Simulation as a Tool for Conspicuity Measurement

Brandon Rishi

CMU-RI-TR-24-58

July 30, 2024

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

Thesis Committee:
Professor Katia Sycara, *chair*
Professor Henny Admoni,
Renos Zabounidis

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

To my favorite family, friends, and educators who have supported me throughout my academic career.

Abstract

The use of unmanned aerial vehicles (UAVs) for time critical tasks is becoming increasingly popular. Operators are expected to use information from these swarms to make real-time and informed decisions. Consequently, detecting and recognizing targets from video is extremely pivotal to the success of these systems. At greater altitudes or with more vehicles, this process becomes increasingly difficult and requires significant scrutiny from an operator. Hence, it is important to try and optimize the information passed between a UAV and the operator. In this work, we present a system that can be used to recreate high-fidelity environments as well as simulate the behavior and data of a UAV swarm. This simulation extends AirSim, a flight simulator plugin for Unreal Engine, to deploy fixedwing aircrafts and other vehicles for the use of conspicuity measurement and other visual salience studies. Through the use of this simulator, we showcase the effects that various factors can have on an operator's target recognition including altitude, velocity, and conspicuity.

Acknowledgments

I would like to express my gratitude to my advisor, Professor Katia Sycara, who has helped guide me and advance my learning throughout this journey. I would also like to thank Professor Michael Lewis from the University of Pittsburgh, who welcomed me into his lab and offered another outlet to research and innovate. Both of whom have offered insightful feedback and given me the ability to learn and grow freely.

I would like to additionally thank the members of the CMU Ultimate Frisbee organizations for their friendship, encouragement, and general support over the last few years. You have made my years at CMU unforgettable.

Finally, I am extraordinarily grateful to my parents and family for their support and encouragement throughout my life and education.

Contents

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

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Multi-Agent Situation Awareness

The use of multi agent unmanned systems is becoming increasingly more popular for time and situation critical tasks. During which, it is important that we are able to effectively communicate with a human operator to have them absorb information about a given scenario as quickly as possible. However, determining exactly which pieces of information to pass an operator can prove to be challenging. When large amounts of information, such as synchronous camera feeds from multiple vehicles, are passed to a human operator, they may struggle to sift through the data to determine what is relevant. On the other hand, if not enough information is given, the operator may then have a difficult time grasping the overall scenario at hand.

These issues are only exacerbated depending on the context. With high speed vehicles, information will be gathered and expected to be processed at a much quicker rate. Similarly, different altitudes serve their own problems. For example, at higher altitudes resolution of any key points of interest will decrease making it more difficult to determine a point of interest. Comparatively, while at lower altitudes, the duration of video for these targets may decrease giving an operator less time to determine the relation between targets or the overall situation they are observing [15].

The piloting of these multi-agent scenarios also has a distinct impact on the amount and quality of information gained by an operator. Specifically, if a group of vehicles traverse paths that are closer together, an operator would gain more

specific insights into a smaller subset of targets. On the other hand, if they were to take very different paths with the intent to improve coverage, there would be much more information about the scenario as a whole, but less involving individual targets [6, 8, 14]. Furthermore, as the vehicles spread out more and more and/or increase in number, the amount of raw information that the operator receives increases which then later needs to be filtered. If at any time an operator is unable to process the information at least as quickly as they are receiving it, they then need to either ignore potentially relevant data to stay as updated as possible, or they will be using outdated intelligence. Hence, it is important to have a variety of behaviors available to a swarm. This way we can tailor our controls as we see fit.

1.2 Situation Awareness and Conspicuity

Situation awareness can be described as the understanding of a person about the environment around them. This concept describes how well an individual recognizes key actors in their environment, their relations to one another, and how this relation will impact the environment in the future [12, 21].

Improving an operator's situation awareness is extremely meaningful in any time dependent task. One of the best ways to do this would be to improve how quickly an operator can recognize a target. However, visual saliency can often be hard to measure. Hence, a metric known as conspicuity helps to quantify a very qualitative measurement. Conspicuity can be measured in one of two ways: either as an angle or as a distance. As an angle, conspicuity refers to the difference in angle between a person's focus of view and an object in which that person is still able to recognize that object. As a distance, conspicuity refers to the distance between a person's focus of view and an object in which that person is still able to recognize the object (see figure 1.1) [16, 25, 34, 35, 38, 41]. This metric plays an especially important role in target identification when an operator is surveying a large environment very quickly, as the operator may not have the ability to focus on the details in an environment.

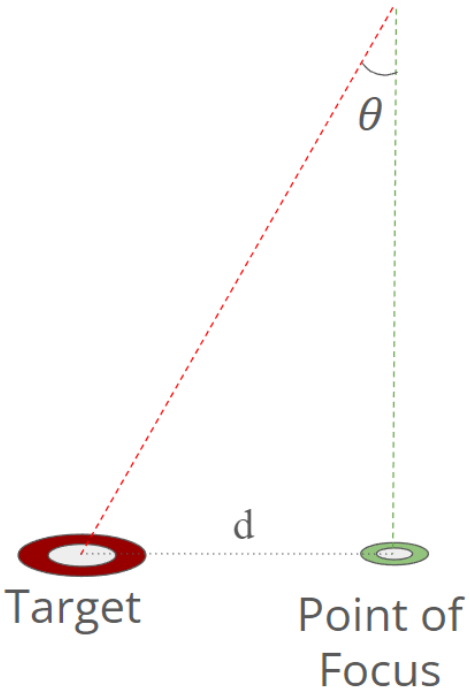

Figure 1.1: Visual representation of conspicuity where θ represents conspicuity as an angle and d represents conspicuity as a distance.

1.3 Motivation

This work was largely motivated by Professor Lewis' lab at the University of Pittsburgh. This lab aims to delve into human information processing specifically with robots and robot swarms. To perform this research, the members of this lab needed a platform that they could use to simulate and record data for later use when conducting studies and uncovering different interactions between human and robot(s). Hence, we worked to develop a simulator that matched the desired characteristics for these tests. This took our work into a variety of different directions, such as simulation, game development, and swarm control, to produce a simulator that matched their needs and an interface to control it. Currently, this work is being adapted for use with the Xprize Challenge [40], a competition to efficiently detect a wildfire scenario in large environments. This will require simulating and coordinating swarms of fixedwing UAVs to quickly survey an environment and effectively provide an operator information such that they can identify a wildfire event.

1.4 Solution Proposition

In this paper, we present a system that can be used to test different methods of relaying information about a scenario to an operator as well as some preliminary results. This system largely makes use of Unreal Engine 5 (UE5) and AirSim (later renamed to Colosseum during the transition from UE4 to UE5). UE5 is a game development engine that simulates an environment. UE5 can be modified to make a variety of different environments and scenarios, and can represent moving and/or time dependent actors. Colosseum is an plugin for UE5 that includes models and API controls for quadrotors and cars. We were able to craft the environments that matched our needs in UE5 and used AirSim to control the UAVs and simulate our tests. From there, we modified AirSim include three main components: heterogeneous vehicle use, fixedwing aircraft simulation, and consensus behaviors for swarms. This allowed us to have a wider and more fitting variety of vehicles and the means to control them collectively. Through the use of this simulator, we were able to develop tests to observe the effect that different variables have on an operator's ability to recognize and observe a target. This included factors such as velocity, altitude, conspicuity, etc.

1.5 Contributions

This work saw a lot of collaboration to produce our results. Most notably, this work is built off of the open source software AirSim [32]. AirSim offers a wide variety of simulation tools including a way to interface with Unreal Engine and control vehicles within it. Further, all of the studies that utilized this platform were undergone by the students in Professor Lewis' lab, while any necessary features of the simulator needed for the studies were developed by us.

