Demonstrating an Equivalent Level of Safety for sUAS in Shielded Environments

Kendy E. Edmonds

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

Master of Science
in
Aerospace and Ocean Engineering

Craig A. Woolsey, Chair
John M. Coggin
Mathieu Joerger

April 28, 2021
Blacksburg, Virginia

Keywords: Unmanned Aircraft Systems, drones, avoidance algorithms, detect and avoid, well clear

Demonstrating an Equivalent Level of Safety for sUAS in Shielded Environments

Kendy E. Edmonds

(ABSTRACT)

The current proposed unmanned aircraft system (UAS) detect and avoid standards require the same safety metrics, even when in close proximity to the ground or structures. This requirement has the potential to hinder low altitude small unmanned aircraft operations, such as local package delivery and utility inspection. One of the main safety metrics for UASs to adhere to is a "well clear" volume that quantifies the vertical and horizontal separation UASs are required to maintain from manned aircraft. The current volume of 2000 feet horizontal and +/- 250 feet vertical does not provide credit for the safety benefit of being close to an obstacle where manned aircraft do not fly and could prove to be too restricting for low-level flight operations (i.e., under 400 feet above ground level). This thesis suggests using smaller safety metric volumes than the well clear volume to demonstrate that operations at lower altitudes can still be proven to be just as safe as if they were held to the larger well clear volume standard by using obstacle and terrain shielding. The research leverages simulation to analyze different safety metrics and provides an example use case in which the methodology of shielded operations is applied to demonstrate how this methodology can be applied for a safety case.

Demonstrating an Equivalent Level of Safety for sUAS in Shielded Environments

Kendy E. Edmonds

(GENERAL AUDIENCE ABSTRACT)

With the development of small unmanned aircraft system (sUAS) technologies have come many practical and regulatory challenges, especially in low altitude airspaces. At lower altitudes, manned aircraft are likely to be operating at lower velocities and restricting standards require UASs to maneuver against aircraft that may not present a significant risk of collision. The excessive avoidance maneuvering can cause the successful execution of even simple operations such as package delivery or survey operations to become difficult. The strict requirements have the potential to specifically inhibit sUAS beyond visual line-of-sight commercial operations, which are of great interest to the industry. This thesis describes a method for demonstrating an equivalent level of safety of small UAS operations when utilizing avoidance algorithms that leverage obstacle and terrain awareness. The purpose of this research is to demonstrate that by remaining close to obstacles, which pose a hazard to other aircraft, an unmanned aircraft can lower the risk of a mid-air collision and to demonstrate an equivalent level of safety for operations using a reduced safety metrics.

I would like to dedicate this thesis to my mother, Dawna Ross, who is always there for me to make an unsolicited lighthearted joke, and to my sisters, Jade and Lawryn, who helped blaze the trail I'm on. "Here's to strong women. May we know them, may we be them, may we raise them."

Acknowledgments

I would like to extend my sincerest gratitude to the Virginia Tech Mid-Atlantic Aviation Partnership for their support throughout this research. In particular, special thanks to: John Coggin for his guidance and mentorship; Andrew Kriz for the development of the simulation tool and for the countless zoom calls to help me fix my code; Mark Blanks for his extended faith in me; Tombo Jones for his leadership; and Robert Briggs for lending his expertise in the development of this thesis. I would also like to thank Dr. Joerger for agreeing to be on my thesis committee and my advisor, Dr. Woolsey, for his invaluable guidance throughout my master's program. Finally, I would like to thank the Virginia Tech College of Engineering and New Horizons Graduate Scholars program for their support.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The unmanned aircraft system (UAS) industry has grown exponentially over the past decade. With this growth have also come great challenges. One of the key challenges that has been exposed with the recent desire for beyond visual line-of-sight (BVLOS) operations is the ability to avoid other aircraft. To this end, UASs will require a detect and avoid (DAA) capability to safely manage encounters with both manned aircraft and other UASs when flown BVLOS of the remote pilot in command.

However, the current alerting and avoidance algorithms for small UAS DAA are fairly immature. Furthermore, it is necessary to develop and test DAA algorithms in a robust simulation environment to validate performance and examine corner-cases, as well as test and evaluate different avoidance techniques. One such technique is an avoidance algorithm that utilizes terrain and obstacle awareness to mitigate the chance of a collision. This thesis specifically explores use cases that are conducted at low altitudes (below 500 feet AGL), such as infrastructure inspection, surveying, and local package delivery. Unmanned-to-unmanned avoidance is out of scope of this research, as current UAS Traffic Management (UTM) research indicates these encounters will be managed through cooperative technologies for compliant

aircraft [10].

This research has two major components: (1) The development and testing of an aircraft avoidance algorithm that can utilize obstacle awareness to mitigate the chance of a midair collision (MAC) or near-midair collision (NMAC) between a small unmanned aircraft and a manned "intruder" aircraft, and (2) exploration of quantitative definitions for an appropriate shielding volume that will result in an equivalent level of safety (ELOS) for reduced well clear volumes.

1.1 Motivation

There are many commercial UAS companies that could benefit from using reduced well clear volumes and shielded avoidance to achieve an equivalent level of safety. There are numerous applications for low-level flight operations where intruder aircraft would be navigating at slower velocities, as opposed to operations at higher altitudes, and obstacles or terrain would be available to be used as shields. One example of an application is a package delivery operation, such as the current expansion of UAV package delivery in Christiansburg, Virginia by the commercial package delivery company Wing [5].

Though traditional avoidance algorithms, such as a an algorithm that commands an immediate landing or a maneuver to turn perpendicular to the intruder aircraft trajectory, could produce the required level of safety, performing significant avoidance maneuvers could hinder the time or energy efficiency of a mission and ultimately mission success. An algorithm that allows for minimal deviation of the original flight path would be beneficial to many UAS operations.

1.2 Problem Statement

The objective of this research is to demonstrate that even if the well clear volume becomes smaller, or the well clear volume is violated more often, operations at lower altitudes can be proven to be just as safe if utilizing reduced well clear volumes in conjunction with obstacle and terrain shielding. To demonstrate an equivalent level of safety, a robust simulation environment is used to generate safety metrics for analysis.

The following are my major contributions within this thesis: (1) Simulation additions to the Mid-Atlantic Aviation Partnership's DAA Simulator: (1a) Development of a shielded avoidance algorithm; (1b) Development of a descend only avoidance algorithm; (1c) Development of various helper functions, including the fly to point function; (2) Demonstration and justification for the use of smaller safety metrics; (3) Use case methodology outline; (4) Demonstration of using traffic density analysis for further justification of safer low-level flight operations

1.3 Aviation Preliminaries

There are a number of standard terms that are considered common within the UAS indsutry that this thesis will use. The term "ownship" will reference the small unmanned aircraft for which avoidance maneuvering will be performed. An "intruder" aircraft is the manned traffic that the ownship is avoiding. This study only considers one intruder and one ownship per encounter. A "shield" is a volume of airspace adjacent to terrain or obstacles, such as a pole or building, where intruders typically do not fly due to regulatory or practical safety constraints. The term "proximity" refers to the the vertical distance from the ground or the horizontal distance from the obstacle that the ownship must remain within to be considered

shielded from an intruder. A "shielding volume" is a group of proximity values. Figure 1.1 depicts a top-down view of different levels for the horizontal shielding volume. Each ring represents a new volume and the obstacle is represented by the back square in the middle of the volume rings. The figure depicts four different levels: 50 foot, 100 foot, 150 foot, and 200 foot shielding volumes.

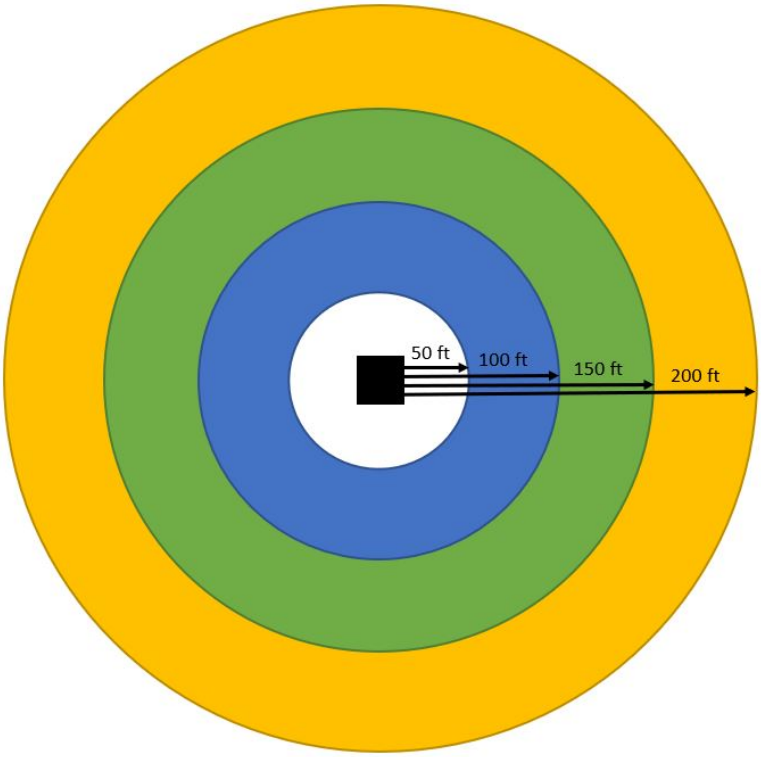


Figure 1.1: Top-Down View of Example Shielding Volumes

A "trajectory" is the path of a single ownship or intruder aircraft. Two trajectories, one ownship and one intruder, are combined to form an "encounter." An encounter typically references one run in the simulator, and the term "simulation" generally references multiple encounters run together to generate statistically significant metrics. The number of encounters run per simulation to develop a given safety metric is always provided. The distance at the closest point of approach, or DCPA, is the closest distance recorded between the ownship and intruder during a given encounter.

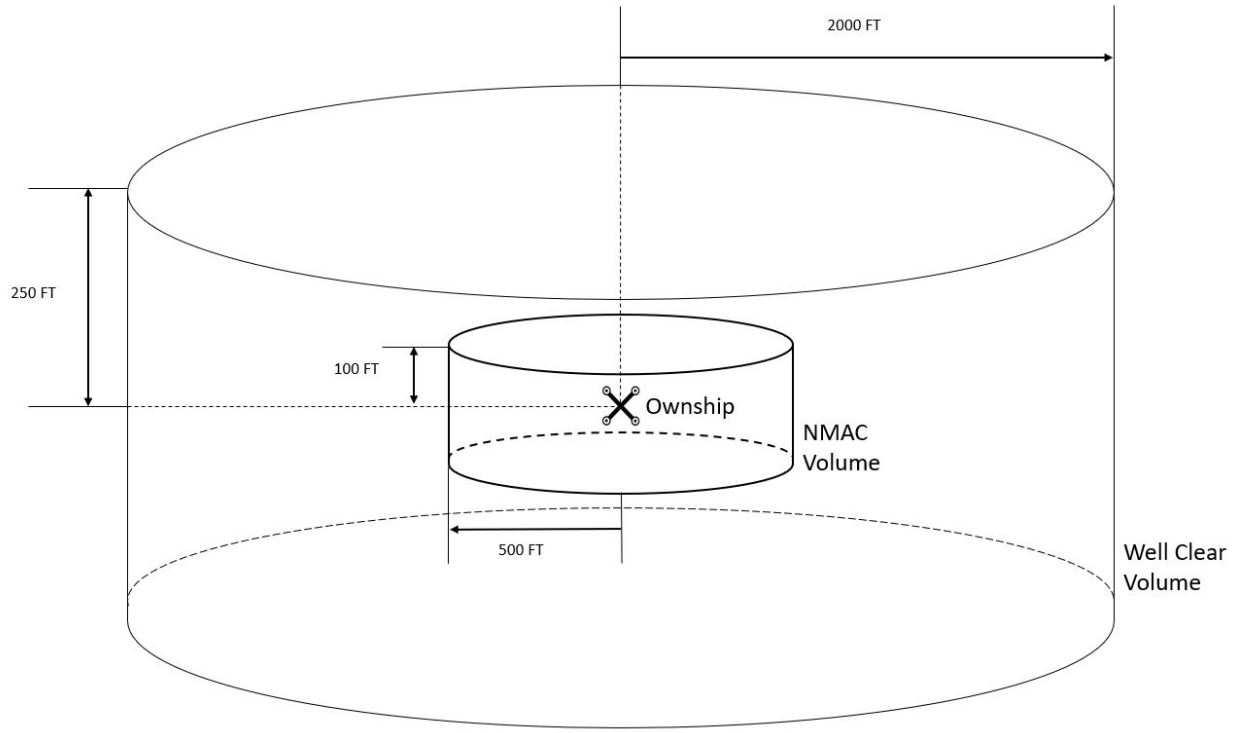


Figure 1.2: Traditional Well Clear and NMAC Volumes

Each encounter has a horizontal, vertical, and slant range DCPA. Each encounter also has an unmitigated and a mitigated DCPA. A mitigated encounter means an encounter in which an avoidance maneuver was performed and an unmitigated encounter is an encounter in which there was no avoidance performed. The unmitigated DCPA is defined as the closest distance between the ownship and intruder during an encounter, if the ownship does not perform an avoidance maneuver. The mitigated DCPA represents the closest distance in the case where the ownship executes an avoidance maneuver.

Conventional metrics used for safety analysis in both manned and unmanned aviation are well clear and near mid-air collision (NMAC) volumes. The traditional well clear will be considered to be, and is defined in all simulations to be, 2,000 feet horizontally and 250 feet vertically. The value for a near mid-air collision (NMAC) metric is 500 feet horizontally and 100 feet vertically. The history of these metrics will be further expanded upon in Section 2.1.

These metrics are depicted in Figure 1.2.

Another metric leveraged for this research is a small NMAC, or sNMAC. This metric is not considered standard, but has recently been used to quantify safety for encounters between two small UASs. The use of this metric will be further expanded upon in Section 3.3.2.

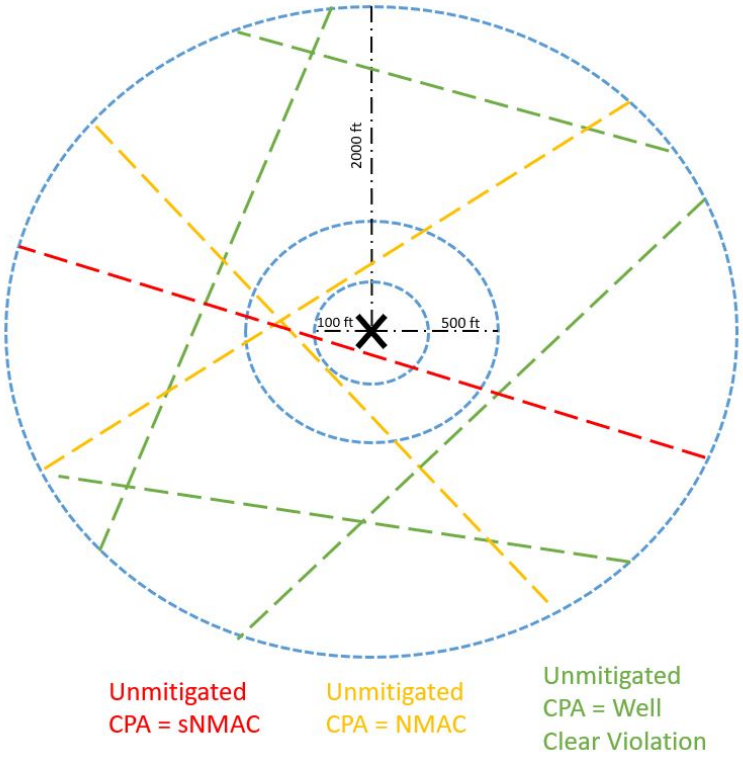


Figure 1.3: Top-Down View of Safety Metrics

Figure 1.3 depicts a top-down view of an ownship with the different horizontal thresholds of each safety metric. The X in the middle represents the ownship and the dotted lines represent different intruder trajectories. The red dotted line represents an sNMAC encounter, given a 100 foot horizontal threshold, the yellow dotted lines represent NMAC encounters, and the green dotted lines represent well clear violation encounters.

Chapter 2

Review of Literature

This chapter provides a detailed literature review of several topics relating to encounter simulations and shielded operations. The topics are broken up into 3 major topics: History of the "well clear" and "NMAC" definitions introduced in Section 1.3, background of avoidance algorithms for low-level flight operations in Section 2.2, and an introduction to shielded operations in the final section.

2.1 Well Clear and NMAC History

The concept of remaining "well clear" has been around since before the inception of the UAS industry, and was traditionally used as a qualitative assessment made by manned aircraft pilots. More recently, with the expansion of beyond visual line-of-sight (BVLOS) operations, a standard quantitative definition is required for operators to adhere to. There has been an abundance of recent literature in this area, with various recommendations for well clear definitions or papers summarizing existing work done by various government and educational organizations: [18], [37], [46] [41], [47].

The FAA has also attempted to define "well clear" for manned and unmanned aircraft. 14 CFR §91.113 states that "...vigilance shall be maintained by each person operating an aircraft so as to see and avoid other aircraft." For unmanned applications, the term "see and avoid" has been adapted to "sense and avoid" or "detect and avoid" (DAA) and there are efforts in place to define specific requirements to ensure the detection equipment is sufficient for remaining well clear of other aircraft in the NAS. Additionally, 14 CFR § 107, the regulation governing unmanned aircraft weighing less than 55 pounds, does not stipulate a specific distance unmanned aircraft need to stay away from manned aircraft, only that "No person may operate a small unmanned aircraft so close to another aircraft as to create a collision hazard" (14 CFR §107.37b). In 2018, the FAA released a draft Advisory Circular (AC 90-WLCLR) titled "Well Clear Definition for Small Unmanned Aircraft Systems Operating Beyond Visual Line of Sight" [50]. This AC proposed quantitative definitions for the horizontal and vertical well clear distances of 2,000 feet horizontally and 250 feet vertically respectively, and would include all operations conducted under §107, 91, or 135 in class G and E airspace and below 1,200 feet AGL. The AC was released as a draft for comment, but was never published by the FAA.

