the aircraft is correctly detected at closer ranges. In edge cases, however, such as when the aircraft descends below the horizon or appears against a less distinct background, the NN tends to fail. Cases with image noise and lower pixel coverage also result in false negatives. Because the focus here is on proof of concept rather than performance optimization, the YOLOv3 implementation used pre-trained NN weights. Retraining with a dataset that contains a variety of small UAS operating in a rich set of scenes and environmental conditions would likely yield more robust NN detection and classification.

3.5.3 Zoom optimization

To make use of the central vision camera's zoom capability, a control strategy was developed to optimize the camera's FOV when acquiring and tracking a threat. A low zoom value (large FOV) reduces the chance that a disturbance to the camera's orientation would cause the threat to be lost from view. On the other hand, a high zoom value (narrow FOV) enables better threat detection and characterization by providing more pixels on the threat. Here, I consider how the zoom setting can be adjusted to optimize the tradeoff between these two concerns.

Given the range r to the threat, which can be estimated using the algorithm described in the next chapter, the FOV coverage C^{c} of the central vision camera is:

$$C^{\mathrm{c}} = \frac{rW_s^{\mathrm{c}}}{f^{\mathrm{c}}} \tag{3.2}$$

where W_s^{c} is the sensor width and f^{c} is the focal length; see Figure 3.10. To prevent the threat from leaving the FOV, the speed of the threat relative to the central vision FOV and the margins around the threat within the FOV should be considered. For example, if the FOV coverage of the central vision camera is 10 m at the threat range, and the threat width is 2 m, the margin of the FOV is 4 m. In this case, if the threat relative speed is more than 4 m/s, then the threat will escape the static camera's FOV in one second. Therefore, the FOV margin should be larger than the (pixel) distance moved by the threat within a given time period γ_{t} (say, 2 seconds), which depends on the system latency as well as the error in the reference cue and in the measurements (e.g., gimbal orientation).

To formulate an optimization problem, consider that the threat size in the image should be maximized to increase the number of pixels on the threat, but the threat should not escape

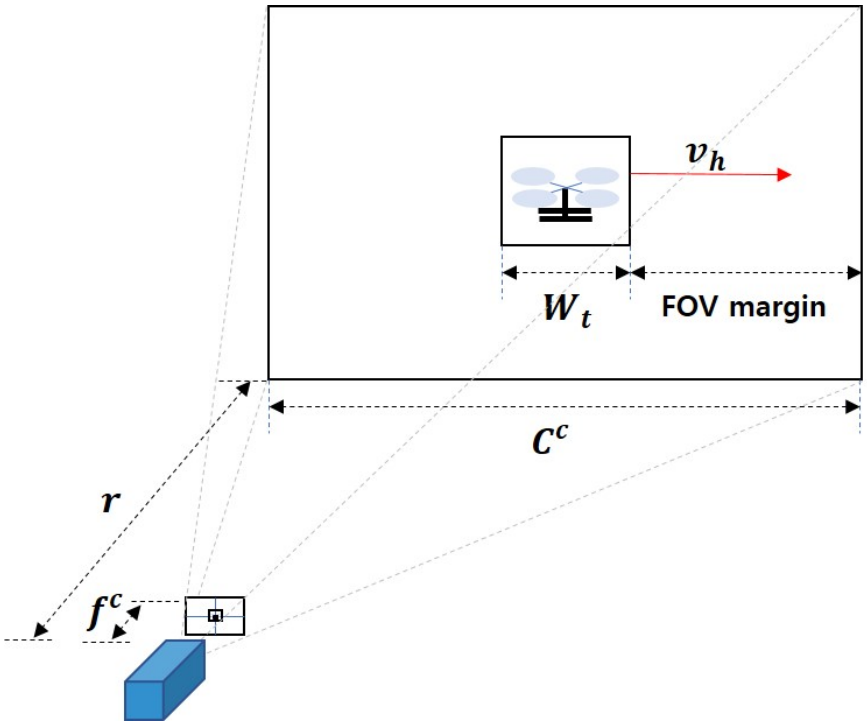


Figure 3.10: Zoom optimization geometry

the camera's FOV in time γ_{t}. These objectives are captured in the following:

$$\begin{aligned} &\text{Maximize} \quad \left(\frac{W_{\mathrm{t}}}{C^{\mathrm{c}}}\right)^2 \\ &\text{s.t.} \quad \gamma_{\mathrm{t}} v_{\mathrm{h}} - \frac{C^{\mathrm{c}} - W_{\mathrm{t}}}{2} = 0 \end{aligned} \tag{3.3}$$

where v_{h} is the relative horizontal speed of the threat and W_{t} is the width of the threat. Formulating the Hamiltonian

$$H = \left(\frac{f^{\mathrm{c}} W_{\mathrm{t}}}{r W_s^{\mathrm{c}}}\right)^2 - \lambda \left(\gamma_{\mathrm{t}} v_{\mathrm{h}} - \frac{r W_s^{\mathrm{c}}}{2 f^{\mathrm{c}}} + \frac{W_{\mathrm{t}}}{2}\right) \tag{3.4}$$

After that, H is differentiated by λ and f^{c}.

$$\frac{\partial H}{\partial \lambda} = \gamma_{\mathrm{t}} v_{\mathrm{h}} - \frac{r W_s^{\mathrm{c}}}{2 f^{\mathrm{c}}} + \frac{W_{\mathrm{t}}}{2} = 0 \tag{3.5}$$

$$\frac{\partial H}{\partial f^{\mathrm{c}}} = \frac{2 f^{\mathrm{c}} W_{\mathrm{t}}^2}{r^2 W_s^{\mathrm{c}2}} - \lambda \left(\frac{r W_s^{\mathrm{c}}}{2 f^{\mathrm{c}2}} \right) = 0 \tag{3.6}$$

Solving (3.6) for the zoom setting gives

$$f^{\mathrm{c}} = r W_s^{\mathrm{c}} \sqrt[3]{\frac{\lambda}{4 W_{\mathrm{t}}^2}} \tag{3.7}$$

Substituting into (3.5) gives

$$\lambda = \frac{4 W_{\mathrm{t}}^2}{(W_{\mathrm{t}} + 2\gamma_{\mathrm{t}} v_{\mathrm{h}})^3} \tag{3.8}$$

Finally, (3.8) is substituted into (3.7) to find the focal length that maximizes the threat size in the FOV while ensuring a sufficient margin to prevent losing the threat:

$$f_{\mathrm{h}}^{\mathrm{c}*} = \frac{r W_s^{\mathrm{c}}}{W_{\mathrm{t}} + 2\gamma_{\mathrm{t}} v_{\mathrm{h}}} \tag{3.9}$$

Note that only the horizontal direction ($W_{\mathrm{t}}, W_s^{\mathrm{c}}$ and v_{h}) is considered in this formulation. The optimal focal length for the vertical direction is computed in the same way:

$$f_{\mathrm{v}}^{\mathrm{c}*} = \frac{r H_s^{\mathrm{c}}}{H_{\mathrm{t}} + 2\gamma_{\mathrm{t}} v_{\mathrm{v}}} \tag{3.10}$$

where H_s^{c} is the sensor height of the central vision camera and H_{t} is the height of the threat. The optimal focal length is then taken as the minimum value of the focal lengths for the horizontal and vertical direction:

$$f^{\mathrm{c}*} = \min(f_{\mathrm{h}}^{\mathrm{c}*}, f_{\mathrm{v}}^{\mathrm{c}*}) \tag{3.11}$$

3.5.4 Monocular vision-based attitude estimation

The central vision camera of the peripheral-central vision system can estimate the attitude of the threat aircraft using an algorithm presented in [16]. Because the central vision camera provides a high-resolution image of the threat, the image can be searched for five distinguished points on the threat aircraft: the nose, the two wing tips, and the two tips of the horizontal stabilizer. (The geometry of the threat aircraft is assumed to be known, or inferred from the high-resolution imagery.) In order to detect these feature points, the image is first cropped to the region of interest and the edge of the threat in the image is extracted using Canny edge detection [156, 157]. Next, the smallest pentagon which includes the extracted edge is found; the vertices of this pentagon are adopted as the relevant feature points of the threat aircraft.

These feature points are then used as inputs to the POSIT algorithm, a commonly used 3D pose estimation method that solves for the rotation matrix R_{TC} describing the threat aircraft's attitude with respect to the central vision camera frame. Once R_{TC} is obtained, a rotation matrix R_{TG} that maps free vectors from the global reference frame to the threat frame is computed:

$$R_{TG} = R_{TC} R_{CG} \tag{3.12}$$

where R_{CG} is the rotation matrix that maps free vectors from the global reference frame to the central vision frame. The matrix R_{TG} represents the attitude of the threat aircraft with respect to the global reference frame. As implied above, this approach to threat aircraft attitude estimation does assume that (1) the five distinguished points of the threat aircraft are clearly visible in the image and (ii) the geometry of the threat aircraft is known.

In [16], a camera imagery of a fixed-wing unmanned aircraft obtained during 16 flight tests is used, to check the performance of the attitude estimation algorithm. Table 3.2 shows the

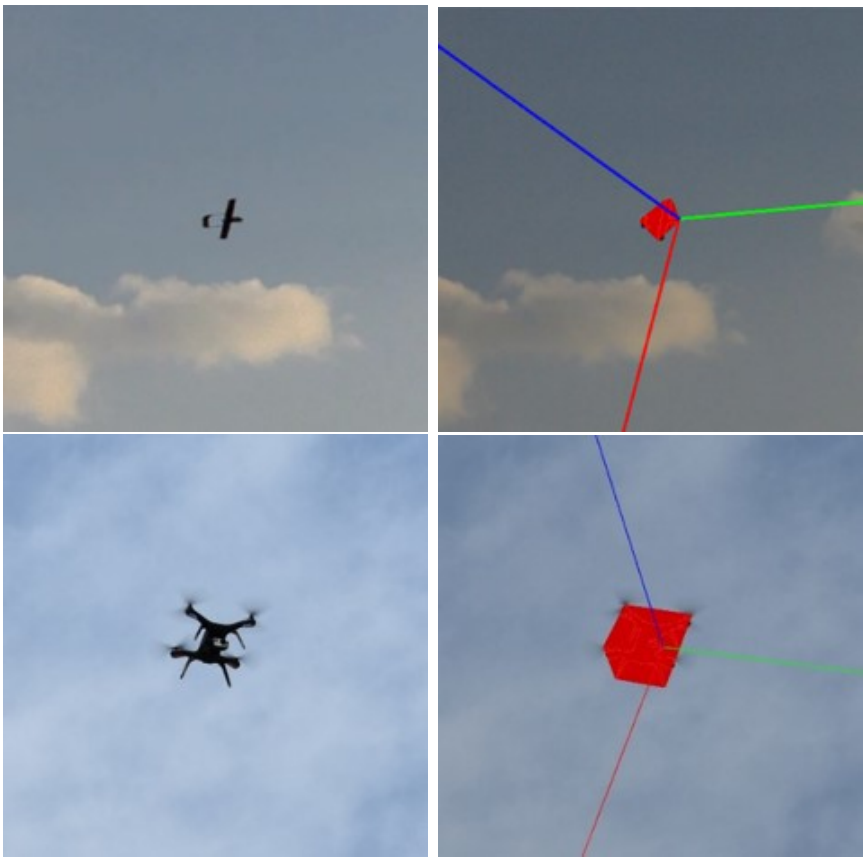


Figure 3.11: Visual pose estimation

average error magnitude between the attitude estimates from the estimation algorithm and direct measurements obtained onboard the threat aircraft. As indicated by these experimental data, the estimation scheme gives reasonably accurate estimates of the threat aircraft attitude. A primary source of the error reported in Table 3.2 is the error in the measured attitude of the central vision camera, which is used to construct the rotation matrix R_{CG} in (3.12).

Table 3.2: Average error of pose estimation experiments

Value	Roll	Pitch	Yaw
Average error magnitude (°)	3.4	2.7	11.6

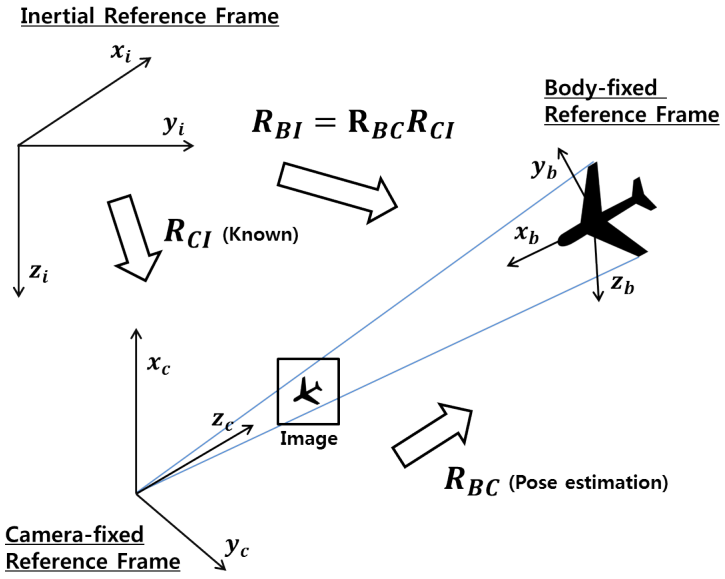


Figure 3.12: Pose estimation geometry

3.6 Summary

This chapter describes a peripheral-central vision (PCV) system to detect, localize, and classify threats. The peripheral vision camera initially detects the threat using optical flow, after image undistorton and stabilization. The threat bearing in the peripheral vision camera-fixed reference frame is then transformed into the central vision camera-fixed reference frame with an optimized assumed range. The central vision camera is then cued to slew toward the threat. Once the threat is acquired by the second camera, the range is estimated using a heterogeneous stereo vision algorithm and the assumed range is replaced with this estimate. The refined range estimate is then used to compute the optimal zoom value. The more detailed image of the threat available from the central vision camera allows one to classify the threat using a deep neural network and also to estimate the pose of the threat. In the next chapter, a dataset is generated using the developed system and algorithms in this chapter, and the threat in the dataset is localized using the heterogeneous stereo vision algorithm.

Chapter 4

Heterogeneous Stereo Vision using PCV System

4.1 Introduction

In the previous chapter, the system algorithms for the PCV system, a heterogeneous stereo vision system comprising the peripheral vision camera and the central vision camera capable of pan-tilt-zoom (PTZ) operation, has been introduced. The peripheral vision camera provides continuous visual coverage of the environment for threat detection, although with relatively low and non-uniform resolution. The central vision camera complements the peripheral vision camera by providing a high-resolution image when cued to observe a threat. The pair of cameras affords the opportunity to use stereo vision for estimating the 3D position of the threat while the high-resolution image of the central vision camera allows monocular vision-based estimation of the threat aircraft attitude. Thus, one may obtain the position of the threat aircraft using the PCV system. In this chapter, the heterogeneous stereo vision algorithm for the PCV system is introduced, and the localization algorithm is implemented using the PCV system dataset. The error of the estimation and the sources of error are discussed, and the system performance is analyzed as well. The dataset generation was conducted with the assistance of collaborators, and my main contribution is on the development of the system algorithms and the software which are described in this chapter. This work is

presented in [17, 18].

4.2 Heterogenous Stereo Vision Algorithm

A camera image is generated when points on 3D objects are projected onto the camera image sensor. As shown in Figure 4.1, the points A, B, and C are projected onto the same image point of the right image sensor. The actual position of the point represented in the image is therefore unknowable, without additional knowledge about the range to the object. This issue is called "range ambiguity." If an additional camera, viewing the same scene from a different perspective, is given, the range ambiguity can be resolved by computing an intersection of two lines of sight from the two cameras. The point B in Figure 4.1 is an example. Conventional stereo vision algorithms normally assume two identical cameras with parallel camera boresight axes. This configuration allows one to determine the range to an object using a simple equation. The formulation must be modified for a heterogeneous stereo vision system, however, because of high distortion in the peripheral vision image and the non-parallel camera boresight axes. An omnidirectional camera calibration procedure published by Scaramuzza *et al.* [143] addresses this issue, enabling conversion from 2D image points on the large FOV camera image to corresponding 3D vectors. The 3D vector pointing toward the threat from a given camera is referred to as a *threat vector*. The 3D position of a threat can be estimated by computing the intersection of the threat vectors from the two cameras.

Figure 4.2 depicts the threat vectors $\vec{r}_{\mathrm{t/p}}$ and $\vec{r}_{\mathrm{t/c}}$ for the peripheral and central vision cameras, respectively. The intersection $\vec{r}_{\mathrm{t/g}}$ of the threat in the global reference frame is then

$$\vec{r}_{\mathrm{t/g}} = \vec{r}_{\mathrm{p/g}} + \frac{\left\|\vec{r}_{\mathrm{t/c}} \times \vec{r}_{\mathrm{p/c}}\right\|}{\left\|\vec{r}_{\mathrm{t/c}} \times \vec{r}_{\mathrm{t/p}}\right\|} \vec{r}_{\mathrm{t/p}} \tag{4.1}$$

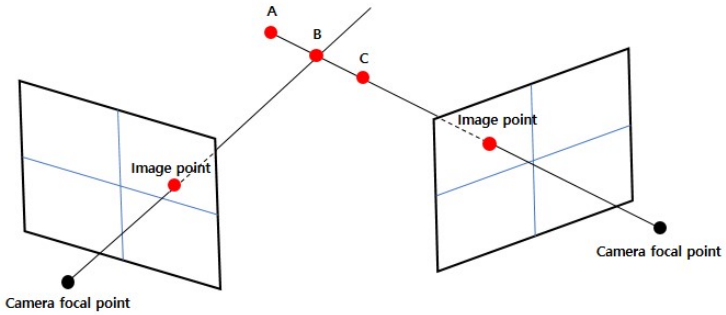


Figure 4.1: Stereo vision

where $\vec{r}_{\mathrm{p/g}}$ and $\vec{r}_{\mathrm{p/c}}$ represent the optical center position of the peripheral vision camera with respect to the global frame and the central vision camera frame, respectively. (All vectors in (4.1) are assumed to be expressed in the global frame.)

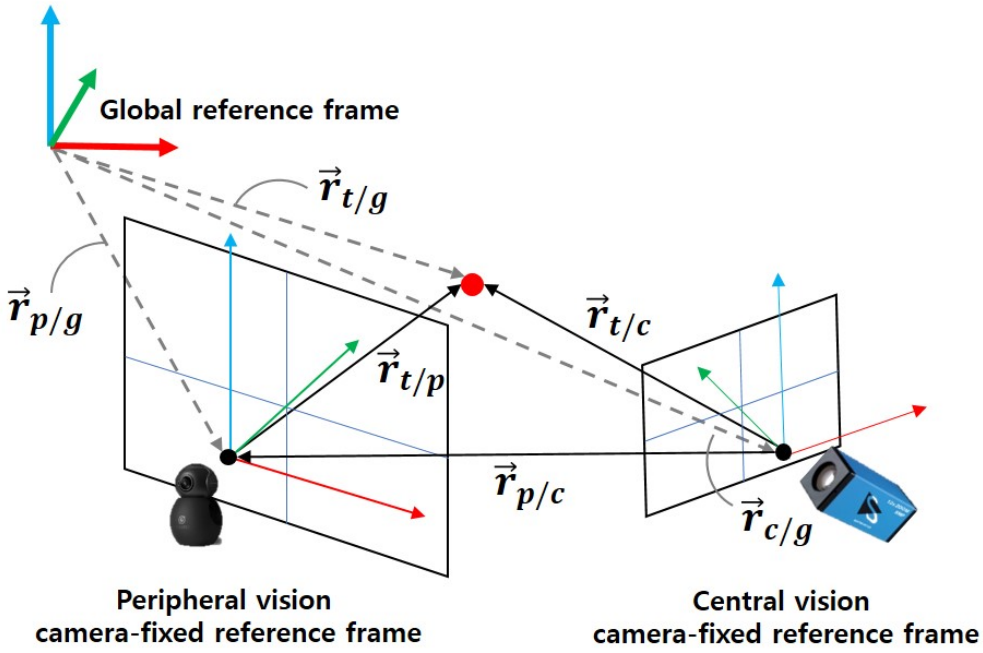


Figure 4.2: Geometry of threat localization

If two 3D threat vectors intersect, the threat position can be easily estimated using (4.1). Several sources of error ensure that two threat vectors will rarely intersect. Following is an incompletel list:

- Lens distortion: the effect of light refraction through the camera lens can be partially corrected through camera calibration, but cannot be removed entirely.

- Camera pose error: to compute the intersection point of threat vectors, the threat vectors should be expressed in a common frame (e.g., the global frame). The position and orientation of each camera frame in the global frame are obtained from an IMU and a GPS sensor attached to each camera, but these measurements are imperfect.

- Feature correspondence: the point on the threat that determines the threat vector for one camera, a point which is determined using a feature detection algorithm (e.g. color detection, corner detector, etc.), may not correspond to the same detected feature point in the other camera image.

Approaches have been suggested to address the non-intersection of threat vectors [158, 159, 160].

Mid-point method

The mid-point method computes the shortest line connecting two 3D vectors and uses a middle point of the connecting line as an intersection as shown in figure 4.3. The mid-point can be computed by finding a point that has the shortest distance from two threat vectors [161, 162]. Let's define an arbitrary point P, and the distance d_p to $\vec{r}_{t/p}$ from P is computed as:

$$d_p = (I - \vec{r}_{t/p}\vec{r}_{t/p}^{\,T})(P - \vec{r}_{p/g}) \tag{4.2}$$

The distance between P and $\vec{r}_{t/c}$ is derived in the same way:

$$d_c = (I - \vec{r}_{t/c}\vec{r}_{t/c}^{\,T})(P - \vec{r}_{c/g}) \tag{4.3}$$

The closest mid-point P^* is then determined as a point that minimizes d_p and d_c.

$$P^* = \arg\min_{P} \left(\|d_p\|^2 + \|d_c\|^2 \right) \tag{4.4}$$

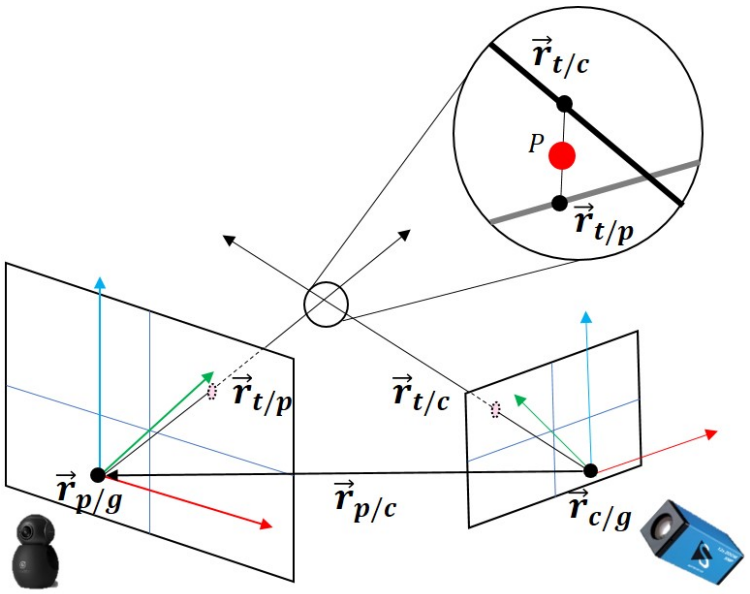


Figure 4.3: Geometry of mid-point approach

This method is relatively easy to implement and fast to compute. However, if the error in the threat vectors is large, the localization error in the stereo ranging method is also large. Moreover, the mid-point method occasionally computes a (non-physical) negative range [163].

Optimal method

An alternative, known as the optimal method, corrects the two 3D vectors based on the epipolar constraint that is satisfied when two 3D vectors are on a common plane (the epipolar plane). The two corrected threat vectors necessarily intersect.

The epipolar constraint is

$$< \vec{r}_{t/p}, E\vec{r}_{t/c} >= 0 \tag{4.5}$$

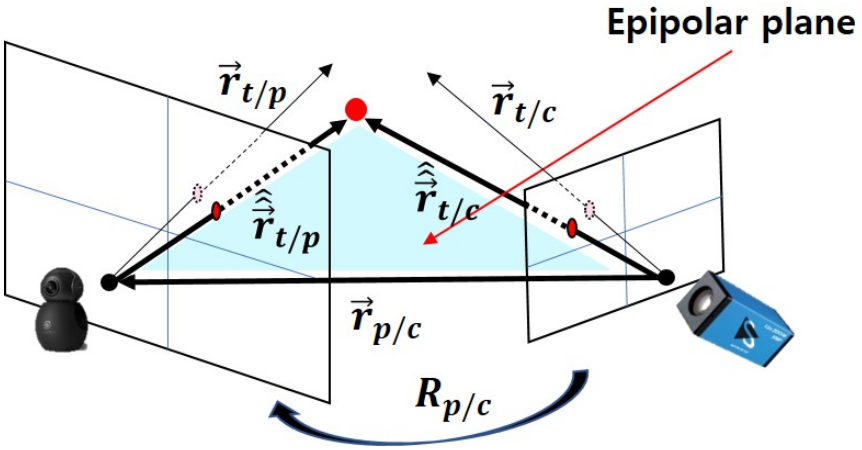


Figure 4.4: Geometry of optimal method

where the bracket indicates the inner product of two vectors, and E is the essential matrix ($E = [\vec{r}_{p/c}]_\times R_{p/c}$). Here, $[\vec{r}_{p/c}]_\times$ represents the skew-symmetric matrix of $\vec{r}_{p/c}$. Let the corrected threat vectors as $\hat{\vec{r}}_{t/p} = \vec{r}_{t/p} - \Delta\vec{r}_{t/p}$, and $\hat{\vec{r}}_{t/c} = \vec{r}_{t/c} - \Delta\vec{r}_{t/c}$, and these two corrected vectors need to be satisfied the epipolar constraint while minimizing $\Delta\vec{r}_{t/p}$ and $\Delta\vec{r}_{t/c}$, which indicate the correction for threat vectors. An optimization problem is thus defined as below.

$$\begin{aligned} &\text{Minimize} \quad \|\Delta\vec{r}_{t/p}\|^2 + \|\Delta\vec{r}_{t/c}\|^2 \\ &\text{s.t} \quad < (\vec{r}_{t/p} - \Delta\vec{r}_{t/p})^T, E(\vec{r}_{t/c} - \Delta\vec{r}_{t/c}) >= 0 \end{aligned} \tag{4.6}$$

In order to solve this problem, a Hamiltonian function is defined.

$$H = \|\Delta\vec{r}_{t/p}\|^2 + \|\Delta\vec{r}_{t/c}\|^2 + \lambda(< (\vec{r}_{t/p} - \Delta\vec{r}_{t/p})^T, E(\vec{r}_{t/c} - \Delta\vec{r}_{t/c}) >) \tag{4.7}$$

This can be solved in the same way as (3.4) for $\Delta\vec{r}_{t/p}$ and $\Delta\vec{r}_{t/c}$.

$$\begin{aligned}\Delta\vec{r}_{t/p} &= \frac{<\vec{r}_{t/p}, E\vec{r}_{t/c}> E\vec{r}_{t/c}}{<E^T\vec{r}_{t/p}, E^T\vec{r}_{t/p}> - <E\vec{r}_{t/c}, E\vec{r}_{t/c}>} \\ \Delta\vec{r}_{t/c} &= \frac{<\vec{r}_{t/p}, E\vec{r}_{t/c}> E^T\vec{r}_{t/p}}{<E^T\vec{r}_{t/p}, E^T\vec{r}_{t/p}> - <E\vec{r}_{t/c}, E\vec{r}_{t/c}>}\end{aligned} \tag{4.8}$$

For more exact vector correction, the iterative higher order correction method is used in a similar way. In this case as well, the corrections should be minimized while satisfying the epipolar constraint for the final corrected vectors, $\hat{\vec{r}}_{t/p} = \vec{r}_{t/p} - \Delta\vec{r}_{t/p}$ and $\hat{\vec{r}}_{t/c} = \vec{r}_{t/c} - \Delta\vec{r}_{t/c}$

$$\begin{aligned}&\text{Minimize} \quad \|\Delta\vec{r}_{t/p} + \Delta\hat{\vec{r}}_{t/p}\|^2 + \|\Delta\vec{r}_{t/c} + \Delta\hat{\vec{r}}_{t/c}\|^2 \\ &\text{s.t} \quad <\hat{\vec{r}}_{t/p} - \Delta\hat{\vec{r}}_{t/p}, E(\hat{\vec{r}}_{t/c} - \Delta\hat{\vec{r}}_{t/c})>= 0\end{aligned} \tag{4.9}$$

In the same way, a Hamiltonian function is defined and defined and solved until the $\Delta\hat{\vec{r}}_{t/p}$ and $\Delta\hat{\vec{r}}_{t/c}$ are converges to ϵ which is a convergence threshold.

$$\begin{aligned}\hat{\hat{\vec{r}}}_{t/p} &= \vec{r}_{t/p} - \frac{\big(<\hat{\vec{r}}_{t/p}, E\hat{\vec{r}}_{t/c}> + <E\hat{\vec{r}}_{t/c}, \tilde{p}_l> + <E^T\hat{\vec{r}}_{t/p}, \tilde{p}_r>\big)E\hat{\vec{r}}_{t/c}}{<E\hat{\vec{r}}_{t/c}, E\hat{\vec{r}}_{t/c}> + <E^T\hat{\vec{r}}_{t/p}, E^T\hat{\vec{r}}_{t/p}>} \\ \hat{\hat{\vec{r}}}_{t/c} &= \vec{r}_{t/c} - \frac{\big(<\hat{\vec{r}}_{t/p}, E\hat{\vec{r}}_{t/c}> + <E\hat{\vec{r}}_{t/c}, \tilde{p}_l> + <E^T\hat{\vec{r}}_{t/p}, \tilde{p}_r>\big)E^T\hat{\vec{r}}_{t/p}}{<E\hat{\vec{r}}_{t/c}, E\hat{\vec{r}}_{t/c}> + <E^T\hat{\vec{r}}_{t/p}, E^T\hat{\vec{r}}_{t/p}>}\end{aligned} \tag{4.10}$$

The threat vectors $\vec{r}_{t/p}$ and $\vec{r}_{t/c}$ are corrected to corrected threat vectors, $\hat{\hat{\vec{r}}}_{t/p}$ and $\hat{\hat{\vec{r}}}_{t/c}$, and these vectors finally intersect. The intersection point is thus able to be estimated using (4.1). For the implementation in this chapter, the optimal method is used to correct the threat vectors since the optimal method gives less localization error than the mid-point method.

4.3 System Architecture and Data Acquisition

The algorithms and hardware are finally integrated into a PCV system to generate a dataset. In this section, the architecture of the PCV system and the data generation setup are illustrated. The dataset is generated using ground-based and air-based PCV systems with various types of mock threats. The pros and cons of each type of the PCV system are described here.

Figure 4.5 depicts the experimental setup for the dataset in which a mock threat aircraft streams its position to a ground station, providing ground truth to assess the localization strategy. For the PCV system, the GPS position of the system and the imagery from the two cameras are acquired and processed on an Nvidia TX2. The hardware and software setup is based on the ROS framework. After processing the data, the pointing cue is sent to the gimbal for the central vision camera. All data are stored in the rosbag file format.