Chapter 2

Related Works

2.1 Recreating High Fidelity Representations

There has been a great deal of work to represent environments and vehicles effectively in simulation. This can be seen with AirSim [32] IsaacSim, and other simulators. However, there is relatively little work focusing on situation awareness with high fidelity visuals from the operator point of view. Human Factor research has focused more on low fidelity simulations such as RESCHU [30] which show targets on a map and use either uncalibrated, unrealistic, or no imagery. The closest work that compares to ours is [17]. In this work, a different simulator, FlightGear, is used to have a human operator control a UAV to find key targets within a given timeframe. Due to its nature, it also focuses on trying to best relay information about a scenario to the operator. However, this work tested different training techniques for the operators of a single aircraft, rather than the effects of different autonomous controls of a fleet of aircrafts on situation awareness.

There are also a few studies that focus on the operator experience for target recognition in slightly different ways than our work [7, 23, 33]. Instead of checking how to best relay information, [10] instead brings to light how boredom and distraction can impact an operator of these systems. This paper discusses how requiring an operator to perform actions can greatly benefit their performance by keeping them engaged.

2.2 Target Recognition and Tracking

Outside of the scope of a human operator, there has been significant work detailing effective ways to identify and track targets using camera videos from UAVs. Understandably, if this can be done without the assistance of an operator in the loop, it would effectively accelerate the process. [36] discusses how the use of neural networks and precise models of the trajectory of a camera can be used to effectively locate targets. From there, their motion can be estimated and can be effectively tracked. [37] and [39] further discuss how the use of additional hardware such as a gimbal can assist with this process further, stabilizing the image while also providing useful analytics in terms of how to best control the quadrotor to track a target.

However, there are limitations in exclusively automating this process. For example, there are some scenarios in which a human may be necessary to make a decision, so their understanding of the situation is critical. Further, the tracking of these targets can aid an operator in understanding the situation, but these autonomous systems are often not able to understand a scenario by themselves.

2.3 Simulating Aircrafts

Due to their rising popularity and use in surveillance, there has been a lot of work that has gone into recreating fixedwing aircrafts in simulation. Many simulators even exist solely to recreate the dynamics of a fixedwing aircraft [24, 26, 31]. However, these simulators often are not well equipped to handle either multi-agent or purely autonomous UAVs. Of the simulators that do offer this capacity, many still lack the high-fidelity environment requirement that we are looking for [3, 5, 9, 11]. Hence, we look to adapt AirSim to take advantage of the realistic appearance and physics of the environments while also maintaining realistic fixedwing physics.

2.4 AirSim

A lot of this work is built on the existing AirSim platform. This is a simulator that leverages Unreal Engine to create a high fidelity visual representations of a variety

of environments [32]. Furthermore, AirSim offers simulation of two main vehicles: cars and quadrotors. These vehicles come with a prebuilt API that handles low level locomotion control (moving to a given position, with a desired velocity and yaw, etc.) as well as an interface for any sensors on the agent. Furthermore, Unreal Engine continues to receive updates and improvements, so the environments that can interact with these vehicles continue to improve as well. However, for this work, a variety of adjustments were necessary to simulate the desired vehicles, swarms, etc. and their corresponding behaviors.

2.5 Consensus Algorithms

Consensus algorithms have already been heavily documented. There are many papers that cover extensively how to get a fleet of agents to work together. The three movements that we cared about most extensively were rendezvous, anti-rendezvous, and biased flocking. [28] discusses in depth how to apply these different algorithms to a swarm that consists of three degree of freedom robots that move according to their heading. These techniques were then modified to match the scenario that we are exploring.

There are many other coverage based algorithms that have been developed for UAVs. Coverage Path Planning (CPP) has been a heavily researched topic due to the capabilities it serves for surveillance, disaster relief, and more. [27] demonstrates how using different formations can be an effective technique for navigating urban environments and simplifying the planning problem while also utilizing multiple aircrafts. [22] comparatively shows how a sampling based approach discretizes the search space more effectively and how it is a useful technique for mapping in three dimensions as opposed to two. Finally, [19] and [29] showcase different technqiues for using quadrotors for CPP while minimizing overlap. Specifically, [29] performs this in an earthquake setting, similar to that of a wildfire scenario that our work aims to eventually tackle. However, these different techniques are largely focused around complete documentation of the area without respect to real time decisions. Each of these papers aims to minimize the time taken to get a complete map or coverage of the area without respect to important points that may improve the overall awareness of the scenario at hand.

Chapter 3

Simulation

3.1 Environment Simulation

One of the major points of interest in this work was generating high fidelity representations of real world environments. This would allow us to create more realistic tests and better learn about the impact that each of the respective factors would have on an operator's ability to learn and retain information. To develop this simulation, we used UE5, an extremely high-fidelity game development platform. Due to the wide variety of preexisting assets that are available for this simulator, we were able to generate environments that matched our needs. Furthermore, with the use of Cesium [20], we were able to match these environments to real world maps for more realistic scenarios.

3.2 Using Hetergeneous Vehicles with AirSim

AirSim natively does not support heterogeneous simulation of vehicles i.e. swarms can only consist of all quadrotors, cars, etc. One of our goals was to allow for the use of different vehicles simultaneously to allow for more potential scenarios to be simulated. Although not used for any studies, this functionality was created for a situation in which coordination between UAVs and Unmanned Ground Vehicles (UGVs) would be beneficial. For example, in the circumstance in which a wildfire limited access

to certain areas, it would be useful to communicate between UAVs and UGVs to simultaneously survey from above while driving to a destination.

To implement this, we leveraged the existing `Multirotor` and `Car` simulation modes. Each simulation mode creates an API client for its respective vehicle type that allows you to interface with vehicles of that type. This is done by establishing a remote procedure call (RPC) socket connection that starts up the respective API server and allows a user to connect to it through a client. By default, AirSim only spawns and manages one server. By modifying the default AirSim methods, we were able to change this behavior to allow for multiple servers to be spawned at the same time. Largely, this consisted of modifying the way that the simulation mode initialized the servers, such that we spawn multiple at once and track the running servers through a list rather than just a single server object. The RPC connections allowed us to communicate with the servers in the same way as if you were to run the vehicle types independently. This can be visualized in figure 3.1.

3.3 UAV Simulation

AirSim does not have any native Group-2 aircrafts (such as fixedwings or other high speed planes), rather it offers quadrotors and cars. Hence, we adapted the preexisting properties of the quadrotors to match the expected values of a fixedwing. This was comprised of three main components: speed, turning radius, and stalling speed.

The main motivation for this component of our work was the expected use of fixedwings in the Xprize challenge. Due to the large environment size and time constraint, high velocity aircrafts such as fixedwings would be a necessity for the challenge. However, we did not need to simulate every behavior of a fixedwing. Rather, we aimed to replicate specific behaviors of fixedwing aircrafts as defined by the requirements for conspicuity experiments and data generation, as well as expected behaviors for Xprize. For example, takeoff of our UAVs still consists of an aircraft raising up into the air as a quadrotor would. For our purposes, we cared specifically about the differences in maximum speed, turning radius, and stalling speed. These properties are critical for control of UAVs as a swarm simulating their expected behaviors.