The International Civil Aviation Organization's (ICAO) Rules of the Air (tenth edition, published in 2005) [26] mentions staying "well clear" multiple times in the standard, but never explicitly defines the term. In section 3.2.1 titled "Proximity," it is stated that "An aircraft shall not be operated in such a proximity to other aircraft as to create a collision hazard." In 2015, ICAO published the Manual on Remotely Piloted Aircraft Systems (RPAS) [27]. In the manual, it is mentioned that the term "well clear" may need to be quantified from a system design perspective, as it is an aspect of "see and avoid" that is subjectively determined by a pilot in manned aviation (section 10.10). The document defines remain-well-clear (RWC) as "the ability to detect, analyze and maneuver to avoid a potential conflict

by applying adjustments to the current flight path in order to prevent the conflict from developing into a collision hazard" (section 10.12.3).

ASTM Standard Specification for Detect and Avoid System Performance Requirements (F3442/F3442M) has established the 2,000 foot horizontal 250 foot vertical metric as the standard for well clear [9]. In F3442/F3442M, the well-clear boundary is defined as above 2,000 feet horizontally and +/- 250 feet vertically for "lower risk airspace." It is also noted that "Remaining well clear is meant to support compliance with 14 CFR §91.111 and §91.113 or §107.37 (or international equivalents) and reduce the chance of creating a collision hazard and therefore a collision." One aspect of remaining well clear that is not addressed, but will possibly be addressed in following versions, is the concept of using shielded operations as a means to RWC.

It is obvious that the well clear definition has become, and will continue to be, a very important metric to quantify safety between unmanned and manned aircraft. One of the main purposes of this work is to evaluate the use of shielding as a means to remain well clear at low altitudes. But the general definitions that have been proposed in the past, specifically by the FAA and SC-228, may prove to be too restricting and not necessary to maintain an equivalent level of safety in certain operational environments.

NMAC is a metric historically used to evaluate risk of a collision between two aircraft. Weinert et al. [58] provide a brief but comprehensive summary of the history of the NMAC definition within the FAA, as well as its use in the development of collision avoidance systems. The standard volume used to determine if an NMAC occurs is 500 feet horizontally and 100 feet vertically.

The FAA has very detailed procedures for reporting an NMAC in Volume 7 Chapter 4 of the FAA Flight Standards Information Management System documentation [7]. It should be

noted that the definition of an NMAC for reporting purposes does not include the separation of vertical and horizontal components; it is defined to be "An incident associated with the operation of an aircraft in which a possibility of collision occurs as a result of proximity of less than 500 feet to another aircraft or where a report is received from a pilot or other flight crewmember stating that a collision hazard existed between two or more aircraft" [6].

Weinert et al. cite the current development and evaluation of ACAS sXu as a technology whose development requires the need for a new quantitative definition to measure collision risk that is "appropriately tailored for operations by these new airspace entrants." This notion will be further expanded upon with discussion of a small NMAC metric in Section 3.3.2.

2.2 Avoidance Algorithms for Low-Level Operations

2.2.1 Terrain Awareness

Utilizing terrain following and terrain avoidance logic on aircraft, manned and unmanned, is not a new concept in the aviation industry. According to Pelosi et. al. [43], terrain masking techniques began emerging during the 1950s with the rise of microwave radar detection air defense weapons. Krachmalnick et. al. [30] describes the design, simulation, and flight testing of an automatic terrain-following flight control system built for a fixed-wing military aircraft in 1968. Additionally, a NASA technical memorandum [21] from 1986 describes research for the development of terrain-following/terrain-avoidance algorithms at Ames Research Center to achieve Nap-of-the-Earth (NOE) flight by helicopter platforms. More recent advances in automation have allowed for this foundation to flourish, and major developments have been made on more capable terrain-avoidance/terrain-following systems.

At the core of UAV terrain awareness is some variation of an avoidance algorithm. In

the subject of terrain considerations, most research and development in this area can fall into one of two categories: NOE flight to reduce the risk of radar vulnerability during flight for military aircraft [43], [30], [21], or simply for avoiding terrain during normal and intruder avoidance flight operations [49], [52], [14], [40]. This research proposes using terrain awareness logic in a novel way, as a means to mitigate the possibility of a mid-air collision when in the vicinity of an intruder aircraft and to examine how flight close to terrain might allow for smaller well clear volumes having an equivalent level of safety.

2.2.2 Intruder Awareness

Another consideration when discussing low-level UAV operations is aircraft-to-aircraft avoidance. There has been a considerable amount of research focused around "cooperative" manned and unmanned intruders, i.e., intruder aircraft that have the capability to transmit avoidance-pertinent information, such as position and ground speed. Krüger et. al. in [31] introduce an algorithm for automatic separation among cooperative UAVs and other air traffic. This work is driven by the assumption that all positions are known via mobile phone network connections. Radanovic et. al. in [45] discuss a scalable UAV-to-UAV conflict management framework by generating spatiotemporal interdependencies, which are characterized by the potential loss of well-clear, should an aircraft choose to maneuver. In this work, it is assumed that all the unmanned aircraft are cooperative and therefore all current positions are known (with no error assumed) to generate predicted trajectories and to determine their effect on air traffic. The research operating under the cooperative traffic assumption has resulted in robust architectures for UAS traffic managment (UTM) systems and corresponding avoidance algorithms [10], [44]. To enable broader application, and for purposes of specific safety cases for implementation of the proposed research, this thesis does not assume a specific detection method, and is focused on encounters between small UA and

all low-level manned aircraft.

Furthermore, assumptions are often made that all manned aircraft will be cooperative when discussing small UAS integration into the National Airspace System (NAS). Since the focus of this research is operations at low altitudes and in airspace where there is not an FAA mandate for transponder equipage, the cooperative assumption will not always hold true. This research is equally applicable to both detection methods, and therefore does not generally assume intruders are cooperative or non-cooperative. This concept will be further expanded upon in Section 3.3.1.

2.2.3 Methods for Autonomous Navigation near Terrain and Obstacles for UAV Flight

One way to implement autonomous navigation in UAV flight is via sensors on-board the vehicle. A common technique to achieve this is with a laser range finder [33], [20], [17]. Though laser range finders are a very useful tool for terrain awareness and have become very mature over the years, the size and cost of the sensors often exclude them from being a realistic method for small UASs.

Another popular method for UAV navigation from on-board sensors is via vision-based sensors [15], [35]. One benefit of using optical sensors is that they are generally cheaper and have a lower size, weight, and power (SWAP) than laser range finder techniques. Though there is an abundance of different kinds of vision-based sensors that can be used for autonomous navigation, one of the drawbacks of vision-based sensors is that longer range estimation is unreliable and might require multiple cameras to improve upon this deficiency. Though this can increase the payload weight on-board the aircraft, the technology is continually becoming lighter and smaller. One application for onboard vision-based sensors is short-range

obstacle avoidance. Newer small UAVs, such as the Skydio 2 [4] and the DJI Mavic Pro [19], have multiple high-resolution cameras to perform obstacle avoidance. At present, however, detection ranges for vision-based systems are limited. For example, sensors onboard the DJI Mavic Pro can "see" obstacles to up to 49 feet away.

An additional tool used for terrain awareness is a publicly available digital terrain elevation database (DTED). Gellerman et. al. [24] utilized the National Elevation Database (NED), made available by USGS, for avoidance algorithm work. The NED was used in conjunction with a Digital Obstacle File (DOF) database, which is a data set of obstacle information for every state in the United States made available by the FAA. The obstacle database has information about every tower, building, etc. that is "of interest to aviation users" [12]. The obstacles are classified by location (latitude and longitude) and altitude in both AGL and MSL. These were merged using Matlab and additional modifications were made to adapt a DAA algorithm developed in previous work [40] to handle terrain avoidance. Using terrain databases has drawbacks, such as the availability and accuracy of the data, but they can be used in conjunction with other sensor based avoidance to become a solid option for obstacle awareness. This multi-pronged approach is leveraged in this research and is further expanded on in Section 3.5.

2.2.4 Avoidance Algorithms

Perhaps the most encompassing research and development done in regards to avoidance algorithms comes from a collaboration between the FAA, MIT Lincoln Laboratories, and Johns Hopkins Applied Physics Lab to develop ACAS sXu [13]. ACAS (Airborne Collision Avoidance System) is an ongoing effort built from the foundation of TCAS (Traffic Collision Avoidance System), which is a deterministic collision avoidance method developed for

manned aircraft that provides pilots with traffic advisories. The ACAS X family was developed based on modernization of Air Traffic Management efforts using probabilistic modeling [36]. ACAS Xa is the general purpose ACAS X designed for large aircraft, and ACAS Xu was specifically developed for large and medium sized unmanned aircraft. From there, ACAS sXu was developed for small unmanned aircraft systems.

TCAS and ACAS are considered the standard in the industry. RTCA Special Committee (SC)-147 has defined performance standards for TCAS and TCAS II, a follow-on from TCAS that provides both traffic advisories and resolution advisories (avoidance suggestions). SC-147 also released DO-385, a Minimum Operational Performance Standards (MOPS) for ACAS X, in 2019 and DO-386 Volume I & II, MOPS for ACAS Xu, in 2020. Volume I of DO-386 is general MOPS and Volume II includes algorithm design guidance. ACAS sXu also has prototype obstacle awareness capabilities, and is under active discussion in RTCA SC-147. SC-147 has not yet issued standards for ACAS sXu, but is scheduled to release an initial version in 2022.

NASA has also performed significant research in developing DAA systems. In an effort to support UAS integration into the NAS, NASA researchers developed their own avoidance algorithm called DAIDALUS (Detect and AvoID Alerting Logic for Unmanned Systems) [42]. DAIDALUS utilizes the mathematical definition of a well clear volume to maneuver around intruding aircraft. This algorithm will be expanded upon in future sections, as the mathematical definition for a well clear boundary violation is leveraged for the avoidance algorithms developed for this research.

A newer innovation in the field of avoidance algorithms is FLARM ("Flight Alarm") [1]. FLARM is similar to ADS-B in that it provides situational awareness to pilots, but can also include predictive tracks for an intruder and collision risk estimates. Additionally, it can perform flight path planning based upon aircraft and flight parameters, such as altitude and

wind speed, and broadcasts the flight path with the unique identification number over a radio channel to other FLARM devices. Newer FLARM devices can also incorporate ADS-B and Mode-S traffic data. They also have the ability to load obstacle and terrain databases onboard the system for obstacle and terrain avoidance. FLARM has a maximum operating range up to 5 nautical miles (compared to the 50 nautical mile range of ADS-B) and only works in the region in which it was purchased. According to Marques et al. [38], FLARM is a suitable candidate for a DAA solution, though it does not offer automatic avoidance and is not yet fully developed.

2.3 Shielded Operations

The basis of the idea behind using shielded operations is that manned aircraft will have stricter limitations than small unmanned aircraft for how close they can fly to structures and terrain. Although shielded operations have not been regulated by the FAA, in 14 CFR §107.51 it is stated that "The altitude of the small unmanned aircraft cannot be higher than 400 feet above ground level, unless the small unmanned aircraft: (1) is flown within a 400-foot radius of a structure; and (2) Does not fly higher than 400 feet above the structure's immediate uppermost limit." Though this regulation is alluding to shielding operations, it is not directly applicable to avoidance operations, as the regulation is specifically referring to altitude limitations and does not address avoidance requirements.

Another reference to shielded operations is found in the Joint Authorities for Rulemaking of Unmanned Systems (JARUS) guidelines on Specific Operations Risk Assessment (SORA) definition of "Atypical Airspace," found in the Annex I - Glossary of terms. Atypical airspace is defined as the following: " ... b) Airspace where normal manned aircraft cannot go (e.g. airspace within 100 ft. of buildings or structures) ..." [8]. Though again, this definition

is not specifically referring to avoidance against manned aircraft operations, the use of a quantitative value for atypical airspace in proximity to buildings could suggest a basis for defining the proximity required to utilize shielding during avoidance maneuvers.

In June 2020, ICAO published the ICAO Model UAS Regulations (§101 and 102) [28]. Within this regulation, "Shielded Operation" is defined as "an operation of an aircraft within 100 meters of, and below the top of, a natural or man-made object." Though this notion of a shielded operation is specifically defined, the only place in the document that the definition is referenced is under section 31: Night Operations, where it is stated that "A person shall not operate a UA at night unless the operation is (1) indoors; or (2) a shielded operation." Though this definition is specific in using the term shielded operations, the term is being used more in regards to airspace permissions than avoidance operations. It is also important to note that the SORA is scoped to cooperative traffic only, generally meaning that the intruder aircraft is equipped with ADS-B Out, a concept that will be further expanded upon in Section 3.3.1.

Perhaps one of the most extensive research papers outlining a foundation for shielded operations for avoidance comes from a paper titled "Three Quantitative Means to Remain Well Clear for Small UAS in the Terminal Area" [32]. In this paper, Lester and Weinert summarize existing well clear definitions and suggest three "means" to ensure that sUAS remain well clear from manned aircraft. One of the mitigation strategies the authors expand upon ("Means #2") is Obstacle Shielding Operations. The quantitative shielding volumes they give are for small UASs to remain within 50 feet vertically and 250 horizontally from "a natural or man-made obstacle" when at least 2,000 feet away from airport movement areas (ramps, helipads, helicopter routes). The authors justified the vertical value by assuming that manned aircraft below 500 feet will still attempt to remain at least 100 feet above terrain and obstacles, resulting in at least 50 feet of separation between manned and unmanned

aircraft. Additionally, the authors utilized flight data collected from manned helicopter air ambulances (HAAs) within or near Boston, MA to support the proposed horizontal shielding criterion. Their analysis of the flight data concluded that HAAs rarely operate below the top of buildings, and are almost always in vertical transit when flying within 250 feet of the center of the nearest building, indicating takeoff or landing.

Table 2.1: Summary of Existing Shielding Volume Definitions/Recommendations

	ASTM DAA Standard	**Part 107**	**JARUS SORA**	**ICAO Part 101**	**3 Quant. Means**	**NCDOT Waiver**
Applicable UAS	Maximum wingspan of 25 feet, operating below 100 knots	Under 55 lbs (MGTOW)	Any	Under 25 kg (55 lbs)	Wingspans of 25 feet or less	Under 55 lbs
Applicable Airspace	"Lower risk" airspace	NA	NA	NA	Any (Typically "lower risk")	1 NM outside airports or heliports
Horiz. Proximity Dist.	Varies	400 feet	400 feet	100 feet	100 feet	50 foot lateral
Vert. Proximity Dist.	0 feet	400 feet	100 feet	0 feet	50 feet	50 foot lateral

On October 1, 2020, the FAA issued a §107 waiver to the North Carolina Department of Transportation. The waiver specifically waived 14 CFR §107.31, which stipulates the requirement of visual line-of-sight operations. Though it is not the first of its kind, it is the first waiver issued by the FAA that does not require the use of a "visual observer" (VO) or surveillance technology (e.g., radar, ADS-B, etc.) during a BVLOS operation [25]. The reason this requirement is relaxed is likely due to the nature of the waiver, which was granted specifically for bridge inspection operations and requires the UA to remain below the deck of the bridge and within 50 feet of the bridge itself. This waiver is the first time the FAA has provided a waiver specifically for what could be considered shielded operations.

The purpose of this thesis is to explore more general shielded operations as a means to remain well clear. Specifically, the goal is to outline a methodology to demonstrate that an equivalent level of safety can be obtained by using obstacles to shield an unmanned aircraft from manned intruders, even if the traditional quantitative definition of the well clear volume is violated.

Chapter 3

Methodology

This chapter will review the methodology for the basis of this research. The first section describes the encounter models leveraged during simulation. The second section will give an overview description of the simulation tool used for all analyses with a detailed description of the main functions used within the simulation. Section 3.3 provides context and explanation for the methods used for demonstrating an equivalent level of safety for operations near terrain and obstacles.

Sections 3.4 and 3.5 provide detail on the two shielding scenarios, vertical proximity and horizontal proximity. The aim of this research was to develop a method for quantitatively defining "proximity" to obstacles and to use that definition to demonstrate an equivalent level of safety. When defining proximity, the metric was broken up into two components: vertical proximity and horizontal proximity. The vertical proximity case references the vertical distance from the ownship to the terrain, and horizontal proximity references the horizontal distance from the ownship to an obstacle. The horizontal proximity case is a more complex problem, and the simplest implementation to analyze different quantitative definitions of horizontal proximity is to simulate a structure (tower, building, etc.) fixed to the ground for

the ownship to maneuver to as a way to stay shielded from a manned intruder.

Section 3.4 will explain the method for testing vertical proximity to the ground, using a simple flat earth case, and Section 3.5 provides explanation and justification for the methodology used for analyzing the horizontal proximity to obstacles.

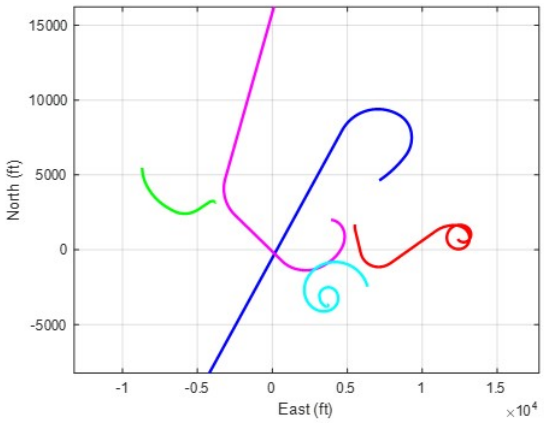
3.1 Encounter Models

In order to analyze an avoidance algorithm, unmitigated encounter models needed to be developed. There has been a robust amount of research in generating representative encounter models both for intruders and UASs to be used for simulation and analysis. Some of the most extensive and open-source modelling for both manned and unmanned aircraft has been done by MIT Lincoln Laboratories (LL): [29], [55], [57], [59], [54], [51]. MIT LL has provided powerful analysis tools for airspace encounter models on their GitHub [53]. These tools can be utilized to model both the ownship and intruder. Additionally, they have created a tool to pair individual trajectories, either from a separate CSV file or from their encounter model network, with one another to form an encounter.