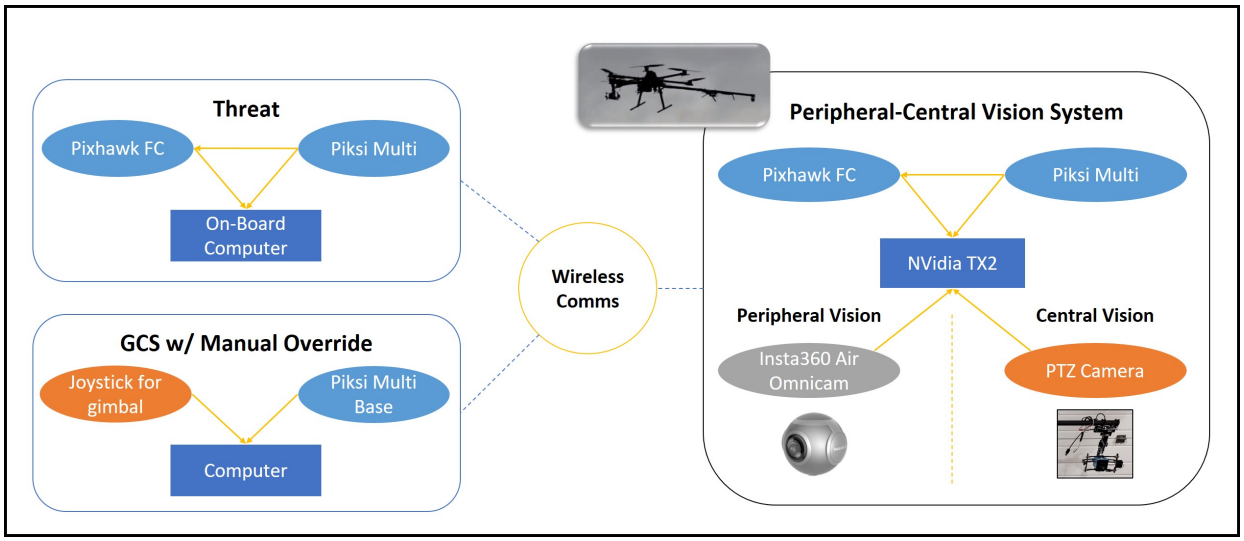


Figure 4.5: Setup for generating image datasets

Experiments were performed using two PCV configurations – ground-based and air-based – and various types of mock threat. In a given experiment, a threat maneuvers within

the detectable range – 100 m for the current system specifications – and the PCV system observes and records the data on the ground or in the air while hovering and maneuvering. Mock threats included: a human, a quadcopter, a fixed-wing UAV, and a manned aircraft. Table 4.1 shows the details of the dataset that was gathered.

Table 4.1: PCV system dataset

Host Type	Threat Type	Time Duration (min)
Single ground-based	Human	20
	Quadcopter	4
Single air-based	Multi-rotor	20
	Fixed-wing UAV	19
Double air-based	Quadcopter	30

Advantages of the ground-based PCV system include unlimited power, low position error for the camera system, and clear, steady imagery. For these and other reasons, the ground-based PCV system is easier to use and its data are easier to process, since there is no need to stabilize the imagery. It is not surprising that more than 70% of commercial Counter-UAS systems are ground-based [12].

The primary disadvantage of a ground-based PCV system is its inability to vary the camera perspective. For some threats in the counter-UAS application, the ability to induce particular relative motions by maneuvering the host could aid detectability and localization accuracy. Moreover, a threat detection and tracking system developed for use on a small unmanned aircraft can serve the dual purpose of ABDAA. One of the virtues of the PCV system is the low SWAP-C that enables it to easily integrated into a small UAS. The performance of the air-based system, on the other hand, is affected by GPS accuracy, image quality from the moving platform, and battery life.

One issue for both of single PCV system configurations that were tested is the short baseline

(2 m) between the central vision camera and the peripheral vision camera. Because the PCV system estimates the threat position based on triangulation, as shown in Section 4.2, a shorter baseline can increase the error in the threat vector error; this issue is discussed further in Section 4.5. To explore the opportunities afforded by larger baselines, a second PCV system is constructed and used in parallel.

Figure 4.6: Example dataset images: synchronous peripheral (left) and central (right) vision images

For the dataset collected using the ground-based system, 2 types of threat were used: a human and a quadcopter. For the air-based system, 2 types of threat were used: a quadcopter and a fixed-wing UAV. While the central vision camera is physically stabilized by the gimbal, the peripheral vision camera was affected by vibration when obtaining data from the air-based system. The peripheral imagery was software-stabilized as described in Section 3.3.1. The dataset includes imagery of the threat from two cameras, the host's GPS location, the threat's GPS location, and the gimbal orientation for the central vision camera. The dataset is used to test the detection algorithms and the localization algorithm. The figure 4.6 shows the example imagery of the dataset.

4.4 Threat Localization Results

The heterogeneous stereo vision algorithm is implemented in ROS-based software by re-playing the rosbag dataset mentioned in the previous section. Figure 4.7 shows a screenshot of the ROS-based threat localization software. Once the threat is detected in both camera images, threat vectors from the two cameras are generated (the rays originating from the two camera frames in Figure 4.7), and the intersection point of two threat vectors is computed to estimate the threat's position. The threat in the dataset has its own GPS, for ground truth, and this independent position measurement is compared with the estimated position to check the localization performance.

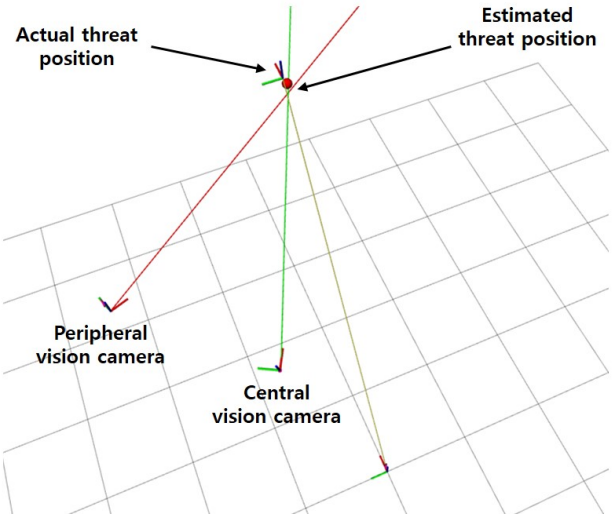
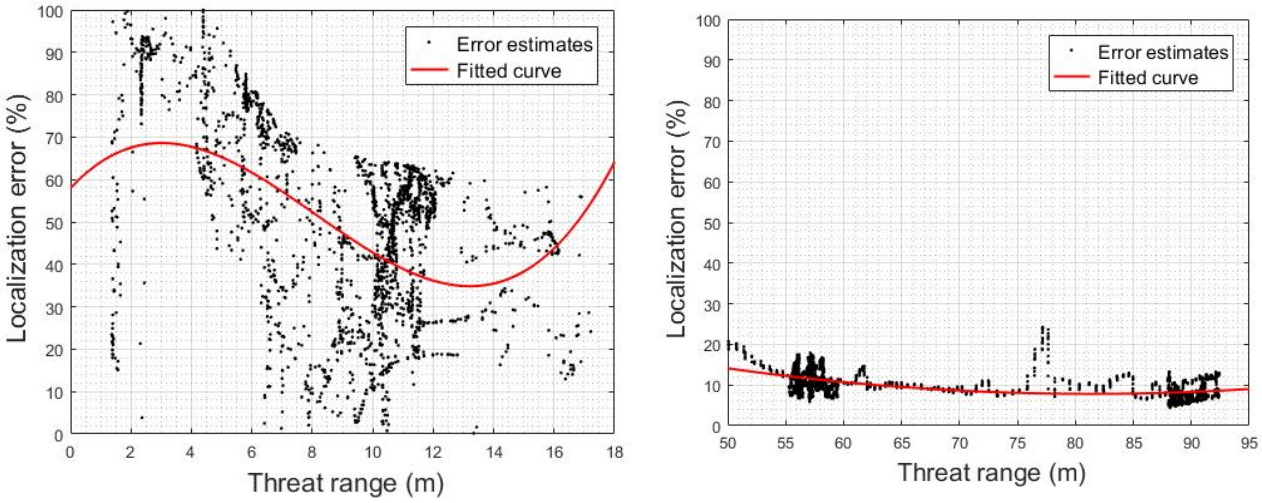


Figure 4.7: Threat localization

Figure 4.8a depicts the localization error (an absolute distance error between the ground truth threat position and the estimated threat position) versus the range of the threat using a single PCV system. The red curve in the figure illustrates a fit to the localization error. As shown in the figure, the fitted localization error is around 50% and the error variation is quite large.



(a) Localization error using single PCV system

(b) Localization error using two PCV systems

Figure 4.8: Threat localization error of the dataset

One major source of error in the threat vectors is the feature extraction error. The feature extraction algorithms applied to both camera images are not guaranteed to detect the threat. If one of the cameras detects the threat incorrectly, the localization error is large, as reflected in Figure 4.8a. Another source of localization error is the relatively short (2 m) camera baseline. (This issue is explored further in the following section.) The localization error for the given PCV system increases rapidly with threat range. For comparison, Figure 4.8b shows the localization error with two PCV systems, where the camera baseline is 90 m. In this figure, a large error variation is still observed due to feature extraction error, but because of the greater range to the threat, the fitted localization error is reduced to around 10%, much less than for the single PCV system. Also, the localization error does not increase as rapidly with increasing range compared with the single PCV system. These results affirm the well-known fact that a longer baseline improves triangulation accuracy. Ongoing work involves the coordinated use of multiple ground- and air-based PCV systems for counter-UAS and cooperative ABDAA applications. As shown in the results, system parameters such as the camera baseline affect localization performance.

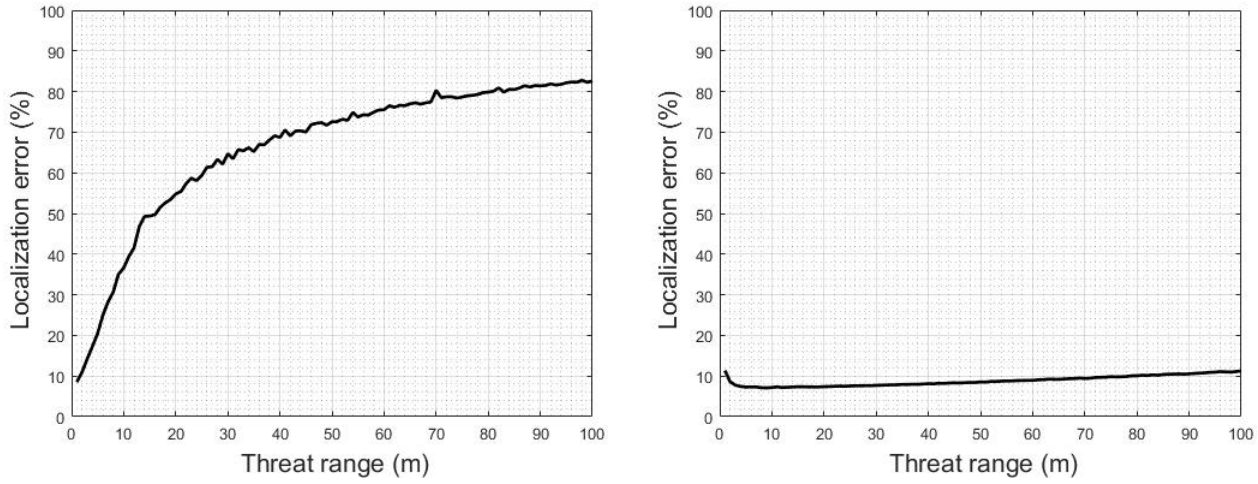
4.5 System Analysis

As mentioned in the previous section, localization error is affected by several system parameters. In this section, parameter effects on localization error is analyzed using Monte Carlo simulation results.

4.5.1 Localization error analysis

Numerous issues contribute to threat localization error including feature extraction error, camera calibration, lens distortion, camera resolution, inertial sensor accuracy, and gimbal sensor accuracy as well as the length of the baseline between the two cameras. The baseline of the single PCV system is 2 m, and the minimum pixel coverage needed to detect a threat using the optical flow algorithm with peripheral vision imagery is about 25 px (5 px wide $\times$ 5 px high). The minimum pixel coverage for detection using Yolov3 with central vision imagery is about 900 px (30 px wide $\times$ 30 px high). Even after undistorting peripheral camera imagery, the low relative pixel density near the edges of the image can prevent the detection of threats in these regions. The problem is compounded because the undistortion function automatically interpolates pixel values, backfilling gaps in regions with sparse pixel coverage. All these sources of error aggregate, resulting in an erroneous threat vector from which the localization error may be assessed.

To investigate measurement uncertainty, Monte Carlo simulations were implemented using the system parameters shown in Table 3.1. Zero-mean Gaussian noise is superimposed to the threat vectors in the horizontal direction (camera sensor width direction) and the vertical direction (camera sensor height direction) of the camera-fixed reference frame, corresponding to a 30 px standard deviation for the peripheral vision image and a 3 px standard deviation for the central vision image, under the assumption that there is no feature extraction error.



(a) Localization error using 2 m baseline (b) Localization error using 100 m baseline

Figure 4.9: Localization error versus threat range from Monte Carlo simulation

This synthetic error was tuned empirically to closely match that of the actual hardware. Localization error estimates with the random Gaussian noise are obtained from 10,000 samples and averaged. The trends observed in these 10,000 samples were evident in the first 1,000 samples, suggesting that 10,000 samples are sufficient. Multiple of these Monte Carlo simulations were conducted by varying the threat range. Figure 4.9 shows the localization error using a 2 m baseline and a 100 m baseline, as obtained from Monte Carlo simulations. The localization error increases with distance to the threat, similar to the experimental results in Figure 4.8. As shown in Figure 4.8 and Figure 4.9, the localization error can be reduced by increasing the baseline of the PCV system. In the following section, a system performance analysis is conducted to see the relationship between system parameters, such as camera position and resolution, and localization accuracy.

4.5.2 System performance analysis

Localization error is affected by a number of system parameters, including the camera baseline, resolution, etc. The quality of the peripheral vision camera, in particular, dominates the performance of the PCV system described here. In order to compare the PCV system performance using different system parameters, the localization error is computed from Monte Carlo simulations using various values for camera location and peripheral vision camera resolution.

Camera location. A formula is given in [136] relating triangulation-based localization error and sensor error:

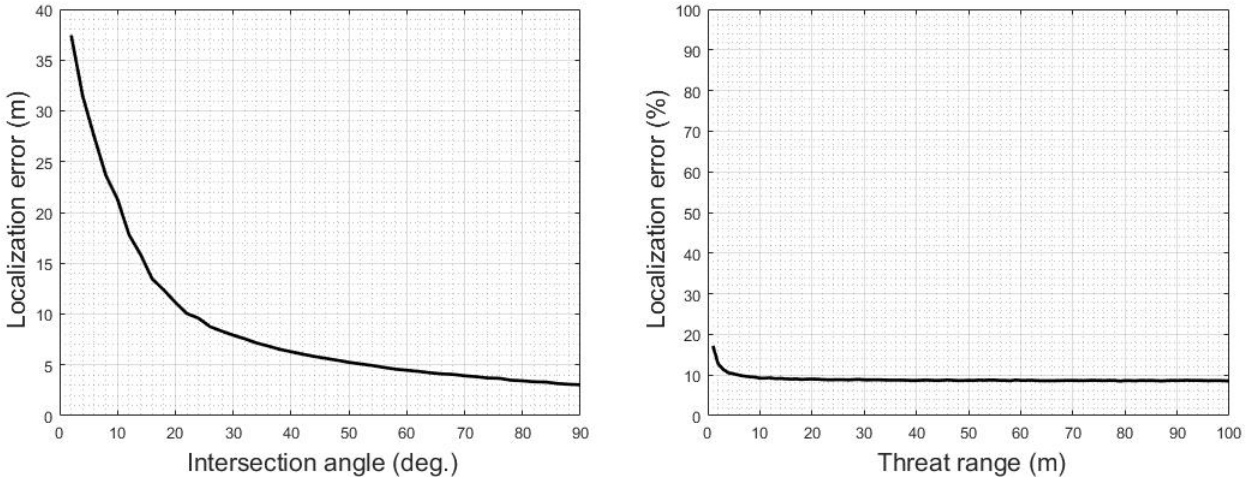
$$\delta_r = \frac{\left\|\vec{r}_{\mathrm{t/p}}\right\| \left\|\vec{r}_{\mathrm{t/c}}\right\|}{\sin\theta}\delta_s \tag{4.11}$$

where δ_r is the magnitude of the localization error, δ_s is the magnitude of threat vector error, and θ is the intersection angle between two threat vectors

$$\theta = \cos^{-1}\left(\frac{\vec{r}_{\mathrm{t/p}} \cdot \vec{r}_{\mathrm{t/c}}}{\left\|\vec{r}_{\mathrm{t/p}}\right\| \left\|\vec{r}_{\mathrm{t/c}}\right\|}\right)$$

Equation (4.11) implies that a shorter range to the threat and a nearly orthogonal viewing angle ensure the smallest localization error. Figure 4.10a shows the localization error obtained from the Monte Carlo simulations in which the threat range is fixed at 50 m but the intersection angle θ is varied. As shown in the plots, localization error decreases with larger intersection angles, which helps to explain why the longer baseline improves localization accuracy in Figures 4.8 and 4.9. Figure 4.10b shows the localization error with a fixed intersection angle $\theta = 90^\circ$ and varying range to the threat. The localization error increases with increasing range, at only about 6 cm per meter. Analysis indicates that both threat range and intersection angle θ are important determinants of localization accuracy, but θ appears to play the more important role. Accuracy degrades quickly for threat vector intersection

angles less than about 30°.



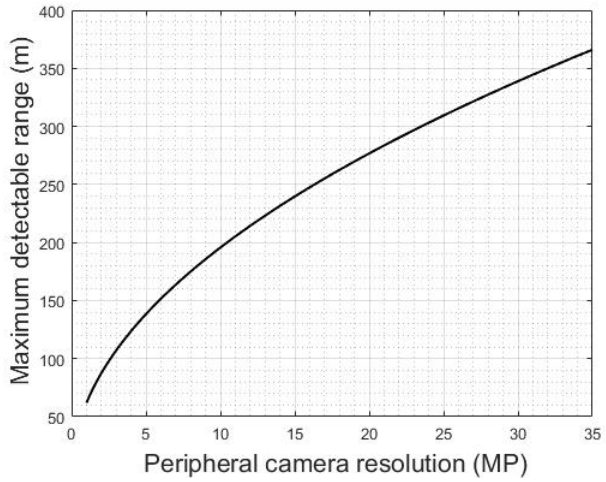
(a) Localization error with constant threat range (50 m)

(b) Localization error with constant intersection angle (90°)

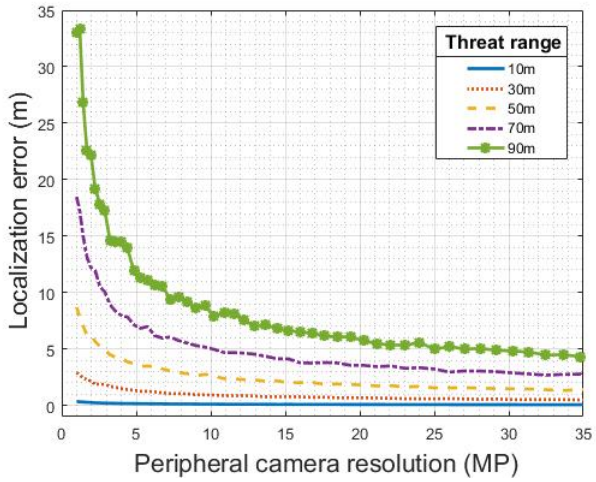
Figure 4.10: Localization error using Monte Carlo simulation

Peripheral vision camera resolution. The peripheral vision camera has a lower resolution than the central vision camera because of the wide FOV. (See figure 4.6.) The PCV system localization performance is thus more affected by the peripheral vision camera resolution than that of the central vision camera. Recalling that localization from a single image is impossible, because of the range ambiguity, it is intuitive that system performance would be limited by the lower resolution camera.

Note that the absolute localization error indicated in these performance analysis results is partly a consequence of system architecture choices unrelated to optical performance, such as the need for a low SWaP-C system that is compatible with a ROS computing framework. The emphasis here is on the relative effects of various system parameters, rather than on absolute performance, which could easily be improved by using higher quality optics.



(a) Maximum detectable range along with the peripheral vision camera resolution



(b) Localization error along with the peripheral vision camera resolution

Figure 4.11: The effect of the peripheral vision camera resolution

The maximum detectable range of the system is estimated as follows:

$$R_{\max} = \frac{W_t \gamma^{\mathrm{p}} f_{\mathrm{p}}}{W_s \gamma_{\mathrm{p}}} \tag{4.12}$$

where W_t and W_s are the width of the threat and the camera sensor, respectively, f_{p} is the focal length of the camera, γ^{p} is the horizontal resolution of the camera (in px), and γ_{p} is the minimum number of pixels required by the detection algorithm. Therefore, the maximum detectable range of the current system for a 1 m-size threat is 100 m. Figures 4.11a and 4.11b illustrate the maximum detectable range and range error, respectively, along with the peripheral vision camera resolution. As shown in figure 4.11a, the maximum detectable range increases almost linearly with resolution, which means a higher resolution peripheral vision camera enables detection of more distant threats. Each plot in figure 4.11b indicates the localization error for several example threat ranges. As shown in the figure, the localization error decreases with increasing camera resolution, but a "knee" is observed which indicates a diminishing return beyond roughly 5 MP. Considering that the resolution of the current

peripheral vision camera is 2.26 MP, increasing the peripheral vision camera's resolution 5 MP would be a reasonable next step for improving system performance.

4.6 Summary

In this chapter, the heterogeneous stereo vision algorithm for the PCV system is explained. In order to assess the threat localization performance, an experimental dataset was generated using a variety of mock threats. Results show that the localization accuracy is quite limited using the current low-cost cameras in the given configuration. Analysis of localization error for the experimental dataset obtained using a single PCV system revealed a large localization error with large variation. The large error variation is due to error in the threat vectors, for which the major contributor is feature extraction error. I also observed, however, that localization error decreases substantially with a longer baseline as obtained in experiments using two PCV systems. Monte Carlo simulations allowed further investigation of the effect of system parameters on the localization error. The results indicate that for threat vector intersection angles smaller than about 30°, localization error increases rapidly. The short baseline configuration of a single PCV system places a fundamental limit on stereo ranging accuracy, but multiple PCV systems operating in concert can provide much more accurate range estimates. This accuracy is also influenced, however, by camera quality and performance of the feature extraction algorithm that helps to define the threat vector.

Chapter 5

Fixed-wing Aircraft Path Prediction Based on Attitude Data

5.1 Introduction

As the personal and professional uses of small unmanned aircraft systems (sUAS) continue to expand, and the number of these aircraft increases, concern is rising about the possibility of malicious mis-use and mid-air collisions. Concern about collisions between sUAS and manned aircraft is supported by recent unauthorized flights of sUAS over major airports that have resulted in airport closures and travel disruptions [9]. Long before concerns about sUAS arose, however, the aviation industry was addressing the risk of mid-air collisions between manned aircraft.

A path prediction method that uses an estimate of the pose of a threat aircraft, focusing on fixed-wing airplanes is proposed in this chapter. The idea that a threat aircraft's pose may be available to inform path prediction is supported by the increasing sophistication of computer vision based sensing systems, as mentioned above. With the development of lightweight, low-power camera technologies, a host aircraft can easily acquire high resolution images from which the attitude of a threat aircraft can be obtained using computer vision and machine learning [16, 17, 164, 165, 166, 167, 168]. Active sensing, such as radar, has shown good performance for position estimation of threat sUAS [27, 169, 170, 171]. Therefore, the

position and attitude of a threat sUAS can be extracted from an on-board sUAS sensing system of a host aircraft. A new model-based path prediction algorithm for a small, fixed-wing, unmanned aircraft which incorporates an estimate of the threat's attitude and its position is introduced.

Here, an algorithm to estimate two types of specific power of a threat aircraft which is the conventional specific excess power and an energy-conserving specific power that accounts for changes in path direction is introduced. These power terms are inferred from position and attitude data obtained from visual imagery, for example, and are then used in a particle dynamic model of the threat aircraft's motion for path prediction. The proposed prediction algorithm and an amended algorithm which also estimates wind velocity are compared with a more conventional position-based prediction method. This work is published in [19].

5.2 Path Prediction Algorithm

Consider a fixed-wing unmanned aircraft modeled as a point mass and let $\vec{r} = [x, y, z]^T$ denote the position of the aircraft in a global frame, which is assumed to be an inertial reference frame. The position variables are assumed to be extracted from visual imagery using the peripheral-central vision system introduced in the previous chapters. Also define the inertial velocity $\vec{v} = [\dot{x}, \dot{y}, \dot{z}]^T$ and acceleration $\vec{a} = [\ddot{x}, \ddot{y}, \ddot{z}]^T$. Finally, define the vector

$$X = [\vec{r}^T, \vec{v}^T, \vec{a}^T]^T \tag{5.1}$$

by concatenating position, velocity, and acceleration. For a small, constant, discrete time step Δt, the aircraft position, velocity, and acceleration at time t_{k+1} can be estimated from

the values at time t_k as follows:

$$X_{k+1} = \begin{bmatrix} I_{3\times3} & \Delta t & \frac{1}{2}(\Delta t)^2 \\ 0 & I_{3\times3} & \Delta t \\ 0 & 0 & I_{3\times3} \end{bmatrix} X_k \tag{5.2}$$

Because the position history obtained from measurements is likely to be noisy, the velocity at time-step k may be estimated using a moving window average of finite differences over the past n time steps

$$\vec{v}_k = \frac{1}{n\Delta t} \sum_{i=k-n+1}^{k} (\vec{r}_i - \vec{r}_{i-1}) \tag{5.3}$$

where $\vec{r}_i$ is the position estimate of the threat aircraft at time step i. For the acceleration, two estimation approaches are considered: (1) using a moving window average of finite differences, as above, and (2) using a particle dynamic model for aircraft motion. The first approach is often used since it requires no model and performs well when the threat aircraft is operating near steady state [172]. However, a sUAS is maneuverable and susceptible to wind disturbances, so its acceleration can change quickly. The second approach uses the position and attitude data, together with aircraft equations of motion, to estimate the aircraft acceleration. This approach responds more quickly to a changes in acceleration than the first method. The latter approach was considered in [16] for 2D flight and was shown to perform better than the first method for predicting steady turning flight paths. In this chapter, the two approaches are compared for more general cases of 3D flight.

Using the second approach, the acceleration $\vec{a}$ can be computed using the threat aircraft velocity and its orientation, with the understanding that the aircraft's attitude influences the aerodynamic forces affecting its motion. Let $\Omega = [0, \dot{\gamma}, \dot{\psi}\cos\gamma]^T$ where γ and ψ are the flight path angle and course angle, respectively. These angles are extracted from the threat's

inertial velocity, which is inferred from measurements by the host aircraft. The algorithm to compute the angular rates is described shortly. The acceleration $\vec{a}$ is

$$\begin{aligned} \vec{a} &= \vec{a}_{\mathrm{t}} + \vec{a}_{\mathrm{n}} \\ &= \vec{a}_{\mathrm{t}} + \Omega \times \vec{v} \end{aligned} \tag{5.4}$$

where $\vec{a}_{\mathrm{t}}$ is the acceleration tangent to the flight path and $\vec{a}_{\mathrm{n}}$ is the acceleration normal to the flight path. Each acceleration vector can be computed using the particle dynamic equations of motion for a fixed wing aircraft:

$$\frac{dx}{dt} = V \cos\gamma \cos\psi \tag{5.5}$$

$$\frac{dy}{dt} = V \cos\gamma \sin\psi \tag{5.6}$$

$$\frac{dz}{dt} = V \sin\gamma \tag{5.7}$$

$$\frac{dV}{dt} = g\left(\frac{T-D}{W} - \sin\gamma\right) \tag{5.8}$$

$$\frac{d\gamma}{dt} = \frac{g}{V}\left(\frac{L}{W}\cos\phi - \cos\gamma\right) \tag{5.9}$$

$$\frac{d\psi}{dt} = \frac{g}{V}\frac{L}{W}\frac{\sin\phi}{\cos\gamma} \tag{5.10}$$

where L, W, T and D are lift, weight, thrust and drag force, respectively, ϕ is the roll angle of the aircraft, $V = \|\vec{v}\|$ is airspeed, assuming flight in still air, and g is the local specific force of gravity (See Figure 5.1.) Using these equations, Ω and $\vec{v}$ in (5.4) are estimated under the assumption that lift is perpendicular to thrust which, in turn, is aligned with the flight path [173, 174].

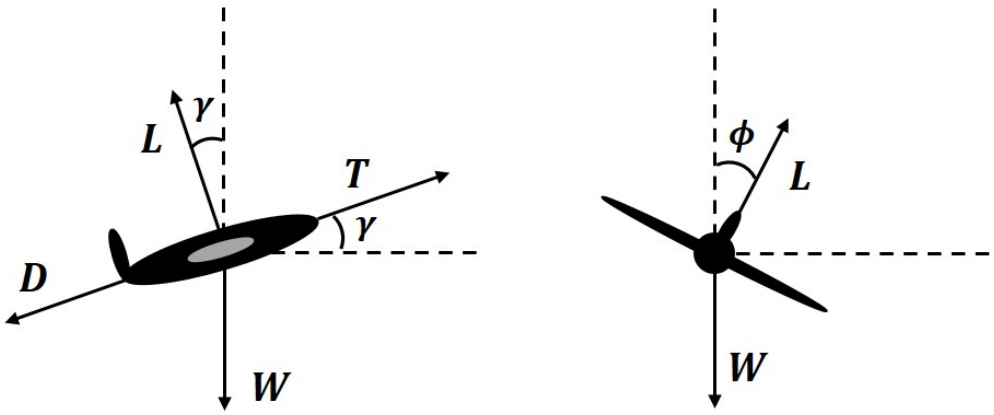


Figure 5.1: Applied forces acting to a fixed-wing aircraft

5.2.1 Acceleration in steady flight

In order to compute acceleration using (5.8)-(5.10), the unknown forces L, W, T and D are required. In this subsection, the simplifying assumption that the threat aircraft is always in a state of steady flight is made; the most general case of which is constant-speed flight at a constant climb angle and a constant turn rate, i.e., constant-speed flight along a vertical helix. Noting that V and γ are constant in steady flight, equations (5.8-5.9) imply that [172, 175, 176, 177]:

$$T = W \sin\gamma + D \tag{5.11}$$

$$L\cos\phi = W\cos\gamma \tag{5.12}$$

Equation (5.10) then gives

$$\dot{\psi} = \frac{g}{V}\tan\phi \tag{5.13}$$

Since $\vec{a}_\mathrm{t}$ and $\dot{\gamma}$ are zero in steady flight,

$$\vec{a} = \Omega \times \vec{v} = \begin{bmatrix} 0 \\ 0 \\ \dot{\psi}\cos\gamma \end{bmatrix} \times \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} -\frac{g}{V}\dot{y}\tan\phi\cos\gamma \\ \frac{g}{V}\dot{x}\tan\phi\cos\gamma \\ 0 \end{bmatrix} \tag{5.14}$$

5.2.2 Estimation of acceleration in unsteady flight

The equations in the previous section work well for wings-level or steady, turning flight. However, sUAS can change their acceleration quickly, due either to control inputs or to disturbances. In these cases, the steady flight assumption is inappropriate. Here, the steady flight assumption is relaxed and the acceleration is estimated using concepts of specific energy and power. First, because the speed of the aircraft may change, $\vec{a}_\mathrm{t}$ in (5.4) is no longer zero:

$$\vec{a} = \vec{a}_\mathrm{t} + \Omega \times \vec{v} = \begin{bmatrix} \ddot{x}_t \\ \ddot{y}_t \\ \ddot{z}_t \end{bmatrix} + \begin{bmatrix} 0 \\ \dot{\gamma} \\ \dot{\psi}\cos\gamma \end{bmatrix} \times \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} \ddot{x}_t - \dot{\psi}\dot{y}\cos\gamma + \dot{\gamma}\dot{z} \\ \ddot{y}_t + \dot{\psi}\dot{x}\cos\gamma \\ \ddot{z}_t - \dot{\gamma}\dot{x} \end{bmatrix} \tag{5.15}$$

In this case, the forces no longer balance as in (5.11) and (5.12).

The applied forces needed to estimate the acceleration of the threat aircraft are unknown. Here, the energy rate of the aircraft is estimated in order to approximate these forces. Reformulating equation (5.8) in terms of specific excess power gives [108, 178, 179, 180]:

$$\frac{(T-D)V}{m} = V\dot{V} + g\dot{z} \tag{5.16}$$

The left-hand side of (5.16) is the specific excess power and the right-hand side is the rate of change of specific energy due to (i) along-track acceleration and (ii) climbing in a gravi-

tational field.