Most notably, we wanted to represent an aircraft that could travel at least 200

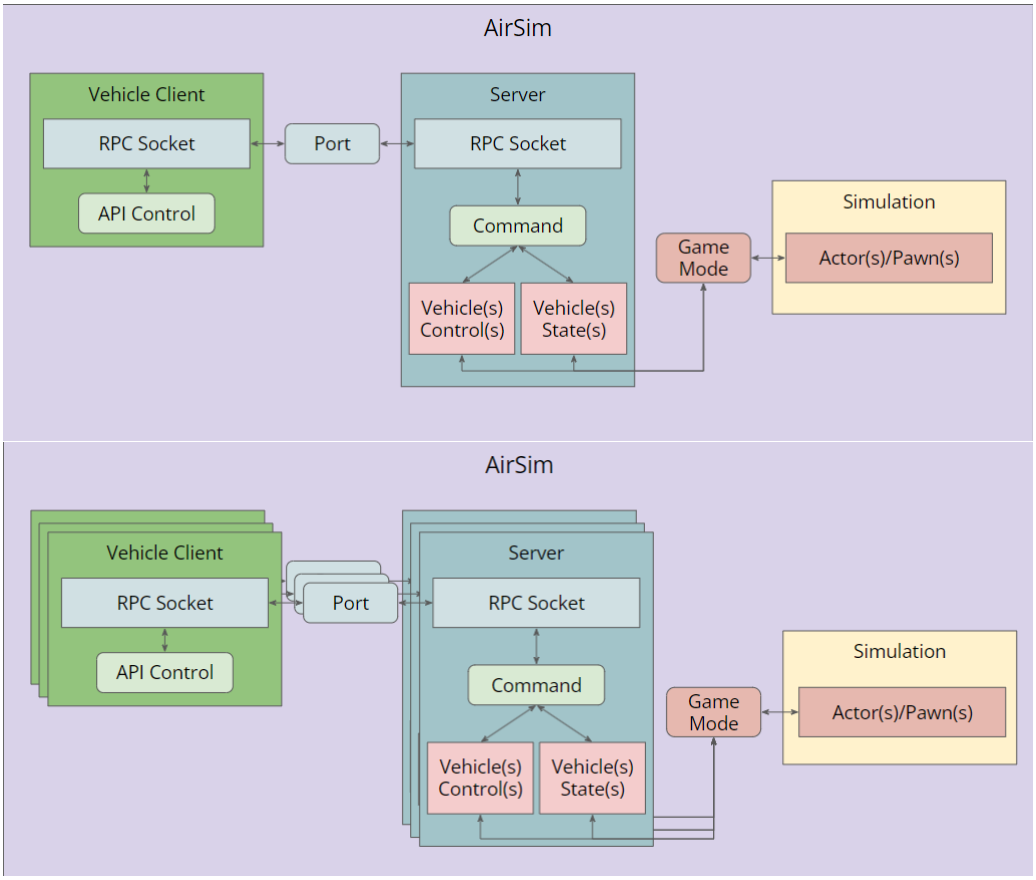

Figure 3.1: AirSim's native API server setup (top) and with tolerance for heterogeneous swarms (bottom).

Figure 3.2: Demo using a quadrotor to help a car navigate obstacles

knots, was limited to a turning radius of $\frac{\pi}{4}$, and would stall out if traveling below the stall speed of an Altius-700 [1] aircraft. We confirmed this behavior using `simGetVehiclePose` and `simGetGroundTruthKinematics` to get the ground truth pose and kinematics (linear velocity, etc.) of the aircraft during its operation and comparing it to our requirements.

3.3.1 Speed

The maximum speed of the quadrotors was significantly less than the desired speed of the fixedwings for our tests. This value was set and calculated according to the dynamics of a quadrotor. Hence, by modifying the parameters associated with the propeller diameter, max RPM, and thrust coefficient, we were able to achieve our desired velocities. We used the equation 3.1 to find the necessary thrust to travel at a given speed [4] (See table 3.1 for descriptors of the symbols).

$$\begin{aligned} C_T &= \frac{T}{\rho n^2 D^4} \\ F_T &= C_T \rho n^2 D^4 \end{aligned} \tag{3.1}$$

Symbol	Description
n	Propeller Rotation Rate (revolutions per second)
D	Propeller Diameter (m)
C_T	Thrust Coefficient
T	Thrust, ambient temperature (N)
ρ	Density of Air ($\frac{kg}{m^3}$)
F_T	Max thrust
R	Turning radius (m)
v	Linear velocity (m/s)
g	Gravity (m/s^2)
θ	Roll (rad)
L	Lift (N)
S	Wing Area of Aircraft (m^2)
C_L	Lift Coefficient
W	Weight (N)

Table 3.1: Description of symbols used to calculate the max dynamics of an aircraft.

The speed of the quadcopter was also able to be increased by adjusting the simulation speed of the AirSim components, effectively causing them to either fly faster or slower depending on the adjustment.

3.3.2 Turning Radius

The turning radius, and overall turning mechanism, of a quadrotor also did not match our intended goals. A quadrotor has a much tighter turning radius and can utilize the four points of lift to more accessibly get into certain positions. However, by limiting the quadrotor's drive train to only allow for forward movement, we can prohibit any strafing movement that the quadrotor may have otherwise performed. Furthermore, by setting a max roll amount as well as turning radius, we were able to simulate the behavior of a fixedwing more closely. This was done by adhering to the equation 3.2 [42] (see table 3.1 for descriptors of symbols).

$$R = \frac{v^2}{g \tan \theta} \tag{3.2}$$

By limiting θ to $\frac{\pi}{4}$, we were able to effectively recreate the fixedwing flight behavior.

3.3.3 Stalling

The final key feature needed for this system is the capacity for the aircrafts to stall when traveling at low speeds. Quadorotors have the capacity to hover whereas fixedwing aircrafts do not. To replicate this behavior, we check the linear velocity and roll angles of the aircrafts while they were flying and compare these values to the minimum velocity needed to maintain lift. If at any point, the linear velocity dropped lower than this value, the UAV begins to stall and pitch downward until it begins rapidly descending.

In this paper, we specifically cared about straight and level stall speed. This can be calculated according to equation 3.3 [2] (see table 3.1 for descriptors of symbols).

$$\begin{aligned} L &= \frac{1}{2}\rho v^2 S C_L \\ W &= mg \\ L &= \frac{W}{\cos(\theta)} \\ v &= \sqrt{\frac{2mg}{\rho S C_L \cos(\theta)}} \end{aligned} \tag{3.3}$$

We used the specifications of the Altius-700 [1] for the aircraft specific qualities, and used [18] to find the coefficient of lift.

3.4 Handling Multiple UAVs

We also wanted to test the capabilities of UE5 and AirSim for controlling multiple UAVs to see how the simulator could handle swarms for Xprize. As the number of UAVs increases, there is an increase in potential maneuvers, area we can cover, and information we can gather. However, due to high-fidelity simulation's high cost, we understood that there likely would be limitations and sought out finding them.

3.4.1 Limitations of UE5 and AirSim

Understandably, simulating a fixedwing UAV can be taxing on the performance of a computer. As described above, each aircraft was not only modelled, but also had

Number of UAVs	Behavior
1-6	Behaves normally, each agent is able to move and be simulated as expected
8-15	Jitter starts to be introduced to the system as AirSim begins to lag causing the controls to fall behind compared to the simulation of the agent
16-22	Agents are able to move but suffer from extreme lag and jitter
22+	Agents are effectively unable to move as the number of resources used exceeds the available amount

Table 3.2: Number of UAVs and the resulting behavior.

controllers that frequently checked the dynamics of each aircraft. Hence, there is a limit in terms of the number of aircrafts that we are able to effectively simulate. Although we did not record metrics on the individual costs of simulating an aircraft, we did find that CPU usage was the main limiting factor. We ran our simulation primarily on a workstation that had a 12th Generation Intel® Core™ i9-12900HK (24 MB cache, 14 cores, up to 5 GHz) CPU, NVIDIA® GeForce® RTX™ 3080 Ti, 16 GB GDDR6 GPU, and 64 GB of DDR5, 4800 MHz, dual-channel RAM.