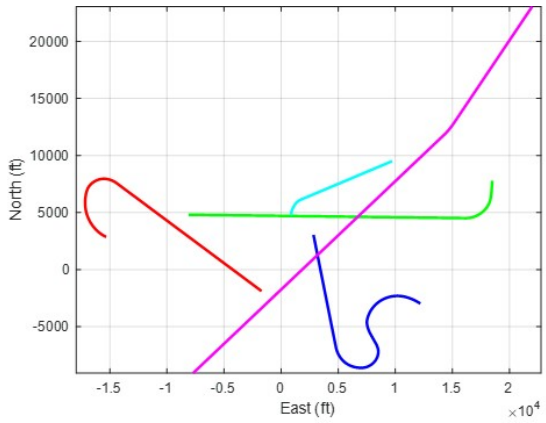
For simplicity, all ownship trajectories used in this research were generated using Matlab. They were CSV files of straight-line trajectories of constant altitude and airspeed. The altitude of the ownship varies between encounters, but remains constant for the duration of a single encounter, until an avoidance maneuver is triggered. Though this behavior is not representative of all ownship trajectories for all use cases, this ownship trajectory can be considered a nominal trajectory for many operations. Such operations may include point-to-point package delivery or utility inspection.

The intruder models used varied between the vertical and horizontal proximity analysis.

For the vertical proximity encounters, the intruder was modelled using MIT LL's Bayesian network. The specific models used were trained using OpenSky Network ADS-B data. OpenSky Network is a Switzerland-based non-profit association that provides access to air traffic control data to the public [2]. The OpenSky Network MIT LL models are broken up into the following categories: fixed wing multi-engine, fixed wing single engine, and rotorcraft. Figure 3.1 depicts example trajectories generated from the MIT LL models. Figure 3.1a represents trajectories from their rotorcraft model and Figure 3.1b represents trajectories from the fixed wing model.



(a) Five Example MIT LL Trajectories, Rotorcraft



(b) Five Example MIT LL Trajectories, Fixed Wing

Figure 3.1: Example Trajectories from OpenSky Network MIT LL Encounter Models

The models are also broken up into sub-categories based upon the Mode 3A/C code squawked by the aircraft's transponder. Each category has a 1200-code only and 1200-code excluded subcategory. If an aircraft is squawking 1200, they are indicating that they are operating under visual flight rules, therefore the 1200-code only represents data in which the intruder was operating under visual flight rules, and the 1200-code excluded data represents intruders that were not necessarily operating under visual flight rules, i.e., possibly operating under a filed instrument flight rules flight plan.

According to Underhill and Weinert [51], the 1200-code only models can be used as representative datasets for low altitude non-cooperative aircraft and the 1200-code excluded models can be used for representative data sets for low altitude cooperative aircraft. Since this research does not make an assumption for the type of detection used, the "all code" version of the data was used, meaning both 1200-code only and 1200-code excluded data were used to generate the intruder trajectories.

For the horizontal proximity analysis, both the intruder and ownship trajectories were generated in Matlab using only simple aircraft kinematics calculations. These trajectories will be further expanded upon in Section 3.5.

3.2 DAA Python Simulation

Both the avoidance algorithm and the simulation used to develop and analyze the algorithms were written in Python version 3.8. The core of the Python simulation tool was developed in-house by a Virginia Tech Mid-Atlantic Aviation Partnership engineer, Andrew Kriz. The simulator is capable of generating encounters dynamically or importing established trajectories from a file (or a combination of both methods). For dynamically generated trajectories, the simulation uses simple aircraft kinematic constraints of bounded (minimum/maximum) accelerations and velocities; there is currently no aircraft performance modelling for the ownship or intruder. Dynamically generated encounters were used for testing the avoidance algorithms, but established trajectory files were used for equivalent level of safety analysis simulations, as described in the previous section.

The simulation environment also has the ability to model the detection probability of different DAA sensors. For the purpose of the majority of this research, all encounters were assumed to have a detect probability of one, i.e., if the intruder is within the well clear surveillance

volume, it will be detected and accurately tracked. In Section 5.2, a specific use case is explored where the assumption of the perfect sensor is relaxed and a radar sensor model within the simulator is used.

3.2.1 Description of Functions

There are a number of core functions used within the Python simulator. This section explains in detail the main functions leveraged for the purposes of this research.

Well Clear Volume Check

The prediction of well clear violation is determined by an algorithm based upon NASA's DAIDALUS well clear logic [42]. The `well clear volume` (`WCV`) function takes the relative positions of the ownship and intruder and determines if the well clear threshold is violated or not. The well clear threshold is split up into the horizontal distance threshold, DTHR, and altitude threshold, ZTHR, and is determined by the user. For all simulations, DTHR was set to 3 times the horizontal component of the well clear volume and ZTHR was set to the vertical component of the well clear volume, or 6000 and 500 feet, respectively.

The original function used in DAIDALUS also has the ability to implement time threshold, `tau`, to predict if the well clear threshold will be violated at a time in the future, given the current locations and velocities of the aircraft. For example, if `tau` was set to 30, the algorithm would calculate the horizontal positions of both aircraft 30 seconds into the future to determine if the distance threshold will be violated using a basic assumption that the aircraft will remain in their current states. Similarly, a user can specify a time to co-altitude threshold, `TCOA`, to determine if the aircraft will be within the altitude threshold at a given time step in the future. For simplicity, and operating under the assumption that the ownship

does not have access to the exact location of the intruder, these features were disabled for all simulations in this thesis.

The `WCV` function is called at every time step and returns a value of True or False. If True is returned, then the well clear threshold has been violated and the ownship will execute an avoidance algorithm. For the purposes of this research, the alerting timeline was solely based upon distance (i.e., tau and TCOA were set to zero and the `WCV` function will only return True if the DTHR and ZTHR are violated). For a more efficient alerting timeline, the function could calculate an estimated future trajectory and only execute an avoidance maneuver once it is determined that well clear will be violated a certain number of seconds (or minutes) in the future.

Maximum Descent

The `maximum descent` function is used in the descend only avoidance algorithm detailed in Subsection 3.2.1. This function is used to command a maximum rate descent as limited by the user-specified variables for maximum descent rate and maximum vertical acceleration. The function starts by calculating the the allowed acceleration change for the current time step, which is the maximum vertical acceleration multiplied by the sampling frequency of the simulation. It then calculates the difference between the current velocity and the desired velocity. If the difference is not zero, the velocity is changed to the current velocity subtracted by the allowed acceleration change. This function was written by Andrew Kriz.

Accelerate Toward

The `accelerate toward` function is used, for the purposes of this research, in conjunction with the `command turn to` function defined in the following subsection. This function is

used to emulate the lateral direction changes of an autopilot-equipped multirotor aircraft, given a desired velocity set-point, while obeying speed and acceleration constraints. The result of this function is an acceleration up to a specified velocity vector with the magnitude of acceleration governed by the user-specified "maximum lateral acceleration" parameter. It first determines the direction of acceleration by taking the difference between current velocity and the specified desired velocity. It then uses basic kinematic equations to command an acceleration in that direction. This function was written by Andrew Kriz.

Command Turn To

The `command turn to` function is used to tell the aircraft to turn to a desired heading with a specified turn rate and direction of turn. There is first a simple "IF" statement to differentiate between the turning parameters of a multirotor verses a fixed wing platform. All simulations for this research assumed a multirotor platform, though it is not expected that results would vary significantly using the fixed wing platform parameters. The function uses simple kinematic equations to calculate the desired east, north, and up components of the velocity, and uses the `accelerate toward` function for multirotor platforms to command a new desired velocity toward the desired direction of turn, based upon the desired heading. This function was written by Andrew Kriz.

Fly To Point

The `fly to point` function was written with the specific application of shielding in mind. It takes the current location of the ownship, the desired final location (the location of the shield for the purpose of this research), and first calculates the current distance between the two positions. If the calculated distance is less than the specified shielding volume,

which is varied throughout the simulations as described in Section 4.2, it will calculate the desired heading toward the shield and command a turn using the `command turn to` function described in the previous subsection. This function is capable of taking the location of the shield in either east, north, up coordinates, or in latitude, longitude, altitude, as provided in the database outlined in Section 3.4. If the calculated distance is less than or equal to the shielding volume, the function forces the velocity to zero to simulate a hovering aircraft.

Descend Only Avoidance Algorithm

The `descend only avoidance` function was written to be a simple avoidance algorithm to descend 20 feet above the ground during an avoidance maneuver while continuing on the predetermined flight path. This is the avoidance algorithm used for the vertical proximity analysis detailed in Section 3.4. The avoidance algorithm first does a well clear volume check, via the `WCV` function as outlined in Subsection 3.2.1. If the well clear threshold is determined to be violated, and the current position of the ownship is above 20 feet, then the algorithm commands a maximum descent using the `maximum descent` function outlined in Subsection 3.2.1. If the well clear threshold is determined to be violated and the ownship is already at or under 20 feet, then the function commands a vertical velocity of 0 knots to keep the ownship at its current altitude while continuing forward on its flight path.

Shielding (as a means of) Avoidance Algorithm

The `shielded avoidance` algorithm is the avoidance algorithm used for horizontal proximity analysis, as outlined in Section 3.5. Similar to the `descent only avoidance` algorithm, it first begins with a `WCV` threshold check. If the threshold is violated, then it first searches through an obstacle database based upon the current location of the ownship. It will first

return a dataframe of all obstacles within a specified distance from the ownship, which is set to 500 feet for this research. This distance varies based upon the type and performance parameters of the ownship.

If multiple obstacles are returned, then it will select the closest obstacle. There is also a part of the function that will check to see if the bearing from the ownship to the shield is in the direction of the intruder, but that feature is not used for this research. This was omitted to generalize the application to consider that the ownship may not have specific information about the location of the intruder, which is more representative of a ground-based detection function that is not fully integrated with the avoidance function.

Once an obstacle is chosen for shielding, the function will use the `fly to point` function to initialize a maneuver toward the shield. The function will only look up a suitable shielding option once, so once a shield is selected, the algorithm will maneuver toward that shield for the duration of avoidance. Once the ownship is determined to be within the shielding volume of the shield by the `fly to point` function, outlined in Subsection 3.2.1, it will hover in that location for the duration of the encounter.

3.3 Equivalent Level of Safety Method

To justify that shielding avoidance is effective, an equivalent level of safety (ELOS) must be demonstrated relative to established requirements. The following subsection expands on how a risk ratio is calculated. Furthermore, in the case of horizontal proximity, intruders flying as close as 500 feet to an obstacle will almost always result in an NMAC violation, as the horizontal threshold is 500 feet. A manned aircraft flying as close as 500 feet, or 300 feet in the case of helicopters, is considered to be an extreme case, and the NMAC criterion would not be considered suitable to properly evaluate safety. Therefore, the use of a small

NMAC is explored in Subsection 3.3.2.

Though a risk ratio metric can show how effective an isolated avoidance maneuver is, it does not take into account other applicable parameters, such as traffic density. The probability that a mid-air collision will happen can be generalized by combining the two biggest factors: (1) the probability that a manned aircraft will enter the flight volume of the UAS and (2) the probability that the UAS will collide with the manned aircraft (i.e., failure of the DAA system). The final subsection explores the use of traffic density as an additional means of demonstrating an equivalent level of safety.

3.3.1 Risk Ratios

Risk ratios have become a standard performance metric for unmanned aircraft DAA safety quantification. This metric has been used to analyze and define well clear recommendations [16], to analyze the effectiveness of collision avoidance systems and their performance requirements [48], [34], [22], and to verify the safety of an updated version of a Traffic Collision Avoidance System (TCAS) II [39], [23]. The definition of a risk ratio can vary slightly depending on the application.

$$\text{NMAC Risk Ratio} = \frac{\text{Number of NMACs with mitigation}}{\text{Number NMACs without mitigation}} \tag{3.1}$$

For the purpose of this research, it is defined as the number of violations of a parameter (well clear, NMAC, or sNMAC) given mitigated encounters for a given set of initial geometries divided by the number of violations given unmitigated encounters for the same initial geometries, as shown in Equation (3.1). Equation (3.1) is written specifically to apply to NMAC, but the same equation can be used for any parameter, such as well clear. The value

generated is an indication to how effective an avoidance algorithm is. For example, a well clear risk ratio of 0.9 would mean that 90% of encounters resulted in a well clear violation, even after mitigation (i.e., avoidance maneuvering) was implemented. A lower value of risk ratio indicates a more effective mitigation. Because the simulation is designed to result in an NMAC or well clear violation for all encounters, the denominator will typically be the total number of encounters simulated.

Standards for risk ratio requirements have been published by ASTM in their Standard Specification for Detect and Avoid System Performance Requirements. These metrics are outlined in Table 3.1.

DAA Quantitative Performance Requirements		
	NMAC RR	Well Clear RR
Intruder Equipped with Transponder or ADS-B Out	≤ 0.18	≤ 0.40
Non-Cooperative Intruder	≤ 0.30	≤ 0.50

Table 3.1: Risk Ratio Metrics

This research does not presume a particular method of detection, as reflected in the left column of Table 3.1, and analysis will be conducted using the risk ratios for both non-cooperative and transponder equipped intruders.

3.3.2 sNMAC

As seen in the horizontal proximity methodology, this research leverages the usage of a small Near Mid-Air Collision or "sNMAC" criterion. The use of sNMAC has not been standardized in the small UAS industry, but with the development of collision avoidance systems for sUAS, such as ACAS sXu, there has become a need for developing such metric. Alvarez et al. [13] used a scale of 1/10 of the standard NMAC volume to define sNMAC for evaluating the

ACAS sXu system via risk ratios. The 1/10 scale was used with the preface that there was no reference volume to measure collision risk between two small UASs at the time.

Weinert et al. [58] introduce an sNMAC evaluation methodology for smaller UAS. The research focused on smaller UAS to smaller UAS encounters, and did not consider smaller UAS to manned aircraft encounters. "Smaller" UAS was defined to be UASs with wingspans of 25 feet or less and under 55 pounds maximum gross take-off weight, with no restrictions on airspeed or vertical rates. This work studied the impact that the size (wingspan and height) had on different unmitigated sNMAC criteria. In their conclusions, the authors had three sets of sNMAC criteria to represent the different scenarios: one where the wingspan distribution was uniform from 1-25 feet, one where there was a higher concentration of smaller (1-5 feet) wingspans, and one where there was a higher concentration of larger (20-25 feet) wingspans. The uniform distribution sNMAC recommendation was 50 feet horizontal and 15 feet vertical to obtain an unmitigated probability of mid-air collision of 10% (0.1).

Weinert et al. propose a minimum horizontal sNMAC metric of 50 feet, because the maximum wingspan considered for "smaller" UAS is 25 feet and it follows their working assumption that the sNMAC dimension should not exceed the sum of the wingspans of the two smaller UASs considered in close encounter. They did not consider the mitigated risk, but they suggested this could be considered in follow-on research for their work.

This thesis proposes using an sNMAC criterion to encompass encounters between small UAS and manned, general aviation aircraft that are flying at low altitudes, where the small UAS can use shielding to avoid a collision. The well clear violation metric historically used for assessing avoidance algorithms is not a direct indication of collision risk for sUAS operations at low altitudes. This thesis uses sNMAC criterion as a direct measure of collision risk. Because Weinert et al. assumed only smaller UAV to smaller UAV encounters, this thesis uses a higher criterion to encompass small UAV to smaller general aviation manned aircraft

encounters. This metric will be further expanded upon and different sNMAC criteria are tested in Section 4.2.2.

3.3.3 Traffic Density

One major factor to be considered in the safety of operations that is not captured in risk ratios is the traffic density for a particular area. One means for evaluating air risk (i.e., risk of an air-to-air collision) is by assigning the area of operation an air-risk class (ARC), as outlined in the JARUS SORA 2.0 [8]. This approach is sound, though it results in a more qualitative assessment of risk. To produce a quantitative level of safety estimate, this research utilizes traffic density metrics in the form of number of aircraft per unit of volume per unit of time.

Unfortunately, traffic density data is not readily available in most areas of interest for small unmanned flight operations. Though data is available for higher altitude operations using radar sources, it is harder to obtain manned flight data at low flight altitudes, especially for operations such as crop dusting and helicopter flights. One set of low altitude data utilized to inform MIT LL Bayesian Network encounter models in [55] is the helicopter air ambulance (HAA) data referenced in Section 2.3 from Lester and Weinert's paper [32]. This helicopter data was unique in the fact that it was not obtained from radar data, but instead from flight operational quality assurance (FOQA) data. MIT LL obtained the data from a Massachusetts-based FAA operator, and is restricted from sharing more than the correlated HAA tracks with other air surveillance sources, per a nondisclosure agreement [56]. The main drawback of this data is that density models and encounter rate estimates could not be generated because the FOQA data only reports information on a single HAA flight in time. Though this data is not perfect for modeling low-level manned flight operations, it could

be useful in obtaining qualitative comparisons between low level unmanned operations and manned HAA flight paths. Unfortunately this data could not be leveraged for the intruder encounter models for this research, as it had not been tested within the encounter generation tool by MIT LL.

This thesis proposes using varying traffic density metrics to analyze how the level of safety is impacted as the density and frequency of manned traffic data increases. This thesis will also establish a method to analyze avoidance algorithms, should the required traffic density data become readily available for specific operation areas. Traffic density analysis is further explored in Section 5.1.

3.4 Vertical Proximity Shielding Analysis

The vertical proximity analysis avoidance algorithm consists of a descent only maneuver. Once it is determined that well clear will be violated, via the well clear volume check, the algorithm begins a descent down to 20 feet AGL via the `maximum descent` function described in Section 3.2.1. For simplicity, it is assumed that the terrain is completely flat, though the same method could be utilized on terrain that is not level with an on-board DTED as described in Section 2.2.3. To isolate horizontal and vertical proximity definitions, the ownship will only perform a descent for avoidance and will not attempt a horizontal maneuver, as illustrated in Figure 3.2. The three manned aircraft in the figure represent four straight-line trajectories of differing altitudes.

