Interception of Automated Adversarial Drone Swarms in Partially Observed Environments

Daniel Saranovic[a], Martin Pavlovski[a], William Power[a], Ivan Stojkovic[a] and Zoran Obradovic[a,*]
[a] *Center for Data Analytics and Biomedical Informatics, Temple University, 1925 N. 12th St. Philadelphia, PA, USA*

Abstract. As the prevalence of drones increases, understanding and preparing for possible adversarial uses of drones and drone swarms is of paramount importance. Correspondingly, developing defensive mechanisms in which swarms can be used to protect against adversarial Unmanned Aerial Vehicles (UAVs) is a problem that requires further attention. Prior work on intercepting UAVs relies mostly on utilizing additional sensors or uses the Hamilton-Jacobi-Bellman equation, for which strong conditions need to be met to guarantee the existence of a saddle-point solution. To that end, this work proposes a novel interception method that utilizes the swarm's onboard PID controllers for setting the drones' states during interception. The drone's states are constrained only by their physical limitations, and only partial feedback of the adversarial drone's positions is assumed. The new framework is evaluated in a virtual environment under different environmental and model settings, using random simulations of more than 165,000 swarm flights. For certain environmental settings, our results indicate that the interception performance of larger swarms under partial observation is comparable to that of a one-drone swarm under full observation of the adversarial drone.

Keywords: UAV Interception, Machine Learning, Adaptive control, Swarm intelligence and decision-making

1 Introduction

The versatility and availability of UAVs make them particularly useful for various tasks such as payload delivery, reconnaissance, infrastructure crack assessment [16, 17], plant recognition in precision agriculture [13], and even facilitating remote unauthorized access to various digital systems [1]. Understanding possible adversarial employment of drones and developing defensive mechanisms against noncompliant drones is an increasingly important issue as drones become more readily available. Consequently, developing drone-on-drone interception mechanisms is instrumental. In the context of this paper, interception is defined as the act of preventing adversarial drones from reaching the desired destination by utilizing friendly drones with a spherical area of influence. By considering the possibility of a drone-on-drone interception, it is possible to leverage existing models of drone behavior to implement a method of interception.

Previous work on UAV interception includes using the Hamilton-Jacobi-Bellman equation [2, 3, 4], for which strong necessary and sufficient conditions need to exist to guarantee the existence of a saddle-point solution. One such condition is that the value function needs to be differentiable or Lipschitz continuous. Other work relies on additional sensor data to intercept the adversarial UAVs [5, 6]. Such sensors include stereo-optical sensors combined with image segmentation algorithms and sensor nodes able to capture the configuration parameters of the evader while also simultaneously providing the uncertainty of sensor measurements. As interception might be performed under hostile or environmentally non-desirable settings, in which one or several sensors might become damaged, it is crucial to have a functioning system even after a sensory failure occurs. The work in [11] focused on developing a mission-level robust, self-adaptive framework that offers a recovery system under a presence of a component fault. Intercepting the adversarial drone with as few sensors as possible is preferable since the adversary can corrupt the sensor data required for its detection. Nevertheless, visual data is very useful for generating obstacle maps and

*Corresponding author. E-mail: zoran.obradovic@temple.edu Tel.: +215.204.6265 FAX:+215.204.5082

can also be used for localization and navigation guidance [14]. Novel path planning algorithms, such as [12], can accompany such visual-based navigation systems. Such work can supplement the work presented in this paper, especially if interception is performed in urban environments with many obstacles. However, relying on visual data alone in fast-paced settings for interception purposes can hinder the chance of successful interception.

In this framework, a drone swarm under control of a defender is able to observe an adversarial swarm and then make predictions about its path, decide on an interception point, and move in such a way as to maximize the chance of collision between drones within the swarms. In addition to such fully observed environment, we also consider the case of a partially observed environment. We define such an environment as one where a friendly (*blue*) swarm has access to an adversarial (*red*) drone's positions for only a portion of its flight. More precisely, in a partially observed scenario, the global positions of the adversarial drone are known up to some critical time. After this critical time, the global positions of the adversarial drone are lost. When this position data is denied, the friendly swarm must make use of a predictive model to estimate the positions of the adversarial drone.

Consequently, this work focuses on a novel interception method and makes use of bootstrap aggregation [7] to predict the adversarial drone's positions when they are not fully observable. The primary novelty of our study is the development of an effective machine learning-based swarm interception method, capable of handling partially observed environments. The main contributions address the following research questions:

- *Can the proposed interception method operate correctly under full observation conditions?*
- *Can the same method operate in a scenario where observations are partially denied?*
- *Under what method parameters and environmental conditions does the proposed approach operate well?*

These questions were answered by performing experiments on simulated flights. A simulator, based on the one described in [19] was used to evaluate the proposed interception model. A comparison was made between runs using full observations and runs made with partial observations by measuring their collision rates. In addition, an investigation was made into the impact of 1) different swarm sizes, 2) parameters controlling different aspects of the simulated environment, and 3) critical parameters of the prediction and interception methods. The results obtained from running over 165,000 simulated flights demonstrate that the proposed method is capable of successful interception in both fully and partially observed environments. Finally, for certain environmental settings, larger swarms under partial observation yielded comparable performance to that of a one-drone swarm operating under full observation of the adversarial drone.

2 Prior Work

Autonomous agents can be leveraged in order to successfully detect and intercept a target [20, 21]. Effort has been made to understand this pursuer-evader problem. Originally, this problem was cast in the world of game theory [2], where the pursuer and evader both have partially observed information, and have opposing goals of evasion and interception. Similarly, such a problem context has been discussed as a partial information Markov game [22]. In partially observed environments, probabilistic trajectory distribution maps have been combined with novel, spline-based, strategy generation methods to provide interception planning [5]. Methods from control theory, like the application of the Hamilton-Jacobi-Bellman equations [23, 4], have been used to generate containment strategies for swarms, even providing methods of interception when the adversary has faster UAVs, given a sufficient number of 'blue' drones in the intercepting swarm [3].

The work presented in this paper can be regarded as a continuation of the one presented in [19]. While [19] focused mainly on trajectory prediction and correction, this paper futher extends the problem by utilizing the aforementioned trajectory prediction to solve the problem of adversarial drone interception. Other works exist that operate under the same assumption of minimal computational and sensor payload require-

ments; the work in [6] being one such example, utilizes simple stereo optical sensors, combined with a novel image segmentation method. Such work demonstrated sufficient tracking capabilities for interception-based tasks.

The ability to predict the trajectory of observed UAV swarms is an important ability to have when playing these purser-evader games. In the context of a single drone, [24] shows that it is possible to leverage deep learning to map observations of an adversary to control a UAV. Again, in the context of a single drone, [25] shows that it is possible, in indoor race environments, to directly control the flight trajectory of a single drone via a combination of convolutional neural networks and a path planning system. Additionally, dynamic models from control theory [26, 27] of quadrotor UAVs, based on neural networks, demonstrate the use of an offline framework for predicting a UAV's trajectory over time without knowing its control system [28]. While these efforts are all novel, they operate in a context different from the goals of this work. The number of training samples, assumptions of hardware availability, and the desire for an online framework preclude directly modifying these attempts.