Similarly, equation (5.9) is reformulated as

$$\frac{LV}{m} = \frac{V}{\cos\phi}\left(V\dot{\gamma} + g\cos\gamma\right) \tag{5.17}$$

Like (5.16), the left-hand side of (5.17) has units of specific power. This power is due to lift, however, rather than excess thrust. It accounts for energy-conserving changes in the flight path direction. The right-hand side of (5.17) can be interpreted as a conservative exchange of energy between vertical and lateral motion.

Alligier et al [178, 179] proposed least squares estimation of the aircraft mass based on the path history for climb path prediction. In their work, T and D are known, allowing one to compute the mass from (5.16). For the case considered in this chapter, T, D and m are all unknown, so the specific excess power

$$P_{\text{excess}} = \frac{(T-D)V}{m} \tag{5.18}$$

is estimated instead of the mass. Equation (5.16) enables one to estimate P_{excess} by computing the specific energy rate $\dot{E}_{\text{excess}} = V\dot{V} + g\dot{z}$ of a data point:

$$P_{\text{excess}} = \dot{E}_{\text{excess}} \tag{5.19}$$

Here, the along-track acceleration $\dot{V}$ is estimated from measurements using a moving window average, as in (5.3). A least-squares estimate based on several consecutive data points is used. In this formulation, P_{excess} is assumed to remain constant over a short time window (e.g., 1 second). Computing the cumulative squared error between the estimated constant value of

P_{excess} and the computed values $\dot{E}_{\text{excess}}$ over n data points up to time step k gives

$$e_{\text{excess}_k} = \frac{1}{n} \sum_{i=k-n+1}^{k} (P_{\text{excess}} - \dot{E}_{\text{excess}_i})^2 \tag{5.20}$$

The best estimate of P_{excess} is the least square solution obtained by solving

$$\frac{de_{\text{excess}_k}}{dP_{\text{excess}}} = 0 \qquad \Rightarrow \qquad P_{\text{excess}_k} = \frac{1}{n} \sum_{i=k-n+1}^{k} \dot{E}_{\text{excess}_i} \tag{5.21}$$

That is, P_{excess_k} is taken as the average specific energy rate over n previous time steps.

In a similar way, referring to (5.17), the specific power acting normal to the flight path is

$$P_{\text{norm}} = \frac{LV}{m} \tag{5.22}$$

This term may be computed in terms of the energy rate

$$\dot{E}_{\text{norm}} = \frac{V}{\cos\phi} \left(V\dot{\gamma} + g\cos\gamma \right) \tag{5.23}$$

As before, P_{norm} is assumed to remain constant over an n-step time horizon yielding the approximation

$$P_{\text{norm}_k} = \frac{1}{n} \sum_{i=k-n+1}^{k} \dot{E}_{\text{norm}_i} \tag{5.24}$$

The resulting specific power terms, P_{excess} and P_{norm}, are substituted directly into the aircraft equations of motion (5.8-5.10) noting that

$$\frac{T-D}{W} = \frac{P_{\text{excess}}}{Vg} \tag{5.25}$$

$$\frac{L}{W} = \frac{P_{\text{norm}}}{Vg} \tag{5.26}$$

Thus,

$$\frac{dV}{dt} = \left(\frac{P_{\text{excess}}}{V} - g\sin\gamma\right) \tag{5.27}$$

$$\frac{d\gamma}{dt} = \frac{1}{V}\left(\frac{P_{\text{norm}}}{V}\cos\phi - g\cos\gamma\right) \tag{5.28}$$

$$\frac{d\psi}{dt} = \frac{P_{\text{norm}}}{V^2}\frac{\sin\phi}{\cos\gamma} \tag{5.29}$$

The acceleration in (5.27) is projected onto each axis to determine the components of tangential acceleration:

$$\begin{aligned}
\ddot{x}_t &= \left(\frac{P_{\text{excess}}}{V} - g\sin\gamma\right)\cos\gamma\cos\psi \\
\ddot{y}_t &= \left(\frac{P_{\text{excess}}}{V} - g\sin\gamma\right)\cos\gamma\sin\psi \\
\ddot{z}_t &= \left(\frac{P_{\text{excess}}}{V} - g\sin\gamma\right)\sin\gamma
\end{aligned} \tag{5.30}$$

In the end, one obtains the following expressions for acceleration of the threat aircraft:

$$\vec{a} = \begin{bmatrix}
\left(\frac{P_{\text{excess}}}{V} - g\sin\gamma\right)\cos\gamma\cos\psi - \frac{P_{\text{norm}}}{V^2}\dot{y}\sin\phi + \frac{1}{V}\left(\frac{P_{\text{norm}}}{V}\cos\phi - g\cos\gamma\right)\dot{z} \\
\left(\frac{P_{\text{excess}}}{V} - g\sin\gamma\right)\cos\gamma\sin\psi + \frac{P_{\text{norm}}}{V^2}\dot{x}\sin\phi \\
\left(\frac{P_{\text{excess}}}{V} - g\sin\gamma\right)\sin\gamma - \frac{1}{V}\left(\frac{P_{\text{norm}}}{V}\cos\phi - g\cos\gamma\right)\dot{x}
\end{bmatrix} \tag{5.31}$$

Finally, the resulting velocity and acceleration are concatenated to form the state vector X_k and the future path of the threat is estimated as in (5.2).

5.2.3 Correcting for wind

To estimate and correct for the wind, the predicted path is compared with the measured path, attributing any error to wind disturbances. More specifically, I formulate an optimization problem to determine the aircraft velocity that minimizes path prediction error. The difference between this minimum-error velocity and the velocity predicted as described above provides an estimate of the ambient wind.

Given the current position $\vec{r}_k$, velocity $\vec{v}_k$ from (5.3), and acceleration $\vec{a}_k$ from (5.31), the position l time steps in the future can be estimated using (5.2):

$$\vec{r}^{\,\mathrm{p}}_{k+l} = \vec{r}_k + \vec{v}_k(l\Delta t) + \frac{1}{2}\vec{a}_k(l\Delta t)^2 \tag{5.32}$$

$$= \vec{r}_k + \vec{v}_k(l\Delta t) + \frac{1}{2}\left(\vec{a}_{\mathrm{t}_k} + \Omega_k \times \vec{v}_k\right)(l\Delta t)^2 \tag{5.33}$$

In order to compare the actual position history and the predicted position history, an averaged distance error at time step k between the two n-step histories is estimated as follows:

$$e^{\mathrm{p}}_k = \frac{1}{n}\sum_{i=k-n+1}^{k} \|\vec{r}^{\,\mathrm{a}}_i - \vec{r}^{\,\mathrm{p}}_i\| \tag{5.34}$$

where $\vec{r}^{\,\mathrm{a}}_i$ is the actual position, and $\vec{r}^{\,\mathrm{p}}_i$ is the predicted position at time i.Having defined an error metric, a velocity value $\vec{v}^{\,*}_k$ which minimizes the error e^{p}_k can be estimated. This optimal velocity is obtained by solving the following equations:

$$\vec{v}^{\,*}_k = \arg\min_{\vec{v_k}} e^{\mathrm{p}}_k = \arg\min_{\vec{v_k}} \frac{1}{n}\sum_{i=k-n+1}^{k} \|\vec{r}^{\,\mathrm{a}}_i - \vec{r}^{\,\mathrm{p}}_i\| \tag{5.35}$$

The difference between the minimum-error velocity $\vec{v}^{\,*}_k$ obtained through the optimization process above and the estimate $\vec{v}^{\,*}_k$ obtained by assuming flight through still air provides an

estimate for the wind velocity:

$$\delta\vec{v}_k = \vec{v}_k^* - \vec{v}_k \tag{5.36}$$

Note that this wind disturbance is based on a moving-window average; implicitly, the wind is assumed to vary slowly in time and space. In the following section, the prediction performance based on the optimized velocity $\vec{v}_k^*$, which attempts to correct for the effect of wind, is compared with the prediction performance using only position or position-plus-attitude measurements.

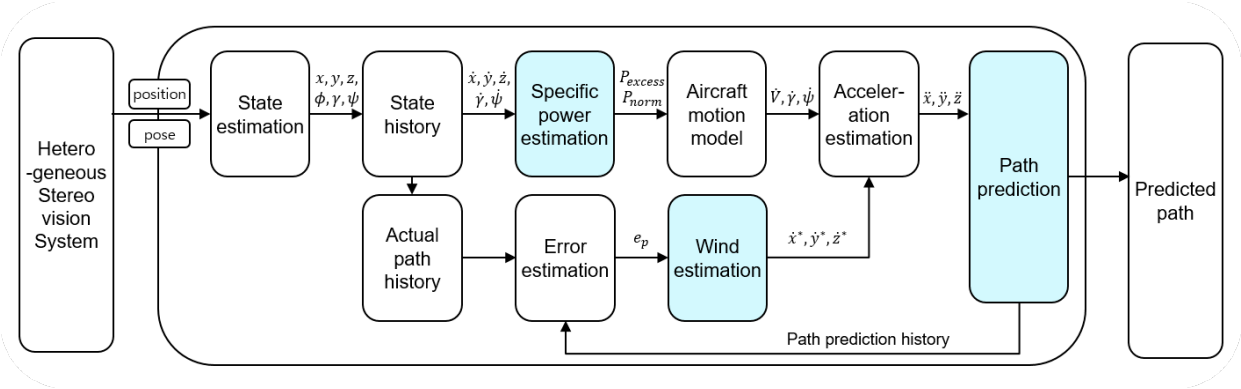


Figure 5.2: Flow chart of the algorithm

5.3 Flight Data

In order to assess the proposed algorithm, experimental flight data for two fixed-wing sUAS, the eSPAARO and HobbyKing Bixler, are used. The publicly accessible Small Aircraft Flight Encounters (SAFE) Data Repository [28] includes flight data for these aircraft.

Each flight data set contains full state history data as well as estimates of wind speed. A flight of a fixed-wing sUAS generally consists of various, distinct segments including straight and level flight, turning flight, and maneuvering flight. In order to see how the path prediction performance differs for each type of flight, the flight data are parsed into these three types,

as shown in Figure 5.3.

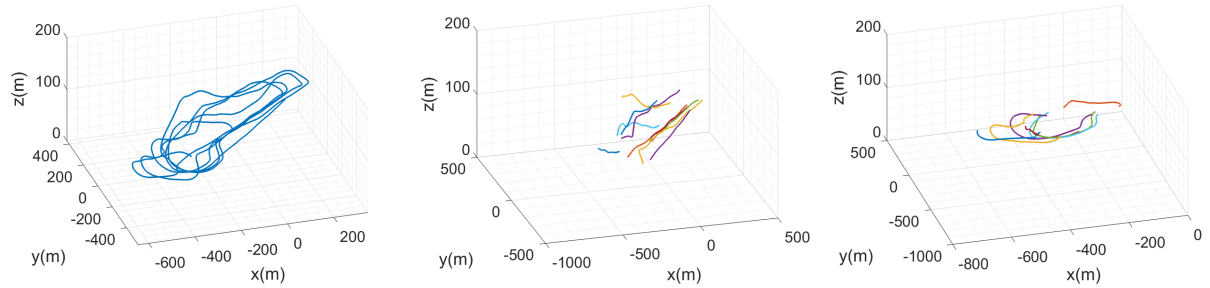


Figure 5.3: Path segmentation for a representative flight test: Complete flight path history (left), straight flight segments (center), and turning flight segments (right)

Table 5.1: Number of paths used for algorithm evaluation

Parameter	eSPAARO	Bixler
Straight flight	28	11
Turning flight	28	12
Maneuvering flight	7	20

Table 5.2: Specifications of aircraft used

Parameter	eSPAARO	Bixler
Length	2.81m	0.95m
Wingspan	3.57m	1.55m
Mass	20kg	1.2kg
Propulsion power	740W	28W
Cruise speed	18m/s	10m/s

Table 5.1 shows the number of each type of flight for each aircraft. As indicated in Table 5.2, the Bixler is lighter and smaller than the eSPAARO, so the Bixler is more maneuverable while the eSPAARO tends to fly more steadily. The Bixler has a greater number of maneuvering flight segments in the dataset than the eSPAARO. After segmenting the flight paths, three path prediction algorithms were applied to each type of flight path to compare

the performance: (i) position-based prediction, (ii) pose-based prediction, and (iii) pose-plus-wind prediction. The position-based prediction algorithm estimates the acceleration by simply computing a discrete-time filtered derivative of velocity from telemetry data. The pose-based method uses the pose and aircraft equations of motion to estimate the acceleration as described in Section 5.2.2. For the purpose of this chapter, pose is taken directly from the threat aircraft data log. In practice, pose would be inferred (with inevitable error) from visual imagery as suggested in [16]. The pose-plus-wind prediction algorithm computes the acceleration in the same way as the pose-based method and corrects the path prediction as described in Section 5.2.3. The flight data includes the position and pose data of the aircraft with the GPS time. Using the GPS time, the algorithm is assessed by re-playing the flight data as shown in Figure 5.4. This figure shows flight path data from a maneuvering flight of the Bixler aircraft. The light blue line depicts the actual path of the aircraft, and the red asterisk shows the aircraft position at the current time-step. The thick blue line indicates the predicted path, and the green line represents the actual path over the prediction time horizon. The following section describes a comparison of the three approaches to path prediction described earlier using flight test data.

5.4 Results

Path prediction performance for the two, small, fixed-wing unmanned aircraft described in Section 5.3 is evaluated by comparing the predicted path to the actual path of the aircraft. The left column of Figure 5.4 shows an example of path prediction using the position-based approach, from two vantage points, and the right column shows the path prediction using the pose-plus-wind approach.

In the figure, the predicted path using the pose-plus-wind approach (right column, thick blue

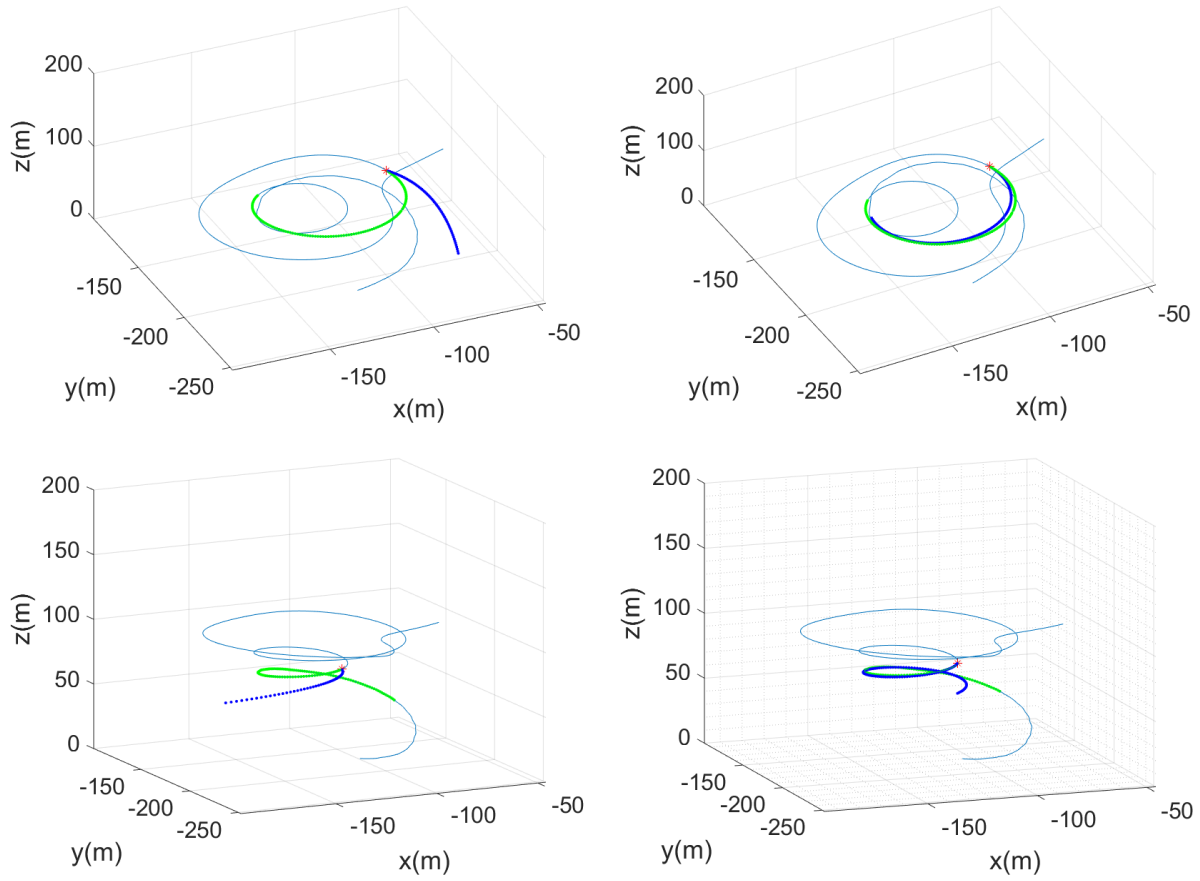


Figure 5.4: Path prediction (blue line) using position data only (left column) and pose-plus-wind (right column). The top and bottom plots show predictions at different times within the same data set, and from different view points.

lines) is more similar to the actual path (green lines) over the prediction time horizon than the one using the position-based approach (left column, thick blue lines). Anecdotally, the pose-plus-wind prediction method performs at least as well – and sometimes significantly better – than the position-based method. Beyond the anecdotal evidence shown in Figure 5.4, a quantitative performance comparison for the flight segments identified in Table 5.1 is presented here.

Although wind estimation is not the focus of this chapter, the wind is a critical factor for sUAS path prediction. Some wind estimation results is thus discussed in this section, as

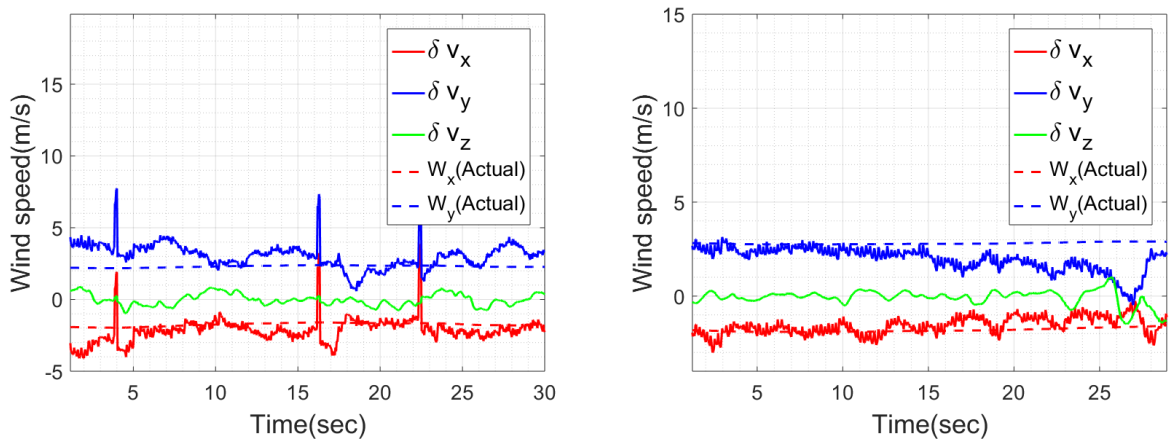


Figure 5.5: Wind estimation for the eSPAARO over two 30-second intervals.

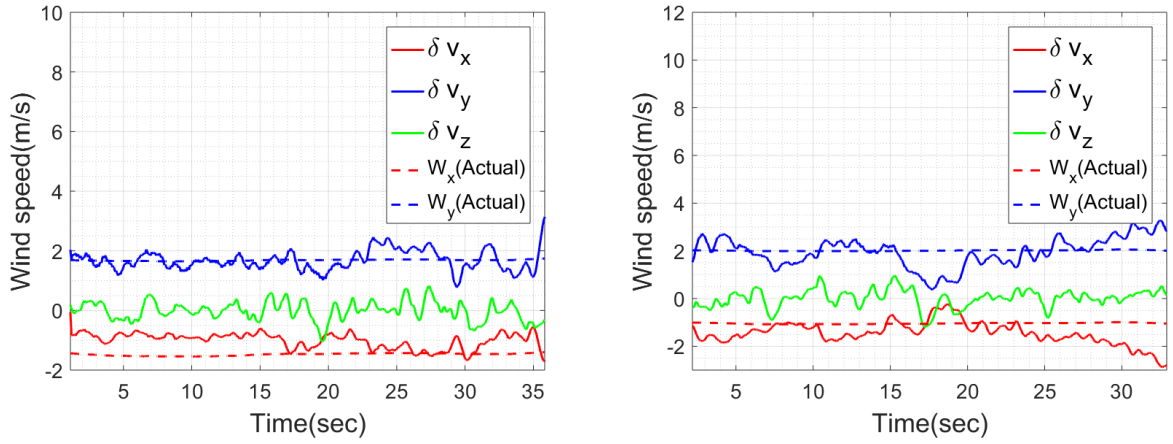


Figure 5.6: Wind estimation for the Bixler over two 30-second intervals.

well. Figures 5.5 and 5.6 show the estimated wind disturbance and the actual wind velocity. (Here, the wind estimate provided by the threat aircraft's Pixhawk 1 autopilot is taken as the true wind velocity.) Red, blue and green lines indicate the three components of $\delta\vec{v}^* = [\delta v_x, \delta v_y, \delta v_z]^T$, respectively. Dashed red and blue lines show the true wind speed in the x direction (W_x) and the y direction (W_y), respectively. As can be seen in the figures, δv_x and W_x are comparable, as are δv_y and W_y, which indicates that the estimated wind disturbance coincides with the actual wind velocity.

The following section provides a quantitative performance comparison for the proposed path

prediction methods, as well as an assessment of wind estimation accuracy for the pose-plus-wind prediction method.

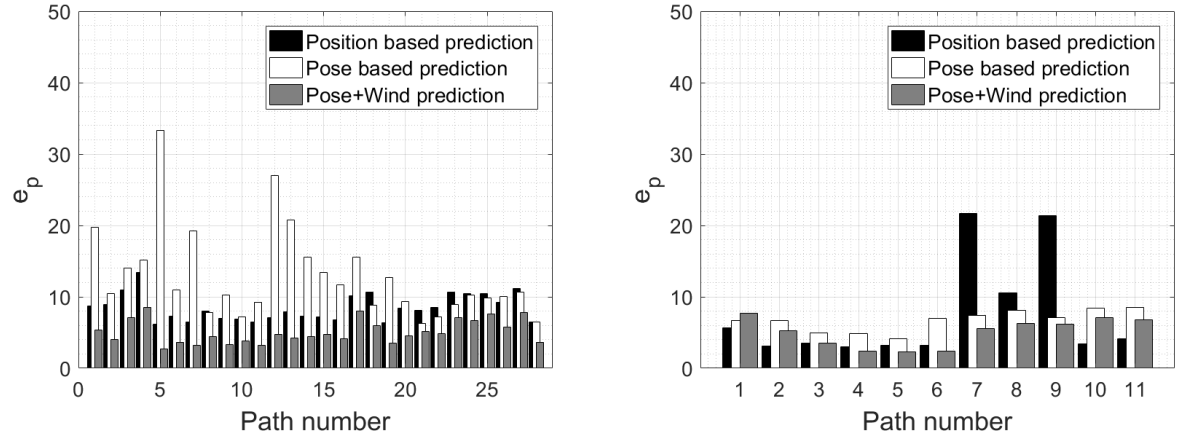


Figure 5.7: Averaged distance error e_p for straight flight of two aircraft without noise (left: eSPAARO, right: Bixler)

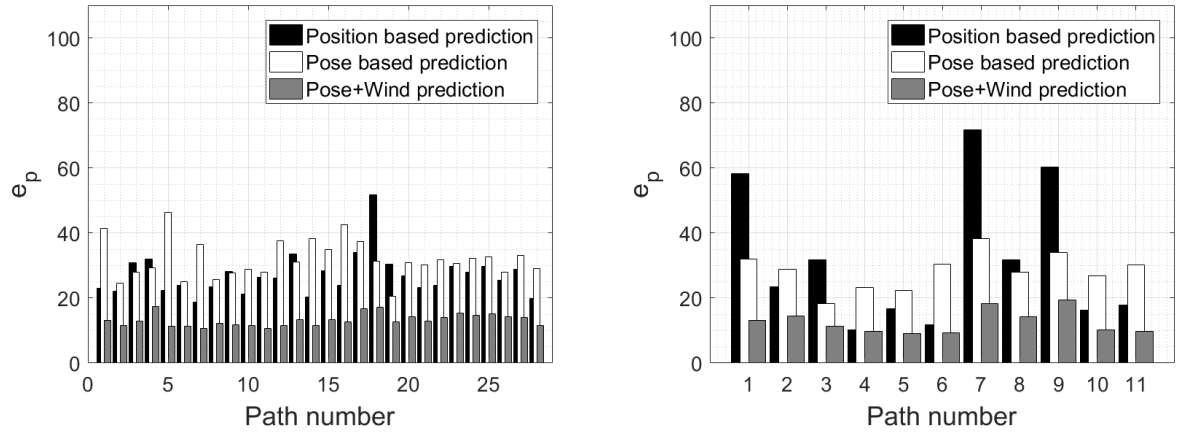


Figure 5.8: Averaged distance error e_p for straight flight of two aircraft with noisy data (left: eSPAARO, right: Bixler)

5.4.1 Prediction performance

In Section 5.2.3, e_k^p is defined as an averaged distance error between the predicted position history and the actual position history at time step k. The averaged e_k^p during a flight is

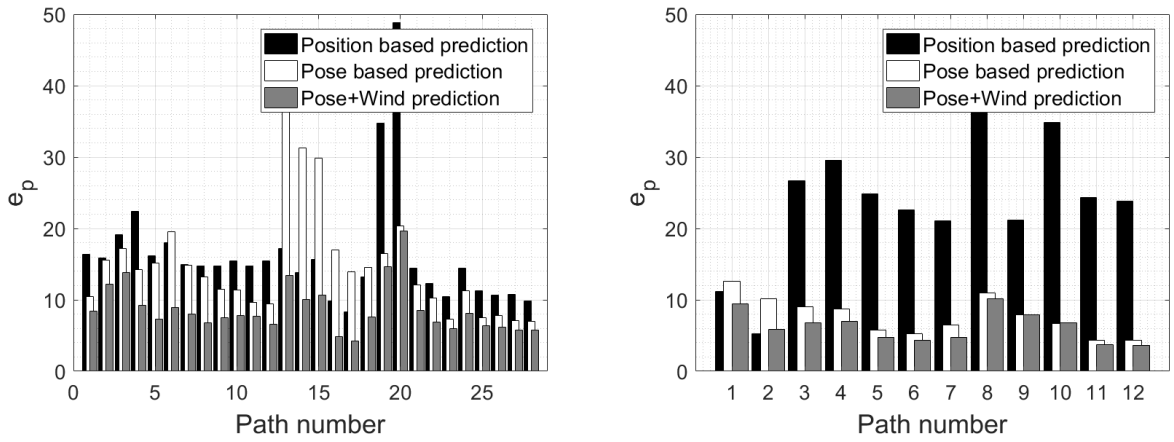


Figure 5.9: Averaged distance error e_p for turning flight of two aircraft without noise (left: eSPAARO, right: Bixler)

defined as e_p, and this is used to evaluate the prediction performance in this section. As indicated in Table 5.1, the entire paths of the eSPAARO and Bixler are segmented to 3 types of flight, and the prediction performance is evaluated for each type of flight.

The measurements of the threat aircraft position and attitude are assumed to be available from an on-board sUAS sensing system of the host aircraft. In practice, this sensing system might be a vision-based system as described in [16]. However, fixed-wing flight imagery that is amenable to the analysis described here, together with the independent motion data that are needed for validation, is currently unavailable to the authors. (A flight campaign is planned to address this data shortage.) The flight data used to assess performance of the algorithms described here were therefore obtained directly from the Pixhawk 1 autopilot data recorded by each threat aircraft. As a result, the flight data used here are likely much more accurate than observations from the on-board sUAS sensing system. My interest here, however, is in comparing the relative performance of three distinct approaches, rather than assessing the absolute performance using a particular measurement system. Having said that, I certainly have an interest in how path prediction performance might vary as a result of noise in the observations. Reference [171] describes the performance of a ground-

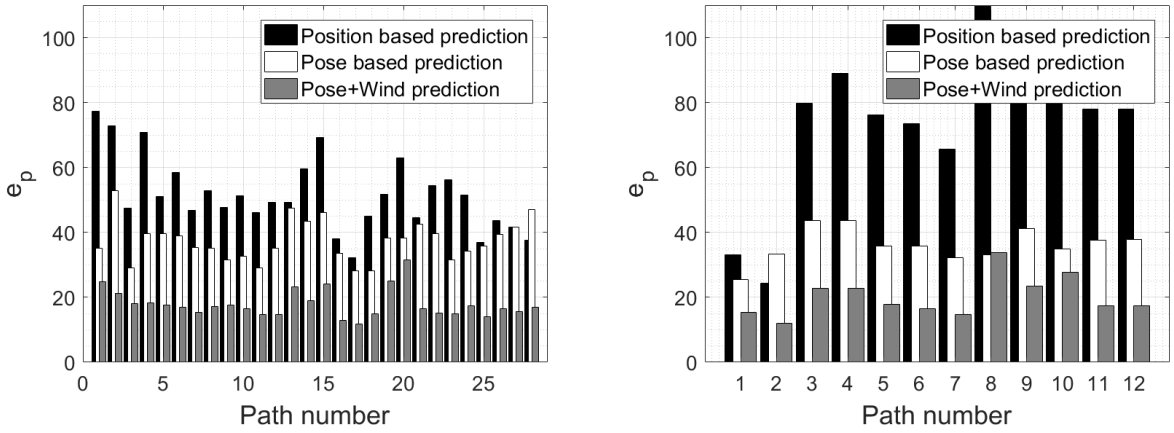


Figure 5.10: Averaged distance error e_p for turning flight of two aircraft with noisy data (left: eSPAARO, right: Bixler)

based radar for detecting and tracking a sUAS (a DJI Phantom 2), citing a range accuracy better than 2 m. By comparison, a small, on-board radar described in [28] provides a position estimation accuracy on the order of 5 m. To simulate measurement uncertainty in the envisioned scenario, I superimpose zero-mean Gaussian noise, with a 5 m standard deviation, on the threat position. Similarly, a zero-mean Gaussian noise to the threat attitude angles is added; the standard deviation is based on measurements from the vision-based pose estimation strategy reported in Table 3.2 of Chapter 3. A Kalman filter is applied to the artificially noise-corrupted measurement data and the prediction algorithms is then applied to these filtered data as described.

Figures 5.7, 5.9 and 5.11 show e_p computed based on uncorrupted telemetry data; Figures 5.8, 5.10 and 5.12 show e_p with artificial noise superimposed on the telemetry data. For these bar graphs, the index for the given path segment appears on the x-axis; the height of the bars represents the prediction error e_p. Black bars show e_p for position-based prediction, white bars show e_p for pose-based prediction (i.e., based on position and attitude), and gray bars show e_p for pose-plus-wind prediction.