AirSim also somewhat creates a bottleneck. AirSim operates with a single server that one or more clients can connect to. This server then internally handles the different physical dynamics of the vehicles being simulated, as well as the controls for each agent. Consequently, as the number of agents increases, AirSim can begin to struggle to maintain an effective simulation of each of them. Table 3.2 shows a comparison between the number of UAVs and the behaviors demonstrated while flocking (described below) on the default Blocks environment.

The number of UAVs that could be run concurrently varied with a variety of different factors. Most notably, the complexity of the environment and the control algorithm had the largest impacts. In an environment with many dynamic actors, the number of UAVs that could be seamlessly simulated decreased. Similarly, algorithm complexity negatively impacted performance. There was no quantitative measurement taken for the visual inaccuracy of the aircrafts.

3.4.2 JSBSim

We also tested one other simulator to try and replicate the flight dynamics of a fixedwing aircraft. JSBSim [13] is an open source flight dynamics model that comes equipped with fixedwing dynamics and controls. Hence, we wanted to try to translate the movements and behaviors from JSBSim to AirSim in an attempt to try to maximize the realness of the simulator. However, JSBSim was developed with the intent to control a single aircraft, whereas our goal was to use multiple. Hence, when we tested JSBSim's capacity to work with multiple aircrafts, it did not perform well. JSBSim has a single server-client pairing that it uses for controls and dynamics. The server will track the location and kinematics of a given aircraft at each simulated time step. When paired with AirSim for visual modelling, this caused discrepancies in timing. JSBSim would often update at a slower rate than AirSim or UE5, resulting in large jerking motions of the aircraft as they moved from location to location. Furthermore, because there is a single client, the dynamics of each aircraft would be updated serially. This resulted in much worse performance as you increased aircrafts. Even attempting to simulate three aircrafts would result in jagged and infrequent movements of the aircrafts.

3.5 Consensus Algorithms

We also looked into implementing a variety of consensus algorithms to effectively coordinate teams of UAVs.This was broken up into three main control schemes: rendezvous, anti-rendezvous, and biased flocking.

This behavior is not native to AirSim and would allow for convenient control of the UAVs when performing large scale surveillance or managing multiple swarms at once. Although this was not necessary for the current studies, this feature will be essential for the Xprize challenge.

3.5.1 Agent Model

Each UAV in the Unmanned Aerial System (UAS) has the following model. x^i, y^i, z^i are the positional components of an agent's pose and θ^i, ϕ^i, and ψ^i are the rotational

Symbol	Description
$\boldsymbol{p}$	Pose
$\boldsymbol{p}^i$	Pose of aircraft i
$\boldsymbol{p}_g$	Goal pose
$\boldsymbol{p}_w$	Pose in world coordinates
$\boldsymbol{p}_r$	Pose in relative coordinates
$N(i)$	Set of agents
s	Speed of agent
s_s	Stall speed of agent
μ	Rendezvous/Anti-Rendezvous coefficient
k_s	Speed coefficient
$\boldsymbol{q}$	Direction that the agents should flock toward

Table 3.3: Description of symbols used to perform consensus algorithms.

components. From there, the UAVs are controlled by giving a target position, (x, y, z) to travel to and a speed to do so. This position is relative to the initial basis of the UAV $(x_s^i, y_s^i, z_s^i, \theta_s^i, \phi_s^i, \psi_s^i)$. Hence, when controls were given in world coordinates, they were transferred to relative coordinates according to equation 3.4 (see table 3.3 for descriptor of symbols).

$$\begin{aligned} x_{r,g}^i &= x_{g,w}^i - x_s^i \\ y_{r,g}^i &= y_{g,w}^i - y_s^i \\ z_{r,g}^i &= z_{g,w}^i - z_s^i \end{aligned} \tag{3.4}$$

3.5.2 Swarm Behaviors

In this section we introduce the different behaviors available to the UAS. This is a set of behaviors that the agents are capable of performing and utilizing for different tasks.

Rendezvous

Rendezvous aims to consolidate the agents into one central location. The first part of generating each of the designated sequences is getting the centroid of the swarm and the relative difference in position of each UAV. From there, we can perform rendezvous. The rate at which the system converges can be altered based on both μ

as well as k_s. This can be represented according to equation 3.5 (see table 3.3 for descriptors of symbols).

$$\begin{aligned} \boldsymbol{p}_{centroid} &= \frac{1}{|N(i)|} \sum_{j \in N(i)} \boldsymbol{p}_{\boldsymbol{w}}^{\boldsymbol{j}} \\ \boldsymbol{p}_{g,w}^{i} &= \mu \left(\boldsymbol{p}_{centroid} - \boldsymbol{p}_{w}^{i} \right) + \boldsymbol{p}_{w}^{i} \\ s^{i} &= \max(s_s, k_s \| \boldsymbol{p}_{w}^{i} - \boldsymbol{p}_{centroid} \|) \end{aligned} \tag{3.5}$$

Anti-Rendezvous

Comparatively, anti-rendezvous aims to spread the agents out as much as possible. To perform anti-rendezvous, we instead set the goal position to be opposite of the direction of the centroid by setting μ to a negative value. However, speed is calculated in the opposite way, where the closer an agent is to the center, the faster it should fly.

$$s^{i} = \max(s_s, k_s \max(\| \boldsymbol{p}_{w}^{\forall j \in N(i)} - \boldsymbol{p}_{centroid} \|) - \| \boldsymbol{p}_{w}^{i} - p_{centroid} \|) \tag{3.6}$$

Biased Flocking

Finally, biased flocking aims to have the agents consolidate and traverse in a general direction as a group. This builds upon rendezvous to provide a direction to the UAS as well. This can be seen according to equation 3.7 (see table 3.3 for descriptors of symbols).

$$\begin{aligned} \boldsymbol{p}_{g,w}^{i} &= \boldsymbol{p}_{g,w,rendezvous}^{i} + \boldsymbol{q} \\ s^{i} &= \max(s_s, k_s \| \boldsymbol{p}_{w}^{i} - \boldsymbol{p}_{g,w}^{i} \|) \end{aligned} \tag{3.7}$$

Algorithm 2 Converge all agents upon one point within a distance R to the center

procedure RENDEZVOUS(N, μ, k_s, R)
    states, centroid, stallSpeeds ← `GetSwarmData`
    dists ← `dist`(states, centroid)
    while dists.`any` > R **do**
        targets ← μ(centroid−states) + states
        speeds ← max($k_s$dists, stallSpeeds)
        `MoveAgents`(N, targets, speeds)
        states, centroid, stallSpeeds ← `GetSwarmData`
        dists ← `dist`(states, centroid)
    end while
end procedure

Algorithm 3 Disperse all agents away from a center point until they are R far apart from their center

procedure ANTI-RENDEZVOUS(N, μ, k_s, R)
    states, centroid, stallSpeeds ← `GetSwarmData`
    dists ← `dist`(states, centroid)
    while dists.`any` < R **do**
        targets ← μ(centroid−states) + states
        speeds ← max(k_s(dists-max(dists), stallSpeeds)
        `MoveAgents`(N, targets, speeds)
        states, centroid, stallSpeeds ← `GetSwarmData`
        dists ← `dist`(states, centroid)
    end while
end procedure

Algorithms

Algorithm 1 Get the states, centroid, and stall speeds of all agents

procedure GETSWARMDATA(N)
    states ← `GetStates`
    centroid ← `mean`(states)
    stallSpeeds ← $\frac{\text{stallConstant}}{\sqrt{\cos(\theta)}}$ **return** states, centroid, stallSpeeds
end procedure