Kalman filter [18] can also be used for predicting the adversarial UAV's trajectory. Utilizing the Kalman filter would require the current position estimate of an adversarial UAV; however, a closed-loop Kalman filter also depends on additional sensor measurements to infer the current from the previous adversarial UAV positions. It is possible to use the Kalman filter in an open-loop setting without taking the additional sensor measurements into account; however, the uncertainty of prediction will grow continuously throughout time. In the partially observed setting, we operate under the assumption of zero sensory measurements of the adversarial UAV. Not only do we not assume the presence of sensors capable of measuring an adversarial UAV's speed in the partially observable settings, but we also regard the variance, mean, or even the distributions under which the environmental effect functions as begin unknown. Moreover, constant velocity and constant acceleration Kalman model would be insufficient since the PIDs directly control UAVs' acceleration in the current simulation system. In addition, setting up the dynamic matrix requires great knowledge of the underlying physical system in many cases.

Operating in partially observed environments has been approached in other work, using bearings to known landmarks for simultaneous localization and mapping (SLAM) [29] and using received signal strength of radio frequency observations for intra-swarm localization [30]. In our work, landmarks and signal strength were not used to estimate a drone's current positions; instead, effects of environmental influences were modeled by learning from an adversarial drone's behavior, as observed during the initial part of the flight.

3 Methodology

3.1 Problem Statement

Formally, a swarm $\mathcal{S} = \{\mathcal{D}_1, \ldots, \mathcal{D}_D\}$ is defined as a collection of D individual drones. If we define the blue swarm, $\mathcal{S}_F$ as friendly, and the red swarm, $\mathcal{S}_A$ as adversarial, the interception task can be formulated as finding a function $f : \mathbb{R}^{\omega_I \times 3} \times \mathbb{R}^3 \rightarrow \mathbb{R}^3$, that maps the ω_I-sized window of past positions of an adversarial drone and the positions of friendly drones, at timestep t, to an estimated position at which the friendly swarm might intercept an adversarial drone. In other words, if $\boldsymbol{P}_F^t$ denotes the blue drones' positions at timestep t, and if $\boldsymbol{\omega}_A^t = [\boldsymbol{p}_A^{t-\omega+1}, \ldots, \boldsymbol{p}_A^t]$ denotes the adversarial drone's ω_I-sized window of past positions at timestep t, then the estimated point of interception at timestep $t+1$ is given by $f(\boldsymbol{\omega}_A^t, P_F^t) = I_{\mathcal{S}_A, \mathcal{S}_F}^{t+1}$.

However, handling estimate updates at each timestep poses a challenge since, in addition to finding the point of interception, it is necessary to provide the state updates for each friendly drone w.r.t. the estimated point of interception.[1] This is a non-trivial problem, since the accumulation of state updates can cause the friendly swarm to overshoot the estimated point of interception or to oscillate about that point considerably. This problem is similar to that of setting the proportional, integral, and derivative terms of a PID controller. PID controller is often used when there is a need to drive a system towards a target posi-

[1] We define the drone's state at timestep t as that drone's position, velocity, acceleration, and current target.

tion or a specific target level. The PID control system achieves that by using a control loop feedback mechanism to control the underlying process variables. The basic form of a PID controller is given by Equation 1, where the error term is defined as $e = \text{Target} - \text{Input}$ and O is the output from the PID controller. For a more comprehensive analysis of the PID control system, we refer the reader to [15].

$$O = K_P + K_I \int e(t)\,dt + K_D \frac{d}{dt} e(t) \tag{1}$$

Interception can be seen as PID tuning and setpoint updating, for which predicting the interception point is required. For this reason, we have decided to define a computationally inexpensive interception module that utilizes the drones' onboard PID controllers to update the swarm's state. The friendly drones' PID controllers' output limits are updated in each timestep, and the output and the integral term are clamped to avoid integral windup. Furthermore, in each timestep, the PID setpoint is set to the latest estimated point of interception with respect to the friendly swarm's structure. For a better understanding on how PID and interception are integrated, we refer the reader to Section 3.2.

3.2 Interception Model

The path prediction method uses the observed (or estimated) positions of an adversarial drone to determine the probable trajectory of that drone. A window of ω_I past positions are tracked, thus creating an $\omega_I \times 3$ matrix, $\boldsymbol{\omega}_A^t$, for which the first principal component is obtained. The first principal component corresponds to a direction in space along which projections of points of the previous ω_I positions have the highest variance. We hypothesize that such a path estimation method is useful for automated drones, which tend to follow a near-linear trajectory offset by environmental deviations. Finding the first principal component of $\boldsymbol{\omega}_A^t$ can be posed as a maximization of the Rayleigh quotient:

$$R(\boldsymbol{\omega}_A^t, v) = \sum_{i=1}^{\omega_I} (\boldsymbol{p}_A^{t-(i-1)} \cdot v)^2$$

$$\arg\max_v R(\boldsymbol{\omega}_A^t, v) = \arg\max_{\|v\|=1} \|\boldsymbol{\omega}_A^t v\|^2 = \arg\max_v \frac{\|\boldsymbol{\omega}_A^t v\|}{\|v\|}$$

Consequently, at a given timestep, our interception framework assumes that this principal vector represents the most likely linear path of the observed swarm in subsequent timesteps.

The interception point is then estimated by projecting the center of the friendly swarm to the line corresponding to the estimated path of an adversarial drone. When the friendly swarm is far away from the adversarial drone, and when environmental deviations are significant, the predicted interception locations will tend to oscillate around the actual interception point. On the other hand, as the friendly swarm approaches the adversarial swarm, the oscillations significantly subside. This behavior of the interception model can be seen in Figures 3a and 3b, where the predicted interception points are depicted as magenta and black points for full and partial observations of the adversarial drone, respectively. Ten most recent predictions for the interception points are shown, with more opaque points corresponding to more recent predictions.

The friendly drones' onboard PID controllers complement the above mentioned interception method since, as friendly drones approach the mean estimated interception point, the PIDs will cause drone accelerations and velocities to oscillate about zero, due to the evenly spread predicted interception locations. This method thus results in a friendly swarm reaching the expected interception point followed by its slight repositioning as the friendly and adversarial swarms draw closer and as the interception prediction becomes closer to the ground truth.

Figure 1 emphasizes this interplay between the PID controllers, interception, and path prediction models in the form of a control scheme. It is important to note that the PID within the simulator is responsible for controlling the UAVs' acceleration, a^{adv} for adversarial's, and $\boldsymbol{a}$ for friendly swarm's acceleration values in a given timestep. The friendly swarm has PID controllers with