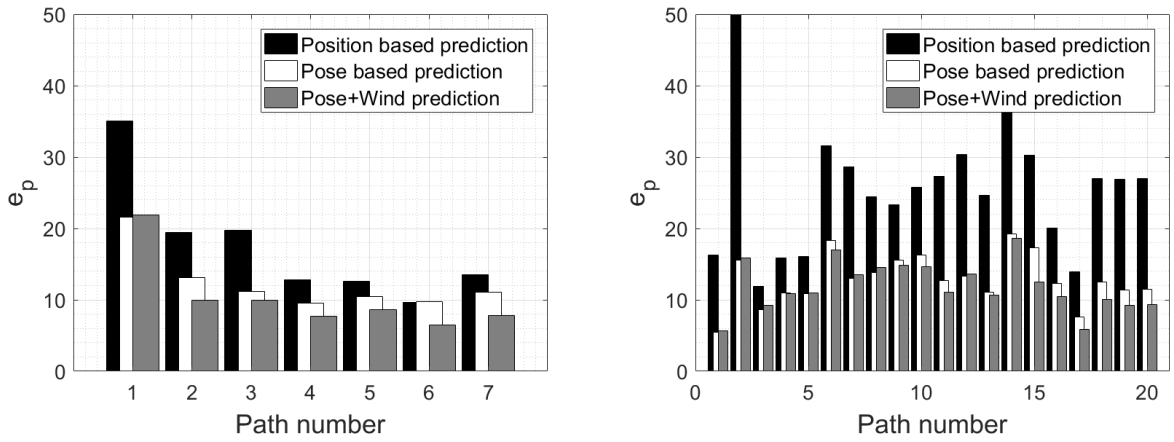


Figure 5.11: Averaged distance error e_p for maneuvering flight of two aircraft without noise (left: eSPAARO, right: Bixler)

In Figure 5.7, which pertains to straight flight path segments, the position-based prediction method generally outperforms pose-based prediction for the eSPAARO (left) and the Bixler (right). Pose-plus-wind prediction generally outperforms both of these other methods for both aircraft. When artificial noise is introduced, the pose-based algorithm performs worse than position-based prediction for both aircraft, as seen in Figure 5.8. Note that pose-based prediction will be in error if the aircraft does not point in the direction it is traveling. This happens, for example, when the aircraft "crabs" into an ambient wind in order to maintain a commanded course.

For turning flight (Figures 5.9 and 5.10), pose-based prediction generally outperforms position-based prediction for the Bixler, with or without noise, and also for the eSPAARO when artificial noise is injected into the measurements.

For maneuvering flight (Figures 5.11 and 5.12), the results generally match those for turning flight. The pose-plus-wind approach does not provide such a dramatic improvement as in the earlier cases, but this approach still consistently outperforms the alternatives.

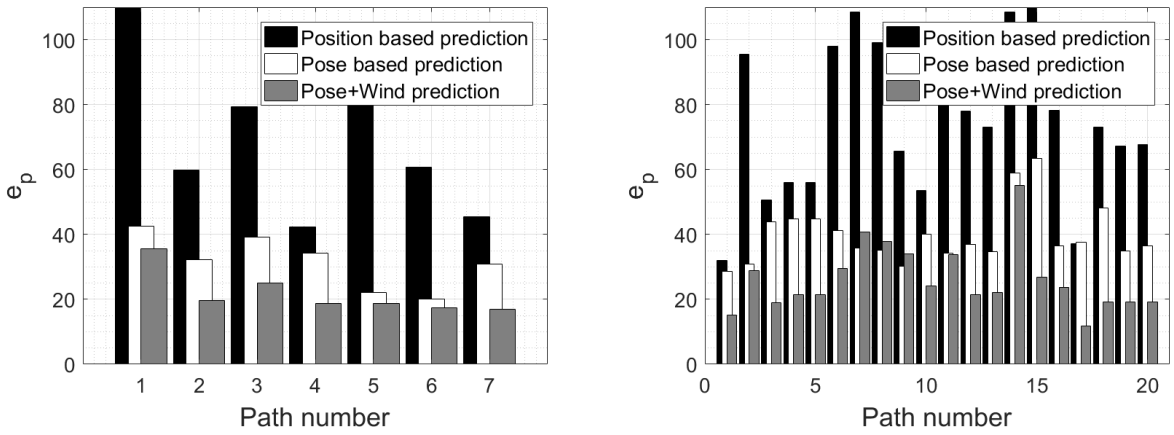


Figure 5.12: Averaged distance error e_{p} for maneuvering flight of two aircraft with noisy data (left: eSPAARO, right: Bixler)

5.4.2 Wind estimation

As stated in Section 5.2.3, the wind disturbance $\delta\vec{v}$ is estimated to correct the path prediction. Here, the estimated wind disturbance is compared with the "true" wind velocity, that is, the wind velocity estimated by the autopilot's integrated wind estimation scheme. Figures 5.13, 5.15 and 5.17 show the averaged wind estimation error based on the pose-plus-wind algorithm applied directly to telemetry data. Figures 5.14, 5.16 and 5.18 show the wind estimation error when artificial noise is injected into the measurements. Black and gray bars indicate the magnitude of the wind estimation error, $|\delta v_x - W_x|$ and $|\delta v_y - W_y|$, respectively. For most cases involving the Bixler, the wind estimation error is quite low compared with that for the eSPAARO aircraft. The author conjecture that because the Bixler is smaller, lighter, and slower than the eSPAARO, it is more susceptible to wind disturbances, making estimation of these wind disturbances easier. The wind estimates predictably degrade with the injection of artificial noise. The results suggest that the wind velocity in the vicinity of a small unmanned aircraft can be estimated using the algorithm suggested here, provided the wind disturbances have a visible influence on aircraft motion and the measurement error is low.

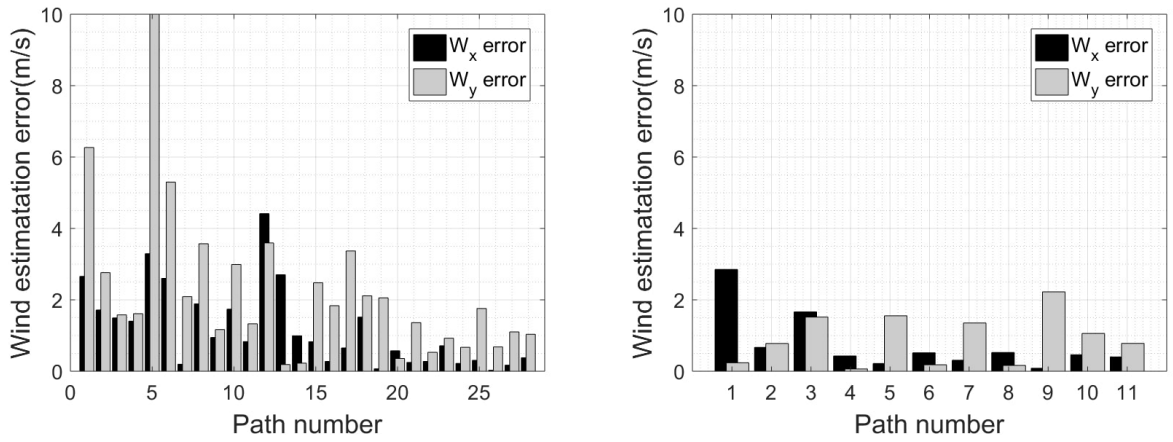


Figure 5.13: Wind estimation error of straight flight from two aircraft without noise (left: eSPAARO, right: Bixler)

Alternatively, if the host aircraft is capable of measuring the wind in its own vicinity, it may be reasonable over short distances to assume this wind field is constant and uniform so that the same wind affects the threat aircraft. In general, though, the ability to infer the wind disturbance acting on a distant aircraft can be useful in predicting its path.

5.5 Summary

Three methods to predict the path of a fixed-wing aircraft from visual observations were proposed and compared. These methods all assume that the aircraft position can be inferred, for example, from a stereo vision system such as a peripheral-central vision system described in earlier chapters. In addition, this vision system may provide the attitude of the observed aircraft which can be used, together with a particle dynamic model for aircraft flight, to predict the path. Moreover, one may obtain even more accurate predictions by using measurement residuals to infer the ambient wind in the vicinity of the threat. To demonstrate and assess the performance of the proposed algorithms, experimental flight data for two small fixed-wing unmanned aircraft were used. The results show that position- and

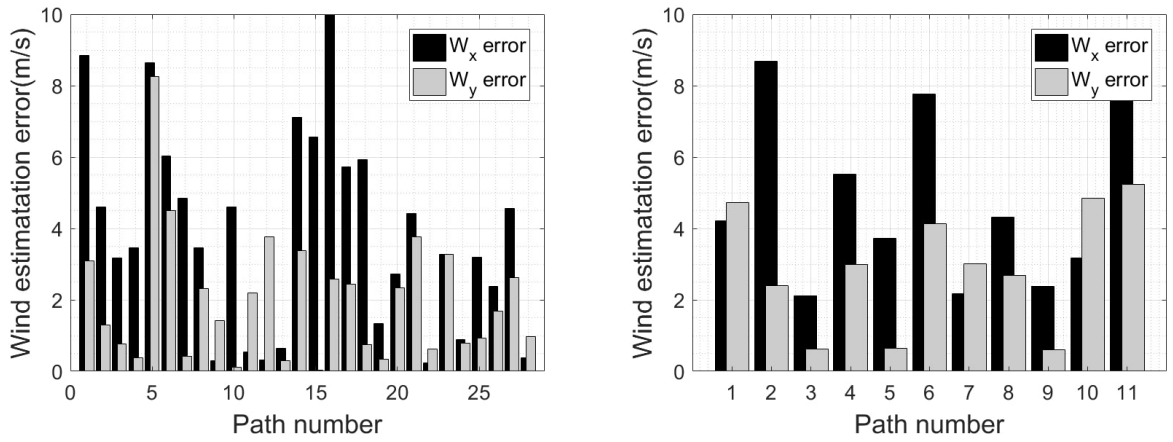


Figure 5.14: Wind estimation error of straight flight from two aircraft with artificial noise (left: eSPAARO, right: Bixler)

pose-based prediction perform comparably for straight flight while pose-based prediction generally outperforms position-based prediction for turning or maneuvering flight. Pose-plus-wind prediction, in which measurements of the aircraft position and attitude are used along with estimates of the wind velocity, shows the best performance among three approaches for all of types of flight and both types of aircraft. Comparisons of the estimated wind disturbance and the actual wind velocity show that the wind estimation error for the smaller, lighter aircraft is quite small relative to the wind estimation error for the larger, faster aircraft. For the smaller aircraft, a given wind disturbances has a greater effect on position and attitude, which are used in the prediction algorithm. Ongoing efforts aim to develop the path prediction algorithm for other types of aircraft, such as multi-rotor aircraft, and to implement the algorithm for real-time operation in experimental hardware.

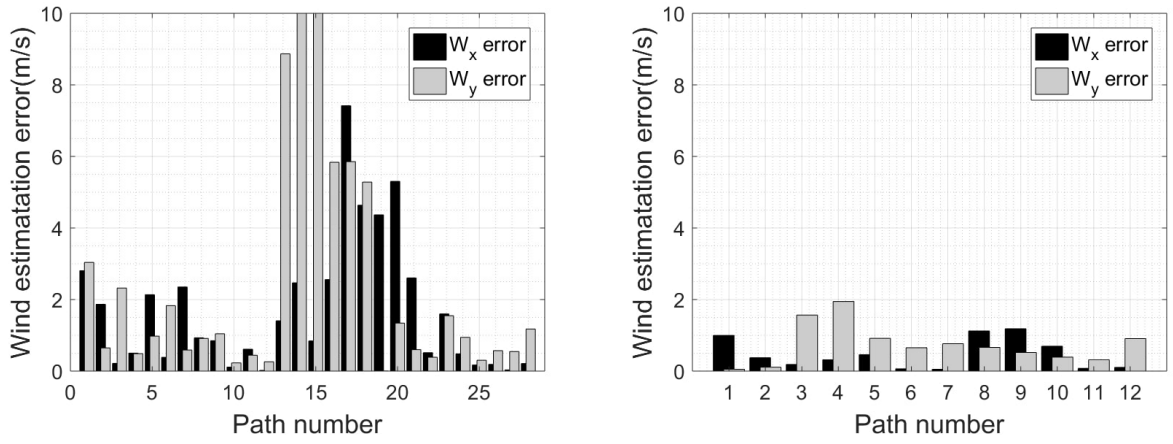


Figure 5.15: Wind estimation error of turning flight from two aircraft without noise (left: eSPAARO, right: Bixler)

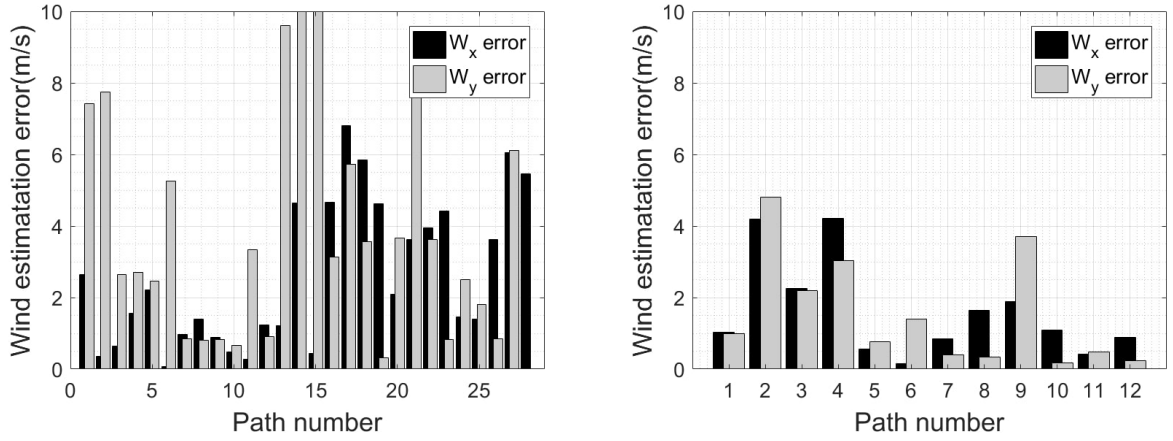


Figure 5.16: Wind estimation error of turning flight from two aircraft with artificial noise (left: eSPAARO, right: Bixler)

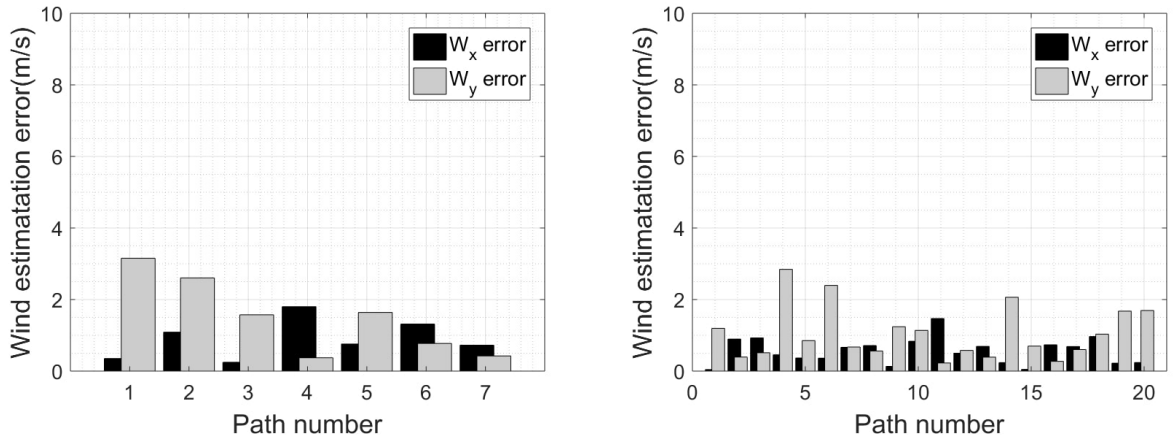


Figure 5.17: Wind estimation error of maneuvering flight from two aircraft without noise (left: eSPAARO, right: Bixler)

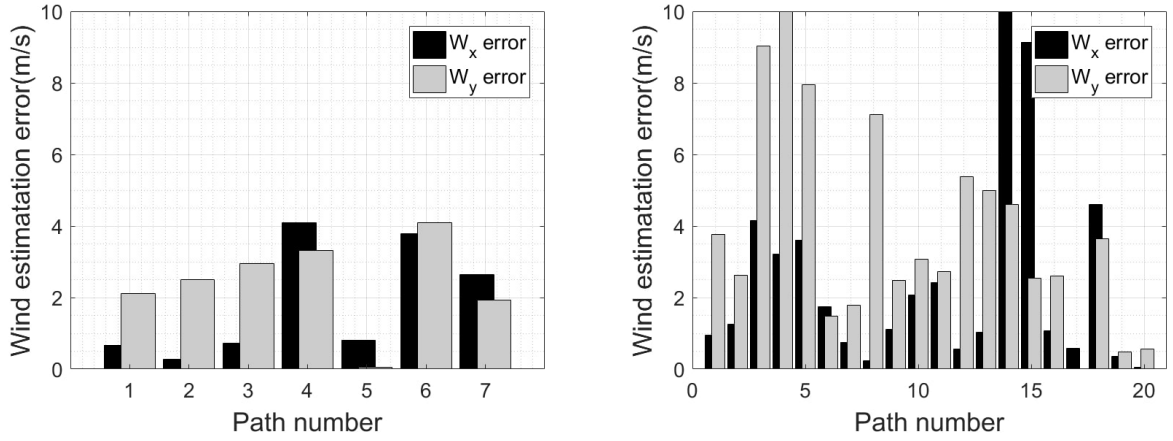


Figure 5.18: Wind estimation error of maneuvering flight from two aircraft with artificial noise (left: eSPAARO, right: Bixler)

Chapter 6

Multiple Threat Tracking Using PCV System

6.1 Introduction

In previous chapters, only a single threat has been assumed. However, this assumption may not be reasonable for an urban environment or more crowed areas in which multiple objects may exist in the air. Also, swarm technology for UAS has been developing, tracking multiple threat simultaneously is a major existing gap to be filled in the commercial C-UAS community [22]. Therefore, the single threat assumption is relaxed in this chapter, and an algorithm to track multiple threats simultaneously is introduced.

Here, an environment with multiple sUAS threats is considered. The peripheral vision camera is able to observe all these threats at once using its large field-of-view (FOV). The central vision camera then slews to observe each threat to obtain detailed data (e.g. position, velocity, aircraft classification, etc.). A threat that is rapidly approaching may need to be observed as soon as possible. However, all of these threats keep moving and changing their state. To keep the threat data updated, the central vision camera needs to keep maneuvering to observe each threat according to a particular "threat schedule" determined by a risk analysis based on the state of threats such as the approach speed and distance to the threat.

Most of the existing threat scheduling algorithms prioritize the risk posed by each threat over other concerns. These scheduling algorithms are especially appropriate for ground-based sensing systems, which are less limited by size, weight, power, and cost (SWaP-C) and which generally have a long sensing range. Scheduling algorithms for these systems do not need to consider the time taken for the central vision camera to move towards threats (e.g. panning/tilting time for a PTZ camera), since the threats can be detected from a long stand-off distance. (For example, the sensing range of the ground-based radar considered in [181, 182] is more than 3 km.) On the other hand, if the sensing range of the peripheral vision camera is short (e.g. the sensing range of the PCV system is 100 m [17]), the time that it takes for the central vision camera to maneuver and image a particular threat is crucial because one of the threats may harm the host system even before it is observed by the central vision camera. Therefore, a threat scheduling algorithm that minimizes both the slewing time required by the central vision camera ("time-efficient observation") and the risk posed by the threats is suggested in this chapter.

Time-efficient observation can be conducted by minimizing the maneuver of the sensor. For example, a threat that is near the current FOV of the central vision camera might pose a relatively low-risk value according to the existing algorithms, so the imaging and characterization of this threat might be postponed while the sensor images a higher risk threat. However, observing this threat as part of the camera's maneuver toward the higher risk threat would be more time-efficient; a suitably designed scheduling algorithm would enable all of the threats to be observed faster and/or more often. To accomplish this, a traveling salesman problem (TSP) [183, 184] is modified, and the solution is used to schedule threats. In this chapter, a novel multiple threat scheduling algorithm for mobile omnidirectional-directional combined sensor systems, that considers not only the risk of threats but also the time duration taken for observation using a modified TSP solution, is

introduced. This work is presented in [20].

6.2 Risk Value Estimation

The definition of "risk" of a threat may differ depending on the purpose of the sensing system. Defining "threat" and "risk" are essential steps in developing C-UAS systems [12, 22, 23, 24, 25]. For example, if the purpose of the sensing system is just to detect and avoid other aircraft, the minimum time to collision might be an appropriate definition of risk. However, if a sensing system supports counter-UAS to protect an area from overflight by hostile drones, then additional parameters should be considered (e.g. threat class/capability, etc.). Other threat scheduling studies mentioned above [112, 113, 114, 115, 120] thus have different definitions of the risk. In this chapter, the risk in the context of a particular scenario is defined:

- The host system is hovering in the air (the host aircraft position does not change), and the purpose of the host system is to detect and characterize the airborne threats inside the detectable range.
- A path prediction algorithm for threat future paths is implemented on the system (e.g. [16, 96, 97, 98, 99, 100, 101]).
- All threats are sUAS.

Given these assumptions, a set of risk parameters associated with each threat is defined. The risk parameter values are computed and summed to generate a risk estimate for each threat. Each parameter is normalized as a unit-free value, between 0 and 1 using a similar method presented in [120].

Range to the threat

A closer threat should be considered riskier. The range to a threat from the host system at time step k is

$$d_k^{\mathrm{t}} = \left\| \vec{r}_k^{\,\mathrm{h}} - \vec{r}_k^{\,\mathrm{t}} \right\| \tag{6.1}$$

where $\vec{r}_k^{\,\mathrm{h}}$ is the host system position, and $\vec{r}_k^{\,\mathrm{t}}$ is the threat position at time step k. The range is normalized using the maximum detectable range of the system ($d_{\max}$). The value of $d_{\max}$ would vary with the system, as optimized for a particular application. The normalized risk parameter corresponding to range to the threat at time step k is

$$\bar{D}_k = \max(0, 1 - \frac{d_k^{\mathrm{t}}}{d_{\max}}) \tag{6.2}$$

Current approach speed

The faster a threat is approaching, the higher the risk that it poses. The threat's approach speed at time step k can be estimated using a moving window average of finite position differences between the host system and a threat over the past n time steps.

$$V_k^t = \frac{1}{n\Delta t} \sum_{i=k-n}^{k} (d_{i-1}^{\mathrm{t}} - d_i^{\mathrm{t}}) \tag{6.3}$$

where Δt is a small, constant, discrete time step. The current approach speed is then normalized using an estimate of the maximum approach speed ($V_{\max}$) of the threat. Since the host is assumed that it is hovering in place, the value of $V_{\max}$ is the maximum speed of the threat; one might choose a large value as a conservative guess, until the threat is imaged and classified. The normalized risk parameter corresponding to the threat's approach speed at time step k is

$$\bar{V}_k^t = \max(0, \frac{V_k^{\mathrm{t}}}{V_{\max}}) \tag{6.4}$$

Predicted approach speed

The predicted path of a threat can be estimated using path prediction algorithms [16, 96, 97, 98, 99, 100, 101]. Figure 6.1 shows an aircraft flying away from the host system ($V^t < 0$). However, the predicted path of the aircraft (blue line), implies that the aircraft will change its heading and possibly approach the host system in the future. Even if a threat does not seem risky now, the threat may become riskier if the threat changes its heading. Therefore, the future maneuver of the threat is an important factor to consider for risk value estimation.

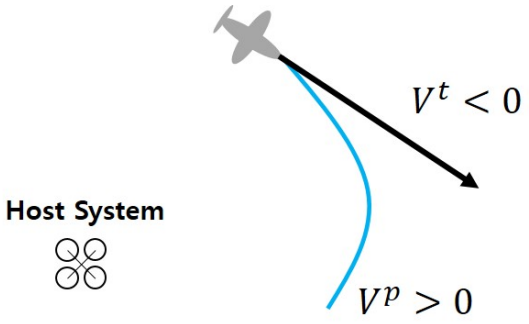


Figure 6.1: Predicted approach speed of a threat

While V_k^t is computed based on the threat position over the past n time steps in (6.3), the predicted approach speed of a threat at time step k is obtained using the predicted position of the threat in n time steps ahead.

$$V_k^{\mathrm{p}} = \frac{1}{n\Delta t} \sum_{i=k+1}^{k+n+1} (d_{i-1}^{\mathrm{p}} - d_i^{\mathrm{p}}) \tag{6.5}$$

where $d_i^{\mathrm{p}} = \sqrt{\left\| \vec{r}_i^{\,\mathrm{h}} - \vec{r}_i^{\,\mathrm{p}} \right\|^2}$, and $\vec{r}_i^{\,\mathrm{p}}$ is the predicted threat position at time step i. One may pick the time step n as the "worst case" value. For example, the turn rate of a fixed-wing

aircraft with a roll angle ϕ is [16]:

$$\dot{\psi} = \frac{g}{V} \tan \phi \tag{6.6}$$

where V is the aircraft speed, and g is the local specific force of gravity. Using this equation and the maximum possible roll angle of the threat aircraft $\phi_{\max}$, the minimum number of time steps for the aircraft to change its heading to a collision course ($\Delta\psi \leq \pi$) can be estimated. (For example, $\Delta\psi$ of the threat aircraft in Figure 6.1 would be about 2.) This value for the worst case scenario can be used as the look-ahead time steps:

$$n = \frac{\Delta\psi V}{g\Delta t \tan \phi_{\max}} \tag{6.7}$$

While it is possible to predict a future closing rate for the threat, this prediction relies on a path prediction that will generally be in error. To reduce the likelihood that a high path prediction error will artificially inflate the risk, the predicted approach speed is discounted based on the current path prediction error. In [16], an averaged distance error at time step k between the n-step threat position history and the predicted position history for the same time horizon is computed as follows:

$$e_k^{\mathrm{p}} = \frac{1}{n} \sum_{i=k-n}^{k} \left\| \vec{r}_i^{\,\mathrm{t}} - \vec{r}_i^{\,\mathrm{p}} \right\| \tag{6.8}$$

This parameter e_k^{p} is used as a parameter to indicate the path prediction error, and this is normalized using the worst-case prediction error ($e_{\max}^{\mathrm{p}}$):

$$\bar{e}_k^{\mathrm{p}} = \min\left(\frac{e_k^{\mathrm{p}}}{e_{\max}^{\mathrm{p}}}, 1\right) \tag{6.9}$$

Considering e^{p} is a mean deviation between the threat position history and the predicted position history, one may choose a conservative mean deviation during a prediction horizon

for $e^{\mathrm{p}}_{\max}$. Therefore, $\bar{e}^{\mathrm{p}}_k$ represents a scale of prediction error which is expected to range from 0 to 1, with 1 indicating that the prediction error is too high, and V^{p}_k would provide a wrong predicted approach speed. To consider the path prediction error, the factor $1-\bar{e}^{\mathrm{p}}_k$ multiplies the predicted approach speed V^{p}_k, and the resulting value is normalized using $V_{\max}$:

$$\bar{V}^{\mathrm{p}}_k = \max(0, \frac{V^{\mathrm{p}}_k}{V_{\max}}(1-\bar{e}^{\mathrm{p}}_k)) \tag{6.10}$$

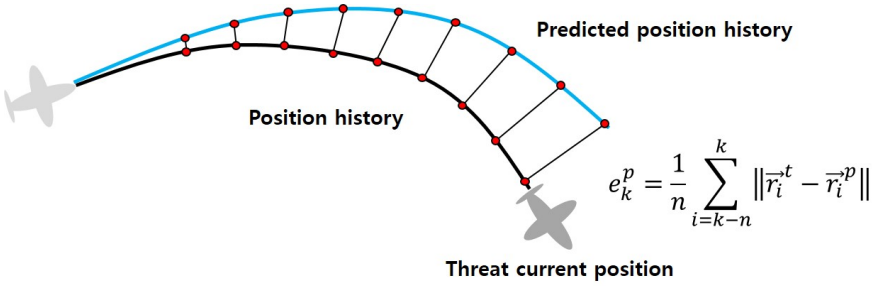


Figure 6.2: e^{p} computation

New threat value

If a new threat is observed by the peripheral vision camera and is not observed by the central vision camera yet, the threat should be observed with a high priority since the detailed data of the new threat is not available yet. A parameter called new threat value (c_{nt}) is defined. If a new threat is detected, then the parameter value for this threat is $c_{\mathrm{nt}} = 1$; otherwise, $c_{\mathrm{nt}} = 0$.

$$c_{\mathrm{nt}} \in \{0, 1\} \tag{6.11}$$

Risk value combination

After estimating all parameters mentioned above, these parameters are combined as a risk

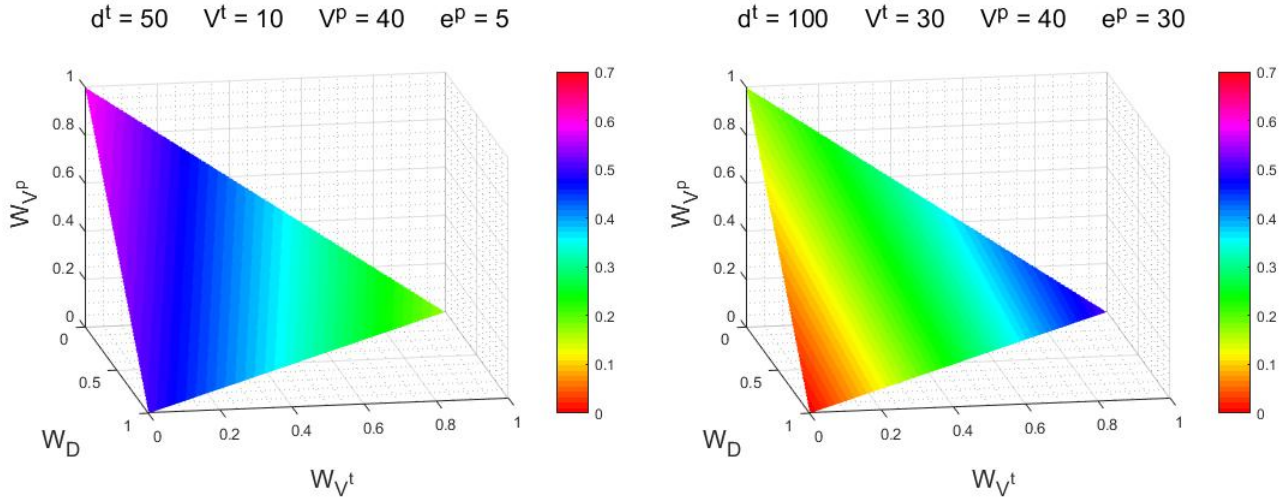
value with weights for each parameter:

$$RV_k = c_{\mathrm{nt}} + (1 - c_{\mathrm{nt}})(W_D \bar{D}_k + W_{V^{\mathrm{t}}} \bar{V}_k^{\mathrm{t}} + W_{V^{\mathrm{p}}} \bar{V}_k^{\mathrm{p}}) \tag{6.12}$$

where the summation of all weight parameters is 1 ($W_D + W_{V^{\mathrm{t}}} + W_{V^{\mathrm{p}}} = 1$). Note that $RV = 1$ when $c_{\mathrm{nt}} = 1$, reflecting the immediate prioritization of previously unobserved threats; the addition and removal of threats detected by the peripheral vision camera are managed by a separated module. Also note that all the parameters are normalized between 0 and 1, and the summation of weight parameters is 1; thus $W_D \bar{D}_k + W_{V^{\mathrm{t}}} \bar{V}_k^{\mathrm{t}} + W_{V^{\mathrm{p}}} \bar{V}_k^{\mathrm{p}}$ cannot exceed 1.