Additionally, the vertical limitation is smaller for both well clear and NMAC, so a vertical maneuver will generally be more effective than a horizontal maneuver, unless both the intruder and ownship are within 250 feet of the terrain. Along this vein, there are three possible scenarios to observe for a vertical proximity case, given a well clear threshold of 250

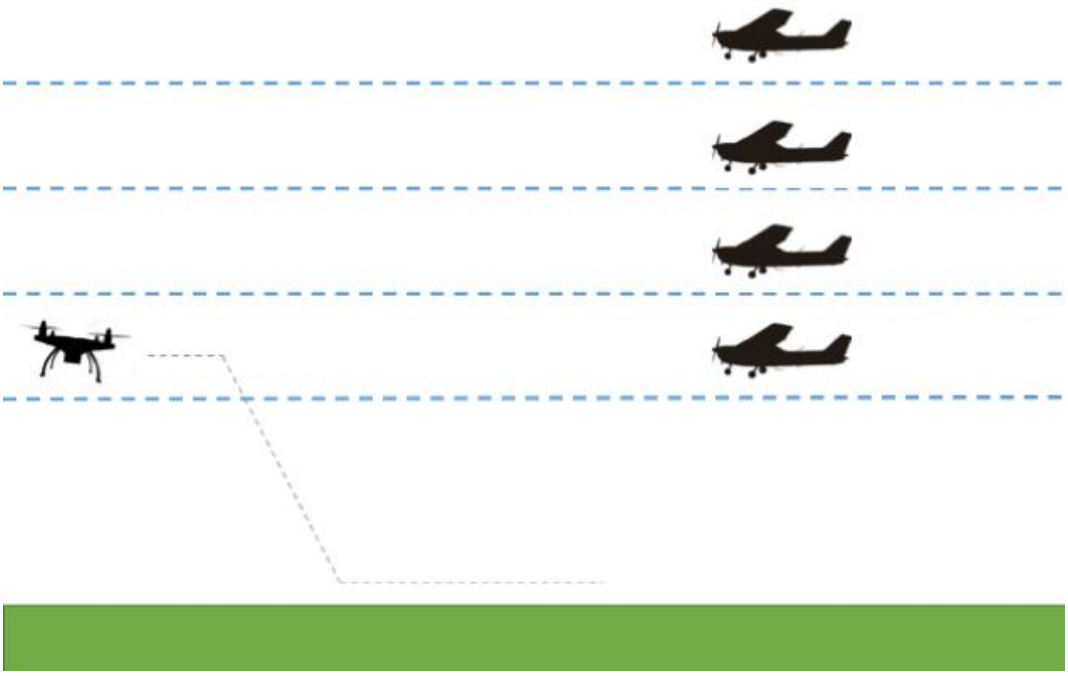


Figure 3.2: Vertical Descend Only Maneuver (Ownship continues on original horizontal path.)

feet and NMAC vertical threshold of 100 feet, where the ownship descends to 20 feet AGL:

1. Intruder is initialized at or above 270 feet:
    - This could allow the ownship to descend to 20 feet and avoid both NMAC and well clear violation
    - This should always result in a risk ratio of 0, but this is highly dependent on aircraft descent performance, sensor range (assumed to be sufficient for all cases in this work), and the algorithm that decides when to initiate an avoidance maneuver.
2. Intruder is initialized between 120-270 feet:
    - The ownship can descend to avoid NMAC, but a well clear violation can not be avoided

3. Intruder is initialized below 120 feet:

    - It is impossible to avoid NMAC (and well clear violation)
    - Risk ratios will always be 1

To verify the avoidance algorithm and approach, randomized "wagon wheel" encounters were performed. The "wagon wheel" approach randomizes the location of the ownship and intruder, initializing them at least 2,000 feet apart, so as to generate a near mid-air collision or well clear violation at the center of the wagon wheel. This method results in a wide variety of different encounter geometries. Based upon the altitude of the intruder, the final descent altitude of the ownship, the horizontal and vertical thresholds that will trigger ownship avoidance, and the descent rate of the ownship, which was set to approximately 10 feet/second, one can verify the outcome (i.e., if there will be a mitigated well clear violation or not) by a simple math problem. Once these predictable encounters were completed and the avoidance algorithm was verified, more complex encounters were used to analyze the descend-only algorithm.

The ownship trajectories used for the vertical proximity shielding analysis case were simple straight line paths and the trajectories used for the intruder were generated using the Bayesian network models from the MIT LL airspace encounter models repository, as described in Section 3.1. Five example encounters that were generated using the MIT LL encounter generator and used in simulation are shown in Figure 3.3. The dashed lines represent intruder trajectories and the solid lines represent the ownship trajectory.

The objective was to generate risk ratios, a metric that will be further discussed in Section 3.3.1, for each altitude bin as shown in Table 3.2.

Each empty cell in the table will be filled with a risk ratio value generated from 1000 encounters of trajectories with the corresponding altitude in the table. An individual table

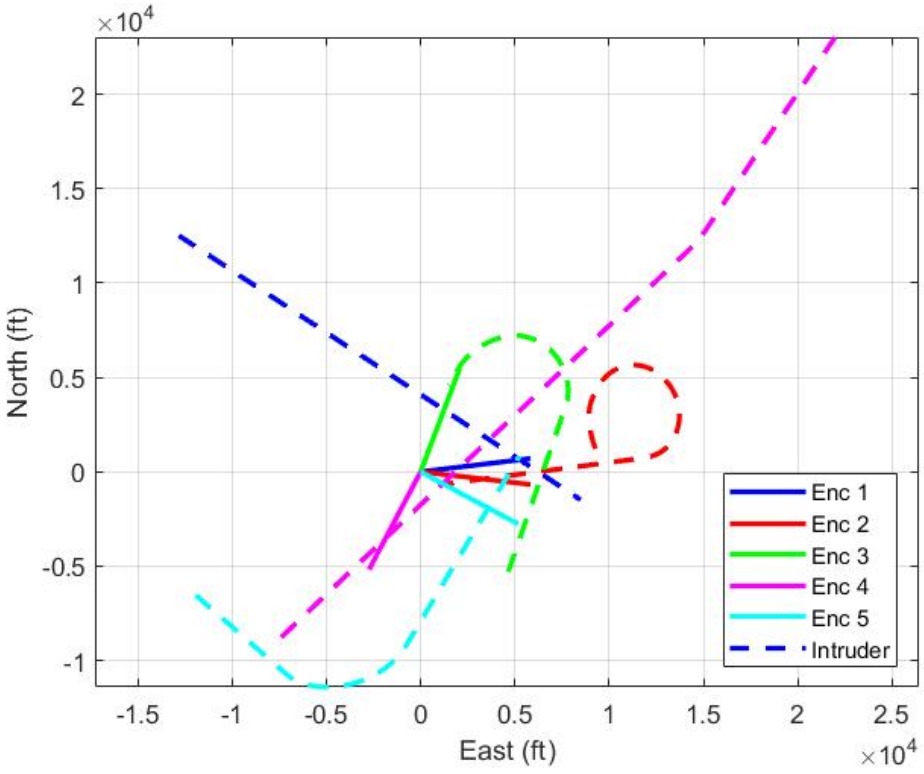


Figure 3.3: Sample Vertical Proximity Encounters, Intruder Trajectories are Dashed Lines, Ownship Trajectories are Solid Lines

	Intruder Altitude (Feet)				
		100-200	200-300	300-400	400-500
Ownship Initial Altitude (Feet)	50-100				
	100-150				
	150-200				
	200-250				
	250-300				
	300-350				
	350-400				

Table 3.2: Example Risk Ratio Output Table – Vertical Proximity

is generated for well clear, NMAC, and sNMAC for each the rotorcraft and fixed wing scenarios.

3.5 Horizontal Proximity Shielding Analysis

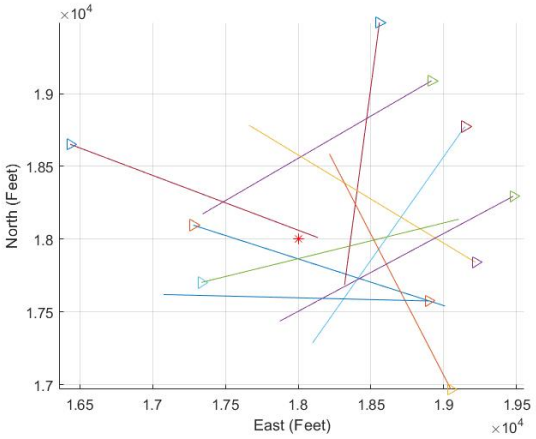
Similar to the vertical proximity shielding analysis avoidance algorithm, the horizontal proximity shielding avoidance algorithm is initiated when it is determined that an intruder aircraft will violate the well clear volume via the `WCV` function. The algorithm then accesses an obstacle database. The current database used is a list of fictitious obstacle data that is formatted based upon the FAA's Digital Obstacle File Database [12], as discussed in Section 2.2.3. The algorithm surveys the obstacle data and will retrieve options within 500 feet of the current ownship location. If multiple obstacles are returned, another function will down-select to the closest obstacle.

If there is no obstacle close enough for shielding, the aircraft will simply stop and descend in place, but these avoidance maneuvers are omitted from the final analysis as they do not represent the effectiveness of shielded avoidance. If an obstacle is determined suitable for shielding, the algorithm calculates the bearing between the current location and the obstacle and executes a maneuver toward the obstacle via the `fly to point` function outlined in Section 3.2.1.

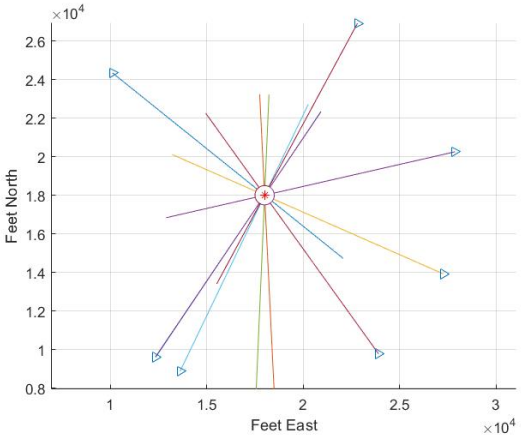
Figure 3.4 illustrates encounters used for horizontal proximity analysis. Figure 3.4a is a plot of example ownship trajectories, and Figure 3.4b is a plot of example intruder trajectories. The triangles on both sub-figures represent the starting location of the trajectories. To generate the encounters seen in Figures 3.4c and 3.4d, first the intruder trajectory is generated. A straight line trajectory is initialized, and the section of the trajectory that is within a certain distance from the obstacle is modified, deviating from the straight path to follow a circular arc that maintains the given distance from the obstacle. The ownship trajectory is then generated based upon the location of the intruder trajectory at the unmitigated closest point of approach (CPA). The unmitigated CPA is arranged to occur at 120 seconds. An

ownship straight line trajectory is then generated to have the same location as the intruder at 120 seconds. A copy of the code used to generate both intruder and ownship trajectories is provided in the Appendix.

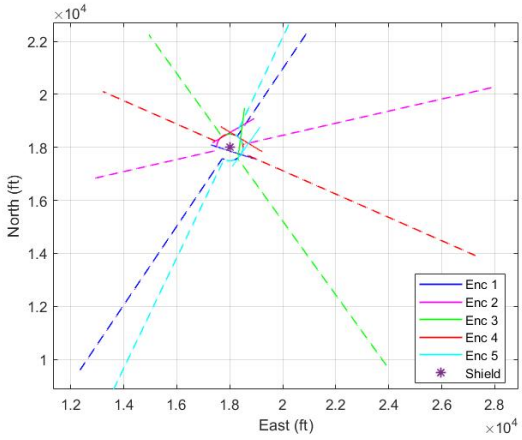
For the purposes of testing the avoidance and generating representative risk ratios, the fictional obstacle database contains only one obstacle and the trajectories are generated to always be within 500 feet of it when avoidance is triggered. Figure 3.4 illustrates the unmitigated trajectories for both the intruder and ownship.



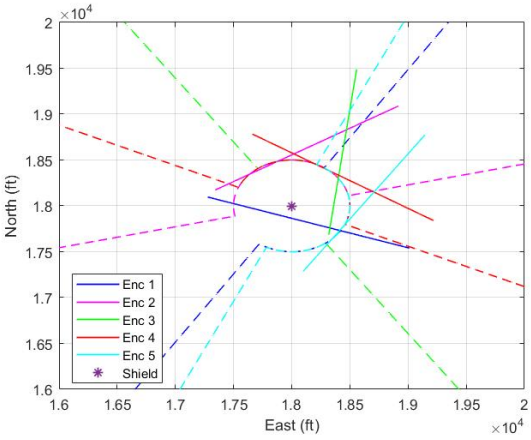
(a) Ownship Example Trajectories



(b) Intruder Example Trajectories



(c) Sample Horiz. Proximity Encounters



(d) Cropped Horiz. Proximity Encounters

Figure 3.4: Horizontal Proximity Unmitigated Example Encounters

In Figure 3.4b, the intruder trajectories are initialized as straight line trajectories until coming within 500 feet of the obstacle. The 500 foot distance was initially decided upon because of the FAA 14 CFR §91.119 minimum safe altitude requirement over "other than congested areas." The regulation states that aircraft must not operate closer than 500 feet to any person, vessel, vehicle, or structure. In conjested areas, or "over any open air assembly of persons," manned aircraft must stay 1000 feet above the highest obstacle within a radius of 2000 feet. The exceptions to this regulation are for helicopters, powered parachutes, and weight-shift-control aircraft.

Helicopters are a major consideration for avoidance at low altitudes and are legally allowed to get closer than 500 feet to obstacles. In 14 CFR §135.203, which applies to commuter and on-demand operations (where pilots are typically required to hold an Air Carrier Certificate), helicopters are granted authorization to operate down to 300 feet above the surface over congested areas under visual flight rules. Under 14 CFR §135.615, prior to flight helicopter air ambulances are required to identify the highest obstacle along a planned route and determine the minimum ceiling required to remain 300 feet above obstacles and terrain under visual flight rules during the day. Because helicopters are a major factor in evaluating risk at low altitudes, intruder trajectories were generated and analysis was performed using both the 500 foot and 300 foot requirement away from obstacles.

One consideration when using an obstacle database is the accuracy of the data within the database. The shielding avoidance algorithm has the ability to eliminate an obstacle as a shielding option based upon accuracy parameters, though accuracy limitations are out of scope for the current analysis of the shielding algorithm. Because this research is a proof of concept, the issue of obstacle avoidance once within the shield and accuracy requirements of the database is not considered. In practice, the ownship would need to determine its position relative to obstacles with sufficient accuracy to select the most effective shield and

to maintain safe flight while approaching it. This would likely require the use of onboard sensors, as discussed in Section 2.2.3.

The goal of the horizontal proximity simulations was to generate risk ratios for each shielding volume value. This idea is illustrated in Table 3.3.

Set Minimum Shielding Volume (Feet)	**NMAC RR**	**sNMAC RR**	**Average Distance From Shield**	**Maximum Distance From Shield**	**Minimum Distance From Shield**
500					
450					
400					
300					
300					
250					
200					
150					
100					
50					

Table 3.3: Example Risk Ratio Output Table – Horizontal Proximity

Chapter 4

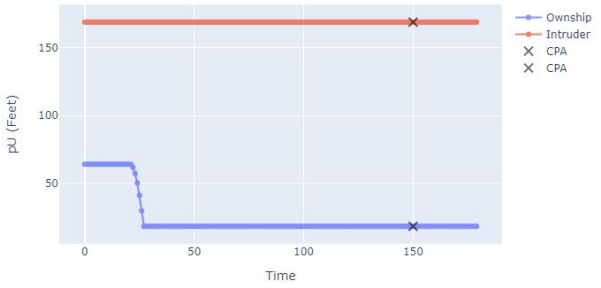
Results and Discussion

4.1 Vertical Proximity Results

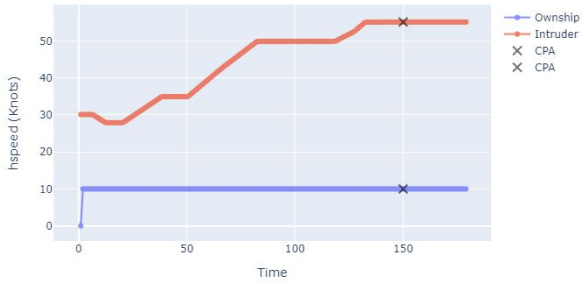
4.1.1 Single Encounter Simulation Results

Figure 4.1 shows an example of the result of running one encounter for the vertical proximity analysis. Figure 4.1a shows the altitude of each aircraft for the duration of the encounter. Note that though this encounter shows the intruder having a constant altitude throughout the entire encounter, not all intruder trajectories have that same property. Figure 4.1b shows the velocity of both aircraft throughout the entire encounter, and Figure 4.1c provides a top view of the position of both the intruder and ownship during a single encounter. Finally, Figure 4.1d is a screen capture of the mitigated and unmitigated distance at the closest point of approaches (DCPAs).

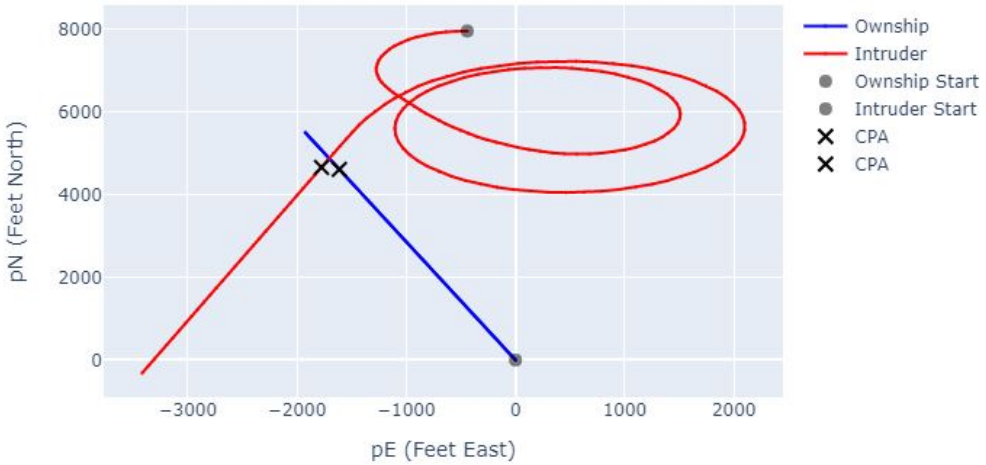
These plots are used to verify that the simulation is performing as expected. In Figure 4.1d, it is clear that there is no horizontal maneuvering, but the mitigated vertical DCPA is



(a) Intruder and Ownship Altitude vs Time



(b) Intruder and Ownship Horizontal Speed vs Time



(c) Intruder and Ownship Position

Unmitigated Horizontal DCPA (Feet)	171.23
Mitigated Horizontal DCPA (Feet)	171.23
Unmitigated Vertical DCPA (Feet)	105.16
Mitigated DCPA (Feet)	151.09
Mitigated Slant DCPA (Feet)	228.47

(d) Vertical Proximity One Simulation Results

Figure 4.1: Vertical Proximity Single Encounter Simulation Results

improved by the descend-only avoidance maneuvering, though in this case, the unmitigated and mitigated encounter would have both resulted in a well clear violation.