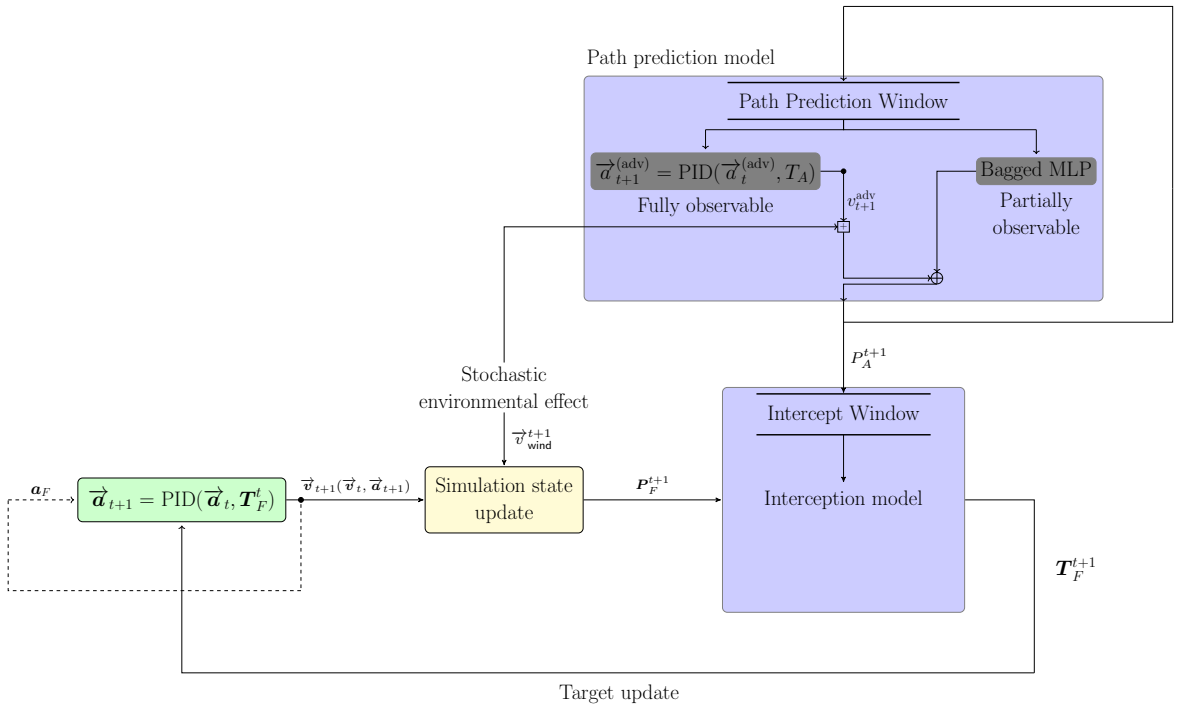


Fig. 1: Interception control scheme. The output from the path prediction model depends on the setting, which is indicated by the $\oplus$ symbol. Stochastic environment also affects the adversarial UAV's final velocity vector, as indicated by the $\boxplus$ symbol.

changing targets over time, $\boldsymbol{T}_F$, while the adversarial UAV's target, T_A, is static. A more realistic setting would, however, include a low-level, UAV-dependent PID, which derives the acceleration by means of controlling the rotor speed and whose output is fed further into the control scheme. This UAV specific implementation was abstracted, and low-level controllers were not used.

Based on Figure 1, the output of friendly UAV PIDs is fed into the simulation model state update since the drafting effect depends on the swarm's collective position at a given time. Furthermore, the path prediction model is used in either the fully observable setting or a partially observable one, based on which the adversarial UAV's position is returned. When operating in a fully observable setting, the interception model receives the actual adversarial position as obtained by its latent PID and environmental deviation model. In a partially observable setting, the bagged MLP regressor's output serves to estimate the final adversarial UAV positions. The interception model's output is fed back to set the new targets of the friendly UAVs' PID controllers. Also, the path prediction output is fed back to itself as the observations and estimates need are saved in a window history.

3.3 Advantages of larger swarms

Let ϵ be the radius of each drone. We will define a collision of two drones as a point in time in which the distance between the centers of the two drones is less than 2ϵ. Moreover, if a friendly swarm, $\mathcal{S}_F$, has a planar structure that is defined by $x \in (\ell_1, \ell_2)$ and $y \in (f_1(x), f_2(x))$ then:

$$p(x) = \int_{max\{c_y - \sqrt{r_e^2 - (x-c_x)^2}, f_1(x)\}}^{min\{c_y + \sqrt{r_e^2 - (x-c_x)^2}, f_2(x)\}} \mathrm{d}y$$

$$E[X] = \frac{1}{r_e^2 \pi} \int_{max\{c_x - r_e, \ell_1\}}^{min\{c_x + r_e, \ell_2\}} p(x)\, \mathrm{d}x \tag{2}$$

where (c_x, c_y) is the center of the friendly swarm and r_e is the error of the estimated interception

point at timestep t_I, and X is a random variable that takes the value of 1 if interception occurred and 0 otherwise. Here timestep t_I is defined as the timestep at which the adversarial drone passes through the plane Π, where $\mathcal{S}_F \subset \Pi$. The expectation $E[X]$ can be thought of as the probability of achieving interception using a swarm whose structure is described with ℓ_1, ℓ_2, $f_1(x)$, and $f_2(x)$, and with an interception point estimate error of r_e at time t_I. In Equation 2, an implicit assumption was made that the friendly swarm has sufficient time to reach the estimated interception point at timestep t_I. Such assumption might not always hold, as it depends on how far in advance the adversarial UAV was spotted, the capabilities of both friendly and adversarial swarms, as well as the distances between the adversarial UAV's target, friendly swarm, and the adversarial UAV itself. In addition, environmental deviations can also impact whether it will be possible to reach the interception point in time. The assumption of this paper is that friendly and adversarial UAVs have similar performance in addition to observing the adversarial UAV sufficiently far in advance.

In particular, let us consider a friendly swarm, $\mathcal{S}_F$ with a square-like planar shape. If d represents the minimum distance of each drone to their neighboring drone in the swarm, the sides of a square planar swarm consisting of n^2 drones have a length of $(n-1)d + 2\epsilon$. Separation parameter, d, should generally be set to the greatest possible value that prevents friendly swarm's mesh penetrability by the adversarial drone. Therefore, the separation parameter, d, should be regarded as a function of ϵ and the friendly swarm's planar structure. If we denote the coordinates of the drone with the lowest x and y coordinates as the origin, then $\mathcal{S}_F$ has a center of $(\frac{(n-1)d}{2} + \epsilon, \frac{(n-1)d}{2} + \epsilon)$. The friendly swarm, $\mathcal{S}_F$, is then defined by $\ell_1 = 0$, $\ell_2 = (n-1)d + 2\epsilon$, $f_1(x) = 0$, and $f_2(x) = (n-1)d + 2\epsilon$.

Therefore, it is possible to evaluate Equation 2, conditioned on knowing the interception point estimate error r_e. For $d = 0.5$, $\epsilon = 0.15$, $E[X]$ is evaluated for different values of r_e, using swarms of different sizes, and the results are depicted in Figure 2. We can observe that there is an evident improvement in the interception performance as more drones are added to $\mathcal{S}_F$.

It is important to mention that directly evaluating Equation 2 is not feasible in most cases, for which no prior knowledge of the introduced interception point estimate error, r_e, exists. However, $E[X]$ can still be evaluated empirically, using multiple simulations performed under similar environmental conditions, as shown in the experimental section of this paper. The interception point estimate error, r_e, consists of both the interception model error, which is in turn also dependent on the path prediction model error. From Figure 4, we can notice that as the path prediction error decreases, by having more timesteps for training, $E[X]$ increases, which corresponds with the trend of $E[X]$ shown in Figure 2.

3.4 Adversarial Drone Path Prediction

The interception model requires a window of an adversarial drone's previous positions in order to estimate an interception point. However, when full observations of the adversarial drone's positions are not available, the interception model requires that those positions be estimated. Due to the assumption of low on-board computing power, as well as the assumption that a relatively low number of training steps are available for learning, deep learning models were not considered for such estimation.

Instead, a Multi-output regressor consisting of small multilayer perceptron (MLP) regressors with one hidden layer of 30 units is selected as our base model [31]. The MLP regressor uses a ω_M-sized window of previous positions of the adversarial drone as its inputs when constructing the training data. Five of these Multi-output regressors were bagged with bootstrap aggregation [7], and the mean prediction was used as the adversarial drone's position estimate when those positions are not available. The MLP is trained using L-BFGS optimization, with a regularization parameter $\alpha = .015$. This model is similar to the unstructured regressor introduced in our related study aimed at inter-swarm collision avoidance in a GPS-denied environment [19].