The weight parameters can be determined depending on the user's intention. If one wants to place twice the importance on the current approach speed as other parameters, then one may choose $W_{V^{\mathrm{t}}} = 0.5$ and $W_D = W_{V^{\mathrm{p}}} = 0.25$. In this case, the current approach speed will dominate when the risk value is estimated. Figure 6.3 shows risk value changes with different weight parameters of some example threats. Each axis represents each weight parameter, and the risk values are represented by the color map on the triangular portion of the plane ($W_D + W_{V^{\mathrm{t}}} + W_{V^{\mathrm{p}}} = 1$). For this example, $d_{\max} = 100$ m, $V_{\max} = 59$ m/s, $e^{\mathrm{p}}_{\max} = 40$. Figure 6.3a shows a threat which is approaching slowly ($\bar{V}^{\mathrm{t}} = 0.2$) from a close distance ($\bar{D} = 0.5$) with a fast predicted approach speed based on an accurately predicted path ($\bar{V}^{\mathrm{p}} = 0.59$.) For this threat, the larger $W_{V^{\mathrm{t}}}$ is, the smaller the risk value is because $\bar{V}^{\mathrm{t}}$ is smaller than other parameters. In another example in Figure 6.3b, the threat is approaching fast ($\bar{V}^{\mathrm{t}} = 0.5$) from a distant location ($\bar{D} = 0$) with a fast predicted approach speed based on an inaccurate path prediction ($\bar{V}^{\mathrm{p}} = 0.17$.) Note that $\bar{V}^{\mathrm{p}}$ is much smaller than the previous example because of the high prediction error even though V^{p} is the same for both cases. For this threat, the larger $W_{V^{\mathrm{t}}}$ is, the larger the risk value is. If the weight parameters are evenly distributed, the risk values of these two threats would be similar. However, if a large value is assigned for $W_{V^{\mathrm{t}}}$, then the risk value of the threat in Figure 6.3b would be higher. As

shown in these examples, the weight parameters determine which parameters will dominate risk value estimation. This can be modified depending on the threat scheduling scenario.



(a) Risk values when $d^{\mathrm{t}} = 50$ m, $V^{\mathrm{t}} = 10$ m/s, $V^{\mathrm{p}} = 40$ m/s, $e^{\mathrm{p}} = 5$

(b) Risk values when $d^{\mathrm{t}} = 100$ m, $V^{\mathrm{t}} = 30$ m/s, $V^{\mathrm{p}} = 40$ m/s, $e^{\mathrm{p}} = 30$

Figure 6.3: Risk value changes with different weight parameters ($d_{\max} = 100$ m, $V_{\max} = 59$ m/s, $e^{\mathrm{p}}_{\max} = 40$)

6.3 Modified Traveling Salesman Problem

The threats can be scheduled for the central vision camera based on the risk values computed in the previous section. The central vision camera then maneuvers to observe the threats using a controller (e.g. PID controller), starting from the most prioritized threat to the least prioritized threat. However, if the threats are scheduled only based on risk values, the threat servicing schedule may be inefficient.

Figure 6.4 shows a planar example of the central vision camera maneuver. The triangles depict the FOV of the sensor. In Figure 6.4a, threats are prioritized only based on the risk values (T1 - T2 - T3); thus the sensor maneuvers starting from threat 1 and moves on to threat 2 and 3. However, the observation can be done in a faster way (T1 - T3 - T2), as

(a) Sensor maneuver only based on risk values

(b) Sensor maneuver based on risk values and the observation time

Figure 6.4: Central vision camera maneuver examples

shown in Figure 6.4b. In this way, threat observation can be conducted in a time-efficient way, so that more threats can be observed in the same time period, and the threat data can be updated more often.

The traveling salesman problem (TSP) involves finding the shortest path for a salesman who wants to visit each city in a set exactly once. Therefore, a normal TSP solution can be used to determine the fastest way to image all threats. The bearing of each threat is used as the position of the 'city' or 'node' for the salesman to visit. In order to estimate the fastest way to visit all n nodes, the cost to travel from threat i to threat j, c_{ij}, is firstly defined. Assuming the sensor moves between threats at a constant rate, and dwells on each threat for the same amount of time, the TSP solution minimizes the total cost of servicing all threats. For the fastest path of the salesman, an integer linear programming problem is formulated as below [183, 184]. A binary variable x_{ij} represents whether the salesman chooses to go from i to j (if the salesman goes from node i to j, x_{ij} is 1, 0 otherwise), and S is a subset

of nodes.

$$\text{Minimize} \quad \sum_{i=1}^{n} \sum_{i \neq j, j=1}^{n} c_{ij} x_{ij} \tag{6.13}$$

$$\text{s.t.} \quad i, j \in \{1, \ldots, n\} \tag{6.14}$$

$$x_{ij} \in \{0, 1\} \tag{6.15}$$

$$\sum_{i=1, i \neq j}^{n} x_{ij} = 1 \tag{6.16}$$

$$\sum_{j=1, j \neq i}^{n} x_{ij} = 1 \tag{6.17}$$

$$\sum_{i \in S} \sum_{j \in S} x_{ij} \leq |S| - 1 \tag{6.18}$$

$$S \subsetneq \{1, \ldots, n\} \tag{6.19}$$

$$2 \leq |S| \tag{6.20}$$

where $|S|$ represents the number of elements of the subset S. For the normal TSP formulation, the Euclidean distance between node i and j is used for c_{ij}. The purpose of the TSP in this chapter is to find the shortest maneuver of the central vision camera. The position of each threat is represented as bearing; thus an angular distance between threat i and j (ω_{ij}) is used for c_{ij} instead of an Euclidean distance (See Figure 6.5.)

$$\omega_{ij} = \frac{2 \sin^{-1} \sqrt{\sin^2(\frac{\Delta\theta_{ij}}{2}) + \cos\theta_i \cos\theta_j \sin^2(\frac{\Delta\psi_{ij}}{2})}}{\pi} \tag{6.21}$$

where $\Delta\theta_{ij}$ and $\Delta\psi_{ij}$ are the absolute difference of elevation and azimuth angles between threat i and j. Note that the angular distance is normalized with π as in Section 6.2 since the maximum angular distance is π. If more than two threats are inside the FOV of the central vision camera, the threats can be observed at once. Therefore, if the angular distance

between threats is smaller than the FOV angle of the central vision camera, the threats are considered as a single threat whose bearing bisects the circular arc between the two threats.

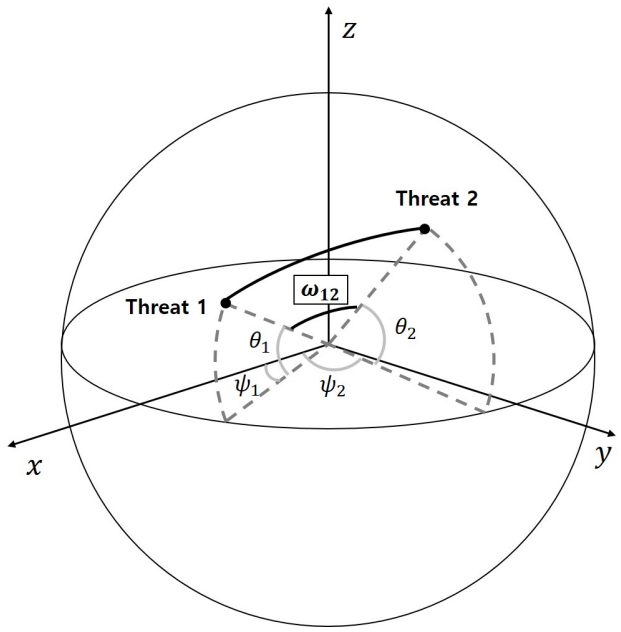


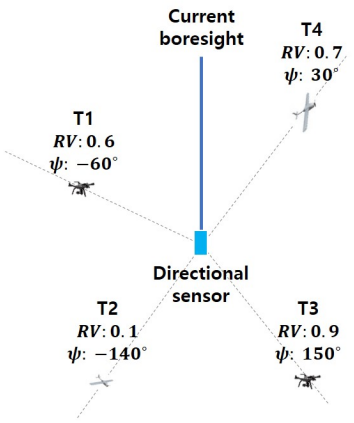
Figure 6.5: Angular distance between two threats

In addition to the observation time, RV should be included in c_{ij} as well. As shown in Figure 6.4b, if RV is similar for threats 2 and 3, then threat 3 can be observed on the way from threat 1 to threat 2. On the other hand, if RV of threat 3 is low (e.g. 0.1), threat 3 should be ignored since threat 2 is much more urgent. To implement this process, the cost of the TSP is modified.

$$c_{ij} = \omega_{ij}(1 - W_{RV}) + k_j W_{RV} RV_j \tag{6.22}$$

where k_j represents the scheduling order of threat j. For example, if threat j is assigned as the first threat to be observed, k_j is 1; if it has second priority, k_j is then 2. The parameter W_{RV} is a weight parameter for the balance between RV and ω, which is between 0 and 1. The greater W_{RV} is, the more RV is considered for threat scheduling in comparison to the

observation time efficiency.



k_j (Threat Order)	Current Position	1	2	3	4
j (Threat #)		3	4	1	2
RV_j		0.9	0.7	0.6	0.1
ω_{ij}		0.833	0.667	0.500	0.444
c_{ij}		**0.866**	**1.033**	**1.150**	**0.422**

$W_{RV} = 0.5$, Total cost: 3.472

k_j (Threat Order)	Current Position	1	2	3	4
j (Threat #)		4	3	2	1
RV_j		0.7	0.9	0.1	0.6
ω_{ij}		0.167	0.667	0.389	0.444
c_{ij}		**0.433**	**1.233**	**1.094**	**0.422**

$W_{RV} = 0.5$, Total cost: 3.433

k_j (Threat Order)	Current Position	1	2	3	4
j (Threat #)		4	3	1	2
RV_j		0.7	0.9	0.6	0.1
ω_{ij}		0.167	0.667	0.833	0.444
c_{ij}		**0.433**	**1.233**	**1.316**	**0.422**

$W_{RV} = 0.5$, Total cost: 3.405

Figure 6.6: Modified TSP example

Figure 6.6 shows a planar example of a modified TSP. There are 4 threats, each with a unique azimuthal position (azimuth angle ψ). The thick blue line represents the current boresight of the central vision camera. The sensor starts to move to observe each threat based on a threat schedule from this position. The table at the top shows the cost computation when the threat order is T3 - T4 - T1 - T2, which is a descending order of risk values. To compute c_{c3}, which is the cost from the current central vision camera position to the T3 position, k_3 is 1, since T3 is assigned as the first threat to observe, and ω_{c3} is 0.833 ($\frac{30°}{180°}$). Therefore, c_{c3} becomes $0.833 \times (1 - 0.5) + 1 \times 0.5 \times 0.9 = 0.866$. In the same way, all costs are computed up to c_{12}, which is $0.444 \times (1 - 0.5) + 4 \times 0.5 \times 0.1 = 0.422$. The total cost of the table at the top is 3.472. If the threat order is changed to T4 - T3 - T2 - T1 as shown in the table in the middle, the total cost decreases to 3.433, because the risk values of T4 and T3 are similar, so observing T4 on the way to T3 is more time-efficient. However, T2

is not an urgent threat ($RV = 0.1$); imaging of this threat could be postponed since T1 is a comparatively urgent threat. Therefore, if the order is changed as the table at the bottom (T4 - T3 - T1 - T2), the cost decreases to 3.405. These examples are using 0.5 for W_{RV}. If one wants to emphasize the observation time efficiency over the risk values by using 0.3 for W_{RV}, then the total cost of the middle case is 2.7269, and 2.8877 for the bottom case. That said, T2 is observed earlier than T1, even with its low RV in this case. Using the W_{RV}, one may choose the balance between RV and ω. As shown in the example above, the modified TSP solution can provide a reasonable solution for threat scheduling that considers both risk values and time-efficient observation. In the next section, the detailed threat scheduling algorithm design is described using the modified TSP solution and the risk value estimation.

6.4 Threat Scheduling Algorithm Design

Based on the risk value estimation and the modified TSP solution introduced in previous sections, the threat scheduling algorithm can now be presented. Figure 6.7 shows the threat scheduling flow chart for the PCV system. The peripheral vision camera provides continuous visual coverage so that it is able to keep providing bearing data of multiple threats. In the first observation cycle, however, the detailed threat data (position, velocity, and predicted path) is not available before the central vision camera observes threats. Thus the central vision camera needs to scan all threats quickly, and the normal TSP solution ($W_{RV} = 0$) is used for the fastest threat scanning.

Once the first observation cycle is over, the state and risk values of all threats are estimated. The estimated threat data is then saved to the "unobserved threat list", and the modified TSP is solved for the threats in the unobserved threat list. Once the threat schedule is determined, the central vision camera starts to slew to the first prioritized threat. Once the

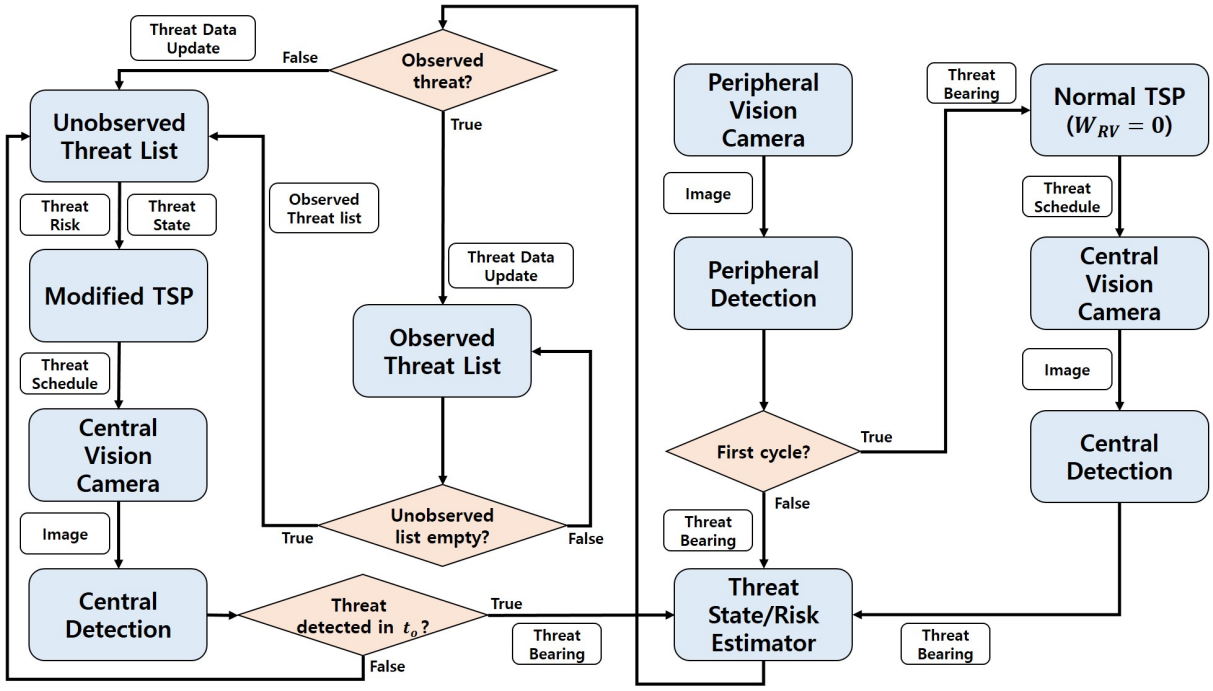


Figure 6.7: Threat scheduling flow chart for the PCV system

threat is detected by the central vision camera, the state (position, velocity, etc.) and the risk value of the threat are updated, and this threat is moved to the "observed threat list" from the unobserved threat list. The threat may be too fast for the central vision camera to track or obscured by obstacles, so that the central vision camera may not detect the threat. In order to consider this situation, the central vision camera moves to the next threat if the threat is not detected in the observation time limit (t_o). When a threat is observed by the central vision camera, the modified TSP is solved for the unobserved threats again. By doing this, even if a new threat appears (which is not detected in the first observation cycle), this new threat can be considered for scheduling immediately. If all threats are observed, and the unobserved threat list becomes empty, the unobserved threat list is reset as the observed threat list. This process keeps running until the host system is turned off. In the following section, the threat scheduling algorithm is implemented in a simulation to assess the performance of the suggested scheduling algorithm.

6.5 Simulation Setup

In this section, a 2D simulation (top view) is conducted to assess the performance of the suggested threat scheduling algorithm. For simplicity, it is assumed that the altitudes of all threats and the PCV system are the same. The threat scheduling process using the PCV system (Figure 6.7) is depicted in the simulation, and the simulation parameters are shown in Table 6.1. The maximum detectable range of the PCV system is 100 m, so $d_{\max}$ is defined as 100 m. All threats are assumed to be sUAS, and the typical racing drone speed of 59 m/s [185] is taken as $V_{\max}$. For $e^{\mathrm{p}}_{\max}$, 40 is chosen because estimated e^{p} using actual sUAS flight data [28] did not exceed 40. To apply the same weight for $\bar{D}$ and $\bar{V}^t$ for threat scheduling, W_D and $W_{V^{\mathrm{t}}}$ are defined as 0.4, and 0.2 is used for $W_{V^{\mathrm{p}}}$ to trust $\bar{D}$ and $\bar{V}^t$ more than $\bar{V}^p$.

Table 6.1: Simulation parameters

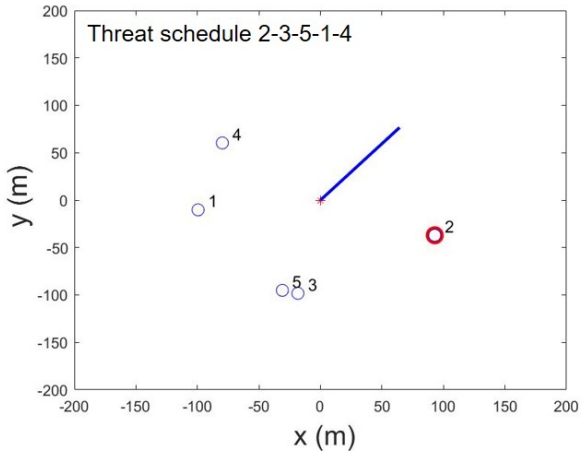
Parameters	Value
Simulation time	5 min
Threat initial range	80 - 100 m
Heading angle	$0 - 360°$
Gimbal maximum angular speed	40°/s
Number of threats	8
$d_{\max}$	100 m
$V_{\max}$	59 m/s
$e^{\mathrm{p}}_{\max}$	40
$W_D, W_{V^{\mathrm{t}}}$	0.4
$W_{V^{\mathrm{p}}}$	0.2

Figure 6.8 shows an example of the simulation. The simulation firstly generates threats with random data (position, speed, and pose) with the simulation parameters shown in Table 6.1 as in Figure 6.8a, and threats randomly change their pose at a random time during the simulation. In the figure, the red asterisk at the center indicates the PCV system, and

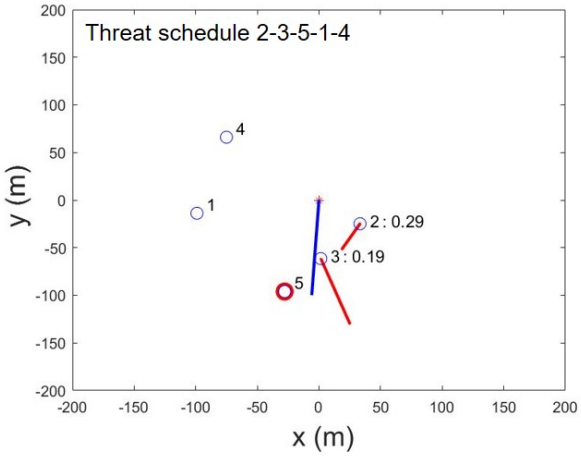
the threats are assumed to be detected by the peripheral vision camera. The only data obtained from the peripheral vision camera is bearing data so that the risk values of threats are not available in the first cycle. Based on the bearing data, the normal TSP solution ($W_{RV} = 0$) is used for fast threat scanning; thus the initial threat schedule is 2-3-5-1-4, which is the fastest way to observe all threats. The boresight of the central vision camera (a thick blue line) starts to move based on the schedule using a PD controller. Once a threat is observed (Figure 6.8b), the position, velocity, and predicted path (red lines) of the threat are estimated. (Note that the position of each threat has been updated to estimated positions in Figure 6.8b.) Having acquired the data, risk values are computed using the approach presented in Section 6.2, and the new threat schedule (4-1-5-2-3) based on the modified TSP solution ($W_{RV} = 0.5$) is then determined (Figure 6.8c). In this example, threat #4 has the highest risk value (this threat is approaching the PCV system from a close location); it thus became the first threat. Threat #1 became the second threat since it is quite close to threat #4. This quick observation is followed by threat #5 that has the second-highest risk value (0.27). Threat #2 became the next threat because threat #3 is not an urgent threat since it has a zero risk value.

6.6 Results

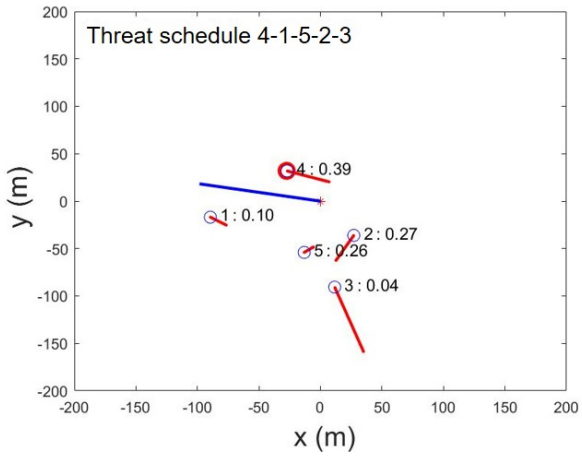
The weight parameter for the risk value (W_{RV}) determines the balance between risk values and time-efficient observation. This weight value plays an essential role in the modified TSP solution since the threat priority can differ depending on the choice of W_{RV} as shown in an example of Section 6.3. Figure 6.9 shows the modified TSP solution with the different W_{RV} in the threat scheduling simulation. The left figure illustrates when $W_{RV} = 0$, and the threat schedule is 2-4-1-5-3 that minimizes the scanning time. This approach enables fast threat



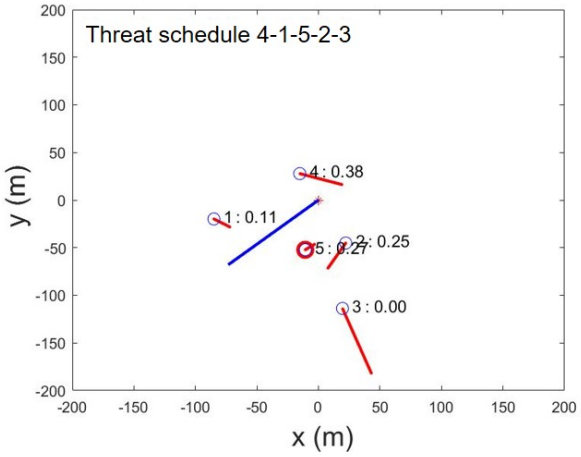
(a) Threats are generated at random positions, and the central vision camera starts to scan all threats (risk values are not available yet, so the initial schedule is not risk-weighted). The scanning threat schedule is 2-3-5-1-4.



(b) Once the central vision camera observes a threat, the position and the predicted path of the threat is estimated, and the risk value (a number next to the threat number) is computed based on the obtained threat data.



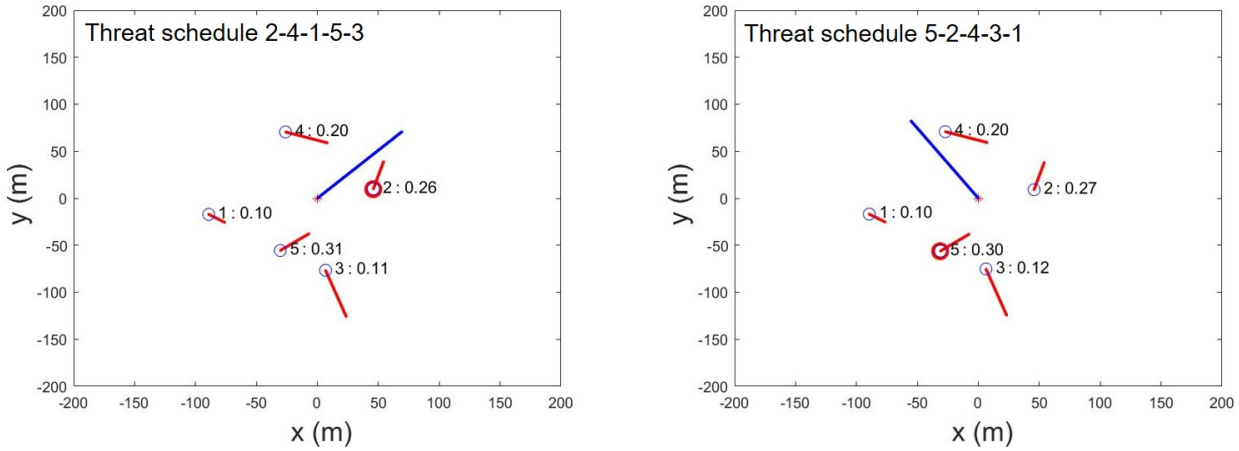
(c) Once the scanning is over, risk values of all threats are computed.



(d) The threat schedule is changed based on the modified TSP (new threat schedule: 4-1-5-2-3).

Figure 6.8: Threat scheduling simulation (red asterisk: PCV system, blue circles: threats, thick blue line: boresight of the central vision camera, red circle: next threat to observe, red lines: predicted threat paths, number: threat number/risk value)

scanning but maybe risky since threats with high-risk values may not be prioritized. The right figure depicts when $W_{RV} = 1$, which means that only risk values are considered for threat scheduling as other studies [112, 115, 120] did. This scheduling approach, however, may result in an inefficient maneuver of the central vision camera (5-2-4-3-1).

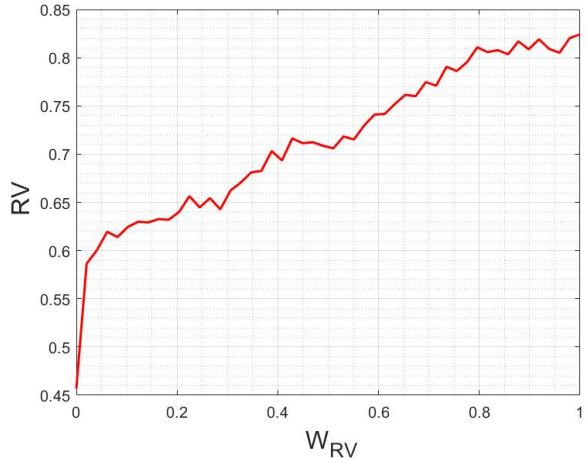


(a) Threat schedule using $W_{RV} = 0$ (b) Threat schedule using $W_{RV} = 1$

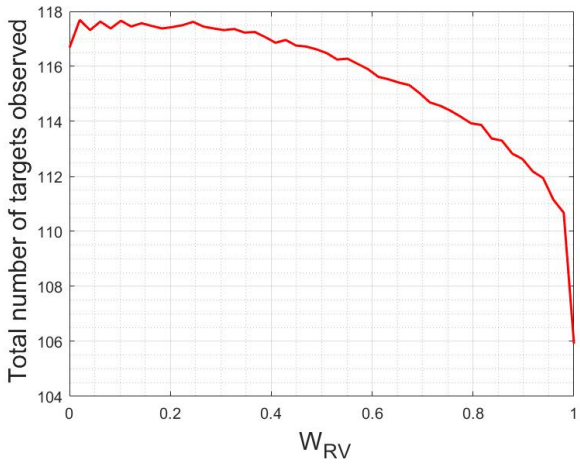
Figure 6.9: Modified TSP solution with different W_{RV} in the simulation (red asterisk: PCV system, blue circles: threats, thick blue line: boresight of the central vision camera, red circle: next threat to observe, red lines: predicted threat paths, number: threat number/risk value)

To assess the performance of the modified TSP algorithm for multiple threat scheduling, some simulations are conducted using the parameters shown in Table 6.1. The purpose of the simulations is to compare the performance between the existing scheduling algorithm ($W_{RV} = 1$) and the suggested algorithm ($0 < W_{RV} < 1$), and to see the effect of W_{RV}. For each simulation, the risk value summation for the initial 50% of prioritized threats (4 threats, since the total number of threats is 8), the number of threats observed during the simulation time (5 min), and the uncertainty of each threat measurement are logged, and this simulation is repeated with different W_{RV}. After that, 8 threats are generated at random positions again and the previous steps are repeated. This simulation is conducted 100 times,

and the results of simulations are averaged. Figure 6.10 shows the averaged values of two logged data for each W_{RV}. (The trends in the figures were similar to those obtained using 30 and 70 simulations, thus the authors conjecture that 100 simulations are sufficient to establish a trend.)



(a) Summed risk values for the initial 50% of serviced threats vs. W_{RV}



(b) Number of threats observed vs. W_{RV}

Figure 6.10: Performance with different W_{RV}

The left figure of Figure 6.10 represents the risk value summation for the initial 50% of prioritized threats. A high sum value indicates that the algorithm imposes a high priority to a threat with a high-risk value. When $W_{RV} = 0$, only the observation time is considered for scheduling, so that the risk value of prioritized threats is the lowest. This risk value almost linearly increases with W_{RV} when $W_{RV} > 0$ and peaks at $W_{RV} = 1$ because the threat schedule is decided just as a descending order of risk values when $W_{RV} = 1$. The right figure shows the total number of threats observed by the central vision camera. In an opposite way to the left plot, the total number of threats observed peaks at $W_{RV} \approx 0$ (because this is the fastest way to observe all threats) and almost linearly decreases with W_{RV}. When $W_{RV} = 1$, the total number of threats observed rapidly decreases. In this particular simulation, each threat could be observed once more when $W_{RV} < 0.8$ than when $W_{RV} = 1$, during 5-minute

simulation time.

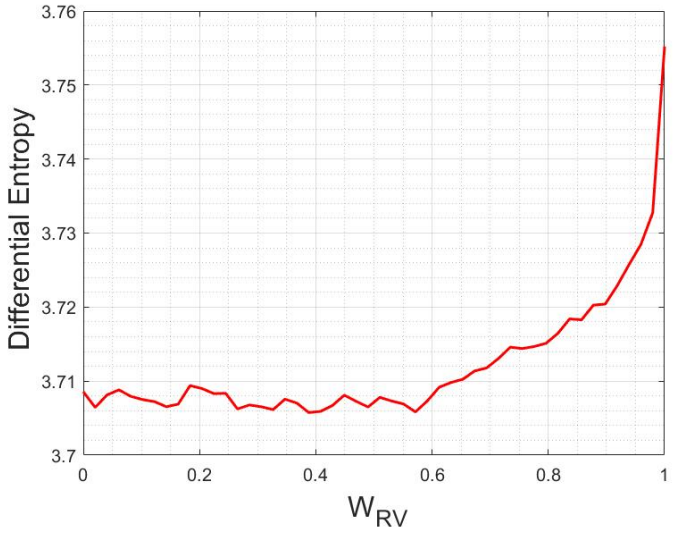


Figure 6.11: Differential entropy with different W_{RV}

If the central vision camera can observe more threats, the threat data can be updated more often, so that the uncertainty of threat data can be then decreased. To see the trends of the 2×2 covariance matrix of the threat position measurement, Σ, the differential entropy of Σ is computed:

$$H = \frac{1}{2}\log[(2\pi e)^2|\Sigma|] \tag{6.23}$$

Figure 6.11 shows the mean differential entropy of the threat positions versus W_{RV}. A high differential entropy represents a high measurement uncertainty in threat position. As shown in the figure, the differential entropy increased with W_{RV} and peaked at $W_{RV} = 1$, which means that the suggested algorithm is capable of decreasing measurement uncertainty of threats by updating the threat data more often than the existing scheduling algorithms. Considering (6.23), it is observed that the determinant of the covariance matrix decreased by 10% when $W_{RV} = 0$ compared to when $W_{RV} = 1$.

As shown in these figures, a low W_{RV} helps the central vision camera to move more efficiently

so that the central vision camera captures more threats in the same period of time, and the measurement uncertainty of threat data decreases. On the other hand, high W_{RV} prioritizes risky threats to be observed earlier, at some cost to the threat imaging efficiency. The modified TSP approach is capable of scheduling threats by considering both observation time efficiency and risk values when $0 < W_{RV} < 1$, and the performance can be adjusted by changing the value of W_{RV}. One may use a high value of W_{RV} to emphasize risk over efficiency, or a low value of W_{RV} to observe more threats faster.