Algorithm 4 Converge all agents within radius R and have them move as a flock in a given direction q

procedure BIASED FLOCKING(N, μ, k_s, R, q)
    states, centroid, stallSpeeds ← `GetSwarmData`
    dists ← `dist`(states, centroid)
    while do
        needConvergence ← dists > R
        targets ← (μ(centroid−states) · needConvergence) + states + q
        speeds ← max(k_s`dist`(targets, states), stallSpeeds)
        `MoveAgents`(N, targets, speeds)
        states, centroid, stallSpeeds ← `GetSwarmData`
        dists ← `dist`(states, centroid)
    end while
end procedure

Chapter 4

Experimentation Using Simulator

This work was adapted to create a tool for testing for different visual saliency and conspicuity trials. The experimentation was was undergone by Professor Lewis and the students of his lab (most notably Xuehang Guo) who used this platform for their experiments. The work done to develop the flight simulator for use in these studies was handled by us, and the studies were performed by the students of the lab. Specifically, this can be seen in [15, 16]. This section showcases how this work was and can be used for the purpose of conspicuity testing as well as for future use in the Xprize challenge.

4.1 Target Conspicuity

The first study that utilized this simulator ([16]) aimed to showcase the effect that viewing angle, distance, and size of target can have on overall conspicuity of that target. Specifically, it highlights a phenomenon known as the Critical Gap. The Critical Gap is an empirical finding that states that the conspicuity of a target, when measured as a distance, remains the same when viewed from any distance beyond a given threshold distance. This can then be expanded to show the relation between conspicuity angle and viewing distance of a target according to equation 4.1.

$$\mathcal{CG} = \tan\alpha \times d = \tan\beta \times h \tag{4.1}$$

By demonstrating this, fewer conspicuity measurements need to be taken as you would not need to record a new measurement for every distance.

4.1.1 Methods

This experiment utilized the AirSim environment to measure conspicuity of a target at different sizes. Thirty trials were performed for each of the sizes of targets, and different viewing distances were used for each of the trials. Data was collected in two different ways, each providing a different interface to the user. The first utilized a camera mounted on a quadrotor to observe a target. The quadrotor moved over the target in one of the different four-linked directions (up, down, left, and right), and when the quadrotor was directly above the target the view camera view would become full screen. The user was instructed to continue looking in the center of the screen and to press any key when the target was no longer visible. This was repeated for each of the four-linked directions per trial. This process was also performed using a UE5 blueprint that would display a top down image similar to the quadrotor, but with additional on screen prompts and without the camera moving. Here, the user was asked to visually track an arrow rather than focus on the center of the screen. Furthermore, they were able to press a key to begin the subtrial in a given direction. The two different prompts can be seen in figure 4.1.

4.1.2 Results and Conclusions

This study confirmed the existence of the Critical Gap phenomenon. The changes in altitude and target size matched the expected results, where a larger and more visible target would result in a larger Critical Gap measurement, but also one with more variability. Comparatively, a smaller target resulted in a smaller Critical Gap value, but stayed relatively consistent past a certain threshold. By confirming these findings, conspicuity measurements beyond a certain distance threshold were proven to be relative to one another according to the relation 4.1. Hence, fewer measurements would need to be taken as you can calculate the critical distance of any distance beyond the threshold with just one measurement.

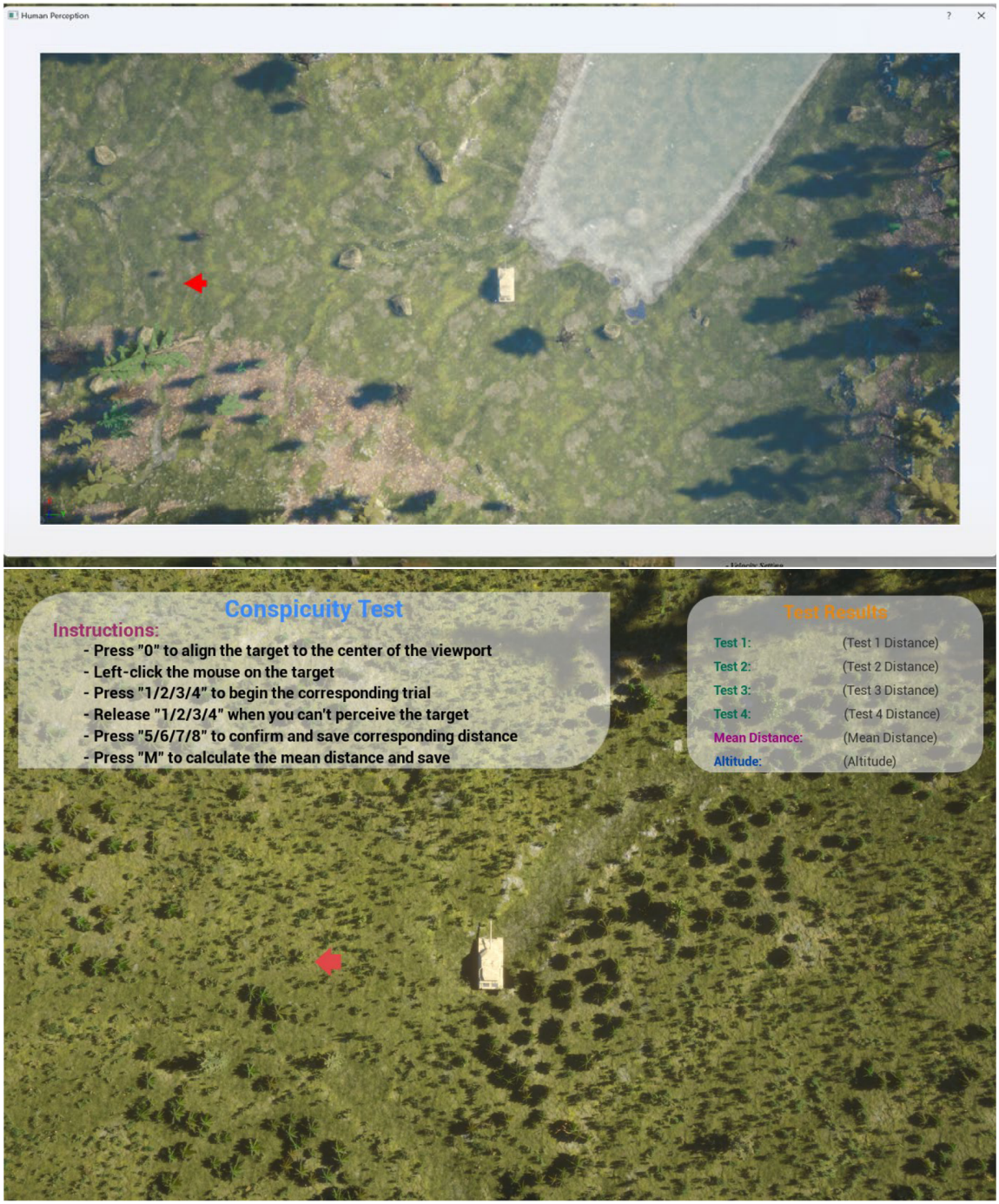

Figure 4.1: Conspicuity test for UAV (top) and UE5 Viewport Blueprint (bottom) [16]

Velocity	Altitude				
	30m	100m	200m	250m	300m
50 kt	1.4s	4.6s	9s	11s	13.6s
100 kt	0.7s	2.3s	4.6s	5.7s	6.8s
200 kt	0.3s	1.1s	2.3s	2.8s	3.4s

Table 4.1: Duration of target visibility for a given speed and altitude [15]

4.2 Effects of Altitude and Speed on Conspicuity

This simulator also was used for a study that focused on the effects of speed and altitude on conspicuity [15]. This experiment was designed to test the effect different factors on conspicuity for the Xprize [40] challenge. At very high speeds and altitudes, it is difficult to observe certain targets from the fixedwing cameras. Hence, this experiment tests how these different parameters modify conspicuity of a target so we can optimize the flight parameters to maximize conspicuity and situation awareness.