The training outside of the simulation would be performed in a distributive fashion, with model sharing between drones. If the training to be performed is longer than one simulation tick, dead reckoning would be used until the training is complete. In the current simulation environment, the assumption is that there is enough computational power shared among drones to complete the train-

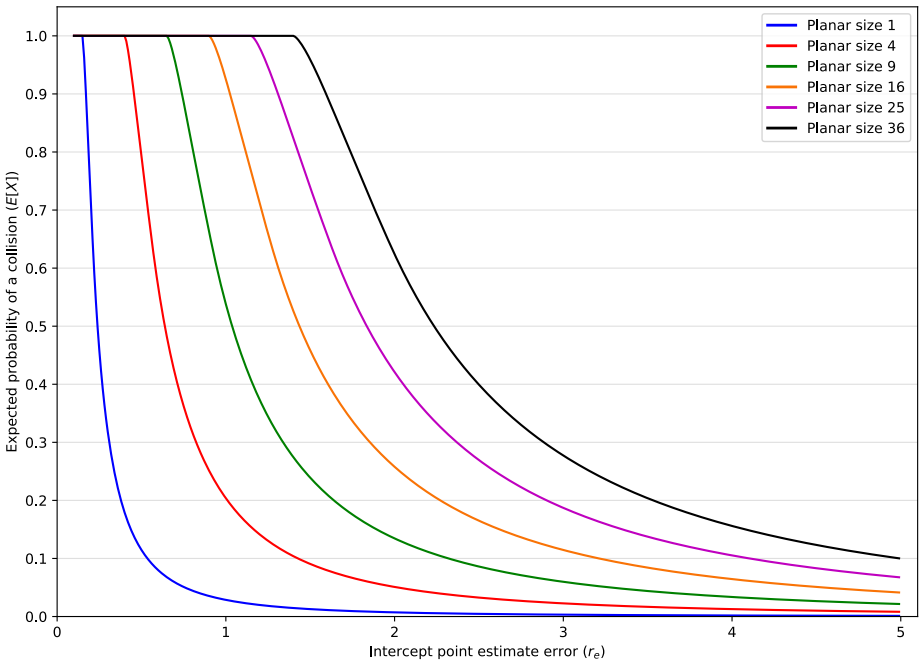


Fig. 2: Comparison of the modeled interception probability using swarms of different sizes. The expected probability of a successful interception is represented on the y-axis, whereas the x-axis represents a particular value of the interception point estimate error, r_e. Equation 2 was used to evaluate the expected probability of an interception, $E[X]$.

ing within one simulation tick.

4 Results

Extensive experiments were conducted to investigate the effectiveness of the proposed prediction and interception method. Additionally, an investigation was made into the parameter space of the environmental model and the impact that various settings have on the ultimate behavior of the interception model. This investigation was done to provide additional support to the hypothesis that the interception model benefits from leveraging greater-sized swarms. In the case of such swarm sizes, the friendly swarm's structure would absorb the spatial error introduced by the interception and path prediction methods, leading to a higher interception accuracy, as we have empirically shown.

4.1 Simulation process

Each class of experiments makes use of the bespoke simulated drone environment, similar to the one described in [19]. This environment can model the trajectories of swarms composed of an arbitrary number of drones. Swarms can be given target locations and are able to host the models described above. A random wind model is used to provide environmental deviations within each run. Such a wind model contains parameters that control distributions governing the intensity, direction, length, and gust frequency. Specifically, the polar and azimuthal angles of wind gusts are sampled from a Gaussian distribution $\theta \sim \mathcal{N}(\mu_\theta, \sigma_\theta)$ and $\phi \sim \mathcal{N}(\mu_\phi, \sigma_\phi)$, respectively; the initial wind settings are set to $\mu_\theta = \mu_\phi = -\frac{\pi}{6}$, and $\sigma_\theta = \sigma_\phi = \frac{\pi}{24}$ radians. The period between gusts, t_{calm}, and the duration of gusts, t_{gusting} are sampled as $t_{\text{calm}} \sim \mathcal{N}(\mu_{\text{calm}}, \sigma_{\text{calm}})$ and $t_{\text{gusting}} \sim \mathcal{N}(\mu_{\text{gusting}}, \sigma_{\text{gusting}})$, where $\mu_{\text{calm}} = \mu_{\text{gusting}} = 6$ and $\sigma_{\text{calm}} = \sigma_{\text{gusting}} = .8$ simulation ticks, by default.

In addition, the wind model is supplemented by a drafting effect, in which drones within the

swarm's structure benefit from experiencing less intense winds than the drones in the outer layers of the swarm that are exposed to the wind. The layers are calculated by recursively considering parts of the convex hull that are directly exposed to the wind, and each successive layer experiences a 10% reduced effect of wind.

The maximum velocity of both friendly and adversarial drones is set to a value of one, and the maximum acceleration is set to a value of two for each direction, for a unit of time. It is assumed that the adversarial drone is moving at full velocity at the beginning of each simulation while the friendly swarm is static. Each drone is considered to have a radius of 0.15. Unless specified differently, a the wind magnitude is sampled as $\mathcal{N}(\mu_w, \sigma_w)$, where $\mu_w = 1$ and $\mu_w = .3$ and applied whenever gusts occur within the simulation. For all simulations, only swarms with a simple planar structure were considered, with a separation between rows and columns of UAVs of 0.45. However, there is no reason to assume that different spacial configurations, aimed towards the task of interception, would not yield similar results. Changing the initial wind angle settings did not significantly change the interception performance due to the path prediction model's in-flight training.

For each unit of time, 20 simulation ticks are performed. The window size used for path prediction, ω_M, and the window size used for interception direction and point estimate ω_I are both initially set to a value of 10 simulation ticks.

4.2 Interception Method Evaluation

In the first set of experiments, a comparison was made between two settings, the difference between them being the type of input provided to the model. In the full observation scenario, the friendly swarm is considered to have perfect observations of the adversarial swarm's positions at all timesteps. In the partial observation setting, the friendly swarm has perfect observations of the adversarial swarm, but only up to a certain critical timestep $t < t_c$, after which it must rely on the outputs of the prediction model. It is useful to note that the interception and path prediction models would work even if we observed an adversarial UAV's true positions briefly or even intermittently after the critical timestep. In a realistic, partially observable setting, such a use case might become useful since adversarial UAV observations might not be unavailable for extended periods, even in urban or occluded environments.

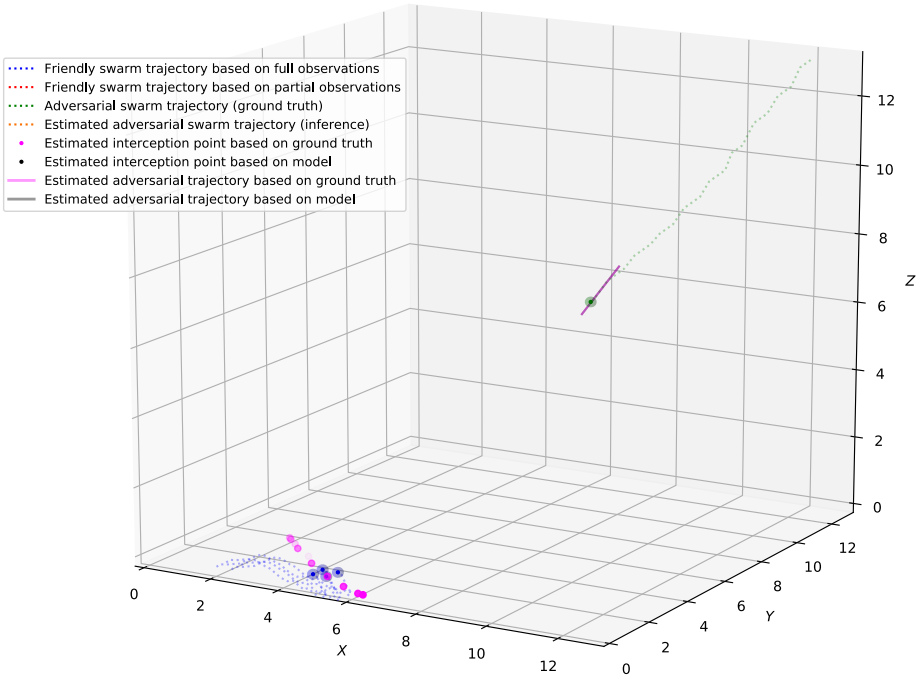
Nevertheless, in all experiments, a blue swarm is attempting to intercept a single, adversarial drone whose positions are entirely unavailable after t_c. An example run of the simulation is shown in Figure 3.