6.7 Summary

In this chapter, a novel multiple threat scheduling algorithm using a modified TSP solution for the PCV system is presented. While various threat scheduling algorithms have been suggested in previous studies, the observation time efficiency has not been considered. A mobile sensing system, such as the peripheral-central vision system mentioned here, has a limited power source, and the detectable range of the system may be comparatively short so that reacting to threats quickly is a high priority. In order to efficiently schedule threats detected by the peripheral vision camera, a modified TSP algorithm is introduced, and the suggested algorithm is implemented in simulations. The simulation results show that the modified TSP algorithm is capable of prioritizing the threats by considering both the time efficiency and the risk values, which existing threat scheduling algorithms do not do. The results also show that the measurement uncertainty of threat data can be decreased by updating the threat data more often.

Chapter 7

Optimal Placement Algorithm for Multiple PCV Systems

7.1 Introduction

As shown in Chapter 4, a short camera baseline generates a large localization error for distant threats. One way to address this issue is to add host aircraft with their own PCV systems so that the camera baseline is no longer limited. Although a long baseline can decrease localization error, however, it does not guarantee small localization error. If the threat is far from the host aircraft, or if the encounter geometry is degenerate, then the localization error may still be large. It is therefore of interest to determine optimal positions for additional host aircraft to minimize threat localization error for C-UAS applications.

The heterogeneous stereo-vision optimal placement (HSOP) algorithm, which enables mobile sensing systems that use heterogeneous stereo vision to minimize the localization error of a threat aircraft is developed in this chapter. I begin by constructing distinct models for the peripheral vision camera and the central vision camera. Each type of camera has a different FOV and detectable range. The peripheral vision camera has low image resolution that is nonuniform over its FOV so the localization error using this camera is comparatively larger than that obtained using a central vision camera. Because the peripheral vision camera has a large FOV, however, it can image multiple threat aircraft simultaneously. Conversely, the

central vision camera can provide better localization accuracy, but the narrow FOV makes it less likely to image multiple threats at once. The HSOP algorithm determines the minimum number of PCV-equipped aircraft required to localize a given number of threat aircraft as well as the optimal locations for those host aircraft and the specific camera (peripheral or central) that each should employ to minimize the cumulative localization error by solving a mixed-integer nonlinear programming problem. The algorithm inputs are the number of threats and a preliminary estimate of their position and velocity. It is assumed that sufficiently many host aircraft are available to accurately localize all threats. This work is presented in [21].

7.2 Camera models and triangulation-based localization

The host aircraft is assumed to be equipped with the PCV system described in previous chapters. The peripheral vision camera comprises two 180°-FOV fisheye lenses to provide a large FOV. The central vision camera has a 53° FOV and is capable of pan-tilt operation. Table 3.1 shows the specifications for the two cameras. Although a specific camera setup is considered in order to describe numerical and experimental results, the analysis described here can be easily extended to a large class of cameras.

7.2.1 Peripheral vision camera model

For the peripheral vision camera model, I use a fisheye lens model available in the OpenCV library [186, 187] which accounts for the refraction of light rays passing through the lens, an effect called lens distortion. As an example, Figure 7.1 illustrates how objects at the edge

of the image are highly distorted so that they appear smaller than if they appeared at the center of the image.

Figure 7.1: Lens distortion of a fisheye lens

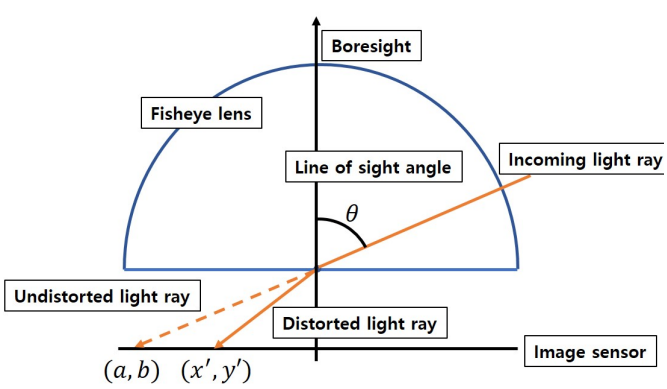
Computer vision based detection algorithms require that the detected object appear in the image with some minimum number of associated pixels. For example, optical flow requires at least 25 px (5 px wide $\times$ 5 px high) to detect a moving object in the image. However, the pixel size of the object on the image can differ depending on the line of sight (LOS) angle to the object. An object of interest that occupies 25 px at the center of the image would occupy fewer pixels at the edge of the image, and hence be undetectable using optical flow. The pixel size of a threat aircraft on the fisheye camera image should be larger than the minimum number of pixels that the detection algorithm requires for a detection. Following is a model for the distortion of a light ray that enters the fisheye lens with line of sight (LOS) angle θ relative to the camera boresight, as shown in Figure 7.2a:

$$\theta_d = \theta(1 + k_1\theta^2 + k_2\theta^4 + k_3\theta^6 + k_4\theta^8) \tag{7.1}$$

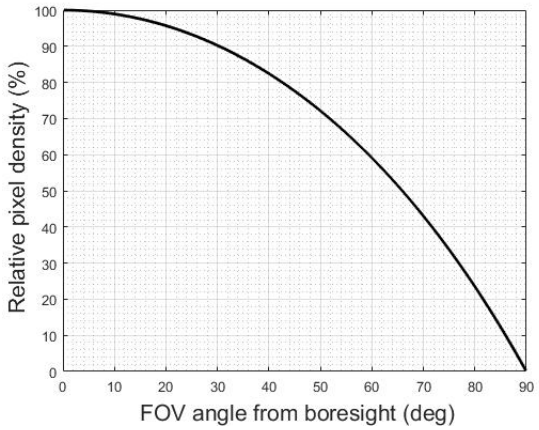
The parameters k_i for $i \in \{1, 2, 3, 4\}$ are the lens distortion coefficients obtained through a camera calibration process [186, 187]. The (distorted) light ray is then projected onto the image pixel coordinates u and v:

$$u = \frac{\theta_d}{\tan\theta} a \quad \text{and} \quad v = \frac{\theta_d}{\tan\theta} b \tag{7.2}$$

where a and b represent the pixel coordinates if the incoming light ray were not distorted by the lens. These equations illustrate how a larger LOS angle results in smaller pixel sizes for objects near the edge of the image. Figure 7.2b shows the relative pixel size of an object in the image versus the LOS angle. At the lens center, there is no distortion, so the relative pixel size is 100%; the relative pixel size decreases to 0% at a 90° LOS angle.



(a) Fisheye distortion geometry



(b) Fisheye pixel density against FOV angle

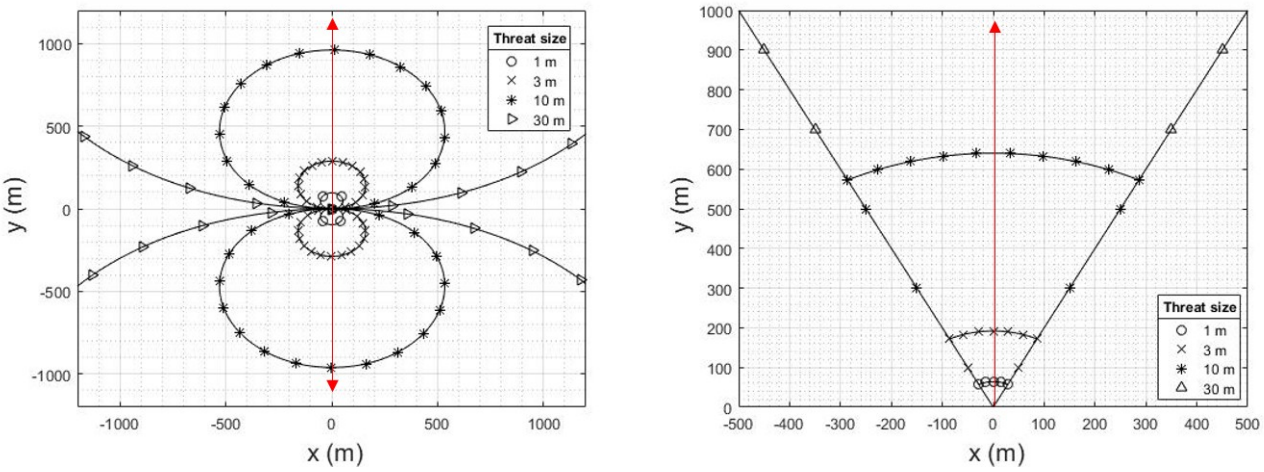
Figure 7.2: Fisheye lens distortion

The pixel size of an object increases when the object moves closer to the camera. The *detectable region* is defined here for a given camera with respect to an object of given size and location relative to the camera boresight. The pixel width of an object represented in

the image can be estimated as:

$$\alpha_{\mathrm{p}} = \frac{f^{\mathrm{p}} W_{\mathrm{t}} \gamma^{\mathrm{p}} \theta_d}{d_{\mathrm{t}} W_s^{\mathrm{p}} \tan\theta} \tag{7.3}$$

where W_{t} is the width of the actual object (the "threat"), f^{p} is the focal length of the peripheral vision camera, γ^{p} is the horizontal resolution (in px) of the peripheral vision camera, d_{t} is the distance to the object, and W_s^{p} is the (physical) width of the peripheral vision camera's sensor. The same equation can be used for the pixel height of the object. The pixel width and height should be larger than the minimum number of pixels required by the detection algorithm.



(a) Detectable region of the peripheral vision camera (red arrows: boresight of two lenses)

(b) Detectable region of the central vision camera (red arrow: boresight of lens)

Figure 7.3: Detectable region of two cameras

Figure 7.3a shows regions in which objects of different size can be detected by the peripheral vision camera using optical flow. (A qualitatively similar figure would result for any fisheye camera; the specific bounding curves depend on the given camera parameters.) As the peripheral vision camera has two fisheye cameras, there are two boresight directions, which are indicated as arrows in the figure. The plot illustrates the detectable region for an object

that is at least as large as indicated in the legend. The figure illustrates the role of object size and LOS angle in the detectability of an object.

7.2.2 Central vision camera model

The central vision camera is a perspective camera. The lens distortion is small and the projection error can be corrected through camera calibration. The pixel size of an object at a given distance therefore remains constant throughout the camera's FOV, in contrast to the peripheral vision camera. A pinhole camera model is adopted for the central vision camera. As shown in Figure 7.4, the pinhole camera model assumes no lens distortion, so the pixel width of an object is proportional to its size:

$$\alpha_{\mathrm{c}} = \frac{f^{\mathrm{c}} W_{\mathrm{t}} \gamma^{\mathrm{c}}}{d_{\mathrm{t}} W_s^{\mathrm{c}}} \tag{7.4}$$

where f^{c} is the focal length of the central vision camera, γ^{c} is the horizontal resolution of the central vision camera, W_s^{c} is the width of the central vision camera sensor. The threat pixel height can be computed in the same way. The pixel width and height of an object must be larger than the minimum number of pixels needed by the central vision detection algorithm.

Because the central vision camera provides a high resolution image, the pixel size of an imaged object may be sufficient to enable classification using a deep neural network algorithm such as YOLO [154, 155]. It is assumed that YOLO is used for central vision detection as described in previous chapters. Earlier work indicates that YOLO requires at least 900 px (30 px wide $\times$ 30 px high) to detect a threat aircraft, so the threat pixel width and height should be larger than 30 px. The detectable region of the central vision camera when YOLO is used for detection is illustrated in Figure 7.3b. Also, because the FOV of the central vision

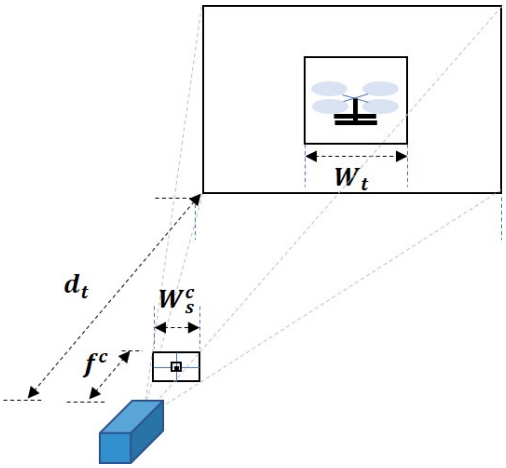


Figure 7.4: The central vision camera model geometry

camera is limited, the bearing angle between the camera boresight and the object should be no more than half the central vision camera's FOV angle.

7.2.3 Triangulation-based localization and error

Recall that the threat localization accuracy differs depending on the camera location and the threat distance (Chapter 4.) If the threat is close, the localization error will decrease; if the distance between the two cameras (the baseline) is too small, the localization error will increase. Reference [136] gives an expression for this triangulation-based localization error:

$$\delta_r = \frac{\left\|\vec{r}_{\mathrm{t/p}}\right\| \left\|\vec{r}_{\mathrm{t/c}}\right\|}{\sin\theta}\delta_s \tag{7.5}$$

where δ_s is the magnitude of the error in the threat location $\vec{r}_{\mathrm{t/g}}$ and θ is the intersection angle between the threat vectors from the two cameras:

$$\theta = \cos^{-1}\left(\frac{\vec{r}_{\mathrm{t/p}} \cdot \vec{r}_{\mathrm{t/c}}}{\left\|\vec{r}_{\mathrm{t/p}}\right\| \left\|\vec{r}_{\mathrm{t/c}}\right\|}\right)$$

See Figure 7.5. Equation (7.5) implies that localization error is minimum when the distance from the cameras to the threat is small and when the threat vectors intersect at a right angle.

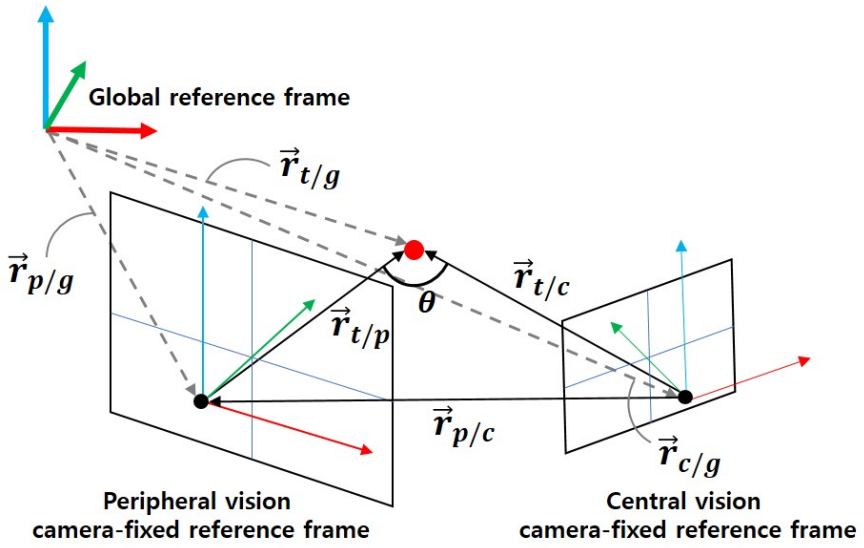


Figure 7.5: Geometry of threat position estimation

7.3 Optimal placement algorithm for multiple heterogeneous stereo vision systems

Considering the two camera models discussed in the previous section, this section describes a heterogeneous stereo vision optimal placement (HSOP) algorithm. The algorithm determines the optimal positions for two or more host aircraft, each equipped with a PCV system, as well as the type of camera to be used, in order to cover the current position (given) and final position (predicted over a given time horizon) of one or more threat aircraft, minimizing the cumulative localization error. The host aircraft are assumed to take the computed positions and to hover there, imaging the threat aircraft over the given time horizon, after which the

optimal host positions are updated by re-applying the HSOP algorithm. The host aircraft continue tracking the threat aircraft until they are no longer a threat. Note that the focus of this chapter is the placement of sensors for optimal localization – the "fix" portion of the "find, fix, finish" chain in a counter-UAS strategy.

The output of the HSOP algorithm can thus be considered as a waypoint generation scheme for host aircraft. Motion of the host aircraft can degrade the performance of camera-based threat detection algorithms. While there are ways to compensate for ownship motion, such as physical or software image stabilization [17], observing and tracking a threat from a static position produces higher quality imagery which improves detection and localization performance.

This section presents the HSOP algorithm in detail. The algorithm requires the initial state (position and velocity) for each threat aircraft. While this requirement may seem circular, the initial threat state might be obtained (with lower accuracy) using a single PCV-equipped host that serves as a counter-UAS sentry in a given region, as described in Chapter 4. The optimization problem is formulated under additional simplifying assumptions outlined below, though each of these can be relaxed with some effort:

- All host and threat aircraft fly at the same altitude.
- The number of available host aircraft is sufficient to cover all threat aircraft.
- The threat aircraft follow a straight path at constant speed (i.e., they are non-cooperative, but non-antagonistic).
- The path traveled by a threat aircraft during the observation time horizon is contained within the detectable range of the imaging sensors.
- Host aircraft can instantly attain waypoints indicated by the HSOP algorithm (i.e.,

the physical placement of host aircraft occurs much faster than the optical motion of threat aircraft).

- The host aircraft can communicate with sufficient speed and volume to cooperatively localize threats.
- All PCV systems are identical; see Table 3.1.

7.3.1 Initial feasibility check

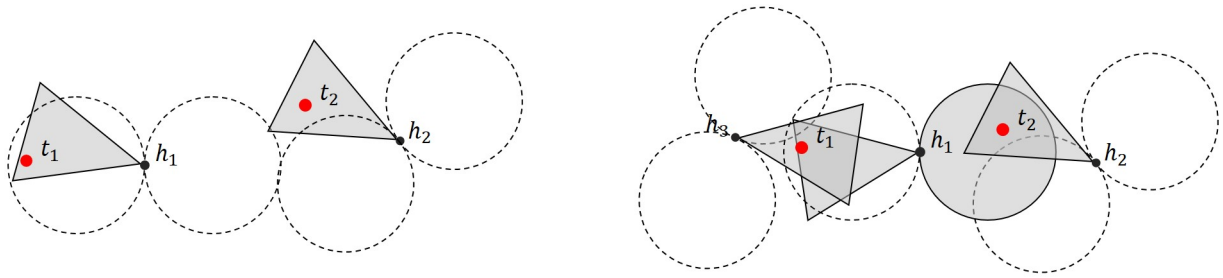
Given initial states for each threat aircraft in a given set T, two host aircraft h_i are assigned to each threat aircraft $t_j \in T$ to enable triangulation. In this way, a *host-threat pair* s comprising the threat and the two assigned host aircraft is generated for each threat aircraft. As an example, suppose there are two threat aircraft and three host aircraft. The nearest unassigned host is first assigned to each given threat as an "initial observer" and that host is labeled accordingly: (h_1, t_1) and (h_2, t_2). The pairs are then completed by assigning a second host aircraft to each threat. A *host-threat pair set* (or just *pair set*) S comprises one host-threat pair for each threat aircraft. For the given example, one possible pair set is $S = \{(t_1, h_1, h_2), (t_2, h_2, h_1)\}$. In this case, only the two host aircraft (h_1, h_2) are used to observe both of the two threat aircraft. Given three available host aircraft for the two detected threats, there are three more possible pair sets: $\{(t_1, h_1, h_2), (t_2, h_2, h_3)\}, \{(t_1, h_1, h_3), (t_2, h_2, h_1)\}, \{(t_1, h_1, h_3), (t_2, h_2, h_3)\}$.

If the number of host and threat aircraft is large, the number of possible pair sets scales as $(N_\mathrm{h}-1)^{N_\mathrm{t}}$, where N_h is the number of the host aircraft and N_t is the number of threat aircraft. To ameliorate this scaling issue, a backtracking approach is used to exclude infeasible pair sets. Algorithm 1 below presents pseudo-code for generating pair sets. Algorithm 2 then culls this set by excluding infeasible pair sets.

Algorithm 1 `Generate-Pair-Sets`(t_i, N_h, SP)

```
tmpSet = {};
for i to N_h do
    // h_f: initial observer of t_i
    if h_i ≠ h_f then
        tmpSet.insert({t_i, h_f, h_i})
    end
end
SP ← Generate-Combinations(tmpSet,SP);
return SP;
```

In Algorithm 1, SP indicates the set of pair sets, `Generate-Combinations` is a combination generation function that can be implemented using, for example, the `nchoosek` function in MATLAB. This function makes combinations of sets in SP and pairs in `tmpSet` and adds the resulting sets to SP.



(a) Infeasible host-threat pair set (b) Feasible host-threat pair set

Figure 7.6: Examples of infeasible and feasible host-threat pair sets

Next, the feasibility of pair sets in SP is determined. An infeasible pair set is one that includes host-threat pairs which are physically impossible. Consider Figure 7.6a, for example, where circles and triangles illustrate the detectable regions of peripheral and central vision cameras,respectively; recall Figures 7.3a and 7.3b. (Dashed lines indicate cameras that are not used for localization.) For the given configuration, the pair set $\{(t_1, h_1, h_2), (t_2, h_2, h_1)\}$ is infeasible because images from at least two host aircraft are needed to triangulate each threat and t_1 and t_2 cannot be covered by h_1 and h_2 simultaneously. On the other hand, the

pair set $\{(t_1, h_1, h_3), (t_2, h_2, h_1)\}$ depicted in Figure 7.6b is feasible.

A simple initial screen for feasibility involves checking whether the assigned threats can be placed within the detectable region of a candidate host's peripheral or central vision camera. In the example of Figure 7.6a, two threat aircraft are assigned to h_1 and h_2, but the distance between two threats is large so that the detectable region of a peripheral vision camera cannot cover both threats at once. In this case, at least four hosts are required to cover each threat from two perspectives. On the other hand, the two threats in Figure 7.6b are closer, which enables h_1 to cover both threats simultaneously; only three hosts are required to cover each threat from two perspectives. In the example of Figure 7.6, the threat aircraft are static. For a moving threat t_i, the assigned host aircraft must be able to cover the path connecting the current threat position $\vec{x}_{t_i}$ to the predicted position after some specified time τ:

$$\vec{x}_{t_i}^{\tau} = \vec{x}_{t_i} + \tau \vec{v}_{t_i} \tag{7.6}$$

where $\vec{v}_{t_i}$ is the threat velocity vector. Rather than require continuous coverage, however, a simpler strategy is adopted and the final threat position is added to the pair s as if it is another threat aircraft to be included in the optimization process. (Given an initial threat pairing $s_1 = (t_1, h_1, h_2)$, for example, an *augmented* threat pairing $s_1^{\tau} = (t_1, t_1^{\tau}, h_1, h_2)$ is defined, where t_1^{τ} denotes the location of threat t_1 after time τ has elapsed.)

For each host h_i, a set T^{h_i} is defined which contains all assigned threats, at both their initial and predicted final positions. The algorithm checks if all threat positions in T^{h_i} can be placed inside the detectable range of the peripheral vision camera (e.g. using the `inpolygon` function in MATLAB). If this is possible for all h_i in the pair set, the pair set is declared feasible. For a feasible pair set, the camera type of each host in the pair set is next determined. If the threats assigned to h_i can all be contained within the detectable region of the central vision camera, then this camera type is chosen for h_i. If they cannot, then

the peripheral vision camera is chosen. Even if a host aircraft's peripheral vision camera can cover all assigned threats, it may also be possible to cover one of these threats using the central vision camera, as shown in Figure 7.6. In this case, because of its greater resolution, the central vision camera is used to observe the threat. If multiple threats can be covered by the central vision camera, then the closest threat is observed by the central vision camera. Pseudo-code for this process is given in Algorithm 2. This initial feasibility check speeds the backtracking process described in the following section.

Algorithm 2 `Feasibility-Check`$(\mathrm{N_h}, T, SP)$

```
for t_i in T do
    SP ← Generate-Pair-Sets(t_i, N_h, SP);
    for each pair set S from SP do
        if S is feasible then
            Determine-Camera-Type(S)
        else
            SP.erase(S)
        end
    end
end
return SP;
```

7.3.2 Backtracking process to remove infeasible branches

As mentioned in the previous section, the possible number of pair sets is $(N_\mathrm{h} - 1)^{N_\mathrm{t}}$. It is expedient to remove all remaining host-threat pair sets that are infeasible before formulating and solving the optimal placement problem. A backtracking approach is used to complete the culling of infeasible pair sets. Backtracking is a constructive algorithm for generating feasible solutions. The process can be visualized as a tree structure in which candidate solutions are the branches that remain after pruning infeasible branches that do not satisfy constraints. Figure 7.7 shows an example of the backtracking process when there are three threat aircraft (t_1, t_2, t_3) and four host aircraft (h_1, h_2, h_3, h_4). Each column shows possible

host-threat pairs for each threat aircraft, starting with threat t_1 at the left. In this case, there are 27 possible combinations for pair sets. Infeasible pair sets are removed through the backtracking process. First, the feasibility check is conducted only for pair combinations of t_1 and t_2. There are 9 possible combinations involving t_1 and t_2 pairs. If one of those combinations is infeasible, the combination is excluded (pruned). In Figure 7.7, for example, the combination $\{(t_1, h_1, h_2), (t_2, h_2, h_1)\}$ is declared infeasible because of the threats t_1 and t_2 are too distant, as in Figure 7.6a. Downstream combinations involving threat t_3 are generated only for feasible host-threat pairings.

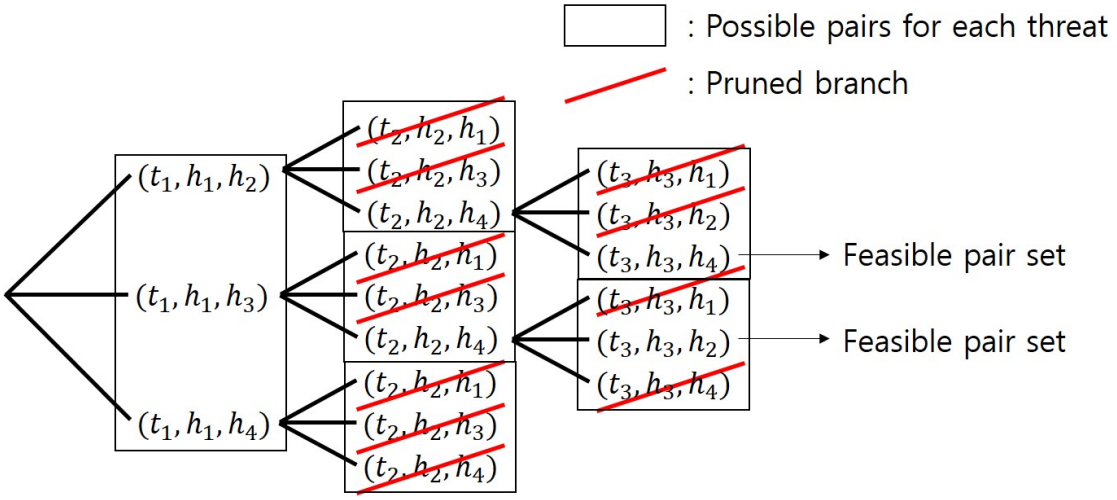


Figure 7.7: Example of backtracking process

7.3.3 Optimization problem statement

Once feasible pair sets are chosen from the backtracking process described in the previous section, the optimization problem is solved for each pair set S. Recall that a host-threat pair s consists of a threat aircraft and two assigned host aircraft and that a pair set S contains a complete set of host-threat pairs, i.e., one and only one pair for each threat. The aim is to minimize the localization error (the sum squared error between the ground

truth threat position vector and the estimated threat position vector) of all threat aircraft in a pair set S. The camera type of the host aircraft h_i is distinguished in the formulation using a binary value $c_{\mathrm{p}}^{h_i}$. If the camera type to be used by host h_i is the peripheral vision camera, then $c_{\mathrm{p}}^{h_i} = 1$; if the camera type to be used is the central vision camera, $c_{\mathrm{p}}^{h_i} = 0$. As described in Section 7.2.3, the intersection angle $\theta_s^{t_j}$ of two threat vectors should be close to 90° to minimize the triangulation-based localization error and the distance $d_{t_j}^{h_i}$ to the threat from the host aircraft should be small. Also, the pixel width ($\alpha_{\mathrm{p}}^{h_i t_j}$ and $\alpha_{\mathrm{c}}^{h_i t_j}$) of the threat aircraft in the peripheral and central vision imagery of host h_i should be larger than the minimum number of pixels (γ_{p} and γ_{c}) required by the peripheral and central vision detection algorithms to ensure detection. To simplify analysis, it is assumed that the width and height of all threat aircraft are equal and that all aircraft operate at the same altitude. In this case, only the threat pixel width in the formulation is considered. Finally, a mixed-integer nonlinear programming problem to find the set X_h of host position vectors which minimizes the localization error of all threat aircraft is formulated below.

$$X_h^* = \arg\min_{X_h} \quad \sum_{s\in S}\sum_{t_j\in s}\left\{\frac{\prod_{h_i\in s}\left\{\left[c_{\mathrm{p}}^{h_i}/\alpha_{\mathrm{p}}^{h_i t_j} + (1-c_{\mathrm{p}}^{h_i})/\alpha_{\mathrm{c}}^{h_i t_j}\right] d_{t_j}^{h_i}\right\}}{\sin\theta_s^{t_j}}\right\} \tag{7.7}$$

$$\text{s.t.} \quad (c_{\mathrm{p}}^{h_i}\alpha_{\mathrm{p}}^{h_i t_j} - \gamma_{\mathrm{p}}) + (1-c_{\mathrm{p}}^{h_i})(\alpha_{\mathrm{c}}^{h_i t_j} - \gamma_{\mathrm{c}}) > 0, \quad h_i \in s,\ t_j \in s,\ s \in S \tag{7.8}$$

$$(1-c_{\mathrm{p}}^{h_i})\left(\frac{\psi_{\mathrm{c}}^{h_i}}{2} - |\psi_{t_j}^{h_i}|\right) \geq 0, \quad h_i \in s,\ t_j \in s,\ s \in S \tag{7.9}$$

$$d_{t_j}^{h_i} \geq R_{\mathrm{s}}, \quad h_i \in s,\ t_j \in s,\ s \in S \tag{7.10}$$

$$c_{\mathrm{p}}^{h_i} \in \{0,1\}, \quad h_i \in s,\ s \in S \tag{7.11}$$

where $\psi_{\mathrm{c}}^{h_i}$ is the FOV angle of the central vision camera for host h_i, $\psi_{t_i}^{h_i}$ is the azimuth angle between h_i and t_i, and R_{s} is a safety distance between host and threat aircraft to prevent

collisions. (It is assumed that collisions between hosts and maneuvering threats are prevented by an over-riding collision avoidance protocol. The objective function (7.7) determines the host locations that maximize the pixels in the image that correspond to the threat aircraft and minimize the error in (7.5). Constraint (7.8) ensures the minimum number of "threat pixels" for detection. Constraint (7.9) ensures that the threat is within the FOV of the central vision camera, if that camera is to be used. The minimum distance between the host aircraft and the threat aircraft is constrained to be larger than the safety distance R_{s} in (7.10). The optimization problem defined above is solved using the interior-point method for each feasible pair set. An example pair set is illustrated in Figure 7.8.