4.2.1 Methods

In this study, a fixedwing aircraft traveled at a fixed velocity and altitude across a given environment. In the environment, a target was placed that the observer needed to identify. Due to the nature of the different altitudes and speeds, the duration of time that the target was visible varied. The duration that target was visible for varied from 0.3 at the minimum all the way to 13.6 seconds at the maximum (see table 4.1). Different targets and levels of conspicuity (target background combinations, visual angles of deviation, etc.) were used for a broader test that incorporated different scenarios. In the circumstance where a target was visible for an extremely short duration, static scenarios were recorded in which the videos were reduced in duration and entry time of the target was more uniform.

4.2.2 Results and Conclusions

This study showcased the impact that several factors had on an operator's ability to recognize a target: velocity, altitude, and conspicuity conditions. After testing,

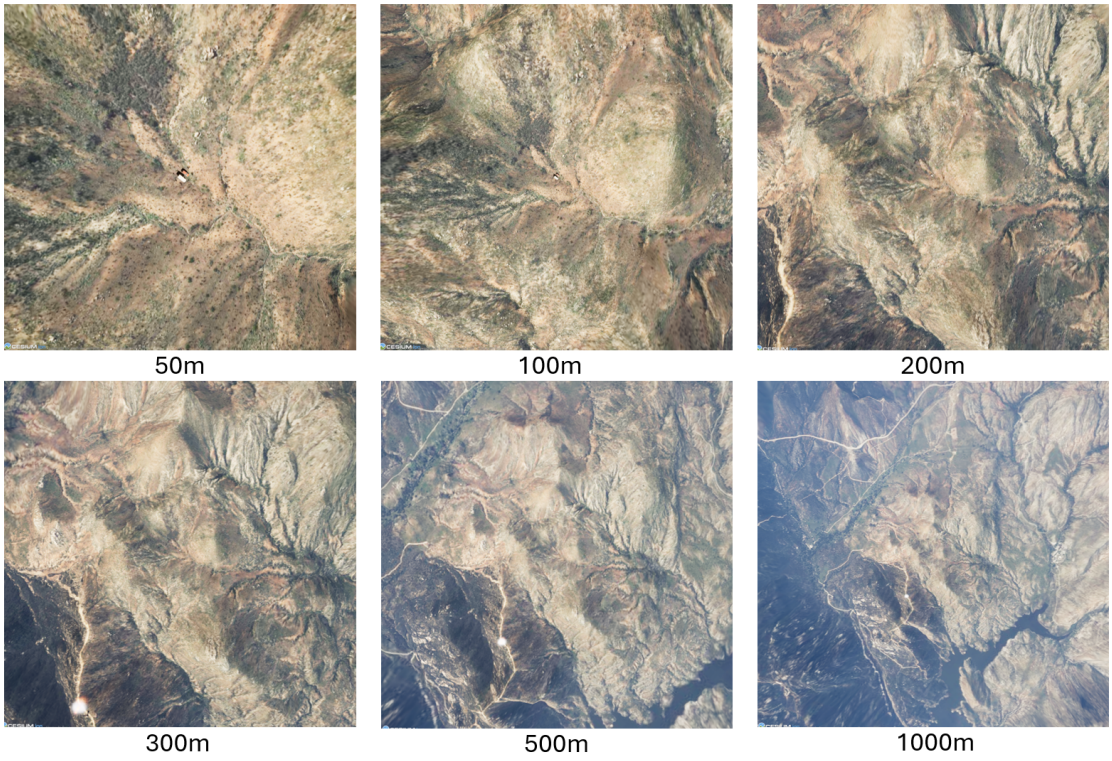

Figure 4.2: Truck at varying altitudes

altitude and conspicuity conditions were shown to have the largest impact on an operator's recognition of a specific target. Lower altitudes, despite the decreased exposure time, showed overall improved performance due to increased detail. Furthermore, increased conspicuity conditions saw a clear increase in recognition accuracy. Finally, velocity seemed to have little impact as a whole (see 4.3).

One important takeaway from this, however, is that as the overall conspicuity of a target increases, even performance at a high altitude, recognition will as well. In the testing setup, a firetruck, which is much more visually distinct compared to the other stimuli, was used for the 250m tests. Consequently, the 250m tests demonstrated a high level of recognition. This is an important distinction as it showcases how we could want to augment our approach to guiding UAVs based on the expectation of the conspicuity of a target. For example, if we have a visually conspicuous target, we could maneuver the aircrafts at a higher altitude to observe more of the landscape at once.