In addition, the current simulator includes a UAV model with a spherical area of influence centered about each UAV. A more general UAV model should account for differently spaced areas of influence and would likely lead to a model in which the center of an ellipsoid does not correspond to the actual UAV location due to its greater front-side influence. Under such assumption, we would need to split the interception point estimate error into its spatial components and look at the projections on the plane spanned by the friendly UAVs. Such a projected ellipsoid would be superimposed over the area of influence of friendly UAVs, and the obtained intersection would be the possible interception area. In case that this intersection is an empty set, no interception would occur. The proportion of such area to the area in which the adversarial UAV might intersect the plane spanned by friendly UAVs provides us with the probability of intersection.

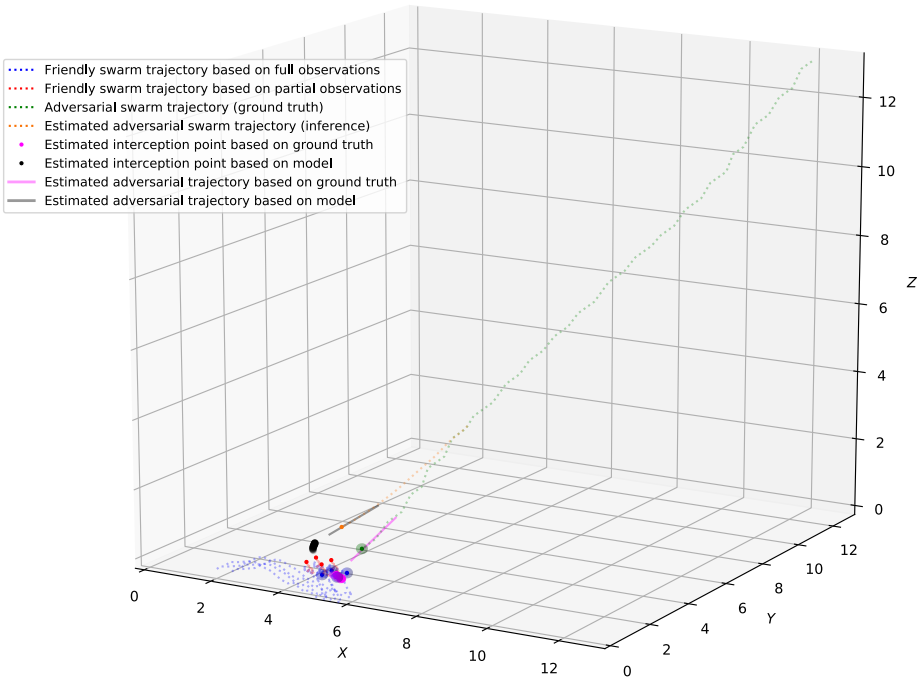
Using such a general interception model is computationally more involved and includes additional parameters that account for differently spaced areas of influence. These changes would require adjusting the simulator, and therefore, such a general model was not used in the current experimental setup.

4.2.1 Full Observation Scenario

In this setting, the friendly swarm is considered to have perfect observations of the adversarial swarm's positions at all timesteps. This experiment serves as a baseline evaluation of the interception procedure itself and helps to determine and compare the interception efficacy using swarms of different sizes. Table 1 reports the interception accuracies, mean collision timesteps, and the standard deviation of collision timesteps for swarms of size $\{1, 4, 9, 16\}$ in a square planar configuration. Hundred simulated flights were run, for each swarm size, to evaluate the percent of



(a)



(b)

Fig. 3: Example of a simulation run during (a) the early phases of the simulation and (b) late stages of the simulation. Both the actual (fully observed) trajectory of the adversarial drone, as well as its inferred trajectory, are depicted in green and orange, respectively. The movement of the friendly swarm when relying on full observations is provided by a blue line, while the red line indicates the friendly swarm's trajectory under partial observation.

Table 1: Percent of runs with successful interceptions when full observations were available to friendly swarms of 1, 4, 9, and 16 drones.

Swarm Size	1	4	9	16
% of successful interceptions	93	100	100	100
Mean interception timestep	207.46	205.05	202.78	200.48
Standard deviation of interception timesteps	3.98	4.29	4.07	4.14

successful collisions.

4.2.2 Partial Observations Scenario

The second experiment proceeds in an identical manner but introduces denial of an adversarial drone's location feedback. In this setting, after timestep t_c, the friendly swarm is given no information about the adversarial drone's actual positions and, instead, must rely on the outputs of the prediction model to intercept it. During the inference steps, i.e., in timesteps $t > t_c$, the inputs to the interception model are the predicted, instead of the actual, positions of the adversarial swarm. Figure 4 presents the percent of successful interceptions over 100 runs, for different values of t_c, starting from $t_c = 50$, to $t_c = 210$, with a step of 5, for a total of $3,300$ simulations per swarm. The same set of swarm sizes as in the first experiment is considered.

An important observation is that the experimental results, shown in Figure 4, follow the modeled interception probability proposed by Equation 2 and presented in Figure 2. As the prediction model uses more timesteps for training, and as the inference window shortens, the introduced interception point estimate error decreases - which is consistent with the findings from Figure 2.

4.3 Parameter Space Exploration

In addition to investigating the efficacy of the proposed methods, an effort was made to explore the impact of environmental and interception model parameters on the outcomes of interception experiments. The parameters governing the wind intensity, wind duration, and interception sensitivity were investigated.

4.3.1 Wind Model Parameters

The simulator utilizes a random wind model to simulate environmental deviations. The basis of this model is a set of governing parameters that control the probability distributions for defining elements of a gust of wind: its intensity, duration, direction, and frequency. Two sets of experiments were run, investigating the effect of varying the parameters for intensity and the parameters for the wind duration. The effect of varying the wind direction parameters was not investigated since swarms have a square planar shape, and thus, due to symmetry, it is expected that varying the wind direction would have little effect on the underlying interception accuracy.

Investigation of Wind Intensity Parameters

In this experiment, the mean and variance governing the intensity of the wind distribution were varied, and observations were made of the resulting interception accuracy. Figure 5 reports the results, in which the horizontal axis represents the standard deviation of the wind intensity, while the vertical axis represents the mean wind intensity. The percentage of runs in which interception occurred is described by heat-maps generated using Gaussian interpolation. The same range of values was selected for testing both the wind intensity means and the wind intensity standard deviations, with values ranging from 0 to 1, with a step size of 0.05, for a total of 441 different combinations. For each tested combination 100 simulations were run, resulting in a total of 44,100 simulations. Only swarms of size 1 and 16 were selected for testing due to the time-consuming nature of this experiment.