4.1.2 Multiple Encounter Simulation Results

As described in Section 3.4, 1000 simulations were run for each altitude bin for the vertical proximity case. The Tables 4.1, 4.2, 4.3, 4.4, 4.5, and 4.6 represent the resulting well clear, NMAC, and sNMAC risk ratios. A value of $1.\overline{0}$ indicates that all mitigated encounters resulted in a violation, and a value of $0.\overline{0}$ indicates that no mitigated encounters resulted in a violation (i.e., perfect avoidance). Empty bins represent encounters in which not enough trajectories could be lined up to create well clear or NMAC violations. That is, the large initial altitude separation already mitigates the chance of collision and 1000 encounters could not be generated for simulation. Note that each risk ratio in the tables below don't encompass all 1000 encounters; out of all 1000 encounters, about half will be classified as "well clear" encounters and the other half will be "NMAC" encounters. A given encounter cannot be both a well clear and NMAC encounter.

Rotorcraft Well Clear Risk Ratios					
	Intruder Altitude (Feet)				
		100-200	**200-300**	**300-400**	**400-500**
Ownship Initial Altitude (Feet)	**50-100**	$1.\overline{0}$	0.71	0.08	-
	100-150	$1.\overline{0}$	0.66	0.08	-
	150-200	$1.\overline{0}$	0.71	0.05	0.09
	200-250	$1.\overline{0}$	0.70	0.06	0.07
	250-300	$1.\overline{0}$	0.72	0.09	0.06
	300-350	$1.\overline{0}$	0.76	0.17	0.05
	350-400	$1.\overline{0}$	0.76	0.27	0.08

Table 4.1: Well Clear Risk Ratios for Differing Ownship and Intruder Altitudes Using a Rotorcraft Intruder Model

There is not much difference between the fixed wing and rotorcraft results. It is clear that when the intruder aircraft is above 300 feet, the well clear risk ratio decreases significantly. Furthermore, the NMAC metric never violates the requirements for either cooperative or

Rotorcraft NMAC Risk Ratios					
	Intruder Altitude (Feet)				
		100-200	**200-300**	**300-400**	**400-500**
Ownship Initial Altitude (Feet)	**50-100**	0.12	$0.\overline{0}$	$0.\overline{0}$	-
	100-150	0.12	$0.\overline{0}$	$0.\overline{0}$	-
	150-200	0.09	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$
	200-250	0.07	0.00	$0.\overline{0}$	$0.\overline{0}$
	250-300	0.11	0.01	$0.\overline{0}$	$0.\overline{0}$
	300-350	0.13	0.02	$0.\overline{0}$	$0.\overline{0}$
	350-400	0.16	0.04	0.01	$0.\overline{0}$

Table 4.2: NMAC Risk Ratios for Differing Ownship and Intruder Altitudes Using a Rotorcraft Intruder Model

Rotorcraft sNMAC Risk Ratios					
	Intruder Altitude (Feet)				
		100-200	**200-300**	**300-400**	**400-500**
Ownship Initial Altitude (Feet)	**50-100**	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	-
	100-150	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	-
	150-200	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$
	200-250	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$
	250-300	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$
	300-350	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$
	350-400	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$

Table 4.3: sNMAC Risk Ratios for Differing Ownship and Intruder Altitudes Using a Rotorcraft Intruder Model

non-cooperative aircraft outlined in Table 3.1, though the well clear risk ratio is almost always violated when the intruder aircraft is below 300 feet. This is to be expected; the ownship would have to land on the ground to not violate well clear for an intruder below 250 feet, as that is the vertical threshold.

Referring to Tables 4.2 and 4.5, for both the fixed wing and rotorcraft NMAC risk ratios, the

Fixed Wing Well Clear Risk Ratios					
	Intruder Altitude (Feet)				
		100-200	**200-300**	**300-400**	**400-500**
Ownship Initial Altitude (Feet)	**50-100**	$1.\overline{0}$	0.75	-	-
	100-150	$1.\overline{0}$	0.70	0.26	-
	150-200	1.00	0.69	0.21	-
	200-250	1.00	0.63	0.18	0.28
	250-300	$1.\overline{0}$	0.67	0.18	0.23
	300-350	1.00	0.75	0.22	0.17
	350-400	$1.\overline{0}$	0.76	0.29	0.19

Table 4.4: Well Clear Risk Ratios for Differing Ownship and Intruder Altitudes Using a Fixed Wing Intruder Model

Fixed Wing NMAC Risk Ratios					
	Intruder Altitude (Feet)				
		100-200	**200-300**	**300-400**	**400-500**
Ownship Initial Altitude (Feet)	**50-100**	0.09	$0.\overline{0}$	-	-
	100-150	0.06	0.00	$0.\overline{0}$	-
	150-200	0.07	$0.\overline{0}$	$0.\overline{0}$	-
	200-250	0.05	0.00	$0.\overline{0}$	$0.\overline{0}$
	250-300	0.10	0.01	$0.\overline{0}$	$0.\overline{0}$
	300-350	0.10	0.01	$0.\overline{0}$	$0.\overline{0}$
	350-400	0.10	0.04	0.01	$0.\overline{0}$

Table 4.5: NMAC Risk Ratios for Differing Ownship and Intruder Altitudes Using a Fixed Wing Intruder Model

100-200 intruder altitude bin has the most variation in the risk ratios between each ownship bin. For example, for the rotorcraft NMAC risk ratios in Table 4.2, the risk ratios decrease for the first four ownship altitude bins, and then begin to increase again. The relatively larger variation in these risk ratios is likely due to the fact that the intruder is generally transiting through the 100-200 foot altitude level, and is rarely maintaining a constant altitude. Rapid descents into the 100-200 foot altitude level would likely result in higher risk ratios, as the

Fixed Wing sNMAC Risk Ratios					
	Intruder Altitude (Feet)				
		100-200	**200-300**	**300-400**	**400-500**
Ownship Initial Altitude (Feet)	**50-100**	$0.\overline{0}$	$0.\overline{0}$	-	-
	100-150	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	-
	150-200	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	-
	200-250	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$
	250-300	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$
	300-350	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$
	350-400	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$

Table 4.6: sNMAC Risk Ratios for Differing Ownship and Intruder Altitudes Using a Fixed Wing Intruder Model

ownship would have less time to maneuver out of the way once the well clear volume threshold is violated.

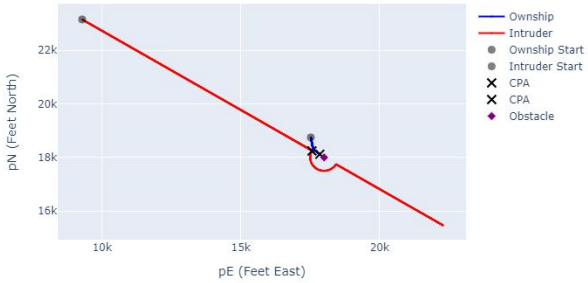
Overall, these results are not surprising and follow the predicted results outlined in Section 3.4. The descend-only maneuver satisfies the NMAC metric in all cases, and is a viable candidate maneuver for avoidance. One drawback of this method is that descending down to 20 feet may not always be an option, especially in urban areas.

4.2 Horizontal Proximity Results

4.2.1 Single Encounter Simulation Results

Figure 4.2 shows an example of one simulation for the horizontal proximity analysis. Figures 4.2a and 4.2c depict the mitigated trajectories of each aircraft, as well as the location of the shield. The two black X markers depict the mitigated CPA locations and represent the same time step within the simulation. The ownship is initialized on a straight line trajectory, and deviates from that trajectory when the well clear threshold is violated (i.e., since the aircraft are co-altitude, the ownship begins its avoidance maneuver when the two aircraft navigate within 6000 feet of each other). Once avoidance is triggered, the ownship immediately begins a maneuver toward the shield, as seen in Figure 4.2c.

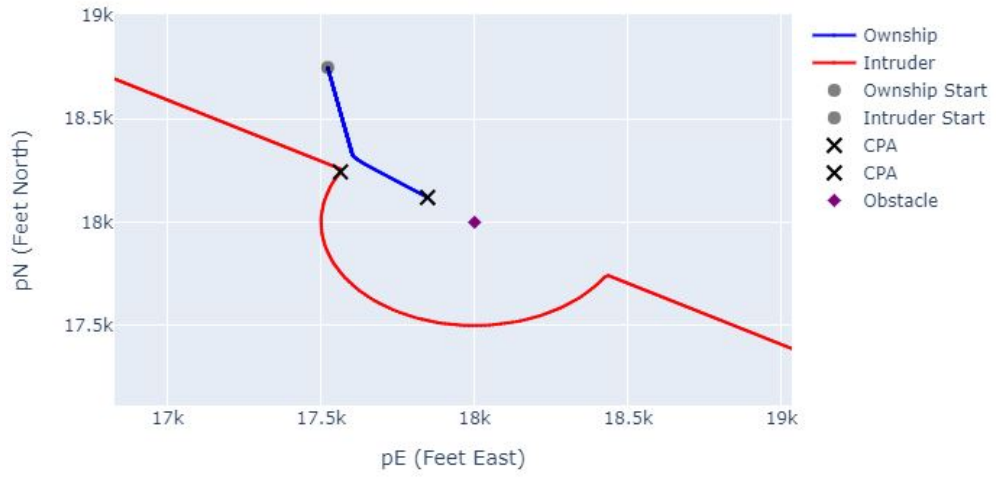
As stated in Section 4.1.1, these plots are used to verify that the simulation is performing as expected, and do not require further analysis. Again note the unmitigated horizontal DCPA improves, but still does not satisfy the NMAC requirement of 500 feet, which is understandable because the intruder is only 500 feet from the obstacle and, in this case, the ownship shielding volume was set to 200 feet.



(a) Horiz. Shielding Intruder and Ownship Position

Unmitigated Horizontal DCPA (Feet)	0
Mitigated Horizontal DCPA (Feet)	309.52
Unmitigated Vertical DCPA (Feet)	0
Mitigated DCPA (Feet)	0
Mitigated Slant DCPA (Feet)	309.52

(b) Horiz. Shielding One Encounter Results



(c) Horiz. Shielding Cropped Intruder and Ownship Positions

Figure 4.2: Horizontal Shielding Single Encounter Simulation Results

4.2.2 Multiple Encounter Simulation Results

For each given shielding proximity, 1000 encounters were simulated to generate risk ratios. The shielding proximity varied from 500 feet to 50 feet in 50 foot increments, as described in Section 3.5. For all encounters, both the intruder and ownship trajectories are initialized at an altitude of 200 feet. The intruder is set at a constant airspeed of 50 knots and the ownship is set to 10 knots. These airspeeds are generally lower than the nominal encounters from Section 4.1, but are meant to be more representative of velocities of low altitude operations around obstacles and terrain.

Tables 4.7 and 4.8 display the results of decreasing the shielding volume metric, which is the value used in the `fly to point` function outlined in Section 3.2.1. These simulations were run for both the 500 and 300 foot intruder radius from the obstacle.

500 Foot Intruder Radius					
Set Shielding Volume (Feet)	**NMAC RR**	**sNMAC RR (50 foot metric)**	**Average Distance From Shield**	**Maximum Distance From Shield**	**Minimum Distance From Shield**
500	1.00	0.75	454.37	499.55	206.97
450	1.00	0.07	417.11	449.99	207.46
400	1.00	$0.\overline{0}$	376.94	399.97	207.46
300	1.00	$0.\overline{0}$	335.24	349.99	207.46
300	1.00	$0.\overline{0}$	291.85	299.99	207.46
250	1.00	$0.\overline{0}$	246.36	249.99	207.46
200	1.00	$0.\overline{0}$	198.38	199.99	196.72
150	1.00	$0.\overline{0}$	148.35	149.99	146.72
100	$1.\overline{0}$	$0.\overline{0}$	98.38	99.99	96.72
50	1.00	$0.\overline{0}$	48.34	49.99	46.72

Table 4.7: Risk Ratios for Differing Shielding Volumes for an Intruder Radius of 500 feet

It is clear that as the value for the shielding volume decreases, the traditional NMAC risk

300 Foot Intruder Radius					
Set Shielding Volume (Feet)	**NMAC RR**	**sNMAC RR (50 foot metric)**	**Average Distance From Shield**	**Maximum Distance From Shield**	**Minimum Distance From Shield**
300 feet	$1.\overline{0}$	0.70	298.40	299.99	296.72
250 feet	$1.\overline{0}$	0.12	248.30	249.99	246.72
200 feet	$1.\overline{0}$	$0.\overline{0}$	198.36	199.99	196.72
150 feet	$1.\overline{0}$	$0.\overline{0}$	148.41	149.99	146.72
100 feet	$1.\overline{0}$	$0.\overline{0}$	98.32	99.99	96.72
50 feet	$1.\overline{0}$	$0.\overline{0}$	48.33	49.99	46.72

Table 4.8: Risk Ratios for Differing Shielding Volumes for an Intruder Radius of 300 feet

ratio does not get smaller. This makes sense because the intruder is only 500 feet away from the obstacle, which is the horizontal threshold value for NMAC. It is also clear that the sNMAC risk ratio value decreases very quickly as the shielding volume gets smaller. To further explore the impact of the shielding volume on sNMAC, risk ratios for various sNMAC candidate metrics were also calculated and recorded in Tables 4.9 and 4.10.

500 Foot Intruder Radius											
			Different sNMAC Metrics (Feet)								
		NMAC RR	**sNMAC 50**	**sNMAC 100**	**sNMAC 150**	**sNMAC 200**	**sNMAC 250**	**sNMAC 300**	**sNMAC 350**	**sNMAC 400**	**sNMAC 450**
Set Shielding Volume (Feet)	**500**	0.98	0.75	0.79	0.82	0.84	0.87	0.89	0.91	0.93	0.95
	450	0.98	0.07	0.79	0.82	0.84	0.87	0.89	0.91	0.93	0.95
	400	0.98	$0.\overline{0}$	0.05	0.81	0.85	0.87	0.89	0.91	0.93	0.95
	350	0.98	$0.\overline{0}$	$0.\overline{0}$	0.05	0.83	0.72	0.89	0.91	0.93	0.95
	300	0.98	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	0.05	0.86	0.89	0.91	0.93	0.95
	250	0.98	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	0.06	0.88	0.91	0.93	1.00
	200	0.98	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	0.06	0.90	0.92	0.96
	150	0.98	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	0.06	0.92	0.96
	100	$1.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	0.03	1.00
	50	1.00	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$

Table 4.9: Risk Ratios for Differing Shielding Volumes and sNMAC metrics for an Intruder Radius of 500 feet

As discussed in Section 3.3.2, sNMAC is directly related to the size of the two aircraft in an encounter. The results from the 300 foot intruder radius with varying sNMAC volumes suggest that if an operator is using a shielding volume of 50 feet, the combined width

300 Foot Intruder Radius											
			Different sNMAC Metrics (Feet)								
		NMAC RR	sNMAC 50	sNMAC 100	sNMAC 150	sNMAC 200	sNMAC 250	sNMAC 300	sNMAC 350	sNMAC 400	sNMAC 450
Set Shielding Volume (Feet)	**300**	$1.\overline{0}$	0.74	0.76	0.80	0.85	0.88	0.91	0.94	0.98	$1.\overline{0}$
	250	$1.\overline{0}$	0.12	0.78	0.80	0.85	0.89	0.92	0.96	$1.\overline{0}$	$1.\overline{0}$
	200	$1.\overline{0}$	$0.\overline{0}$	0.10	0.80	0.85	0.90	0.94	0.99	$1.\overline{0}$	$1.\overline{0}$
	150	$1.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	0.11	0.84	0.91	0.95	$1.\overline{0}$	$1.\overline{0}$	$1.\overline{0}$
	100	$1.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	0.14	0.89	0.97	$1.\overline{0}$	$1.\overline{0}$	$1.\overline{0}$
	50	$1.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	$0.\overline{0}$	0.20	1.00	$1.\overline{0}$	$1.\overline{0}$	$1.\overline{0}$

Table 4.10: Risk Ratios for Differing Shielding Volumes and sNMAC metrics for an Intruder Radius of 300 feet

(wingspan) and height of the sUAS ownship and manned intruder could be up to 200 feet while still achieving a 0 sNMAC risk ratio.

Chapter 5

Analysis

5.1 Analysis with Traffic Density

Though risk ratios can be a good indication of how effective an avoidance algorithm is, they don't always give a holistic view on how save an entire operation is. Risk ratios can be used in conjunction with traffic density data to further strengthen a safety case. Though low-level traffic density metrics can be costly to obtain, if a small UAV operator can quantitatively assess the historical number of manned aircraft in their operational area, those metrics can be used in conjunction with risk ratios obtained from analyzing an avoidance algorithm to significantly strengthen a safety case.

For example, if data is available such as in Figure 5.1, then further assessment can be made using risk ratios obtained in Section 4.1.2. Figure 5.1 represents an example of traffic density data for the operational volume of the UAV. It is not real data, but is representative of how traffic density data could be collected for the application of strengthening a safety case. The applicable metrics that would accompany this type of data would be a unit of time in which

the data was collected and the volume of the area in which it was surveyed.