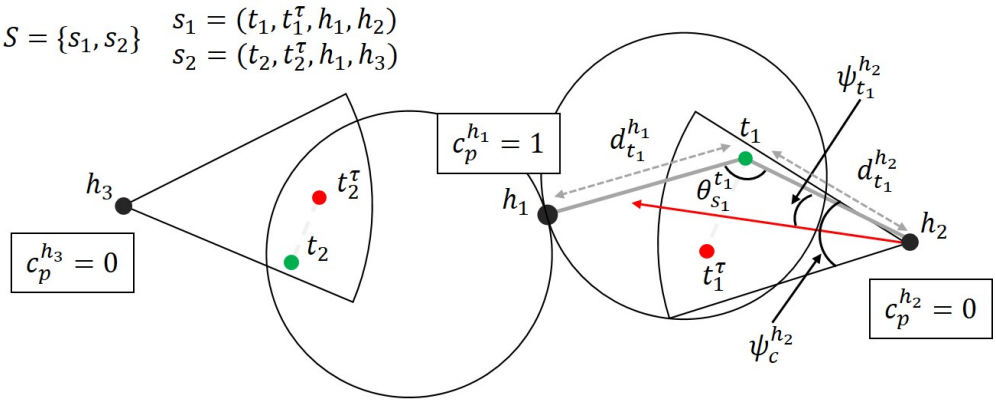


Figure 7.8: Geometry of a pair set

The HSOP algorithm is given in Algorithm 3. To minimize the number of host aircraft involved, the HSOP algorithm initially generates pair sets using only initial observers ($N_{\mathrm{h}} \leq N_{\mathrm{t}}$). If there exists no feasible pair set in SP, the algorithm adds host aircraft until a feasible pair set is found. (Note that the maximum number of hosts N_{h} required to image N_{t} threats is $2N_{\mathrm{t}}$.) Once a feasible set is found, the optimization problem is solved using the pair set. If the cost J of the resulting placement set X_h is smaller than the minimum cost seen so far, and if X_h is determined to satisfy all constraints (7.8)-(7.11) using the `Constraints-Check` function, then the optimal placement set X_h^* and cost J^* are updated

Algorithm 3 `HSOP algorithm`$(\mathrm{N_h}, T)$

```
SP = {};
while X_h^* is empty do
    SP ← Feasibility-Check(N_h, T, SP)
    if SP is not empty then
        for each pair set S in SP do
            [X_h, J] ← Solve-Problem(S)
            if Constraints-Check(X_h) & J < J* then
                X_h^* ← X_h;
                J* ← J;
            end
        end
    else
        N_h ← N_h + 1
    end
end
return X_h^*;
```

accordingly. The resulting X_h^* indicates the optimal locations for the minimum number of host aircraft which cover the current and final positions of all threat aircraft and which also minimize the cumulative threat localization error, including current and final threat positions. In the next iteration, final threat state estimates are taken as initial states and the HSOP algorithm computes the new optimal positions for the host aircraft.

7.4 Results

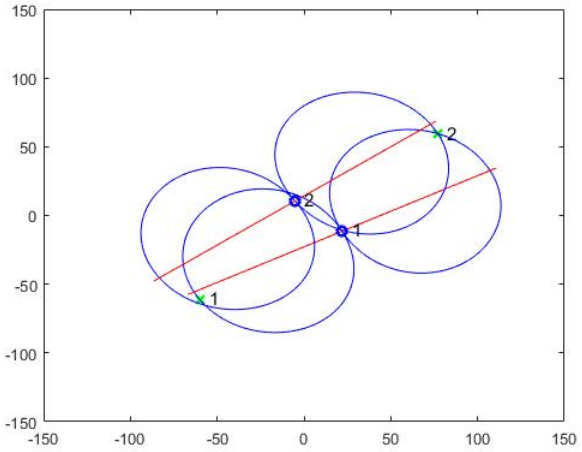
Because of the logistical challenges associated with multi-host, multi-threat detection and localization experiments, simulations were used to assess the performance of the HSOP algorithm. In these simulations, the HSOP algorithm is used to compute host aircraft positions corresponding to randomly generated threat states (positions and velocities). The localization error for each threat aircraft position is then estimated. (The localization error is the sum squared error between the ground truth threat position and the estimated threat

position.) The simulation parameters are shown in Table 7.1. To simulate the localization error, a zero-mean Gaussian noise is superimposed to the threat image in the horizontal and vertical directions with a 30 px standard deviation for the peripheral vision camera and a 3 px standard deviation for the central vision camera. (This empirically tuned synthetic error closely matches the error seen in experimental measurements.) The localization error is then estimated using the noise-corrupted threat vectors. This process is repeated 10,000 times for each simulation and the localization error is averaged. It is concluded that 10,000 data points are enough to assess the localization error with the assumed standard deviation because a similar averaged error is obtained using 1000 data points. References [121, 122, 123, 124, 125, 126, 127, 128] provide host aircraft placement algorithms for threat coverage, but localization error of threats is not considered in determining host placement. To illustrate how well the HSOP algorithm reduces localization error compared with algorithms that consider only threat coverage, the HSOP algorithm is compared with such a "coverage only (CO)" approach obtained by eliminating the localization cost from the HSOP algorithm and using the same constraints (7.8)-(7.11).

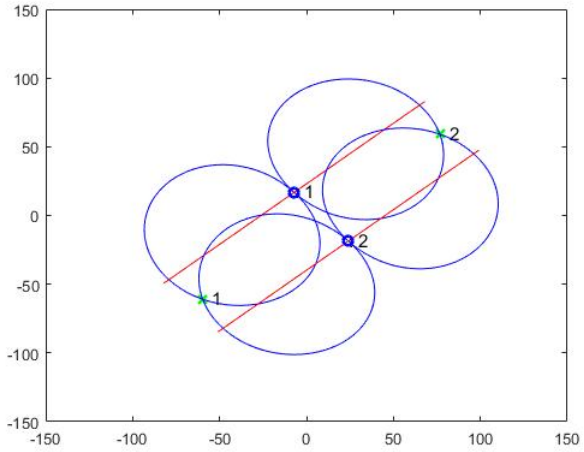
Table 7.1: Simulation parameters

Parameter	Value
Threat speed (m/s)	0 - 5
Prediction horizon (sec)	10
R_{s} (m)	20
γ_{p} (px)	5
γ_{c} (px)	30
$\psi_{\mathrm{c}}^{h_i}$ (°)	53

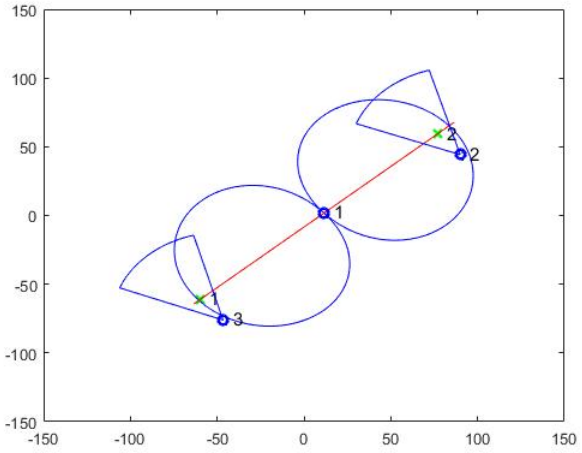
Figure 7.9 shows results of the CO approach and the HSOP algorithm in a sample simulation using two static threat aircraft. In the CO approach, shown in Figure 7.9a, the algorithm successfully covers the two threat aircraft using two host aircraft, but this algorithm does not consider the localization error that results when the hosts triangulate the position of the



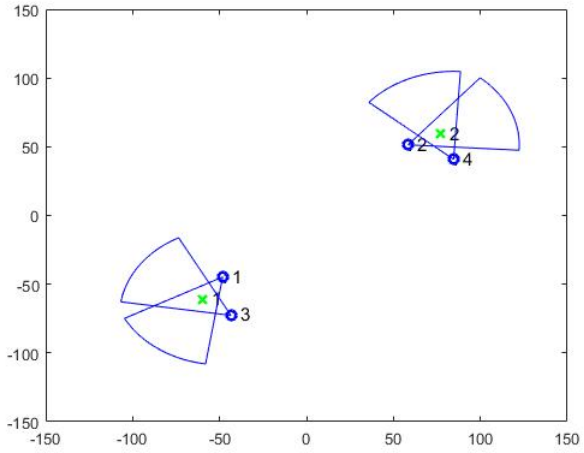
(a) Placement of two host aircraft using the CO approach



(b) Placement of two host aircraft using the HSOP algorithm



(c) Placement of three host aircraft using the HSOP algorithm



(d) Placement of four host aircraft using the HSOP algorithm

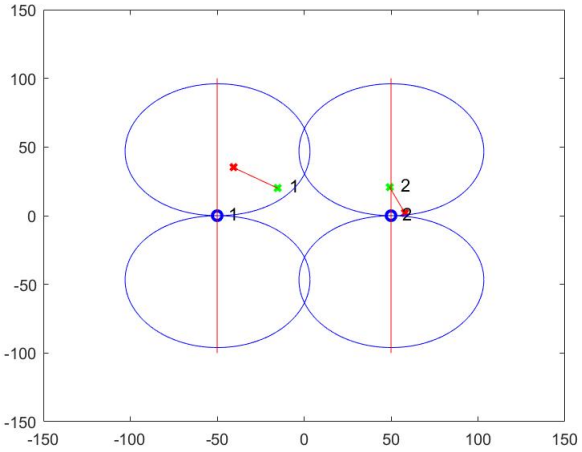
Figure 7.9: Simulation results for optimal host aircraft placement using two approaches for static threats. (Small blue circles: host aircraft positions. Green x marks: threat positions. Red lines: boresight directions of peripheral vision cameras. Large blue circles: detectable regions of peripheral vision cameras. Blue circular sectors: detectable regions of central vision cameras.)

threats. The HSOP algorithm, whose results are shown in Figure 7.9b, simultaneously covers the threat aircraft, with a minimum distance to the threats, and maximizes the intersection angles of the threat vectors from the host aircraft to the two threats.

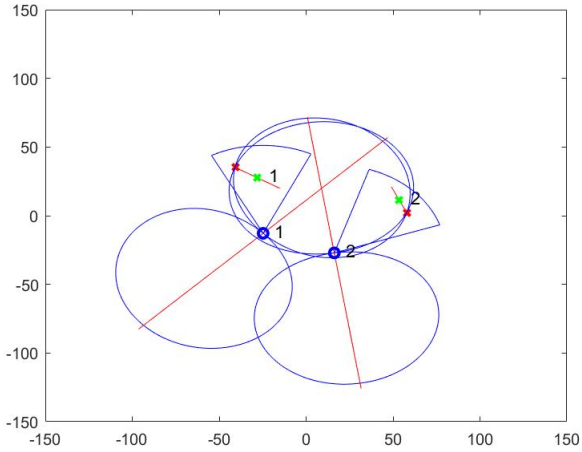
As mentioned in the previous section, the HSOP algorithm provides positions for the minimum number of host aircraft. In this example scenario, the minimum number of host aircraft needed is 2. To see how the localization error changes if more host aircraft are allowed, the HSOP algorithm is implemented using $N_{\mathrm{h}} = 3$ and 4 and again estimated the localization error. Figures 7.9c and 7.9d show the results of the HSOP algorithm when there are three and four host aircraft, respectively. Note that type of camera selected from the hosts' PCV systems changes to minimize the localization error. Also note that the intersection angles are 90°.

Figure 7.10 shows results of the HSOP algorithm with two mobile threat aircraft. The threats are initially detected by two host aircraft and their future path is predicted based on the initial detection data. The HSOP algorithm successfully covers the entire path of the threat aircraft throughout the prediction horizon and optimizes the host position to minimize the localization error. Once the prediction time (τ) has elapsed, a new final position is computed corresponding to the updated threat position and velocity. The HSOP algorithm then generates new host positions based on the new threat paths. Note that the camera types and the number of hosts needed change as the threats move.

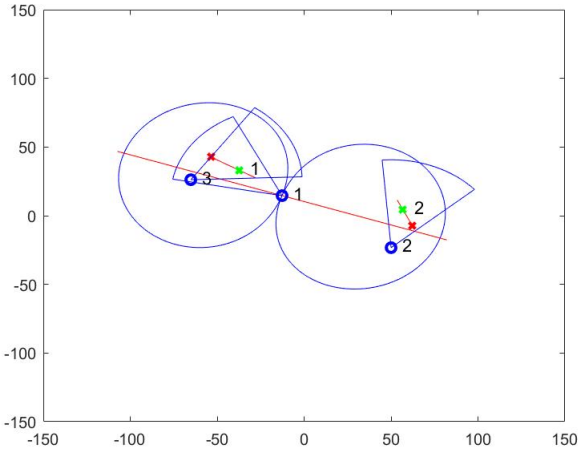
To better appreciate the localization error obtained using the HSOP algorithm, the simulation is repeated 100 times with different threat positions and various numbers of host aircraft. The localization error of all threat aircraft is estimated in each simulation and averaged. Figure 7.11 shows the averaged localization error using the CO approach and using the HSOP algorithm versus the number of host aircraft. The three subfigures show that the localization error of the HSOP algorithm decreases with increasing numbers of host aircraft.



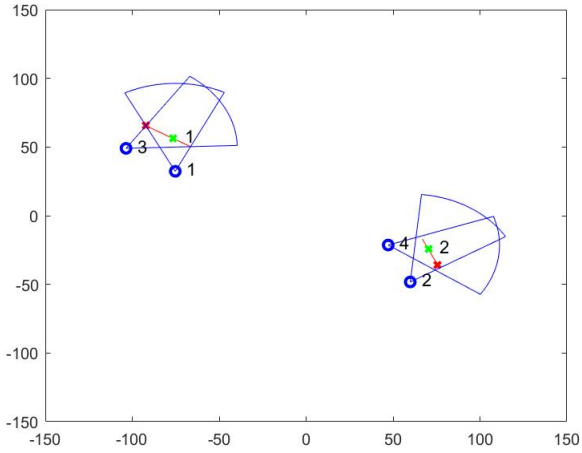
(a) Initialization (two mobile threats are detected by two hosts.)



(b) Placement of two host aircraft using the HSOP algorithm

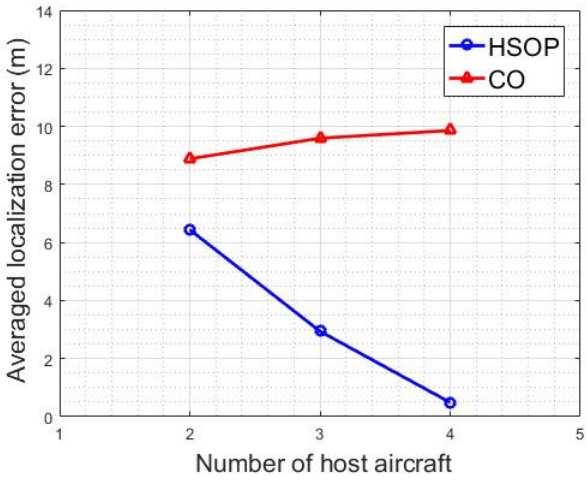


(c) Placement of three host aircraft using the HSOP algorithm

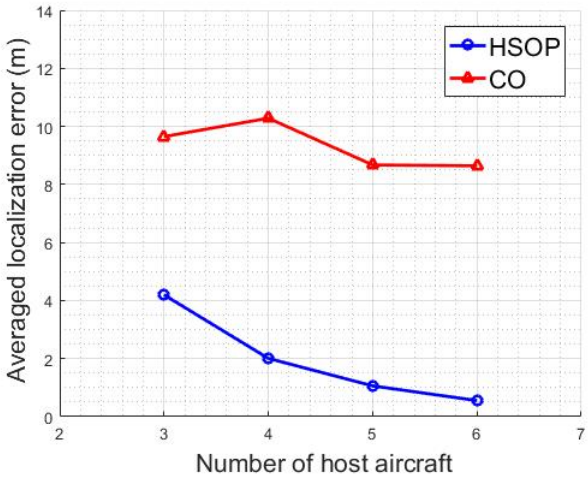


(d) Placement of four host aircraft using the HSOP algorithm

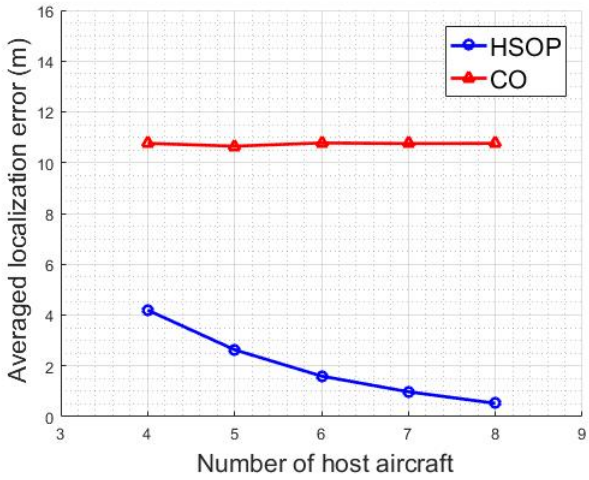
Figure 7.10: Simulation results for optimal host aircraft placement using the HSOP algorithm for mobile threats. (Small blue circles: host aircraft positions. Green x marks: current threat positions. Red x marks: final threat positions. Red lines: boresight directions of peripheral vision cameras. Large blue circles: detectable regions of peripheral vision cameras. Blue circular sectors: detectable regions of central vision cameras.)



(a) Localization error with two threat aircraft



(b) Localization error with three threat aircraft



(c) Localization error with four threat aircraft

Figure 7.11: Localization error using CO and HSOP algorithm

When fewer host aircraft are available, the peripheral vision cameras must be used to cover all of the threat aircraft, as shown in Figures 7.9 and 7.10. If more host aircraft are available, the central vision camera is selected by the algorithm and the localization error decreases because of the camera's higher resolution. On the other hand, the localization error using the CO approach does not change with the number of host aircraft since the CO approach only considers the threat coverage.

7.5 Summary

This chapter suggests an algorithm for heterogeneous stereo vision system placement for airborne counter-UAS applications, a variant on the well-studied problem of placing drones for optimal coverage of ground targets. The algorithm ensures that host aircraft equipped with two different types of camera – an omnidirectional "peripheral vision" camera and a perspective view "central vision" camera – can cover and localize a set of threat aircraft with minimum localization error. Simulation results illustrate that the heterogeneous stereo vision optimal placement (HSOP) algorithm reduces localization error. For the specific aircraft and sensing systems considered here, the HSOP algorithm reduced the averaged localization error to less than 1 m in simulations, compared with a 9-10 m localization error associated with a coverage-only approach.

In considering the optimal placement of drones to localize moving threats, the host aircraft are assumed to be sufficiently fast to attain the commanded positions quickly relative to threat aircraft motion. This assumption may be unrealistic in some scenarios. Future work might address this assumption in terms of optimal motion planning, although the image processing quality may suffer from camera motion. Also, while simulations suggest that the HSOP algorithm is effective, in practice, the algorithm does not always provide a glob-

ally optimum solution nor is it guaranteed to converge because the objective function of the optimization problem is non-convex. One might define a heuristic method to initialize the optimization problem in order to avoid these issues. Another potential concern is the algorithm's computational complexity. Although infeasible host configurations is pruned to speed up the algorithm, if the number of host and threat aircraft is large (of order 10, for example), the time required to compute a solution may prohibit its use for real-time operation.

Chapter 8

Conclusions

A peripheral-central vision (PCV) system to detect and characterize airborne threat has been developed, including specialized algorithms, for use in airborne detect-and-avoid and counter-UAS applications. The peripheral vision camera initially detects the threat, and the central vision camera is then cued to slew toward the threat. Once the threat is observed by the central vision camera, the more detailed image of the threat available from the central vision camera allows one to classify the threat using a deep neural network and also to estimate the pose of the threat. Finally, the 3D position of the threat is estimated using the heterogeneous stereo vision algorithm. In order to assess the threat localization performance, an experimental dataset was generated using a variety of mock threats. Results showed that the localization accuracy is quite limited using the current low-cost cameras in the given configuration, but also showed that the localization error decreases substantially with a longer baseline using two PCV systems. The system parameter analysis indicated that this accuracy is also influenced, however, by camera quality and performance of the feature extraction algorithm that helps to define the threat vector.

For better threat tracking, a model-based path prediction algorithm was developed for small fixed-wing unmanned aircraft. Assuming that the pose (position and attitude) of the aircraft is obtained by the PCV system, the future path of the fixed-wing unmanned aircraft is predicted based on the aircraft's dynamics. To assess the algorithm's performance, predictions using the developed path prediction (pose-plus-wind prediction) algorithm are compared

with two prediction approaches based solely on position data and on pose data for a large experimental data set. Pose-plus-wind prediction showed the best performance among three approaches for all of types of flight and two types of aircraft in the data set.

To extend the problem to the multiple threat problem, a novel multiple threat scheduling algorithm for the PCV system based on modified travelling salesman problem (TSP) solution was devised. The simulation results show that the modified TSP algorithm can prioritize the threats by considering both the time efficiency and the risk values, which existing threat scheduling algorithms do not do. The results also show that the measurement uncertainty of threat data can be decreased by updating the threat data more often.

Additional efforts focused on the multiple host aircraft placement problem. To solve this problem, the heterogeneous stereo-vision optimal placement (HSOP) algorithm was developed. The HSOP algorithm determines the optimal place and the camera type for each of two or more PCV systems that cover and localize given threat aircraft with minimum localization error. The simulation results showed that the HSOP algorithm was able to reduced the localization error compared to when only threat coverage was considered. In most cases, the HSOP algorithm provided an effective solution. However, the algorithm did not always provide a globally optimum solution nor is it guaranteed to converge. Also, the computational complexity of the algorithm may prohibit the real-time operation if the number of host and threat aircraft is large.

There are a number of potential research topics that may extend from this work. Although the localization error can be reduced using multiple host aircraft as shown in Chapter 4 and 7, a single PCV system still showed a high localization error. As analyzed in Chapter 4, the current version of the PCV system could be improved with better hardware. Also, for the threat scheduling algorithm and the HSOP algorithm, the host aircraft maneuver was not considered. The host aircraft was assumed to hover for the threat scheduling algorithm;

the maneuvering strategy of the host aircraft between waypoints was not designed for the HSOP algorithm. Developing these algorithms with the host aircraft maneuver strategies would make the algorithms more practical. Also, these algorithms were designed for the PCV systems, but implemented only in the simulation environment. Incorporating these algorithms into the actual PCV systems and implementing them in real-time would further prove their effectiveness.

Bibliography

[1] Zraick, K., "Like 'Uber for organs': drone delivers kidney to Maryland woman," *The New York Times*, April 30, 2019, p. 5.

[2] FAA, "Forecast, FAA aerospace. "Fiscal years 2019-2039."," Retrieved from https://www.faa.gov/data_research/aviation/aerospace_forecasts/media/FY2019-39_FAA_Aerospace_Forecast.pdf, 2019.

[3] Wingfield, N., and Scott, M., "In major step for drone delivery, Amazon flies package to customer in England," *The New York Times*, December 14, 2015, p. B4.

[4] Administration, F. A., "Administrator's fact book," Retrieved from https://www.faa.gov/news/media/2019_Administrators_Fact_Book.pdf, 2019.

[5] Goglia, J., "NTSB finds drone pilot at fault for midair collision with army helicopter," *Forbes*, December 14, 2017, p. A1.

[6] Olivares, G., Gomez, L., Espinosa de los Monteros, J., Baldridge, R. J., Zinzuwadia, C., and Aldag, T., *Volume II–UAS airborne collision severity evaluation–quadcopter*, Federal Aviation Administration, Washington, DC, 2017.

[7] Olivares, G., Lacy, T., Gomez, L., de los Monteros, J. E., Baldridge, R. J., Zinzuwadia, C., Aldag, T., Kota, K. R., Ricks, T., and Jayakody, N., *Volume III–UAS airborne collision severity evaluation–fixed-Wing*, Federal Aviation Administration, Washington, DC, 2017.

[8] D'Souza, K., Lyons, T., Lacy, T., and Kota, K. R., *Volume IV–UAS airborne collision*

severity evaluation–engine ingestion, Federal Aviation Administration, Washington, DC, 2017.

[9] Mueller, B., and Tsang, A., "At a busy airport in Britain, only pesky drones are flying," *The New York Times*, December 20, 2018, p. A1.

[10] Murphy, H., "The F.A.A. wants to start tracking drones' locations," *The New York Times*, December 28, 2019, p. 3.

[11] Pérez-Peña, R., Tsang, A., and Mueller, B., "Drone disrupts Heathrow, prompting a scramble to improve security," *The New York Times*, January 8, 2019, p. 6.

[12] Michel, A. H., "Counter-drone systems 2nd edition," Center for the Study of the Drone at Bard College, 2019.

[13] Singh, K., "Gatwick, Heathrow airports order military-grade anti-drone equipment," *Reuters*, January 3, 2019.

[14] Gayle, D., "Man fined for flying drone at football matches and Buckingham Palace," *The Guardian*, September 15, 2015.

[15] Cosier, C., "'I don't know whether it's a bomb or not': Train driver flummoxed after drone hits Sydney Harbour Bridge," *The Sydney Morning Herald*, November 26, 2013.

[16] Kang, C., Davis, J., Woolsey, C. A., and Choi, S., "Sense and avoid based on visual pose estimation for small UAS," *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Vancouver, Canada, September 24–28, 2017, pp. 3473–3478.

[17] Kang, C., Chaudhry, H., Woolsey, C. A., and Kochersberger, K. B., "Development of a peripheral-central vision system for small UAS tracking," *AIAA SciTech*, San Diego, California, January 7-11, 2019.

[18] Kang, C., and Woolsey, C. A., "Development of a peripheral-central vision system for small UAS tracking," *In review*, 2020.

[19] Kang, C., and Woolsey, C. A., "Model-based path prediction for fixed-wing unmanned aircraft using pose estimates," *Aerospace Science and Technology*, Vol. 105, 2020, p. 106030.

[20] Kang, C., and Woolsey, C. A., "Scheduled imaging of multiple threat aircraft using modified traveling salesman problem," *In review*, 2020.

[21] Kang, C., and Woolsey, C. A., "Optimal placement algorithm for multiple heterogeneous stereo vision systems," *In review*, 2020.

[22] Herrera, G. J., Dechant, J. A., Green, E. K., and Klein, E. A., "Technology trends in small unmanned aircraft systems (sUAS) and counter-UAS: A five year outlook (No. IDA-P-8823, H-17-000624)," Institute for Defense Analyses Alexandria, 2017.

[23] Birch, G. C., Griffin, J. C., and Erdman, M. K., "UAS detection, classification, and neutralization: Market survey 2015," Sandia National Laboratories, 2015.

[24] Michel, A. H., "Counter-drone systems," Center for the Study of the Drone at Bard College, 2018.

[25] Tyurin, V., Martyniuk, O., Mirnenko, V., Open'ko, P., and Korenivska, I., "General approach to counter unmanned aerial vehicles," *2019 IEEE 5th International Conference Actual Problems of Unmanned Aerial Vehicles Developments (APUAVD)*, Kyiv, Ukraine, October 22-24, 2019, pp. 75–78.

[26] Csengeri, J., "Counter-drone activity as a system," *Security Future*, Vol. 3(1), 2019, pp. 31–34.

[27] Samaras, S., Diamantidou, E., Ataloglou, D., Sakellariou, N., Vafeiadis, A., Magoulianitis, V., Lalas, A., Dimou, A., Zarpalas, D., Votis, K., and Daras, P., "Deep learning on multi sensor data for counter UAV applications—A systematic review," *Sensors*, Vol. 19(22), 2019, p. 4837.

[28] McClelland, H. G., Kang, C., Woolsey, C. A., Roberts, A. K., Buck, D., Cheney, T., and Warnick, K., "Small aircraft flight encounters database for UAS sense and avoid," *In AIAA SciTech*, Grapevine, Texas, January 9-13, 2017.

[29] Klare, J., Biallawons, O., and Cerutti-Maori, D., "Detection of UAVs using the MIMO radar MIRA-CLE Ka," *EUSAR 2016: 11th European Conference on Synthetic Aperture Radar*, Hamburg, Germany, June 6-9, 2016, pp. 1–4.

[30] Klare, J., Biallawons, O., and Cerutti-Maori, D., "UAV detection with MIMO radar," *2017 18th International Radar Symposium (IRS)*, Prague, Czech Republic, June 28-30, 2017, pp. 1–8.

[31] Park, S., and Park, S. O., "Configuration of an X-band FMCW radar targeted for drone detection," *2017 International Symposium on Antennas and Propagation (ISAP)*, Phuket, Thailand, Oct 30 - Nov 2, 2017, pp. 1–2.

[32] Oh, B. S., Guo, X., and Lin, Z., "A UAV classification system based on FMCW radar micro-Doppler signature analysis," *Expert Systems with Applications*, Vol. 132, 2019, pp. 239–255.

[33] Liang, C., Cao, N., Lu, X., and Ye, Y., "UAV detection using continuous wave radar," *2018 IEEE International Conference on Information Communication and Signal Processing (ICICSP)*, Singapore, September 28-30, 2018, pp. 1–5.

[34] Samaras, S., Magoulianitis, V., Dimou, A., Zarpalas, D., and Daras, P., "UAV classification with deep learning using surveillance radar data," *International Conference on Computer Vision Systems*, Thessaloniki, Greece, September 23-25, 2019, pp. 744–753.

[35] Solodov, A., Williams, A., Al Hanaei, S., and Goddard, B., "Analyzing the threat of unmanned aerial vehicles (UAV) to nuclear facilities," *Security Journal*, Vol. 31(1), 2018, pp. 305–324.

[36] Nguyen, P., Ravindranatha, M., Nguyen, A., Han, R., and Vu, T., "Investigating cost-effective RF-based detection of drones," *The 2nd Workshop on Micro Aerial Vehicle Networks, Systems, and Applications for Civilian Use*, Singapore, June 26, 2016, pp. 17–22.

[37] Ezuma, M., Erden, F., Anjinappa, C. K., Ozdemir, O., and Guvenc, I., "Detection and classification of UAVs using RF fingerprints in the presence of Wi-Fi and Bluetooth interference," *IEEE Open Journal of the Communications Society*, Vol. 1, 2019, pp. 60–76.

[38] Ezuma, M., Erden, F., Anjinappa, C. K., Ozdemir, O., and Guvenc, I., "Micro-UAV detection and classification from RF fingerprints using machine learning techniques," *2019 IEEE Aerospace Conference*, Big Sky, Montana, March 2-9, 2019, pp. 1–13.

[39] Al-Sa'd, M. F., Al-Ali, A., Mohamed, A., Khattab, T., and Erbad, A., "RF-based drone detection and identification using deep learning approaches: An initiative towards a large open source drone database," *Future Generation Computer Systems*, Vol. 100, 2019, pp. 86–97.

[40] Sedunov, A., Haddad, D., Sutin, A., and Sedunov, N., "Stevens drone detection acoustic system and experiments in acoustics UAV tracking," *2019 IEEE International Sym-*

posium on Technologies for Homeland Security (HST), Woburn, MA, November 5 - 6, 2019, pp. 5–6.

[41] Kim, J., Park, C., Ahn, J., Ko, Y., Park, J., and Gallagher, J. C., "Real-time UAV sound detection and analysis system," *2017 IEEE Sensors Applications Symposium (SAS)*, Glassboro, New Jersey, March 13-15, 2017, pp. 1–5.

[42] Harvey, B., and O'Young, S., "Acoustic detection of a fixed-wing UAV," *Drones*, Vol. 2(1), 2018, p. 4.

[43] Case, E. E., Zelnio, A. M., and Rigling, B. D., "Low-cost acoustic array for small UAV detection and tracking," *2008 IEEE National Aerospace and Electronics Conference*, Dayton, Ohio, July 15 – 19 , 2019, pp. 110–113.

[44] Jin, H., "Design of UAV detection scheme based on passive acoustic detection," *IOP Conference Series: Materials Science and Engineering*, Vol. 563(4), 2019, p. 042085.

[45] Didkovskyi, V., Kozeruk, S., and Korzhik, O., "Simple acoustic array for small UAV detection," *2019 IEEE 39th International Conference on Electronics and Nanotechnology (ELNANO)*, Kyiv, Ukraine, April 16-18, 2019, pp. 656–659.

[46] Borghgraef, A., and Vandewal, M., "Evaluation of acoustic detection of UAVs using machine learning methods," *Counterterrorism, Crime Fighting, Forensics, and Surveillance Technologies III*, Vol. 11166, 2019, p. 111660H.

[47] Jin, H., and Zhang, Y., "Research on feature recognition of UAV acoustic signal based on SVM," *Journal of Physics: Conference Series*, Vol. 1302(2), 2019, p. 022037.

[48] Bernardini, A., Mangiatordi, F., Pallotti, E., and Capodiferro, L., "Drone detection by acoustic signature identification," *Electronic Imaging 2017*, Vol. 2017(10), 2017, pp. 60–64.

[49] Rehman, S. U., and Iqbal, M. A., "Feature extraction and classification of UAV's acoustic signal using 4-microphones array in a real noisy environment," *11th International Conference on Signal Processing Systems (ICSPS 2019)*, Chengdu, China, Nov 15-17, 2019, p. 113840E.