The negative effect of the wind intensity is evident when the prediction model is run on a friendly swarm consisting of a small number of drones. Wind with a high mean intensity causes significant deviations from the expected linear trajectory, which the adversarial drone is assumed to follow. Consequently, the introduced error results in a limited interception success when full observations are unavailable, and when small-sized swarms are utilized under strong wind conditions.

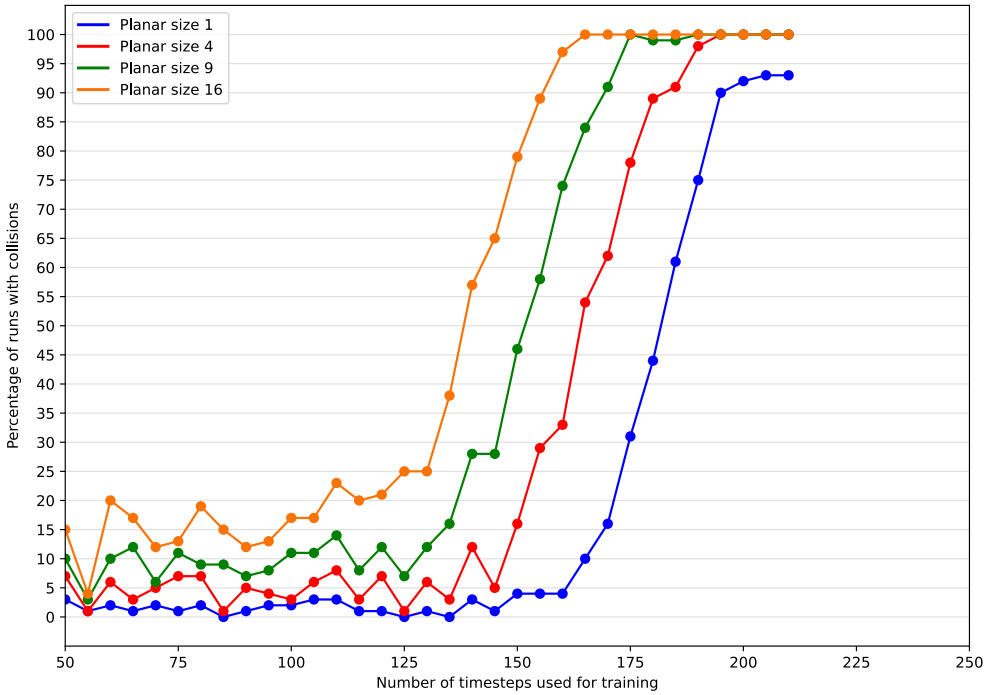


Fig. 4: Interception accuracy comparison in a partially observed scenario. Blue, red, green, and orange lines correspond to friendly swarms of size 1, 4, 9, 16, respectively. The percentage of runs with successful interceptions is represented by the y-axis, whereas the x-axis represents a particular value of t_c from which locations of the adversarial drone were unavailable.

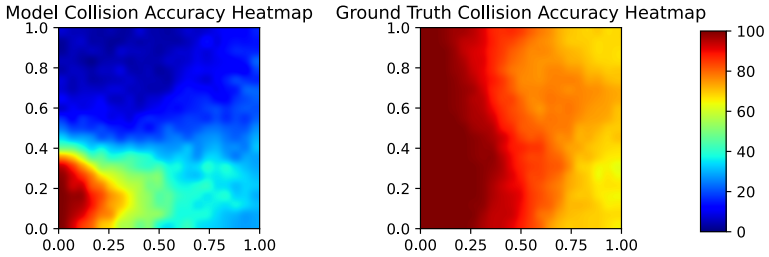
However, as observed from Figure 5, the interception accuracy heat-map of a one-drone swarm, under perfect observation, is comparable to that of a swarm consisting of 16 drones under partial observation. This serves as an indicator that adding more drones to the friendly swarm can help overcome the above-mentioned negative effect of wind on the interception accuracy.

Investigation of Wind Duration Parameters
Similar to investigating the wind intensity parameters, an experiment was run where the values of the mean duration and the variance of gusts were perturbed. The wind duration mean was varied from 0 to 20, with a step of 1, and the wind duration variance was varied from 0 to 10, with a step of 0.5. Consequently, 441 different combinations were tested, and 100 simulations were run for each combination. However, since perturbing the variance in the given range had no effect on the percentage of successful collisions, only the graph describing the relation between the interception accuracy and the wind duration means, for a one-drone swarm, is depicted. Figure 6a reports these results.

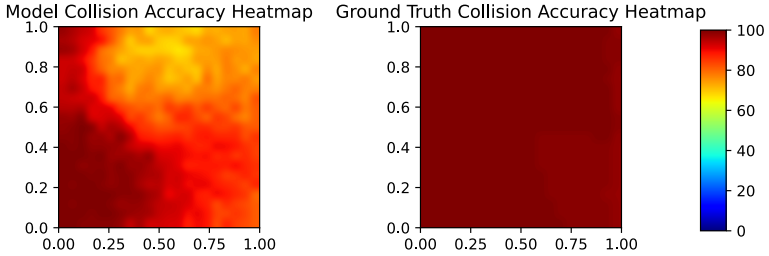
4.3.2 Interception Model Parameters

The main governing parameter of the interception model is the window size, ω_I, used for calculating the principal component vector that corresponds to the estimated trajectory of an adversarial drone. Experiments were run using interception windows with previous points of different sizes. Windows of size $\omega_I = 2$, up to $\omega_I = 50$, with a step of 1, were tested. The mean duration of gusts and the mean period between gusts was set to 6 timesteps. In order to compare the effect of different interception window sizes on swarms of different sizes, the number of timesteps used for training, t_c, was set to a value such that the swarm has an interception accuracy of nearly 50% when $\omega_I = 10$, for each swarm respectively. For each choice of ω_I, 100 simulations were run per swarm type, with a total of 4,900 simulations per swarm type. The results of these experiments are depicted in Figure 6b.

From Figure 6b, we can observe that the interception accuracy has a local maximum for $\omega_I \approx 5$, and a local minimum for $\omega_I \approx 13$, after which the interception accuracy rises gradually. These val-



(a) Swarm of size 1



(b) Swarm of size 16

Fig. 5: Comparison of wind intensity parameters on swarms of size (a) one and (b) 16. The x-axis values correspond to values of standard deviations of wind intensity, while the y-axis values correspond to mean values of wind intensity. The left column contains a heat-map generated under partial observations, while the right column contains a heat-map generated under full observations of the adversarial swarm.