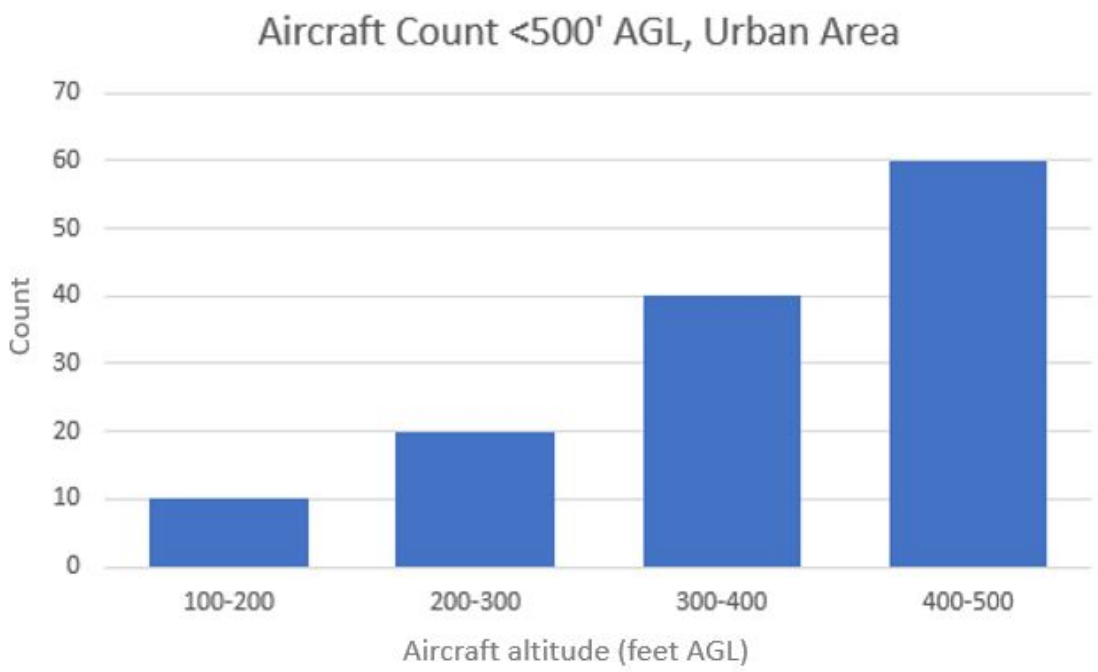


Figure 5.1: Sample Aircraft Traffic Density Chart

Using the NMAC risk ratios from Table 4.2, equivalent level of safety metrics can be obtained in terms of probability of NMAC per unit of time. If the cruise component of the ownship flight plan is set between an operational altitude of 150-300 feet, Table 5.1 displays the equivalent level of safety metrics for the volume surveyed in Figure 5.1, which should cover the entire operation area for the UAV mission.

From this data, a user can select the operation altitude that will mitigate risk. For example, from the data in Table 5.1, it would make most sense for the operator to choose an operating altitude of 200-250 feet to result in a mitigated risk of 0.476 NMACs per the amount of time the data was collected in from Figure 5.1, though it should be noted that the intruder models used to obtain the original risk ratios were not location-specific and may not be

Rotorcraft Probability of NMAC per Unit of Time					
	Intruder Altitude (Feet)				
		100-200	**200-300**	**300-400**	**400-500**
Ownship Altitude (Feet)	**150-200**	0.74	$0.\overline{0}$	$0.\overline{0}$	-
	200-250	0.48	0.04	$0.\overline{0}$	$0.\overline{0}$
	250-300	0.99	0.27	$0.\overline{0}$	$0.\overline{0}$

Table 5.1: Using traffic density metrics example

representative of intruder geometry in all airspaces. Therefore if the traffic density data that was collected in Figure 5.1 was taken over the span of 6 months, then the probability of an NMAC using the descend-only avoidance for the specific area surveyed would be less than 1 per year.

Furthermore, traffic density data can be used in conjunction with the sNMAC metric to more closely obtain a more direct indication of collision risk. Using Figure 5.1 and Table 4.10, one can assess the risk of collision by selecting an appropriate sNMAC criterion, based upon the size of the UAV platform and expected manned traffic in the area, and the shielding volume they are able to maintain in the precense of obstacles. For example, if the ownship wingspan is 20 feet, then an sNMAC of at least 100 feet should be used and as long as the ownship platform is able to maintain 150 foot proximity to obstacles, then the resulting sNMAC risk ratio will be 0.0 at any operating altitude. If the ownship can only maintain 200 foot proximity to obstacles, then the resulting risk ratio is 0.1, and the risk of collision would vary depending on the operating altitude. For an operating altitude of 100-200 feet, according to the traffic density values in Figure 5.1, the collision rate would be 2 per year (for a collection timeline of 6 months), which is very high. Therefore, the ownship would need to be able to maintain a closer proximity than 200 feet to the obstacle to be able to obtain an equivalent level of safety.

5.2 Use Case Example

The methods used for safety analysis could be used to support a safety case for a particular application. This section will outline an example safety case in which the shielding avoidance method could be used to demonstrate an equivalent level of safety.

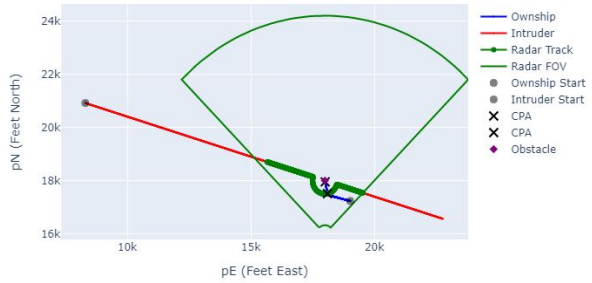
5.2.1 Package delivery in urban areas

Consider the following hypothetical use case:

A company wants to use a shielding avoidance algorithm to obtain an equivalent level of safety with reduced well clear volumes over an urban area. Though the FAA requirements for manned aircraft are to remain 1000 feet above the highest obstacle within 2000 feet for the entire package delivery flight operations area, the company has decided to be conservative and assume manned aircraft will maintain a radius of 500 feet from all obstacles. The company's detection system consists of an ADS-B receiver and radar along all flight paths. Their platform, which has a wingspan of less than 5 feet, has hovering capabilities and has the ability to sense obstacles up to 50 feet away and avoid them.

Based upon the criteria above, the company determines that it can maintain a shielding volume of 50 feet. Using their shielding avoidance algorithm, the FAA digital obstacle file database [12], an accurate terrain database for their flight area, and a sensor model of their radar developed through flight testing, the company can simulate sNMAC risk ratios for their small UAV. Because their flight operation is set to be in an urban area, the intruder is simulated to maintain 500 feet from the shield.

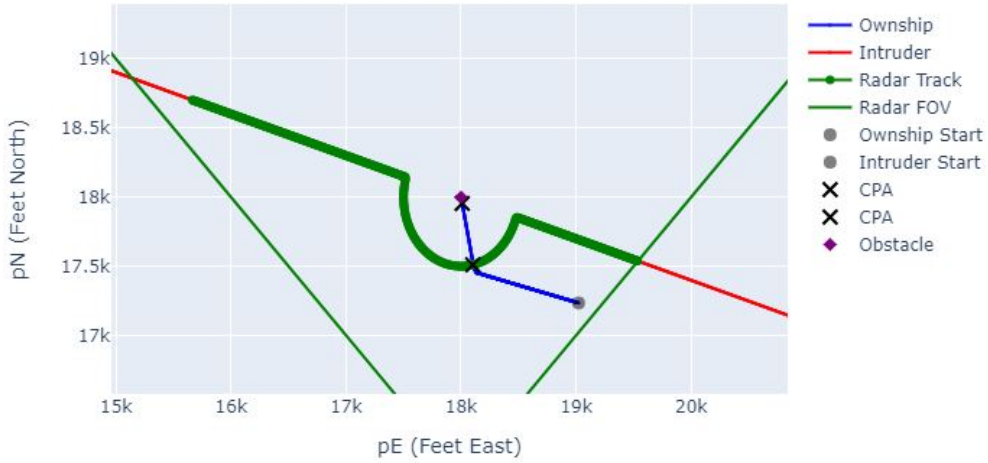
To create an even stronger safety case, the company can obtain low level flight radar data that quantifies the true low altitude manned air traffic in the given region, as discussed in



(a) Full view of encounter

Unmitigated Horizontal DCPA (Feet)	0
Mitigated Horizontal DCPA (Feet)	452.94
Unmitigated Vertical DCPA (Feet)	0
Mitigated DCPA (Feet)	0
Mitigated Slant DCPA (Feet)	452.94

(b) Results with use of radar model



(c) Cropped view of encounter with radar model

Figure 5.2: Single Encounter Simulation Results with Use of Radar Model

Section 5.1, though this data is not readily available and can be costly to obtain.

Figure 5.2 depicts one simulation using a sensor model that was developed during flight testing. As clearly viewed in Figure 5.2c, the track for the intruder is not immediately formed when the intruder enters the field of view on the left side of the sensor volume. The radar model used in simulation will impact how soon a track is formed, which will in turn impact how much time the ownship has to execute an avoidance maneuver.

The resulting risk ratios from running 1000 simulations of this test case are displayed in Table 5.2. An sNMAC metric of 100 feet was used because of the UAS platform's small wingspan. This choice of sNMAC metric is considered conservative, as many general aviation

manned aircraft that would likely be encountered at low altitudes have a wingspan of less than 50 feet. For completeness, the risk ratio values for an sNMAC metric of both 50 and 150 feet are also listed in Table 5.2 as well.

Because the radar model is used for all simulations, the risk ratios represent metrics for non-cooperative aircraft. Though the sNMAC values are not 0.0 as they were for many of the similar simulation results from Section 4.2.2, this is to be expected using an accurate radar sensor model. Though sNMAC does not have required standards like NMAC and well clear, it would follow the trend seen in Table 3.1 that the acceptable sNMAC risk ratio would be slightly higher when evaluating solely non-cooperative aircraft.

sNMAC Metric	**Set Shielding Volume**	**NMAC RR**	**sNMAC RR**	**Average Distance From Shield**	**Maximum Distance From Shield**	**Minimum Distance From Shield**
50 feet	50 feet	$1.\overline{0}$	0.00	48.48	182.35	46.72
100 feet	50 feet	$1.\overline{0}$	0.01	48.48	182.35	46.72
150 feet	50 feet	$1.\overline{0}$	0.01	48.48	182.35	46.72

Table 5.2: sNMAC and Shielding Metrics Using a Radar Sensor Model

The average, maximum, and minimum shielding distance values displayed in the last three columns of Table 5.2 are used in part to verify that the ownship is completing a shielded avoidance maneuver toward the obstacles. In this case, the maximum distance from the shield is much higher than the nominal results seen in Section 4.2.2, most likely because the alerting timeline is delayed due to the radar sensor model. The sensor model was developed based upon flight testing and was designed to more accurately represent the timeline from when the intruder enters the field of view of the radar, and when the radar actually forms a track that would be received by the avoidance system onboard the ownship.

5.3 Comparison to Other Avoidance Algorithms

It is expected that other avoidance algorithms will result in much better risk ratios than the shielded avoidance algorithm. For example, one of the standard options for an avoidance algorithm in a Pixhawk autopilot [3] has been added as an avoidance function within the Python simulation. The code for the avoidance algorithm implemented in the simulator is made available by ArduPilot on GitHub [11]. The avoidance maneuver is a horizontal maneuver in the opposite direction of the intruder. The encounters were simulated using the same 1000 trajectories of the 500 foot intruder radius horizontal proximity case. The NMAC and sNMAC risk ratios end up being perfect (0.0), which is due to the fact that the perfect sensor assumption.

The difference between the results of the Pixhawk avoidance and the shielded avoidance algorithms is that the Pixhawk avoidance calculates a heading that is in the opposite direction of the intruder, even if this heading causes the ownship to deviate off of its mission course. Figure 5.3 clearly depicts this behavior. The encounter depicted is the same encounter trajectories used in Figure 4.2 with the avoidance algorithm switched to a Pixhawk avoidance instead of the shielding avoidance. This deviation from the flight plan can be problematic for missions which have a time or energy constraint. Shielding avoidance will not always achieve the best well clear or NMAC risk ratios, but because the ownship will only search for obstacles or terrain already near its course, it will almost always be the most time and battery efficient solution.

Furthermore, both the vertical and horizontal proximity cases operated under the assumption that only a vertical and horizontal maneuver could be conducted. In reality, the company could further strengthen the avoidance by performing both a vertical and horizontal avoidance maneuver toward an obstacle, though it is difficult to develop representative intruder

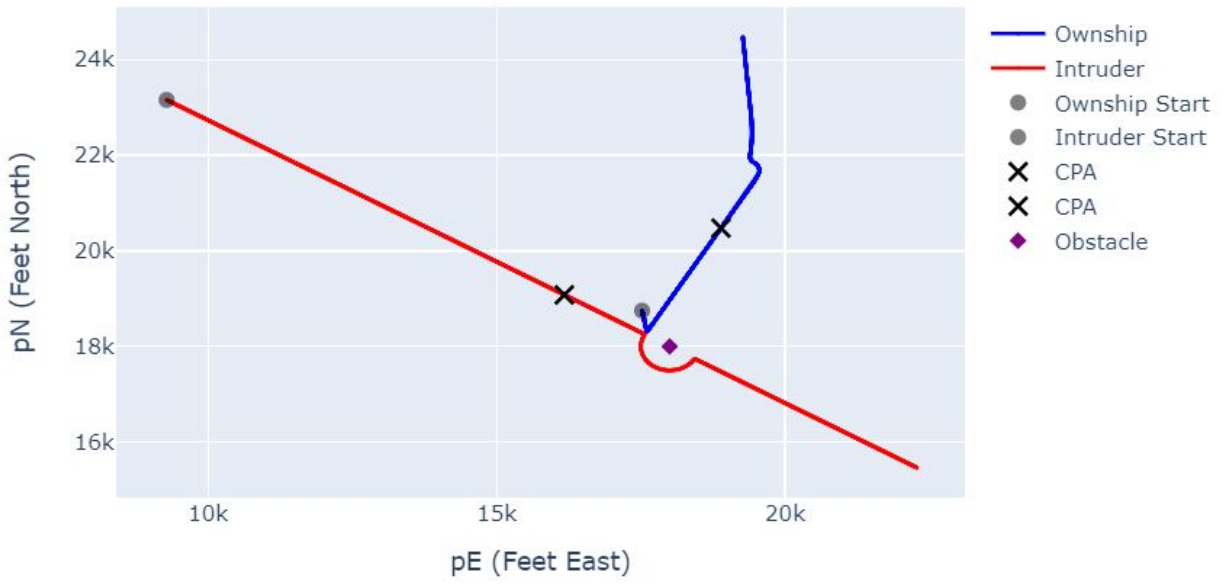


Figure 5.3: Sample Aircraft Traffic Density Chart

trajectories for nominal operations around an obstacle to properly analyze avoidance maneuvers in both the horizontal and vertical planes.

Chapter 6

Conclusions and Future Work

The objective of this research was to explore the use of shielded avoidance and to demonstrate that operations at lower altitudes where shielded avoidance can be used can be proven to be performed safely without satisfying well clear risk ratios requirements. It has been demonstrated that utilizing a horizontal-only maneuver for shielding will generally result in unacceptable well clear and NMAC risk ratios. The use of an sNMAC metric is appropriate for the case of shielded avoidance and by adopting this metric, one can establish a level of safety that is equivalent to FAA requirements for more conventional scenarios.

This study was limited in that it assumed only a horizontal or a vertical maneuver. In reality, the ownship will almost always be able to use a combined maneuver to maximize the separation. However, it is difficult to develop representative encounters for this case. Further research may include a study of representative trajectories of manned aircraft in the proximity of obstacles over urban and rural areas. These trajectories could help further inform the use and acceptance of shielded avoidance maneuvers.

Current DAA standards do not fully address shielded avoidance operations, though it is a

topic being considered for future standards. From the results of this work, it is recommended that an sNMAC value be allowed for evaluating and characterizing risk of a collision in the future. The sNMAC value will likely be correlated with the wingspan and height of the ownship aircraft, as well as the expected manned traffic in the particular airspace.

The use of an sNMAC value, and the corresponding acceptable risk ratio values, needs to be further explored. The method for quantifying an sNMAC volume has historically only been performed for smaller UAS to smaller UAS encounters. This methodology could be further expanded into encounters between small UAS and manned, general aviation aircraft that are flying at low altitudes. This sNMAC value could be tailored for different sizes of smaller UASs, as it is highly dependent upon the combined wingspan and height of the two aircraft in an encounter.

Bibliography

[1] Atom UAV – FLARM for drones. URL https://flarm.com/products/uav/atom-uav-flarm-for-drones/.

[2] The OpenSky Network. URL https://opensky-network.org/about/about-us.

[3] Flight features– ADS-B. URL https://ardupilot.org/copter/docs/common-ads-b-receiver.html.

[4] Skydio - how does Skydio 2 work? URL https://support.skydio.com/hc/en-us/articles/360036116834-How-does-Skydio-2-work-.

[5] Wing - United States — Virginia. URL https://wing.com/united-states/virginia/.

[6] FAA order. Technical Report 8020.11D, Federal Aviation Administration, May 2018. URL https://www.faa.gov/documentLibrary/media/Order/FAA_Order_8020.11D.pdf.

[7] FAA flight standards information management system (FSIMS). Technical Report 8900.1, Federal Aviation Administration, 2019. URL https://fsims.faa.gov/PICDetail.aspx?docId=8900.1,Vol.7,Ch4,Sec1.

[8] JARUS guidelines on specific operations risk assessment (SORA). Technical Report JAR-doc-06, Joint Authorities for Rulemaking of Unmanned Systems, 2019.

[9] Standard specification for detect and avoid system performance requirements. Technical Report F3442/F3442M 20, ASTM, 100 Barr Harbor Drive, PO Box C700, West Conshohocken, PA 19428-2959. United States, July 2020.

[10] ConOps for UAS UTM. Technical report, U.S. Department of Transportation, 03 2020.

[11] ArduPilot - ArduCopter - avoidance. https://github.com/ArduPilot/ardupilot/blob/master/ArduCopter/avoidance_adsb.cpp, 2020.