[50] Casasanta, G., Petenko, I., Mastrantonio, G., Bucci, S., ad Andrea M. Di Lellis, A. C., Sfoglietti, G., and Argentini, S., "Consumer drones targeting by sodar (acoustic radar)," *IEEE Geoscience and Remote Sensing Letters*, Vol. 15(11), 2018, pp. 1692–1694.

[51] Busset, J., Perrodin, F., Wellig, P., Ott, B., Heutschi, K., Rühl, T., and Nussbaumer, T., "Detection and tracking of drones using advanced acoustic cameras," *Unmanned/Unattended Sensors and Sensor Networks XI; and Advanced Free-Space Optical Communication Techniques and Applications*, Vol. 9647, 2015, p. 96470F.

[52] Du, L., Gao, C., Feng, Q., Wang, C., and Liu, J., "Small UAV detection in videos from a single moving camera," *CCF Chinese Conference on Computer Vision*, Tianjin, China, October 11–14, 2017, pp. 187–197.

[53] Saqib, M., Khan, S. D., Sharma, N., and Blumenstein, M., "A study on detecting drones using deep convolutional neural networks," *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, Lecce, Italy, August 29 - September 1, 2017.

[54] Sommer, L., Schumann, A., Müller, T., Schuchert, T., and Beyerer, J., "Flying object detection for automatic UAV recognition," *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, Lecce, Italy, August 29 - September 1, 2017.

[55] Coluccia, A., Fascista, A., Schumann, A., Sommer, L., Ghenescu, M., Piatrik, T., De Cubber, G., Nalamati, M., Kapoor, A., Saqib, M., and Sharma, N., "Drone-vs-Bird detection challenge at IEEE AVSS2019," *2019 16th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, Taipei, Taiwan, September 18-21, 2019.

[56] Aker, C., and Kalkan, S., "Using deep networks for drone detection," *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, Lecce, Italy, August 29 - September 1, 2017.

[57] Schumann, A., Sommer, L., Klatte, J., Schuchert, T., and Beyerer, J., "Deep cross-domain flying object classification for robust UAV detection," *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, Lecce, Italy, August 29 - September 1, 2017.

[58] Unlu, E., Zenou, E., and Rivière, N., "Using shape descriptors for UAV detection," *Electronic Imaging*, Vol. 2018(9), 2018, pp. 128–1.

[59] Rozantsev, A., Lepetit, V., and Fua, P., "Detecting flying objects using a single moving camera," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 39(5), 2016, pp. 879–892.

[60] Rozantsev, A., Lepetit, V., and Fua, P., "Flying objects detection from a single moving camera." *IEEE Conference on Computer Vision and Pattern Recognition*, Boston, MA, June 7-12, 2015, pp. 4128–4136.

[61] Hu, S., Goldman, G. H., and Borel-Donohue, C. C., "Detection of unmanned aerial vehicles using a visible camera system," *Applied Optics*, Vol. 56(3), 2017, pp. B214–B221.

[62] Sapkota, K. R., Roelofsen, S., Rozantsev, A., Lepetit, V., Gillet, D., Fua, P., and Martinoli, A., "Vision-based unmanned aerial vehicle detection and tracking for sense and avoid systems," *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Daejeon, South Korea, October 9-14, 2016.

[63] Mcfadyen, A., and Mejias, L., "A survey of autonomous vision-based see and avoid for unmanned aircraft systems," *Progress in Aerospace Sciences*, Vol. 80, 2016, pp. 1–17.

[64] Lyu, Y., Pan, Q., Zhao, C., and Hu, J., "Autonomous stereo vision based collision avoid system for small UAV," *AIAA SciTech*, Grapevine, Texas, January 9-13, 2017.

[65] Birch, G. C., and Woo, B. L., "Counter unmanned aerial systems testing: evaluation of VIS SWIR MWIR and LWIR passive imagers," *Sandia Report*, Vol. 921, 2017.

[66] Lyu, Y., Pan, Q., Zhao, C., and Hu, J., "Autonomous stereo vision based collision avoid system for small UAV," *In AIAA Information Systems-AIAA Infotech@ Aerospace*, Grapevine, Texas, January 9-13, 2017, p. 1150.

[67] Drulea, M., Szakats, I., Vatavu, A., and Nedevschi, S., "Omnidirectional stereo vision using fisheye lenses," *In Intelligent Computer Communication and Processing (ICCP), 2014 IEEE International Conference*, Cluj Napoca, Romania, September 4-6, 2014, pp. 251–258.

[68] Kita, N., and Kita, Y., "Reference plane based fisheye stereo epipolar rectification," *In 12th International Conference on Computer Vision Theory and Applications (VISAPP)*, Porto, Portugal, February 27 - March 1, 2017, pp. 308–320.

[69] Chen, C. H., Yao, Y., Page, D., Abidi, B., Koschan, A., and Abidi, M., "Heterogeneous fusion of omnidirectional and PTZ cameras for multiple object tracking," *IEEE*

Transactions on Circuits and Systems for Video Technology, Vol. 18(8), 2008, pp. 1052–1063.

[70] Yu, M. S., Wu, H., and Lin, H. Y., "A visual surveillance system for mobile robot using omnidirectional and PTZ cameras," *SICE Annual Conference 2010*, Taipei, Taiwan, August 18-21, 2010, pp. 37–42.

[71] Cui, Y., Samarasckera, S., Huang, Q., and Greiffenhagen, M., "Indoor monitoring via the collaboration between a peripheral sensor and a foveal sensor," *1998 IEEE Workshop on Visual Surveillance*, Bombay, India, January 2, 1998, pp. 2–9.

[72] Scotti, G., Marcenaro, L., Coelho, C., Selvaggi, F., and Regazzoni, C. S., "Dual camera intelligent sensor for high definition 360 degrees surveillance," *IEE Proceedings-Vision, Image and Signal Processing*, Vol. 152(2), 2005, pp. 250–257.

[73] Iraqui, A., Dupuis, Y., Boutteau, R., Ertaud, J. Y., and Savatier, X., "Fusion of omnidirectional and PTZ cameras for face detection and tracking," *2010 International Conference on Emerging Security Technologies*, Canterbury, UK, September 6-7, 2010, pp. 18–23.

[74] Fahn, C. S., and Lo, C. S., "A high-definition human face tracking system using the fusion of omni-directional and PTZ cameras mounted on a mobile robot," *2010 5th IEEE Conference on Industrial Electronics and Applications*, Taichung, Taiwan, June 15-17, 2010, pp. 6–11.

[75] Baris, . B. Y., I., "Classification and tracking of traffic scene objects with hybrid camera systems," *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, Yokohama, Japan, October 16-19, 2017, pp. 1–6.

[76] Muñoz-Salinas, R., Medina-Carnicer, R., Madrid-Cuevas, F. J., and Carmona-Poyato,

A., "Particle filtering with multiple and heterogeneous cameras," *Pattern Recognition*, Vol. 43(7), 2010, pp. 2390–2405.

[77] Rathnayaka, P., Baek, S. H., and Park, S. Y., "An efficient calibration method for a stereo camera system with heterogeneous lenses using an embedded checkerboard pattern," *Journal of Sensors*, Vol. 2017, 2017.

[78] Eynard, D., Vasseur, P., Demonceaux, C., and Frémont, V., "UAV altitude estimation by mixed stereoscopic vision," *In Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, Taipei, Taiwan, October 18-22, 2010, pp. 646–651.

[79] Eynard, D., Demonceaux, C., Vasseur, P., and Fremont, V., "UAV motion estimation using hybrid stereoscopic vision," *In Machine Vision Applications (MVA), 2011 IAPR Conference on*, Nara City, Japan, June 13-15, 2011, pp. 340–343.

[80] Eynard, D., Vasseur, P., Demonceaux, C., and Frémont, V., "Real time UAV altitude, attitude and motion estimation from hybrid stereovision," *Autonomous Robots*, Vol. 33(1-2), 2012, pp. 157–172.

[81] Lin, C. E., and Lai, Y. H., "UAV path prediction for CD&R to manned aircraft in a confined airspace for cooperative mission," *International Journal of Aerospace Engineering*, Vol. 2018(1), 2018, pp. 1–9.

[82] Kuchar, J. K., and Yang, L. C., "A review of conflict detection and resolution modeling methods," *IEEE Transactions on Intelligent Transportation Systems*, Vol. 1(4), 2000, pp. 179–189.

[83] Olive, X., and Morio, J., "Trajectory clustering of air traffic flows around airports," *Aerospace Science and Technology*, Vol. 84, 2019, pp. 776–781.

[84] Pierpaoli, P., and Rahmani, A., "UAV collision avoidance exploitation for noncooperative trajectory modification," *Aerospace Science and Technology*, Vol. 73, 2018, pp. 173–183.

[85] Lin, . S. S., Y., "Path planning using 3D Dubins curve for unmanned aerial vehicles," *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*, Orlando, Florida, May 27-30, 2014, pp. 296–304.

[86] Laurenzis, M., Rebert, M., Schertzer, S., and Christnacher, F., "Tracking and prediction of small unmanned aerial vehicles' flight behavior and three-dimensional flight path," *SPIE Defense + Commercial Sensing*, Baltimore, Maryland, April 16 - 18, 2019.

[87] Canolla, A. C., Jamoom, M. B., and Pervan, B., "Unmanned aircraft systems detect and avoid sensor hybrid estimation error analysis," *17th AIAA Aviation Technology, Integration, and Operations Conference*, Denver, Colorado, June 5-9, 2017, p. 4384.

[88] Kanneganti, S. T., Chilson, P. B., and Huck, R., "Visualization and prediction of aircraft trajectory using ADS-B," Dayton, Ohio, July 23-26, 2018, pp. 529–532.

[89] Lin, Y., and Saripalli, S., "Sampling-based path planning for UAV collision avoidance," *IEEE Transactions on Intelligent Transportation Systems*, Vol. 18(11), 2017, pp. 3179 – 3192.

[90] Hammer, M., Hebel, M., Laurenzis, M., and Arens, M., "Lidar-based detection and tracking of small UAVs," *Emerging Imaging and Sensing Technologies for Security and Defence III; and Unmanned Sensors, Systems, and Countermeasures*, Berlin, Germany, September 10 - 13, 2018.

[91] Wu, Z., Li, J., Zuo, J., and Li, S., "Path planning of UAVs based on collision probability and Kalman filter," *IEEE Access*, Vol. 6, 2018, pp. 34237–34245.

[92] Sakthivel, C., Sureshkumar, B., Ramkumar, R., and Yokeswaran, R., "Detection and path prediction of aircraft based on acoustics and vibration," *Materials Today: Proceedings*, Vol. 21, 202, pp. 588–591.

[93] Zhang, X., Fan, C., Fang, J., Xu, S., and Du, J., "Tracking prediction to avoid obstacle path of agricultural unmanned aerial vehicle based on particle filter," *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, Vol. 232(4), 2018, pp. 408–416.

[94] Lin, Y., and Saripalli, S., "Collision avoidance for UAVs using reachable sets," *2015 International Conference on Unmanned Aircraft Systems (ICUAS)*, Denver, Colorado, June 9-12, 2015, pp. 226–235.

[95] Ren, L., Castillo-Effen, M., Yu, H., Yoon, Y., Nakamura, T., Johnson, E. N., and Ippolito, C. A., "Small unmanned aircraft system (sUAS) trajectory modeling in support of UAS traffic management (UTM)," *In 17th AIAA Aviation Technology, Integration, and Operations Conference*, Denver, Colorado, June 5-9, 2017.

[96] Slattery, R., and Zhao, Y., "Trajectory synthesis for air traffic automation," *Journal of Guidance, Control, and Dynamics*, Vol. 20(2), 1997, pp. 232–238.

[97] Vilardaga, S., and Prats, X., "Mass estimation for an adaptive trajectory predictor using optimal control," *In Proceedings of the 5th International Conference on Application and Theory of Automation in Command and Control Systems*, Toulouse, France, September 30 - October 2, 2015, pp. 75–84.

[98] Glover, W., and Lygeros, J., "A multi-aircraft model for conflict detection and resolution algorithm evaluation," *Technical Report WP1, Deliverable D1.3, Version 1.3. HYBRIDGE*, 2003.

[99] Ayhan, S., and Samet, H., "Aircraft trajectory prediction made easy with predictive analytics," *In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, California, August 13-17, 2016, pp. 21–30.

[100] Lymperopoulos, I., Lygeros, J., and Lecchini, A., "Model based aircraft trajectory prediction during takeoff," *In AIAA Guidance, Navigation, and Control Conference and Exhibit*, Keystone, Colorado, August 21-24, 2006.

[101] Zhang, J., Jie Liu, R. H., and Zhu, H., "Online four dimensional trajectory prediction method based on aircraft intent updating," *Aerospace Science and Technology*, Vol. 77, 2018, pp. 774–787.

[102] Chan, W., Bach, R., and Walton, J., "Improving and validating CTAS performance models," *In AIAA Guidance, Navigation, and Control Conference and Exhibit*, Denver, Colorado, August 14-17, 2000.

[103] Warren, A. W., and Ebrahimi, Y. S., "Vertical path trajectory prediction for next generation ATM," *In Proceedings of the 17th DASC. AIAA/IEEE/SAE. Digital Avionics Systems Conference*, Vol. 2, Bellevue, Washington, October 31-November 7, 1998, pp. F11/1 – F11/8.

[104] Hadjaz, A., Marceau, G., Savéant, P., and Schoenauer, M., "Online learning for ground trajectory prediction," *arXiv Preprint*, 2012, arXiv:1212.3998.

[105] Lee, H. T., and Chatterji, G., "Closed-form takeoff weight estimation model for air

transportation simulation," *In 10th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference*, Fort Worth, Texas, September 13-15, 2010.

[106] Lopez-Lago, M., Rafael Casado, A. B., and Serna, J., "A predictive model for risk assessment on imminent bird strikes on airport areas," *Aerospace Science and Technology*, Vol. 62, 2017, pp. 19–30.

[107] Alligier, R., Gianazza, D., and Durand, N., "Machine learning and mass estimation methods for ground-based aircraft climb prediction," *IEEE Transactions on Intelligent Transportation Systems*, Vol. 16(6), 2015, pp. 3138–3149.

[108] Thipphavong, D. P., Schultz, C. A., Lee, A. G., and Chan, S. H., "Adaptive algorithm to improve trajectory prediction accuracy of climbing aircraft," *Journal of Guidance, Control, and Dynamics*, Vol. 36(1), 2012, pp. 15–24.

[109] Slater, G. L., "Adaptive improvement of aircraft climb performance for air traffic control applications," *In Proceedings of the IEEE International Symposium on Intelligent Control*, Vancouver, Canada, Oct 30, 2002, pp. 602–607.

[110] Godbole, A. A., *Adaptive Improvement of Climb Performance (Master's thesis, University of Cincinnati)*, Retrieved from https://etd.ohiolink.edu/pg_10?0::NO:10:P10_ACCESSION_NUM:ucin1061303791, 2003.

[111] Nuic, A., *User manual for the base of aircraft data (BADA) revision 3.10*, Atmosphere, 2010.

[112] Miranda, S., Baker, C., Woodbridge, K., and Griffiths, H., "Knowledge-based resource management for multifunction radar," *IEEE Signal Processing Magazine*, Vol. 23(1), 2006, pp. 66–76.

[113] Miranda, S. L., Baker, C. J., Woodbridge, K., and Griffiths, H. D., "Simulation methods for prioritizing tasks and sectors of surveillance in phased array radar," *International Journal of Simulation*, Vol. 5(1-2), 2004, pp. 18–25.

[114] Miranda, S. L. C., Baker, C. J., Woodbridge, K., and Griffiths, H., "Fuzzy logic approach for prioritisation of radar tasks and sectors of surveillance in multifunction radar," *IET Radar, Sonar Navigation*, Vol. 1(2), 2007, pp. 131–141.

[115] Ding, Z., and Moo, P., "Benefits of target prioritization for phased array radar resource management," *2017 18th International Radar Symposium (IRS)*, Prague, Czech Republic, June 28-30, 2017, pp. 1–7.

[116] Marques, T., Lukic, L., and Gaspar, J., "Observation functions in an information theoretic approach for scheduling pan-tilt-zoom cameras in multi-target tracking applications," *Robot 2015: Second Iberian Robotics Conference*, Lisbon, Portugal, November 19-21, 2015, pp. 503–515.

[117] Sommerlade, E., and Reid, I., "Probabilistic surveillance with multiple active cameras," *2010 IEEE International Conference on Robotics and Automation (ICRA)*, Anchorage, Alaska, May 3-8, 2010, pp. 440–445.

[118] Sommerlade, E., and Reid, I., "Cooperative surveillance of multiple targets using mutual information," *Workshop on Multi-camera and Multi-modal Sensor Fusion Algorithms and Applications-M2SFA2 2008*, Marseille, France, October 18, 2008.

[119] Zhang, C., and Hwang, I., "Decentralized multi-sensor scheduling for multi-target tracking and identity management," *2019 18th European Control Conference (ECC)*, Naples, Italy, 25-28 June 2019, pp. 1804–1809.

[120] Khosla, D., Huber, D. J., and Chen, Y., "Automated scheduling of radar-cued camera

system for optimizing visual inspection and detection of radar targets," *2017 IEEE International Symposium on Technologies for Homeland Security (HST)*, Waltham, Massachusetts, April 25-26, 2017, pp. 1–5.

[121] Savkin, A. V., and Huang, H., "Proactive deployment of aerial drones for coverage over very uneven terrains: A version of the 3d art gallery problem," *Sensors*, Vol. 19(6), 2019, p. 1438.

[122] Tuba, E., Capor-Hrosik, R., Alihodzic, A., and Tuba, M., "Drone placement for optimal coverage by brain storm optimization algorithm," *In International Conference on Health Information Science*, Moscow, Russia, October 7-9, 2017, pp. 167–176.

[123] Strumberger, I., Sarac, M., Markovic, D., and Bacanin, N., "Moth search algorithm for drone placement problem," *International Journal of Computers*, Vol. 3, 2018.

[124] Tuba, E., Tuba, I., Dolicanin-Djekic, D., Alihodzic, A., and Tuba, M., "Efficient drone placement for wireless sensor networks coverage by bare bones fireworks algorithm," *In 2018 6th International Symposium on Digital Forensic and Security (ISDFS)*, Antalya, Turkey, March 22-25, 2018, pp. 1–5.

[125] Zorbas, D., Razafindralambo, T., Pugliese, L. D. P., and Guerriero, F., "Energy efficient mobile target tracking using flying drones," *Procedia Computer Science*, Vol. 19, 2013, pp. 80–87.

[126] Pugliese, L. D. P., Guerriero, F., Zorbas, D., and Razafindralambo, T., "Modelling the mobile target covering problem using flying drones," *Optimization Letters*, Vol. 10(5), 2016, pp. 1021–1052.

[127] Zorbas, D., Pugliese, L. D. P., Razafindralambo, T., and Guerriero, F., "Optimal

drone placement and cost-efficient target coverage," *Journal of Network and Computer Applications*, Vol. 75, 2016, pp. 16–31.

[128] Al-Turjman, F., Zahmatkesh, H., Al-Oqily, I., and Daboul, R., "Optimized unmanned aerial vehicles deployment for static and mobile targets' monitoring," *Computer Communications*, Vol. 149, 2020, pp. 27–35.

[129] Fiore, L., Somasundaram, G., Drenner, A., and Papanikolopoulos, N., "Optimal camera placement with adaptation to dynamic scenes," *In 2008 IEEE International Conference on Robotics and Automation*, Pasadena, California, May 19-23, 2008, pp. 956–961.

[130] Bodor, R., Drenner, A., Janssen, M., Schrater, P., and Papanikolopoulos, N., "Mobile camera positioning to optimize the observability of human activity recognition tasks," *In 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Edmonton, Canada, August 2-6, 2005, pp. 1564–1569.

[131] Bodor, R., Drenner, A., Schrater, P., and Papanikolopoulos, N., "Optimal camera placement for automated surveillance tasks," *Journal of Intelligent and Robotic Systems*, Vol. 50(3), 2007, pp. 257–295.

[132] Chen, J., Khatibi, S., and Kulesza, W., "Planning of a multi stereo visual sensor system-depth accuracy and variable baseline approach," *In 2007 3DTV Conference*, Kos Island, Greece, May 7-9, 2007, pp. 1–4.

[133] Kulesza, W., Chen, J., Khatibi, S., and Bhatti, A., "Arrangement of a multi stereo visual sensor system for a human activities space," *Stereo Vision*, IntechOpen, 2008, pp. 153–172.

[134] Malik, R., and Bajcsy, P., "Automated placement of multiple stereo cameras," *The 8th*

Workshop on Omnidirectional Vision, Camera Networks and Non-classical Cameras, Marseille, France, October 17, 2008.

[135] Rahimian, P., and Kearney, J. K., "Optimal camera placement for motion capture systems in the presence of dynamic occlusion," *In Proceedings of the 21st ACM Symposium on Virtual Reality Software and Technology*, Beijing, China, November 13-15, 2015, pp. 129–138.

[136] Kelly, A., "Precision dilution in triangulation based mobile robot position estimation," *Intelligent Autonomous Systems*, 2003, pp. 1046–1053.

[137] Sanders-Reed, J. N., "Triangulation position error analysis for closely spaced imagers," *SAE Transactions*, Vol. 111, 2002, pp. 978–984.

[138] Sanders-Reed, J. N., "Impact of tracking system knowledge on multisensor 3D triangulation," *AeroSense 2002*, Orlando, Florida, July 1, 2002, pp. 33–41.

[139] Shih, S. E., and Tsai, W. H., "Optimal design and placement of omni-cameras in binocular vision systems for accurate 3-D data measurement," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 23(11), 2013, pp. 1911–1926.

[140] Altahir, A. A., Asirvadam, V. S., Hamid, N. H., Sebastian, P., Saad, N., Ibrahim, R., and Dass, S. C., "Modeling multicamera coverage for placement optimization," *IEEE Sensors Letters*, Vol. 1(6), 2017, pp. 1–4.

[141] Yildiz, E., Akkaya, K., Sisikoglu, E., and Sir, M. Y., "Optimal camera placement for providing angular coverage in wireless video sensor networks," *IEEE Transactions on Computers*, Vol. 63(7), 2013, pp. 1812–1825.

[142] Gonzalez-Barbosa, J. J., García-Ramírez, T., Salas, J., and Hurtado-Ramos, J. B.,

"Optimal camera placement for total coverage." *In 2009 IEEE International Conference on Robotics and Automation*, Kobe, Japan, May 12-17, 2009, pp. 844–848.

[143] Scaramuzza, D., Martinelli, A., and Siegwart, R., "A toolbox for easily calibrating omnidirectional cameras," *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Beijing, China, October 9-15, 2006, pp. 5695–5701.

[144] Lucas, B. D., and Kanade, T., "An iterative image registration technique with an application to stereo vision," *Proceedings of the International Joint Conference on Artificial Intelligence*, 1981.

[145] Lee, J. J., and Kim, G., "Robust estimation of camera homography using fuzzy RANSAC," *International Conference on Computational Science and Its Applications*, Kuala Lumpur, Malaysia, August 26-29, 2007, pp. 992–1002.

[146] Monnin, D., Bieber, E., Schmitt, G., and Schneider, A., "An effective rigidity constraint for improving RANSAC in homography estimation," *International Conference on Advanced Concepts for Intelligent Vision Systems*, Sydney, Australia, December 13-16, 2010, pp. 203–214.

[147] Lusk, P. C., and Beard, R. W., "Visual multiple target tracking from a descending aerial platform," *In 2018 Annual American Control Conference (ACC)*, Milwaukee, Wisconsin, June 27-29, 2018, pp. 5088–5093.

[148] Harris, C., and Stephens, M., "A combined corner and edge detector," *Alvey Vision Conference*, Vol. 15, 1988, pp. 10–5244.

[149] Fischler, M. A., and Bolles, R. C., "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, Vol. 24(6), 1981, pp. 381–395.

[150] Lin, J., Ji, X., Xu, W., and Dai, Q., "Absolute depth estimation from a single defocused image," *IEEE Transactions on Image Processing*, Vol. 22(11), 2013, pp. 4545–4550.

[151] Rajagopalan, A. N., and Chaudhuri, S., "An MRF model-based approach to simultaneous recovery of depth and restoration from defocused images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 21(7), 1999, pp. 577–589.

[152] Hiura, S., and Matsuyama, T., "Depth measurement by the multi-focus camera," *1998 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Santa Barbara, California, June 25, 1998, pp. 953–959.

[153] Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., and Adam, H., "MobileNets: efficient convolutional neural networks for mobile vision applications," *ArXiv e-prints*, 2017.

[154] Redmon, J., and Farhadi, A., "YOLO9000: Better, faster, stronger," *arXiv preprint*, 2016.

[155] Redmon, J., and Farhadi, A., "YOLOv3: An incremental improvement," *arXiv preprint*, 2018.

[156] Zhao, X. M., Wang, W. X., and Wang, L. P., "Parameter optimal determination for canny edge detection," *Imaging Science Journal*, Vol. 59(6), 2011, pp. 332–341.

[157] Canny, J., "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 6, 1986, pp. 679–698.

[158] Fooladgar, F., Samavi, S., Soroushmehr, R., S. M., and Shirani, S., "Geometrical analysis of localization error in stereo vision systems," *IEEE Sensors Journal*, Vol. 13(11), 2013, pp. 4236–4246.

[159] Mandun, Z., Lichao, Q., Guodong, C., and Ming, Y., "A triangulation method in 3D reconstruction from image sequences," *2009 Second International Conference on Intelligent Networks and Intelligent Systems*, Tianjian, China, November 1-3, 2009, pp. 306–308.

[160] Kanatani, K., Sugaya, Y., and Niitsuma, H., "Triangulation from two views revisited: Hartley-Sturm vs. Optimal correction," *19th British Machine Vision Conference (BMVC 2008)*, Leeds, U.K., September 1–4, 2008, p. 173–182.

[161] Chen, J., Wu, D., Song, P., Deng, F., He, Y., and Pang, S., "Multi-view triangulation: Systematic comparison and an improved method," *IEEE Access,*, Vol. 8, 2020, pp. 21017–21027.

[162] Li, Y., Zhang, J., and Tian, J., "Error analysis in stereo vision for location measurement of 3D point," *Ninth International Symposium on Multispectral Image Processing and Pattern Recognition (MIPPR2015)*, Enshi, China, October 31 - Novemeber 1, 2015.

[163] Lee, S. H., and Civera, J., "Triangulation: Why optimize?" *arXiv preprint*, 2019, arXiv:1907.11917.

[164] Lin, C. M., Tsai, C. Y., Lai, Y. C., Li, S. A., and Wong, C. C., "Visual object recognition and pose estimation based on a deep semantic segmentation network," *IEEE Sensors Journal*, Vol. 18(22), 2018, pp. 9370–9381.

[165] Wang, Y., Tan, X., Yang, Y., Liu, X., Ding, E., Zhou, F., and Davis, L. S., "3D pose estimation for fine-grained object categories," *European Conference on Computer Vision (ECCV)*, Munich, Germany, September 8-14, 2018.

[166] Poirson, P., Ammirato, P., Fu, C. Y., Liu, W., Kosecka, J., and Berg, A. C., "Fast

single shot detection and pose estimation," *2016 Fourth International Conference on 3D Vision (3DV)*, Stanford, California, October 25-28, 2016, pp. 676–684.

[167] Billings, G., and Johnson-Roberson, M., "Silhonet: An RGB method for 3D object pose estimation and grasp planning," *arXiv preprint*, 2018, arXiv:1809.06893.

[168] Xiang, Y., Kim, W., Chen, W., Ji, J., Choy, C., Su, H., Mottaghi, R., Guibas, L., and Savarese., S., "Objectnet3D: A large scale database for 3D object recognition," *European Conference on Computer Vision(ECCV)*, Amsterdam, Netherlands, October 11-14, 2016, pp. 160–176.

[169] Wellig, P., Speirs, P., Schuepbach, C., Oechslin, R., Renker, M., Boeniger, U., and Pratisto, H., "Radar systems and challenges for C-UAV," *2018 19th International Radar Symposium (IRS)*, Bonn, Germany, June 20-22, 2018, pp. 1–8.

[170] Rovkin, M. E., Khlusov, V. A., Malyutin, N. D., Hristenko, A. V., Novikov, A. S., Nosov, D. M., Osipov, M. V., Konovalenko, M. O., Marchenko, A. O., and Ilchenkoy, V. E., "Radar detection of small-size UAVs," *2018 Ural Symposium on Biomedical Engineering, Radioelectronics and Information Technology (USBEREIT)*, Yekaterinburg, Russia, May 7-8, 2018, pp. 371–374.

[171] Poitevin, P., Pelletier, M., and Lamontagne, P., "Challenges in detecting UAS with radar," *2017 International Carnahan Conference on Security Technology (ICCST)*, Madrid, Spain, October 23-26, 2017, pp. 1–6.

[172] Canolla, A. C., Jamoom, M. B., and Pervan, B., "Unmanned aircraft systems detect and avoid sensor hybrid estimation error analysis," *In 17th AIAA Aviation Technology, Integration, and Operations Conference*, Denver, Colorado, June 5-9, 2017.

[173] Anderson, J., *Introduction to flight 3rd edition*, McGraw-Hill, New York City, New York, 2008.

[174] Vinh, N. X., *Flight mechanics of high-performance aircraft*, Cambridge University Press, Cambridge, United Kingdom, 1995, Vol. 4.

[175] Sahawneh, L. R., and Beard, R. W., "Path planning in the local-level frame for small unmanned aircraft systems," *Kinematics*, IntechOpen, 2017.

[176] Li, X. R., and Jilkov, V. P., "Survey of maneuvering target tracking. Part I. Dynamic models," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 39(4), 2003, pp. 1333–1364.

[177] Maeder, U., Morari, M., and Baumgartner, T. I., "Trajectory prediction for light aircraft," *Journal of Guidance, Control, and Dynamics*, Vol. 34(4), 2011, pp. 1112–1119.

[178] Alligier, R., Gianazza, D., Hamed, M. G., and Durand, N., "Comparison of two ground-based mass estimation methods on real data," *In ICRAT 2014, 6th International Conference on Research in Air Transportation*, Istanbul, Turkey, May 26-30, 2014.

[179] Alligier, R., Gianazza, D., and Durand, N., "Energy rate prediction using an equivalent thrust setting profile," *In ICRAT 2012, 5th International Conference on Research in Air Transportation*, Berkeley, California, May 22-25, 2012.

[180] Schultz, C., Thipphavong, D., and Erzberger, H., "Adaptive trajectory prediction algorithm for climbing flights," *In AIAA Guidance, Navigation, and Control Conference*, Minneapolis, Minnesota, August 13-16, 2012.

[181] Poitevin, P., Pelletier, M., and Lamontagne, P., "Challenges in detecting UAS with

radar," *2017 International Carnahan Conference on Security Technology (ICCST)*, Madrid, Spain, October 23-26, 2017.

[182] Wellig, P., Speirs, P., Schuepbach, C., Oechslin, R., Renker, M., Boeniger, U., and Pratisto, H., "Radar systems and challenges for C-UAV," *2018 19th International Radar Symposium (IRS)*, Bonn, Germany, June 20-22, 2018.

[183] Dantzig, G., Fulkerson, R., and Johnson, S., "Solution of a large-scale traveling-salesman problem," *Journal of the Operations Research Society of America*, Vol. 2(4), 1954, pp. 393–410.

[184] Laporte, G., "The traveling salesman problem: An overview of exact and approximate algorithms," *European Journal of Operational Research*, Vol. 59(2), 1992, pp. 231–247.

[185] Sewart, T., Partridge, L., Cornwell, D., Arcas, D. A., and Wilcox, C., "Quick quad-copters: top speed of a racing drone," *Physics Special Topics*, Vol. 18(1), 2019.

[186] Bradski., G., "The OpenCV library," *Dr. Dobb's Journal of Software Tools*, 2000.

[187] Bradski, G., and Kaehler, A. (eds.), *Learning OpenCV: Computer vision with the OpenCV library*, O'Reilly Media, Inc., 2008.