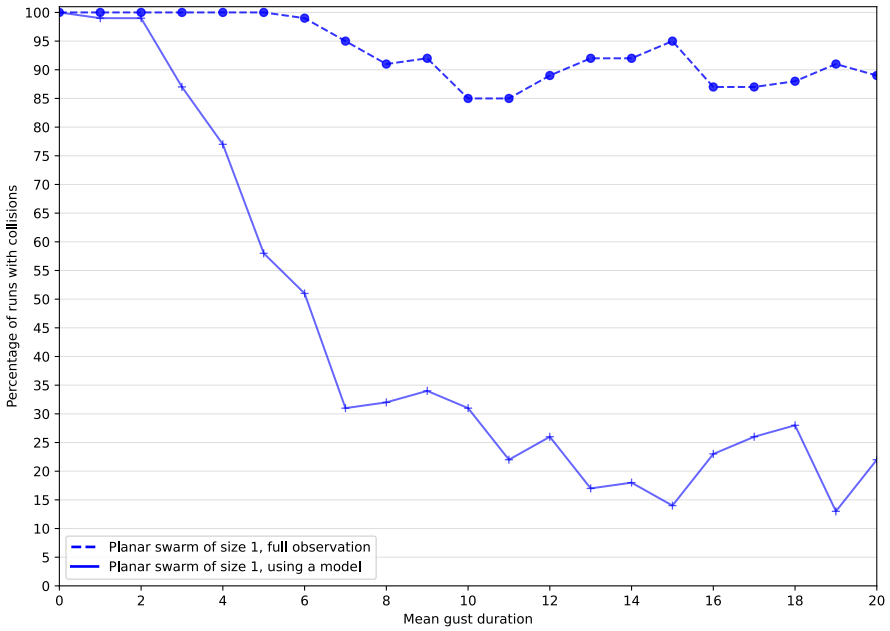
ues are related to the mean gust duration and the mean periods between gusts. These results show that in order to maximize the probability of a collision, under the current problem settings, ω_I can be set to either: 1) a large value, several times greater than the mean gust duration or mean period between gusts, or 2) a small value closely related to the mean gust duration and mean period between gusts. Optimization of ω_I would include collecting the simulated flight data in an offline setting, under expected environmental conditions of the friendly UAV to obtain an estimate of collision probabilities w.r.t. intercept window and friendly swarm size settings, ω_I and D, respectively, as shown in Figure 6. The function estimate is saved on the friendly UAV model, which sets the maximum allowable ω_I based on the adversarial swarm's observed distance. Then, the optimal window size ω_I is obtained by performing an optimization over the function estimate constrained on the maximum intercept window size.

This finding can be used to the friendly swarm's advantage when intercepting. If the friendly swarm observes the adversarial swarm sufficiently far in advance, a large value for ω_I can be used. However, large ω_I can also cause late predictions by the interception module, as the interception window requires a greater number of previous adversarial drone's positions. In such cases, setting ω_I to a small value that corresponds to the local maxima, similar to the one in Figure 6b, is preferred. Notably, the findings observed in Figure 6b assume an adversarial drone with a near-linear trajectory.

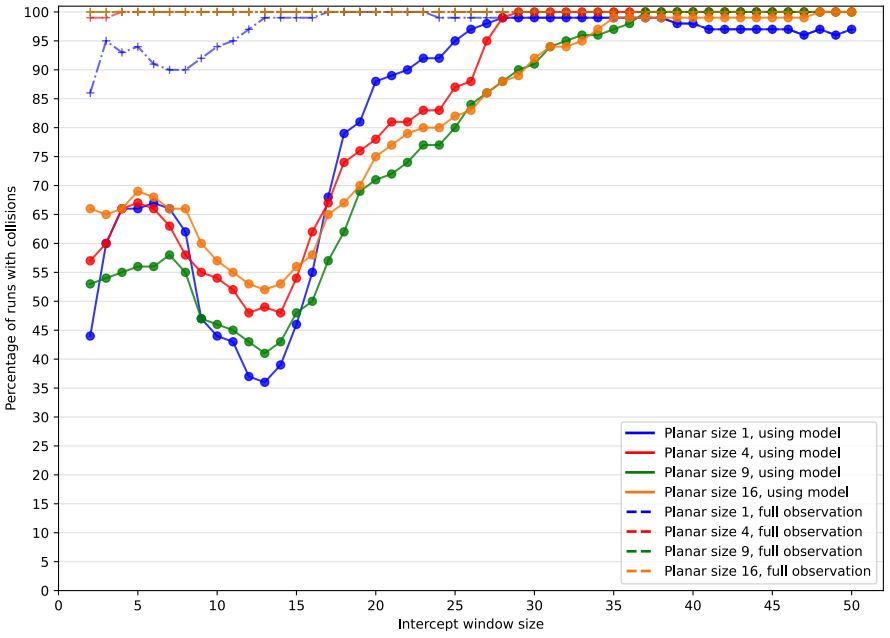
4.4 Extension to latent adversarial target selection and larger adversarial swarms

The aforementioned experimental section assumed a single adversarial UAV moving in a near-linear fashion, offset by environmental deviations. The severity of deviation of an adversarial UAV's trajectory from a linear one depended purely on the harshness of the environmental conditions. However, a more applicable scenario would likely include a swarm of adversarial UAVs in which the swarm consists of more than a single drone, in addition to allowing for non-linear adversarial trajectories. Also, such flight paths are often highly dependent on the adversarial swarm's target.

Previously, the adversarial UAV's target was fixed throughout all experiments in such a way



(a)



(b)

Fig. 6: Percentage of runs with successful interceptions w.r.t a) wind duration means and b) different choices for interception window size. In a partially observed scenario, when the model is used to infer an adversarial drone's path, the percentage of runs with successful interceptions is depicted by solid lines, while dashed lines were used when full observations of the adversarial drone were available.

that prevents an interception occurring by chance, considering the initial friendly swarm's movement direction. However, a more appropriate setting would allow for multiple probable and known adversarial swarm targets, in addition to having a mechanism of latent adversarial target selection.

Given that the model described in this paper is highly modular, we can easily exchange and add specific components that would account for such, more general behavior of an adversarial UAV swarm. Experiments on such an extended problem were not performed due to the significant computational complexity of running numerous simulations under vastly different environmental settings. Such complexity stems from the requirement of performing structured learning in addition to training a per-drone regressor, in the case of larger adversarial swarms. In the case of latent adversarial target selection, an offline data collection and classifier training step would be required, along with training a regressor conditioned on those targets.

4.4.1 Larger adversarial swarms

Given an adversarial swarm that maintains a particular internal structure between drones over time, the current model would likely prove insufficient for the task of interception. In [19], a swarm trajectory prediction task under partial observations was resolved using structured learning. Similarly, a Multi-Target Gaussian Conditional Random Fields (MT-GCRF) model can be used to predict the individual locations of adversarial UAVs under partial observations while simultaneously accounting for their spatial structure. Alike the model in this paper, MT-GCRF also utilizes a window of previous adversarial UAV positions, ω_M. The underlying regressor used by MT-GCRF can be any parameterized function that outputs the current UAV location estimate given its window of previous positions, ω_M. As a result, MT-GCRF can also be used when non-linear adversarial trajectories are observed by utilizing more elaborate models.

4.4.2 Latent adversarial target selection

In a more practical setting, adversarial UAV flight paths are often highly dependent on the swarm's underlying target. Such targets are usually latent from the friendly swarm's perspective; however, we are often aware of potential targets in a specific area. Utilizing powerful supervised classification algorithms to infer the actual adversarial swarm target could prove highly beneficial when combined with a trajectory prediction model. Training such classification algorithms would be done offline, after collecting various adversarial flight data in similar-behaving simulated environments. A critical constraint placed on such classification algorithms would be having fast inference times as they are employed in fast-paced environments.

For example, in [8], the authors introduced a Neural Dynamic Classification (NDC) algorithm, which utilizes the available training data to learn a transformation function that transforms the data such that the probability of correct classification is increased. Since the parameters of such transformation functions are established in a way that creates large margins between clusters and small margins between classmates, such an algorithm is also useful for inspecting what adversarial modes of behavior would lead to a particular target selection. In a scenario where the window size, ω_M, is large, inference could turn out to be slow even when accounting for the friendly swarm's distributed computing power; therefore, NDC works best on relatively small window sizes. Similarly, enhanced PNNs (EPNN) [9] could be utilized, when used in conjunction with small window sizes. EPNN's classification power over different signal-to-noise ratio (SNR) levels and train-to-test data ratios is especially useful if the model is to be used in unknown environments. Finally, if explainability and fast model updates are prioritized, a finite element machine for fast learning (FEMa) [10] can be used. FEMa has the advantage of evaluating the certainty of classification and, consequently, computing a probability map during the inference phase. In addition to being non-parameterized, FEMa might also not require a training step if interpolation and partition of unity assumptions are imposed on the basis functions, thus allowing for fast model updates.