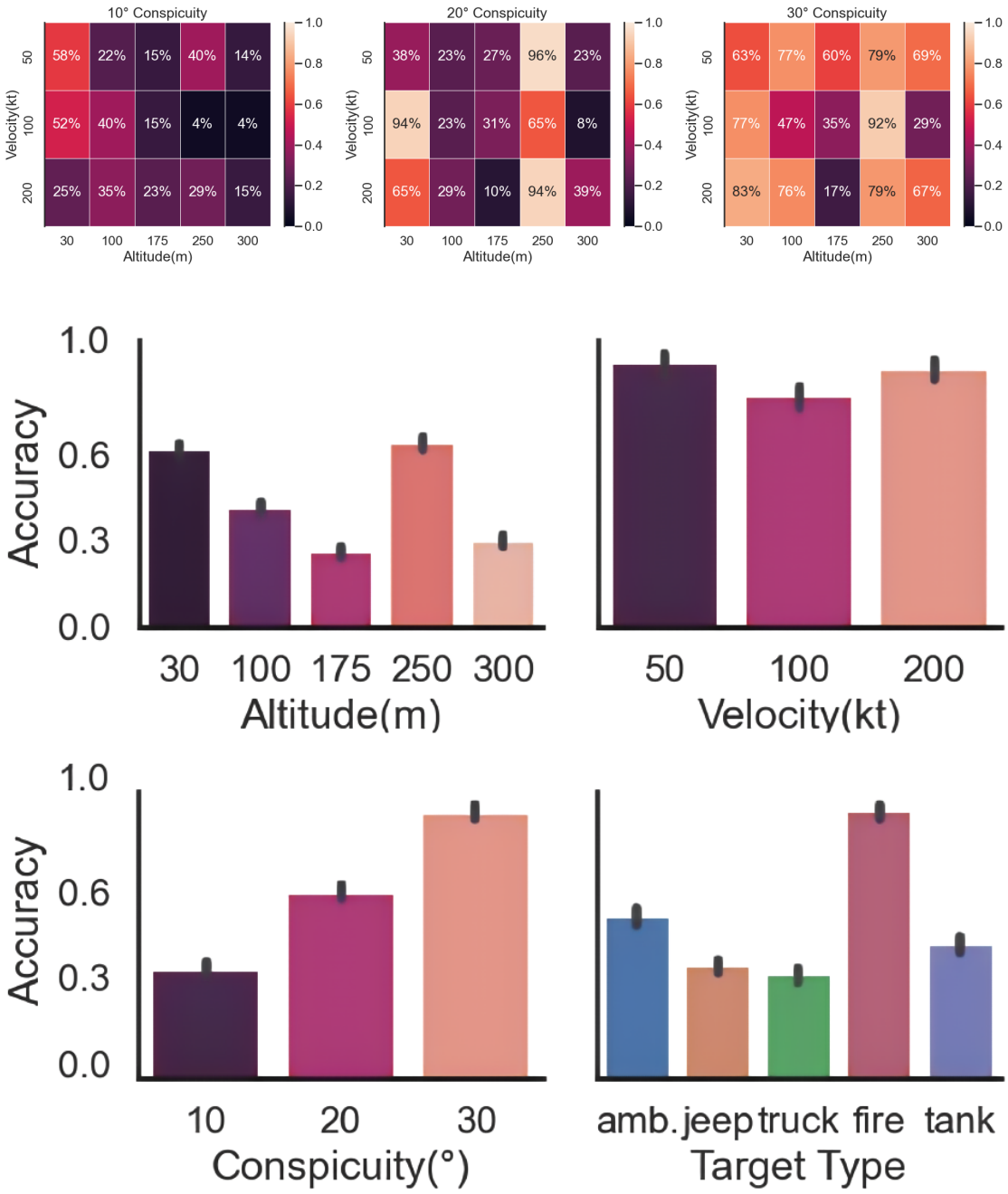

Figure 4.3: Relation between velocity, altitude, conspicuity, and recognition [15]

Chapter 5

Future Work

This simulator will eventually be used for the Xprize challenge, which will utilize the different functionalities that we have developed. This task will require a swarm of fixedwing UAVs to navigate a 1000 km^2 environment in a ten minute window. During this time, an operator will be collecting data on the environment with the intent to recognize if there is a wildfire event happening. Due to the nature of the task, this will require not only smart routing and maneuvering of the aircrafts, but also meaningful data that is sent to an operator. According to Xprize, there is a high number of false positives amongst wildfire detection which can lead to wasteful deployments and uses of resources. Furthermore, the fastest methods of detection currently sit in around the 40 minute time-frame [40], so a ten minute detection would be four times faster. Although many additions will be added to this work, such as interfaces for operators, specific coverage plans using UASs, etc., this work should stand as a basis for simulation and control such that others can develop the surrounding features more conveniently.

Chapter 6

Conclusions

We present a high-fidelity simulator that is able to effectively model real world scenarios and can be used for a variety of conspicuity testing. It has the capacity to model a variety of vehicles, notably cars, quadrotors, and fixedwing aircrafts, and provides the functionality to control each vehicle as a user would see fit. This system also provides functional consensus algorithms for a swarm of vehicles, which can be used for coverage and control. Finally, this simulator was utilized for conspicuity testing to help gain key insights into the factors that may impact a user's ability to recognize a target. Overall, the system is well equipped to handle testing use cases such as the ones defined above, but will need slightly more development for larger testing requirements, such as Xprize.

Bibliography

[1] The agile-launched, tactically-integrated unmanned system. URL https://www.anduril.com/hardware/altius/. 3.3, 3.3.3

[2] Accelerated stall speed. URL https://academicflight.com/articles/accelerated-stall-speed/. 3.3.3

[3] Amjed Al-Mousa, Belal H Sababha, Nailah Al-Madi, Amro Barghouthi, and Remah Younisse. Utsim: A framework and simulator for uav air traffic integration, control, and communication. *International Journal of Advanced Robotic Systems*, 16(5):1729881419870937, 2019. doi: 10.1177/1729881419870937. URL https://doi.org/10.1177/1729881419870937. 2.3

[4] J.B. Brandt, R.W. Deters, G.K. Ananda, O.D. Dantsker, and M.S. Selig. Uiuc propeller database vols 1-4. URL https://m-selig.ae.illinois.edu/props/propDB.html. 3.3.1

[5] Eitan Bulka and Meyer Nahon. Autonomous fixed-wing aerobatics: From theory to flight. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6573–6580, 2018. doi: 10.1109/ICRA.2018.8460610. 2.3

[6] Tauã M. Cabreira, Paulo R. Ferreira, Carmelo Di Franco, and Giorgio C. Buttazzo. Grid-based coverage path planning with minimum energy over irregular-shaped areas with uavs. In *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 758–767, 2019. doi: 10.1109/ICUAS.2019.8797937. 1.1

[7] S Carpin, Todor Stoyanov, Yashodhan Nevatia, Michael Lewis, and Jijun Wang. Quantitative assessments of usarsim accuracy. 01 2006. 2.1

[8] Jinchao Chen, Chenglie Du, Ying Zhang, Pengcheng Han, and Wei Wei. A clustering-based coverage path planning method for autonomous heterogeneous uavs. *IEEE Transactions on Intelligent Transportation Systems*, 23(12):25546–25556, 2022. doi: 10.1109/TITS.2021.3066240. 1.1

[9] Calvin Coopmans, Michal Podhradský, and Nathan V. Hoffer. Software- and hardware-in-the-loop verification of flight dynamics model and flight control simulation of a fixed-wing unmanned aerial vehicle. In *2015 Workshop on Research, Education and Development of Unmanned Aerial Systems (RED-UAS)*,

pages 115–122, 2015. doi: 10.1109/RED-UAS.2015.7440998. 2.3

[10] M.L. Cummings, C. Mastracchio, K.M. Thornburg, and A. Mkrtchyan. Boredom and Distraction in Multiple Unmanned Vehicle Supervisory Control. *Interacting with Computers*, 25(1):34–47, 01 2013. ISSN 0953-5438. doi: 10.1093/iwc/iws011. URL https://doi.org/10.1093/iwc/iws011. 2.1

[11] Francesco d'Apolito. Reinforcement learning training environment for fixed wing uav collision avoidance. *IFAC-PapersOnLine*, 55(39):281–285, 2022. ISSN 2405-8963. doi: https://doi.org/10.1016/j.ifacol.2022.12.035. URL https://www.sciencedirect.com/science/article/pii/S2405896322030750. 21st IFAC Conference on Technology, Culture and International Stability TECIS 2022. 2.3

[12] Mica R. Endsley. Toward a theory of situation awareness in dynamic systems. *Human Factors*, 37(1):32–64, 1995. doi: 10.1518/001872095779049543. URL https://doi.org/10.1518/001872095779049543. 1.2

[13] Berndt et al. Jsbsim. https://github.com/JSBSim-Team/jsbsim, 2023. 3.4.2

[14] Alia Ghaddar and Ahmad Merei. Eaoa: Energy-aware grid-based 3d-obstacle avoidance in coverage path planning for uavs. *Future Internet*, 12(2), 2020. ISSN 1999-5903. doi: 10.3390/fi12020029. URL https://www.mdpi.com/1999-5903/12/2/29. 1.1

[15] Xuehang Guo, Huao Li, Anfeng Peng, Lesong Jia, Brandon Rishi, and Michael Lewis. Determining human perceptual envelope in fixed wing uav surveillance. pages 1–6, 05 2024. doi: 10.1109/ICHMS59971.2024.10555714. (document), 1.1, 4, 4.1, 4.2, 4.3

[16] Xuehang Guo, Anfeng Peng, Lesong Jia, and Michael Lewis. Target conspicuity for human-uav visual perception. In *2024 IEEE 4th International Conference on Human-Machine Systems (ICHMS)*, pages 1–6, 2024. doi: 10.1109/ICHMS59971.2024.10555699. (document), 1.2, 4, 4.1, 4.1

[17] Svyatoslav Guznov, Gerald Matthews, Joel S. Warm, and Marc Pfahler. Training techniques for visual search in complex task environments. *Human Factors*, 59 (7):1139–1152, 2017. doi: 10.1177/0018720817712307. URL https://doi.org/10.1177/0018720817712307. PMID: 28609643. 2.1