[12] Federal Aviation Administration. Digital obstacle file (DOF). Available at https://www.faa.gov/air_traffic/flight_info/aeronav/Digital_Products/dof/, 2021.

[13] L. E. Alvarez, I. Jessen, M. P. Owen, J. Silbermann, and P. Wood. ACAS sXu: Robust decentralized detect and avoid for small unmanned aircraft systems. In *2019 IEEE/AIAA 38th Digital Avionics Systems Conference (DASC)*, pages 1–9, 2019. doi: 10.1109/DASC43569.2019.9081631.

[14] M. Bagherian. Unmanned aerial vehicle terrain following/terrain avoidance/threat avoidance trajectory planning using fuzzy logic. *Journal of Intelligent & Fuzzy Systems*, 34:1791–1799, 03 2018. doi: 10.3233/JIFS-161977.

[15] H. Chao, Y. Gu, and M. Napolitano. A survey of optical flow techniques for UAV navigation applications. In *2013 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 710–716, 2013.

[16] C.C. Chen, B. Gill, M.W. Edwards, S. Smearcheck, T. Adami, S. Calhoun, M. G Wu, A.C. Cone, and S. Lee. Defining well clear separation for unmanned aircraft systems operating with non-cooperative aircraft. In *AIAA Aviation 2019 Forum*, 2019.

[17] M. Clark and R. C. Roberts. Autonomous quadrotor terrain-following with a laser rangefinder and gimbal system. In *2017 IEEE SENSORS*, pages 1–3, 2017.

[18] S. P. Cook, D. Brooks, R. Cole, D. Hackenberg, and V. Raska. *Defining Well Clear for Unmanned Aircraft Systems*. 2015. doi: 10.2514/6.2015-0481. URL https://arc.aiaa.org/doi/abs/10.2514/6.2015-0481.

[19] Fintan Corrigan. Mavic pro – highlights you need to know. URL https://store.dji.com/guides/mavic-pro-highlights-need-know/.

[20] M. Dekan, D. František, B. Andrej, R. Jozef, R. Dávid, and M. Josip. Moving obstacles detection based on laser range finder measurements. *International Journal of Advanced Robotic Systems*, 15(1):1729881417748132, 2018. doi: 10.1177/1729881417748132. URL https://doi.org/10.1177/1729881417748132.

[21] D. Dorr. Rotary-wing aircraft terrain-following/terrain-avoidance system development. 1986.

[22] M.W. Edwards and J.K. Mackay. Determining required surveillance performance for unmanned aircraft sense and avoid. In *17th AIAA Aviation Technology, Integration, and Operations Conference*, 2017. URL https://arc.aiaa.org/doi/pdf/10.2514/6.2017-4385.

[23] L.P Espindle, J.D. Griffith, and J.K. Kuchar. Safety analysis of upgrading to TCAS version 7.1 using the 2008 u.s. correlated encounter model. 05 2009.

[24] N. Gellerman, N. Kaabouch, and W. Semke. A terrain avoidance algorithm based on the requirements of terrain awareness and warning systems. In *2015 IEEE Aerospace Conference*, pages 1–6, March 2015. doi: 10.1109/AERO.2015.7119160.

[25] B. Groves. NCDOT and Skydio secure the first true BVLOS waiver under part 107, October 6 2020. URL https://www.suasnews.com/2020/10/ncdot-and-skydio-secure-the-first-true-bvlos-waiver-under-part-107/.

[26] ICAO. Rules of the air. Available at https://www.icao.int/Meetings/anconf12/Document%20Archive/an02_cons%5B1%5D.pdf, 2005.

[27] ICAO. Manual on remotely piloted aircraft systems (RPAS). Available at https://skybrary.aero/bookshelf/books/4053.pdf, 2015.

[28] ICAO. ICAO model UAS regulations. Available at https://www.icao.int/safety/UA/UAID/Documents/Final%20Model%20UAS%20Regulations2%20-%20Parts%20101%20and%20102.pdf, 2020.

[29] M. Kochenderfer, M. Edwards, L. Espindle, J. Kuchar, and J. Griffith. Airspace encounter models for estimating collision risk. *Journal of Guidance Control and Dynamics*, 33:487–499, 03 2010. doi: 10.2514/1.44867.

[30] F. M. Krachmalnick, G. J. Vetsch, and M. J. Wendl. Automatic flight control system for automatic terrain-following. *Journal of Aircraft*, 5(2):168–175, 1968. doi: 10.2514/3.43925. URL https://doi.org/10.2514/3.43925.

[31] T. Krüger, B. Blom, and T. Feuerle. Evaluation of an automatic separation algorithm for unmanned aircraft systems. *Transportation Research Procedia*, 43:309–318, 2019.

[32] E. T. Lester and A. Weinert. Three quantitative means to remain well clear for small UAS in the terminal area. In *2019 Integrated Communications, Navigation and Surveillance Conference (ICNS)*, pages 1–17, 2019.

[33] A. Livshitz and M. Idan. Low-cost laser range-measurement-based terrain-following

concept and error analysis. *Journal of Guidance, Control, and Dynamics*, 41(4):1006–1014, 2018.

[34] E.H. Londner. Collision avoidance system effectiveness on low performance unmanned aircraft. In *AIAA SciTech 2016 Forum*, 2016. URL https://arc.aiaa.org/doi/pdf/10.2514/6.2016-1987.

[35] Y. Lu, Z. Xue, G. Xia, and L. Zhang. A survey on vision-based UAV navigation. *Geospatial Information Science*, 21(1):21–32, 2018. doi: 10.1080/10095020.2017.1420509. URL https://doi.org/10.1080/10095020.2017.1420509.

[36] G. Manfredi and Y. Jestin. An introduction to ACAS Xu and the challenges ahead. pages 1–9, 09 2016. doi: 10.1109/DASC.2016.7778055.

[37] G. Manfredi and Y. Jestin. Are you clear about "well clear"? In *2018 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 599–605, 2018.

[38] M. Marques, A. Brum, S. Antunes, and J. G. Mota. Sense and avoid implementation in a small unmanned aerial vehicle. In *2018 13th APCA International Conference on Automatic Control and Soft Computing (CONTROLO)*, pages 395–400, 2018. doi: 10.1109/CONTROLO.2018.8514548.

[39] M.F. Mclaughin and A.D. Zeitlin. Safety study of TCAS II for logic version 6.04. 07 1992.

[40] M. Mullins, K. Foerster, N. Kaabouch, and W. Semke. Incorporating terrain avoidance into a small UAS sense and avoid system. 06 2012. ISBN 978-1-60086-939-6. doi: 10.2514/6.2012-2504.

[41] C. Muñoz, A. Narkawicz, J. Chamberlain, M. Consiglio, and J.M. Upchurch. A family of well-clear boundary models for the integration of UAS in the NAS. 2014.

[42] C. Muñoz, A. Narkawicz, G. Hagen, J.M. Upchurch, A. Dutle, M.C. Consiglio, and J. Chamberlain. DAIDALUS: Detect and avoid alerting logic for unmanned systems. *2015 IEEE/AIAA 34th Digital Avionics Systems Conference (DASC)*, pages 5A1–1–5A1–12, 2015.

[43] Pelosi, C. Kopp, and M. Brown. Range limited UAV trajectory using terrain masking under radar detection risk. *Applied Artificial Intelligence: An International Journal*, 26:743–759, 09 2012. doi: 10.1080/08839514.2012.713308.

[44] T. Prevot, J. Rios, P. Kopardekar, J. Robinson III, M. Johnson, and J. Jung. UAS traffic management (UTM) concept of operations to safely enable low altitude flight operations. 06 2016. doi: 10.2514/6.2016-3292.

[45] M. Radanovic, M. Omeri, and M.A. Piera. Test analysis of a scalable UAV conflict management framework. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, 233:6076 – 6088, 2019.

[46] R.Weibel, M. Edwards, and C. Fernandes. *Establishing a Risk-Based Separation Standard for Unmanned Aircraft Self Separation*. 2011. doi: 10.2514/6.2011-6921. URL https://arc.aiaa.org/doi/abs/10.2514/6.2011-6921.

[47] S. Smearcheck, S. Calhoun, and T. Adami. *Analysis of Quantitative Terminal Area Well-Clear Volumes for Unmanned Aircraft Systems*, pages 2780–2795. 2018. doi: https://doi.org/10.33012/2018.16065. URL https://www.ion.org/publications/abstract.cfm?articleID=16065.

[48] S. Temizer, M.J. Kochenderfer, L.P. Kaelbling, T. Lozano-Perez, and J.K. Kuchar. Collision avoidance for unmanned aircraft using markov decision processes. In *AIAA Guidance, Navigation, and Control 2010 Conference*, 2010. URL https://arc.aiaa.org/doi/pdf/10.2514/6.2010-8040.

[49] E. Theunissen, J. Koeners, R.M. Rademaker, R. Jinkins, and Tim Etherington. Terrain following and terrain avoidance with synthetic vision. pages 4.D.2 – 41, 01 2005. ISBN 0-7803-9307-4. doi: 10.1109/DASC.2005.1563363.

[50] J. Trock and G. Keithley. Operating BVLOS: FAA proposes guidance on staying "well clear" – the hockey puck of clearance, 2018. URL https://www.uasinsights.com/2018/12/13/156/.

[51] N. Underhill and A. Weinert. Applicability and surrogacy of uncorrelated airspace encounter models at low altitudes, 2021.

[52] S. Wallace. Development of a small and inexpensive terrain avoidance system for an unmanned areal vehicle via potential function guidance algorithm. *Master's Theses and Project Reports*, 09 2010.

[53] A. Weinert and M. Kochenderfer. Airspace encounter models. https://github.com/Airspace-Encounter-Models, 2021.

[54] A. Weinert and N. Underhill. Generating representative small UAS trajectories using open source data. In *2018 IEEE/AIAA 37th Digital Avionics Systems Conference (DASC)*, pages 1–10, 2018. doi: 10.1109/DASC.2018.8569745.

[55] A. Weinert, E. Harkleroad, J.D Griffith, M. Edwards, and M. Kochenderfer. Uncorrelated encounter model of the national airspace system version 2.0, 08 2013.

[56] A. Weinert, S. Campbell, A. Vela, D. Schuldt, and J. Kurucar. Well-clear recommendation for small unmanned aircraft systems based on unmitigated collision risk. *Journal of Air Transportation*, 26:1–10, 08 2018. doi: 10.2514/1.D0091.

[57] A. Weinert, N. Underhill, and A. Wicks. Developing a low altitude manned encounter

model using ADS-B observations. In *2019 IEEE Aerospace Conference*, pages 1–8, 2019. doi: 10.1109/AERO.2019.8741848.

[58] A. Weinert, L. Alvarez, M. Owen, and B. Zintak. A quantitatively derived NMAC analog for smaller unmanned aircraft systems based on unmitigated collision risk. 2020. doi: 10.5281/zenodo.3727764. URL https://doi.org/10.5281/zenodo.3727764.

[59] A. Weinert, M. Edwards, L. Alvarez, and S. Katz. Representative small UAS trajectories for encounter modeling. 01 2020. doi: 10.2514/6.2020-0741.

Appendices

Appendix A

Matlab Trajectory Generation Code

A.1 Intruder Trajectory Generation Code

```
clear;
tic;
format long
```

Notes:

takes about 35 minutes to generate 1000 encounters

Intruder obstacle avoidance trajectory generator

This code is to generate intruder trajectories near an obstacle at (0,0)

Wagon wheel trajectories with CPA near (0,0) at 150 seconds (total encounter 180 sec); sample rate is 10 Hz (better resolution for manuvering around obstacle)

Parameters user may want to change:

- *write_to_file* when ready to write all trajectories in .csv format
- *path* to location where .csv files will be saved
- *num_runs* to how many trajectories are desired
- *dist_away_from_obs* to the radius of the intruder from the obstacle
- *alt* to desired altitude bin
- *traj_len* currently set to match MIT LL encounter generator (182 seconds)
- *vel* to ownship velocity value

Units: feet, knots; these are converted to meters in the Python simulation

```
% set write_to_file == true when ready to export trajectories
write_to_file = false;
```

Set up parameters

```
% change path ever time so that previous trajectories are not overwritten
% path = ['D:\Kendy_VT\Code\enc_repo\shielding\intruder_300ft\'];
path = ['C:\Users\kendye\Desktop\temp\'];
% change num_runs variable to the desired number of trajectories
num_runs = 10;

dist_away_from_obs = 500; % intruder must stay 500ft away from obstacles


% change altitude values in brackets to change the altitude bin of
% trajectories - this will generate randome altitudes for each trajectory
% alt = randi([100 500], num_runs,1);
alt = 200*ones(num_runs,1);
% MIT LL enc. gen. requires altitude in MSL. This conversion is for (0,0)
alt_msl = alt + 18.3296*3.28084; % MSL at (0,0) is 18 meters
% trajectory length of 182 seconds is used for MIT LL encounter generator
traj_len = 182; % seconds
% velocity is set at a constant value of 10 kts
vel = 50*1.688; % 50 kts (converted to ft/sec for sim)
% randomize a theta vector to change direction of flight
theta = (2*pi)*rand(1,num_runs); % rad
deg = (180/pi) * theta;

tstep = 0.1;
tCPA = 120;

% create random encounters
time = 0:tstep:traj_len;

% initiate a starting point for encounters, velocity*150 seconds away from
% origin
start_loc_dist = vel*tCPA;
start_pos_E = 18000 + start_loc_dist*cos(theta)';
start_pos_N = 18000 + start_loc_dist*sin(theta)';
% start_pos_E = zeros(size(theta))';
% start_pos_N = zeros(size(theta))';
wgs84 = wgs84Ellipsoid;
```

Develop nominal trajectory (no obstacle avoidance)

```
% outer loop for each ENCOUNTER, inner loop for each time step within a
% single encounter

for j = 1:length(theta)
    trajEN(1,1,j) = start_pos_E(j);
```

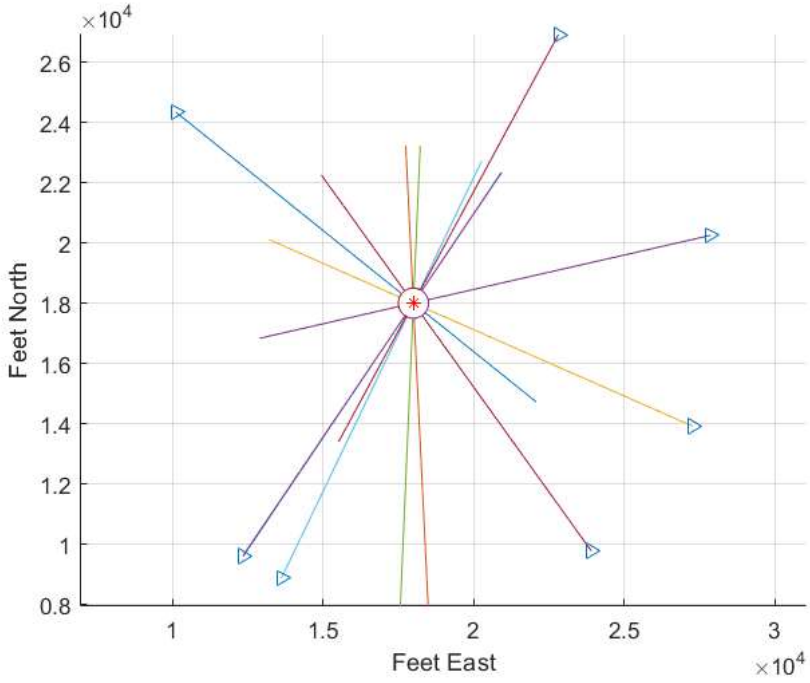
```
    trajEN(2,1,j) = start_pos_N(j);

    % initialize heading
    heading(1,j) = theta(j) *(180/pi); % set heading to theta, change to deg
    dheading(1,j) = 0; % straight line trajectories have a dh of 0

    for i = 1:length(time)-1

        deltax = vel*cos(theta(j))*tstep;
        deltay = vel*sin(theta(j))*tstep;

        E_val = trajEN(1,i,j);
        N_val = trajEN(2,i,j);

        % new position = old position + delta
        trajEN(1,i+1,j) = trajEN(1,i,j) - deltax;
        trajEN(2,i+1,j) = trajEN(2,i,j) - deltay;

        % save lat and lon (scaling purposes only, actual location doesn't
        % matter)
%         [trajLatLon(1,i+1,j), trajLatLon(2,i+1,j), ~] = enu2geodetic(E_val, N_val, alt(j), 0, 0, 0, wgs84);

        % heading is a matrix of headings (should all be the same for each
        % encounter) row# = time step; column# = encounter#
        heading(i+1,j) = theta(j)*(180/pi); % constant heading based on theta
        dheading(i+1,j) = 0;
    end
end

% figure(1)
% hold on
% for ii = 1:length(theta)
% %     plot(trajLatLon(1,:,ii),trajLatLon(2,:,ii))
%     plot(trajEN(1,:,ii),trajEN(2,:,ii), 18000,18000,'o', start_pos_E, start_pos_N, '>')
% end
% grid on; xlabel('East (feet)'); ylabel('North (feet)');
% title('Trajectories w/ no obstacle avoidance')
% hold off
```