5 Conclusion

In this study, an interception method that utilizes on-board PID controllers was proposed to intercept an adversarial drone under both full and partial observation of the adversarial drone's positions. The efficacy of the method was assessed in a virtual environment under different environmental and model settings. When simulated in over 165,000 flights, the proposed method was able to achieve successful interception in the baseline

and partially denied scenarios. Additionally, a parameter space exploration was conducted, the findings of which supported the hypothesis that the friendly swarms of greater sizes increases the efficacy of the proposed method. These experiments reveal an interesting regime for the window size parameter, suggesting possible criteria for improving the interception efficacy. Future work will focus on intercepting smart, interactive adversarial agents using a more general interception UAV model.

Acknowledgements

This research was supported in part by the AFRL Contract No. FA8750-18-C-0041, subcontract 555027-78056.

References

[1] Tassey M, Perkins R. Wireless aerial surveillance platform. In: DEFCON Conference, Las Vegas, NV; 2011.

[2] Basar T, Olsder GJ. Dynamic noncooperative game theory. vol. 23. Siam; 1999.

[3] Reimann J, Vachtsevanos G. UAVs in Urban Operations: Target Interception and Containment. J Intell Robot Syst. 2006 Nov;47(4):383–396.

[4] Sethian JA, Vladimirsky A. Ordered upwind methods for static Hamilton–Jacobi equations: Theory and algorithms. SIAM Journal on Numerical Analysis. 2003;41(1):325–363.

[5] Koyuncu E, Inalhan G. Exploiting Delayed and Imperfect Information for Generating Approximate UAV Target Interception Strategy. J Intell Robot Syst. 2013 Jan;69(1-4):313–329.

[6] Strydom R, Thurrowgood S, Denuelle A, Srinivasan MV. UAV Guidance: A Stereo-Based Technique for Interception of Stationary or Moving Targets. In: Dixon C, Tuyls K, editors. Towards Autonomous Robotic Systems. vol. 9287. Cham: Springer International Publishing; 2015. p. 258–269. Series Title: Lecture Notes in Computer Science.

[7] Breiman L. Bagging predictors. Machine learning. 1996;24(2):123–140.

[8] Rafiei M. H., Adeli H. A new neural dynamic classification algorithm. IEEE transactions on neural networks and learning systems. 2017;28(12):3074–3083.

[9] Ahmadlou M., Adeli H. Enhanced probabilistic neural network with local decision circles: A robust classifier. Integrated Computer-Aided Engineering. 2010;17(3):197–210.

[10] Pereira D., Piteri M. A., Souza A., Papa J. P., Adeli H. FEMa: a finite element machine for fast learning. Neural Computing and Applications. 2020;32(10):6393–6404.

[11] Hernández C., Bermejo-Alonso J., Sanz R. A self-adaptation framework based on functional knowledge for augmented autonomy in robots. Integrated Computer-Aided Engineering. 2018;25(2):157–172.

[12] Pérez-Hurtado I., Martínez-del-Amor M., Zhang G., Neri F., Pérez-Jiménez M. J A membrane parallel rapidly-exploring random tree algorithm for robotic motion planning. Integrated Computer-Aided Engineering. 2020;27(2):1–18.

[13] Bonet I., Caraffini F., Pena A., Puerta A., Gongora M. (2020) "Oil Palm Detection via Deep Transfer Learning," Proc. Congress on Evolutionary Computation (CEC), pp. 1–8, IEEE.

[14] Al-Kaff A., Armingol J. M., de La Escalera A. A vision-based navigation system for unmanned aerial vehicles (uavs). Integrated Computer-Aided Engineering. 2019;26(3):297–310.

[15] Ang K. H., Chong G., Li Y. PID control system analysis, design, and technology. IEEE transactions on control systems technology. 2005;13(4):559–576.

[16] Jiang S., Zhang J. Real-time crack assessment using deep neural networks with wall-climbing unmanned aerial system. Computer-Aided Civil and Infrastructure Engineering. 2020;35(6):549–564.

[17] Liu Y., Nie X., Fan J., Liu X. Image-based crack assessment of bridge piers using unmanned aerial vehicles and three-dimensional scene reconstruction. Computer-Aided Civil and Infrastructure Engineering. 2020;35(5):511–529.

[18] Kalman R. E. A new approach to linear filtering and prediction problems. 1960.

[19] Power W., Pavlovski M., Saranovic D., Stojkovic I., Obradovic Z. (2020) "Autonomous Navigation for Drone Swarms in GPS-Denied Environments Using Structured Learning," Proc. 16th IFIP WG 12.5 Int'l Conf. Artificial Intelligence Applications and Innovations (AIAI), Neos Marmaras, Greece, June 2020, Part II, pp. 219–231, Springer.

[20] Kunwar F, Sheridan PK, Benhabib B. Predictive guidance-based navigation for mobile robots: A novel strategy for target interception on realistic terrains. Journal of Intelligent & Robotic Systems. 2010;59(3-4):367–398.

[21] Wise R, Rysdyk R. UAV coordination for autonomous target tracking. In: AIAA Guidance, Navigation, and Control Conference and Exhibit; 2006. p. 6453.

[22] Hespanha JP, Prandini M, Sastry S. Probabilistic pursuit-evasion games: A one-step nash approach. In: Proceedings of the 39th IEEE Conference on Decision and Control (Cat. No. 00CH37187). vol. 3. IEEE; 2000. p. 2272–2277.

[23] Kao CY, Osher S, Qian J. Lax–Friedrichs sweeping scheme for static Hamilton–Jacobi equations. Journal of Computational physics. 2004;196(1):367–391. Publisher: Elsevier.

[24] Arola S, Akhloufi MA. UAV Pursuit-Evasion using Deep Learning and Search Area Proposal. In: Proceedings of the IEEE International Conference on Robotics and Automation; 2019.

[25] Kaufmann E, Loquercio A, Ranftl R, Dosovitskiy A, Koltun V, Scaramuzza D. Deep drone racing: Learning agile flight in dynamic environments. arXiv preprint arXiv:180608548; 2018.

[26] Franklin GF, Powell JD, Emami-Naeini A, Powell JD. Feedback control of dynamic systems. vol. 3. Addison-Wesley Reading, MA; 1994.

[27] Whitehead B, Bieniawski S. Model reference adaptive control of a quadrotor UAV. In: AIAA Guidance, Navigation, and Control Conference; 2010. p. 8148.

[28] Xue M. UAV trajectory modeling using neural networks. In: 17th AIAA Aviation Technology, Integration, and Operations Conference; 2017. p. 3072.

[29] Saska M, Baca T, Thomas J, Chudoba J, Preucil L, Krajnik T, et al. System for deployment of groups of unmanned micro aerial vehicles in GPS-denied environments using onboard visual relative localization. Auton Robot. 2017 Apr;41(4):919–944.

[30] Lockspeiser JR, Don ML, Hamaoui M. Radio Frequency Ranging for Swarm Relative Localization. US Army Research Laboratory; 2017. ARL-TR-8194.

[31] Pal S, Mitra S. Multilayer perceptron, fuzzy sets, and classification. IEEE Transactions on Neural Networks. 1992;3(5):683–697.