[18] Carol Hodanbosi and Jonathan G. Fairman. Lift, 08 1996. URL https://www.grc.nasa.gov/WWW/K-12/WindTunnel/Activities/lift_formula.html. 3.3.3

[19] Wenjian Hu, Yao Yu, Shumei Liu, Changyang She, Lei Guo, Branka Vucetic, and Yonghui Li. Multi-uav coverage path planning: A distributed online cooperation method. *IEEE Transactions on Vehicular Technology*, 72(9):11727–11740, 2023. doi: 10.1109/TVT.2023.3266817. 2.5

[20] Cesium GS INC. Cesium: The platform for 3d geospatial, 2023. URL https://cesium.com/. 3.1

[21] Lesong Jia, Anfeng Peng, Huao Li, Xuehang Guo, and Michael Lewis. Situation theory based query generation for determining situation awareness across distributed data streams. pages 1–6, 05 2024. doi: 10.1109/ICHMS59971.2024.10555793. 1.2

[22] Wei Jing, Di Deng, Yan Wu, and Kenji Shimada. Multi-uav coverage path planning for the inspection of large and complex structures. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1480–1486, 2020. doi: 10.1109/IROS45743.2020.9341089. 2.5

[23] Lauren Reinerman-Jones Julian Abich IV and Gerald Matthews. Impact of three task demand factors on simulated unmanned system intelligence, surveillance, and reconnaissance operations. *Ergonomics*, 60(6):791–809, 2017. doi: 10.1080/00140139.2016.1216171. URL https://doi.org/10.1080/00140139.2016.1216171. PMID: 27557433. 2.1

[24] Waqas Khan and Meyer Nahon. Real-time modeling of agile fixed-wing uav aerodynamics. In *2015 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1188–1195, 2015. doi: 10.1109/ICUAS.2015.7152411. 2.3

[25] Vassilios Krassanakis, Matthieu Perreira Da Silva, and Vincent Ricordel. Monitoring human visual behavior during the observation of unmanned aerial vehicles (uavs) videos. *Drones*, 2(4), 2018. ISSN 2504-446X. doi: 10.3390/drones2040036. URL https://www.mdpi.com/2504-446X/2/4/36. 1.2

[26] Scott A. Morton and David R. McDaniel. A fixed-wing aircraft simulation tool for improving dod acquisition efficiency. *Computing in Science & Engineering*, 18(1):25–31, 2016. doi: 10.1109/MCSE.2015.133. 2.3

[27] Javier Muñoz, Blanca López, Fernando Quevedo, Concepción A. Monje, Santiago Garrido, and Luis E. Moreno. Multi uav coverage path planning in urban environments. *Sensors*, 21(21), 2021. ISSN 1424-8220. doi: 10.3390/s21217365. URL https://www.mdpi.com/1424-8220/21/21/7365. 2.5

[28] Sasanka Nagavalli, Nilanjan Chakraborty, and Katia Sycara. Automated sequencing of swarm behaviors for supervisory control of robotic swarms. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2674–2681, 2017. doi: 10.1109/ICRA.2017.7989312. 2.5

[29] Arman Nedjati, Gokhan Izbirak, Bela Vizvari, and Jamal Arkat. Complete coverage path planning for a multi-uav response system in post-earthquake assessment. *Robotics*, 5(4), 2016. ISSN 2218-6581. doi: 10.3390/robotics5040026. URL https://www.mdpi.com/2218-6581/5/4/26. 2.5

[30] Carl E. Nehme, Ryan M. Kilgore, and M. L. Cummings. Predicting the im-

pact of heterogeneity on unmanned-vehicle team performance. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 52(13):917–921, 2008. doi: 10.1177/154193120805201304. URL https://doi.org/10.1177/154193120805201304. 2.1

[31] David Orbea, Jessica Moposita, Wilbert G. Aguilar, Manolo Paredes, Gustavo León, and Aníbal Jara-Olmedo. Math model of uav multi rotor prototype with fixed wing aerodynamic structure for a flight simulator. In Lucio Tommaso De Paolis, Patrick Bourdot, and Antonio Mongelli, editors, *Augmented Reality, Virtual Reality, and Computer Graphics*, pages 199–211, Cham, 2017. Springer International Publishing. ISBN 978-3-319-60922-5. 2.3

[32] Shital Shah, Debadeepta Dey, Chris Lovett, and Ashish Kapoor. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and Service Robotics*, 2017. URL https://arxiv.org/abs/1705.05065. 1.5, 2.1, 2.4

[33] Florian Jentsch Thomas Fincannon, Joseph R. Keebler and Michael Curtis. The influence of camouflage, obstruction, familiarity and spatial ability on target identification from an unmanned ground vehicle. *Ergonomics*, 56(5):739–751, 2013. doi: 10.1080/00140139.2013.771218. URL https://doi.org/10.1080/00140139.2013.771218. PMID: 23514129. 2.1

[34] Alexander Toet. Computational versus psychophysical bottom-up image saliency: A comparative evaluation study. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(11):2131–2146, 2011. doi: 10.1109/TPAMI.2011.53. 1.2

[35] Erik Van der Burg, Jay Yu, Maarten Hogervorst, Bin Lee, Joanne Culpepper, and Alexander Toet. The relation between visual search and visual conspicuity for moving targets. page 9, 09 2021. doi: 10.1117/12.2600307. 1.2

[36] Jiabao Wang, Yafei Zhang, Jianjiang Lu, and Weiguang Xu. *A Framework for Moving Target Detection, Recognition and Tracking in UAV Videos*, pages 69–76. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. ISBN 978-3-642-27866-2. doi: 10.1007/978-3-642-27866-2_9. URL https://doi.org/10.1007/978-3-642-27866-2_9. 2.2

[37] Shuaijun Wang, Fan Jiang, Bin Zhang, Rui Ma, and Qi Hao. Development of uav-based target tracking and recognition systems. *IEEE Transactions on Intelligent Transportation Systems*, 21(8):3409–3422, 2020. doi: 10.1109/TITS.2019.2927838. 2.2

[38] A. H. Wertheim. Visual conspicuity: A new simple standard, its reliability, validity and applicability. *Ergonomics*, 53(3):421–442, 2010. doi: 10.1080/00140130903483705. URL https://doi.org/10.1080/00140130903483705. PMID: 20191416. 1.2

[39] Tian Xiang, Fan Jiang, Gongjin Lan, Jiaming Sun, Guocheng Liu, Qi Hao, and Cong Wang. Uav based target tracking and recognition. In *2016 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, pages 400–405, 2016. doi: 10.1109/MFI.2016.7849521. 2.2

[40] Xprize. Xprize wildfire, 2023. URL https://www.xprize.org/. 1.3, 4.2, 5

[41] Kao Zhang, Zhenzhong Chen, Songnan Li, and Shan Liu. An efficient saliency prediction model for unmanned aerial vehicle video. *ISPRS Journal of Photogrammetry and Remote Sensing*, 194:152–166, 2022. ISSN 0924-2716. doi: https://doi.org/10.1016/j.isprsjprs.2022.10.008. URL https://www.sciencedirect.com/science/article/pii/S0924271622002763. 1.2

[42] Qianyu Zhou, Li-Yu Lo, Bailun Jiang, Ching-Wei Chang, Chih-Yung Wen, Chih-Keng Chen, and Weifeng Zhou. Development of fixed-wing uav 3d coverage paths for urban air quality profiling. *Sensors*, 22(10), 2022. ISSN 1424-8220. doi: 10.3390/s22103630. URL https://www.mdpi.com/1424-8220/22/10/3630. 3.3.2