Create obstacle avoidance

This is written to be able to process more than one obstacle, but has currently only been tested using one obstacle

```
% one obstacle at the unmitigated CPA (18000, 18000)
obs_loc = [18000; 18000];


% now, when the intruder flies within 500 feet of the obstacle, move it:

% for each obstacle, check each time step in each encounter to see if the
% intruder gets close to the obstacle:

% first loop-- obstacle(mm)
for mm = 1:size(obs_loc,2)

    obs_temp_E = obs_loc(1,mm);
    obs_temp_N = obs_loc(2,mm);

    % second loop-- one trajectory(jj)
    for jj = 1:num_runs

        % initialize trajectory lat,lon
%         [trajLatLon(1,1,jj), trajLatLon(2,1,jj), ~] = enu2geodetic(trajEN(1,1,jj), trajEN(2,1,jj), alt(jj), 0, 0, 0, wgs84);
        frst_mvmnt = true;
        % final loop-- each time step in a trajectory(kk)
        for kk = 1:length(time)
            % Current E,N position
            E_val = trajEN(1,kk,jj);
            N_val = trajEN(2,kk,jj);

            dist_btwn = sqrt((E_val-obs_temp_E)^(2) + (N_val-obs_temp_N)^(2));

            if dist_btwn <= dist_away_from_obs
                radius = dist_away_from_obs;
                % generate delta theta based upon arc length formula
                % (breaking up pi radians into equal sections based upon
                % velocity)
                dtheta = (pi / (2*radius / vel)) * tstep;

                if frst_mvmnt == true
```

```
                    prev_crcl_theta = atan2((N_val-obs_temp_N),(E_val-obs_temp_E)); % in radians
                    x = obs_temp_E + radius.*cos(prev_crcl_theta + dtheta);
                    y = obs_temp_N + radius.*sin(prev_crcl_theta + dtheta);

                    trajEN_obs(1,kk,jj) = x;
                    trajEN_obs(2,kk,jj) = y;

                    frst_mvmnt = false;
                else
                    % calculate the angle from the previous move
                    prev_crcl_theta = atan2((y-obs_temp_N),(x-obs_temp_E)); % in radians

                    x = obs_temp_E + radius*cos(prev_crcl_theta + dtheta);
                    y = obs_temp_N + radius*sin(prev_crcl_theta + dtheta);

                    trajEN_obs(1,kk,jj) = x;
                    trajEN_obs(2,kk,jj) = y;

                end


                % update dh != 0
                dheading(kk,jj) = 4; % dheading is changing if ac turns
            else
                % if the point isn't within 500 feet of the obstacle, leave
                % it alone
                trajEN_obs(1,kk,jj) = trajEN(1,kk,jj);
                trajEN_obs(2,kk,jj) = trajEN(2,kk,jj);

%                 heading(kk, jj) = atan2((start_pos_N(jj) - N_val),(start_pos_E(jj)-E_val));
            end
            heading(kk, jj) = atan2((start_pos_N(jj) - trajEN_obs(2,kk,jj)),(start_pos_E(jj)-trajEN_obs(1,kk,jj)))*(180/pi);
            % save location in lat, lon format
            % format for lat lon conversion:
            % [lat,lon,h] = enu2geodetic(xEast,yNorth,zUp,lat0,lon0,h0,spheroid)
            [trajLatLon(1,kk,jj), trajLatLon(2,kk,jj), ~] = enu2geodetic(trajEN_obs(1,kk,jj), trajEN_obs(2,kk,jj), alt(jj), 0, 0, 0, wgs84

        end

    end


end


figure(2)
hold on
for ii = 1:length(theta)
    plot(trajEN_obs(1,:,ii),trajEN_obs(2,:,ii), start_pos_E, start_pos_N, '>', ...
        obs_loc(1), obs_loc(2), 'r*', 'markersize', 6)
%         start_pos_E(ii), start_pos_N(ii), '^')
end
grid on; xlabel('Feet East'); ylabel('Feet North'); axis equal;
% title('Intruder Trajectories for Obstacle Shielding Testing')
hold off
```



Set up the data to save to a file

```
header = {'NAME', 'east', 'north', 'alt', 'hdg', 'speed', 'dh', 'time'};
units = {'unitless', '[ft]', '[ft]', '[ft]', '[rad]', '[fps]', '[fps]', '[s]'};

% header = {'ID', 'Time', 'lat', 'lon', 'alt_AGL_ft', 'speed_kts', 'heading_deg', 'dh_fpm', 'alt_MSL_ft'};
% units = {'unitless', '[sec]', '[]', '[]', '[]', '[]', '[]', '[]'};

% alt = num2cell(alt);
% save column vectors of data that is the same for each encounter
name = repelem([{'INTRUDER'}], [traj_len/tstep+1])';
name = cellstr(name);


speed =  num2cell(repelem(vel, traj_len/tstep+1))';
time = num2cell(time)'; % make time a column vector cell

ID = 1;         % initialize the ID number


% outer For loop creates the metadata file, inner For loop creates each
% individual trajectory file
% for nn = 1:size(theta,2)
for mm = 1:size(theta,2)

    % set up params to save matrix of each encounter
    east = num2cell(trajEN_obs(1,:,mm))';
    north = num2cell(trajEN_obs(2,:,mm))';
    hdg = num2cell(heading(:,mm));
    dh = num2cell(dheading(:,mm));

    lat = num2cell(trajLatLon(1,:,mm))';
    lon = num2cell(trajLatLon(2,:,mm))';


    % alt AGL in feet
    alt_val = alt(mm,1);
    alt_vec = ones(traj_len/tstep+1, 1)*alt_val;
    alt_vec = num2cell(alt_vec);

    % alt MSL in feet
    alt_msl_val = alt_msl(mm,1);
    alt_msl_vec = ones(traj_len/tstep+1, 1)*alt_msl_val;
    alt_msl_vec = num2cell(alt_msl_vec);

    % make vector of ID values
    ID_vec = num2cell(ones(traj_len/tstep+1, 1)*ID);

    % put together the entire matrix of the intruder encounter
```

```
    data_all = [name east north alt_vec hdg speed dh time];
    int_mat = [header; units;  data_all];
%     data_all = [ID_vec time lat lon alt_vec speed hdg dh alt_msl_vec];
%     int_mat = [header;  data_all];

    filename = [num2str(mm),'.csv'];
%     filename = [num2str(mm),'.txt'];
    path_plus_filename = [path,filename];

    if write_to_file == true
        writecell(int_mat, path_plus_filename)
    end


    ID = ID+1;
end
```

Create metadata file

This format follows the required format for MIT LL enc. gen.

Note: This section is not needed unless trajectories will be used in the MIT LL encounter generator

```
% % save the header row for metadata
% metadata_header = {'filename', 'Track_length', 'Min_Alt_msl', 'Max_Alt_msl', 'Min_Speed', 'Max_Speed', 'Min_Alt_agl', 'Max_Alt_agl', 'tr
%
% ID = 1;
% for m = 1:size(theta,2)
%     fname = [num2str(m),'.csv'];
%     fname_vec{m} = fname;
%
%         % save in metadata
%     track_len = traj_len/tstep+1;
%     meta_alt_vec = ones(traj_len/tstep+1, 1)*alt(m,1);
%     meta_alt_msl_vec = ones(traj_len/tstep+1, 1)*alt_msl(m,1);
%     min_alt_msl = min(meta_alt_msl_vec);
%     max_alt_msl = max(meta_alt_msl_vec);
%     min_alt_agl = min(meta_alt_vec);
%     max_alt_agl = max(meta_alt_vec);
%     min_speed = min(cell2mat(speed));
%     max_speed = max(cell2mat(speed));
%     trackID = 1;
%
%     metadata(m,:) = [track_len, min_alt_msl, max_alt_msl, min_speed, max_speed, min_alt_agl, max_alt_agl, ID];
%
%     ID = ID+1;
% end
%
% fname_vec = fname_vec';
% metadata = num2cell(metadata);
% metadata_wfname = [fname_vec, metadata];
% metadata_all = [metadata_header; metadata_wfname];
% path_plus_filename_metadata = [path,'metadata.csv'];
%
% if write_to_file == true
%     writecell(metadata_all, path_plus_filename_metadata)
% end

toc
```

```
Elapsed time is 27.024213 seconds.
```

A.2 Ownship Trajectory Generation Code

```
clear;
tic;
```

Notes:

- currently formatted to output trajectories to be injected directly in Python sim (using the same format as the MIT LL enc gen output)
- took about 35.4 minutes to generate 1000 encounters

Ownship straight line trajectory generator

This code is just to get straight line trajectories based upon the intruder location from intruder_maneuver_around_one_obs_v1.mlx

Parameters user may want to change:

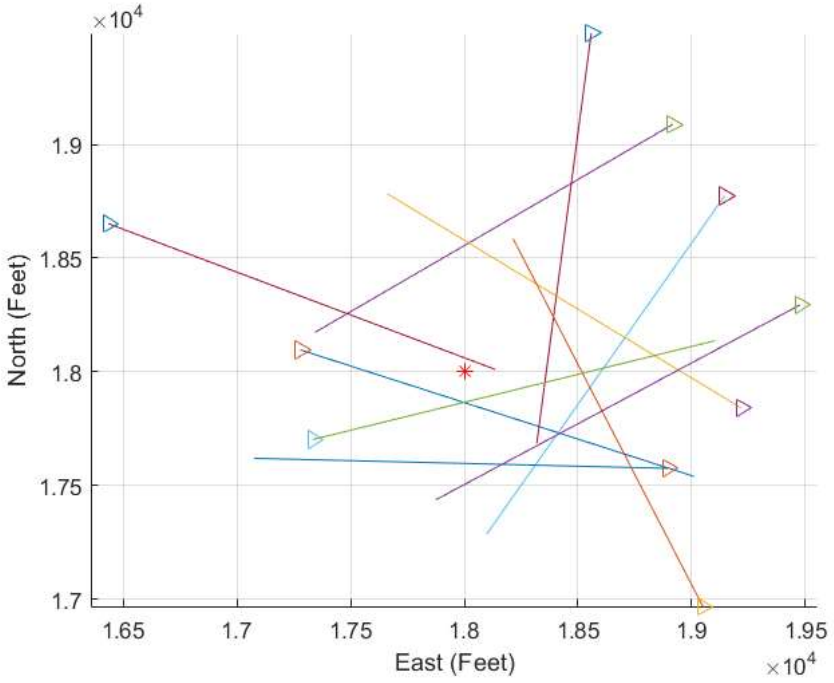
- *write_to_file* when ready to write all trajectories in .csv format
- *path* to location where .csv files will be saved
- *alt* to desired altitude bin
- *traj_len* currently set to match MIT LL encounter generator (182 seconds)
- *vel* to ownship velocity value

```
% set write_to_file == true when ready to export trajectories
write_to_file = false;
```

Set up parameters

```
% change path ever time so that previous trajectories are not overwritten
path = ['D:\Kendy_VT\Code\enc_repo\shielding\ownship_tstttt\'];

% path where the intruder files are
int_dir = 'D:\Kendy_VT\Code\enc_repo\shielding\intruder_10000\';

% set up files to iterate through them to grab location at CPA
intFilePattern = fullfile(int_dir, '*.csv');
theFiles1 = dir(intFilePattern);
dates1 = [theFiles1.datenum];
[~,order1] = sort(dates1);
int_list = theFiles1(order1);
int_east = [];
int_north = [];
% iterate through trajectory directory
for j = 1: length(int_list)
    int_baseFileName = int_list(j).name;
    int_fullFileName = fullfile(int_list(j).folder, int_baseFileName);

    % get the j'th intruder file:
    int_temp = readmatrix(int_fullFileName);

    int_east(j) = int_temp(1201,2);
    int_north(j) = int_temp(1201,3);
end
% change num_runs variable to the desired number of trajectories
num_runs = length(int_list);


% change altitude values in brackets to change the altitude bin of
% trajectories - this will generate randome altitudes for each trajectory
% alt = randi([100 400], num_runs,1);
alt = 200*ones(num_runs,1);
% MIT LL enc. gen. requires altitude in MSL. This conversion is for (0,0)
alt_msl = alt + 18.3296*3.28084; % MSL at (0.0) is 18 meters
% trajectory length of 182 seconds is used for MIT LL encounter generator
traj_len = 182; % seconds
% velocity is set at a constant value of 10 kts
vel = 10; % kts
% randomize a theta vector to change direction of flight
theta = (2*pi)*rand(1,num_runs); % rad
deg = (180/pi) * theta;

tstep = 0.1;

% create random encounters
time = 0:tstep:traj_len;

% % have all encounters start at (0,0). The actual location shouldn't matter once the
% % trajectories are injected into the MIT LL encounter generator
% start_pos_E = zeros(size(theta))';
% start_pos_N = zeros(size(theta))';
```

```
% want all encounters to be at (18000, 18000) at tCPA=120
tCPA = 120;
% initiate a starting point for encounters, velocity*150 seconds away from
% origin
start_loc_dist = vel*tCPA;

% the following code will generate the location of CPA to be at the
% intruder's location at 120 seconds
start_pos_E = int_east + start_loc_dist*cos(theta);
start_pos_N = int_north + start_loc_dist*sin(theta);

wgs84 = wgs84Ellipsoid;


% outer loop for each ENCOUNTER, inner loop for each time step within a
% single encounter

for j = 1:length(theta)
    trajEN(1,1,j) = start_pos_E(j);
    trajEN(2,1,j) = start_pos_N(j);

    % initialize heading
    heading(1,j) = theta(j) *(180/pi); % set heading to theta, change to deg

    for i = 1:length(time)-1

        deltax = vel*cos(theta(j))*tstep;
        deltay = vel*sin(theta(j))*tstep;

        E_val = trajEN(1,i,j);
        N_val = trajEN(2,i,j);

        % new position = old position + delta
        trajEN(1,i+1,j) = trajEN(1,i,j) - deltax;
        trajEN(2,i+1,j) = trajEN(2,i,j) - deltay;

        % save lat and lon (scaling purposes only, actual location doesn't
        % matter)
        [trajLatLon(1,i+1,j), trajLatLon(2,i+1,j), ~] = enu2geodetic(E_val, N_val, alt(j), 0, 0, 0, wgs84);

        % heading is a matrix of headings (should all be the same for each
        % encounter) row# = time step; column# = encounter#
        heading(i+1,j) = theta(j)*(180/pi); % constant heading based on theta
    end
end

hold on
for ii = 1:length(theta)
    plot(trajEN(1,:,ii),trajEN(2,:,ii), start_pos_E(ii), start_pos_N(ii), '>', 18000, 18000, 'r*', 'markersize', 7)
end
grid on; xlabel('East (Feet)'); ylabel('North (Feet)'); axis equal;
% title('Ownship Trajectories for Obstacle Shielding Testing')
hold off
```



Set up the data to save to a file

```
% Python Sim format
header = {'NAME', 'east', 'north', 'alt', 'hdg', 'speed', 'dh', 'time'};
units = {'unitless', '[ft]', '[ft]', '[ft]', '[rad]', '[fps]', '[fps]', '[s]'};

% MIT LL format
% header = {'ID', 'Time', 'lat', 'lon', 'alt_AGL_ft', 'speed_kts', 'heading_deg', 'dh_fpm', 'alt_MSL_ft'};
% % units = {'unitless', '[sec]', '[]', '[]', '[]', '[]', '[]', '[]'};

% alt = num2cell(alt);
% save column vectors of data that is the same for each encounter
name = repelem([{'OWNSHIP'}], [traj_len/tstep+1])';
name = cellstr(name);


speed =  num2cell(repelem(vel, traj_len/tstep+1))';
time = num2cell(time)'; % make time a column vector cell

dh = num2cell(zeros(traj_len/tstep+1, 1)); % change in heading = 0 for straight line trajectories

% use for MIT LL format
ID = 1;         % initialize the ID number


% outer For loop creates the metadata file, inner For loop creates each
% individual trajectory file
% for nn = 1:size(theta,2)
for mm = 1:size(theta,2)

    % set up params to save matrix of each encounter
    east = num2cell(trajEN(1,:,mm))';
    north = num2cell(trajEN(2,:,mm))';
    hdg = num2cell(heading(:,mm));

    lat = num2cell(trajLatLon(1,:,mm))';
    lon = num2cell(trajLatLon(2,:,mm))';

%     lat = north;
%     lon = east;

    % alt AGL in feet
    alt_val = alt(mm,1);
    alt_vec = ones(traj_len/tstep+1, 1)*alt_val;
    alt_vec = num2cell(alt_vec);

    % alt MSL in feet
    alt_msl_val = alt_msl(mm,1);
    alt_msl_vec = ones(traj_len/tstep+1, 1)*alt_msl_val;
    alt_msl_vec = num2cell(alt_msl_vec);
```

```
    % make vector of ID values
    ID_vec = num2cell(ones(traj_len/tstep+1, 1)*ID);

    % put together the entire matrix of the intruder encounter
    % Python Sim format
    data_all = [name east north alt_vec hdg speed dh time];
    int_mat = [header; units;  data_all];
    % MIT LL format
%     data_all = [ID_vec time lat lon alt_vec speed hdg dh alt_msl_vec];
%     int_mat = [header;  data_all];

    filename = [num2str(mm),'.csv'];
    path_plus_filename = [path,filename];

    if write_to_file == true
        writecell(int_mat, path_plus_filename)
    end


    ID = ID+1;
end
```

Create metadata file

This format follows the required format for MIT LL enc. gen.

Note: This section is not needed unless trajectories will be used in the MIT LL encounter generator

```
% % save the header row for metadata
% metadata_header = {'filename', 'Track_length', 'Min_Alt_msl', 'Max_Alt_msl', 'Min_Speed', 'Max_Speed', 'Min_Alt_agl', 'Max_Alt_agl', 'tr
%
% ID = 1;
% for m = 1:size(theta,2)
%     fname = [num2str(m),'.csv'];
%     fname_vec{m} = fname;
%
%         % save in metadata
%     track_len = traj_len+1;
%     meta_alt_vec = ones(traj_len+1, 1)*alt(m,1);
%     meta_alt_msl_vec = ones(traj_len+1, 1)*alt_msl(m,1);
%     min_alt_msl = min(meta_alt_msl_vec);
%     max_alt_msl = max(meta_alt_msl_vec);
%     min_alt_agl = min(meta_alt_vec);
%     max_alt_agl = max(meta_alt_vec);
%     min_speed = min(cell2mat(speed));
%     max_speed = max(cell2mat(speed));
%     trackID = 1;
%
%     metadata(m,:) = [track_len, min_alt_msl, max_alt_msl, min_speed, max_speed, min_alt_agl, max_alt_agl, ID];
%
%     ID = ID+1;
% end
%
% fname_vec = fname_vec';
% metadata = num2cell(metadata);
% metadata_wfname = [fname_vec, metadata];
% metadata_all = [metadata_header; metadata_wfname];
% path_plus_filename_metadata = [path,'metadata.csv'];
%
% if write_to_file == true
%     writecell(metadata_all, path_plus_filename_metadata)
% end

toc
```

```
Elapsed time is 26.446286 seconds.
```