Learning Cooperative Strategies for Drone Swarms using Multi-Agent Reinforcement Learning

Christian Llanes[1], Kyle A. Williams[2], Spencer W. Jensen[2], and Samuel Coogan[1]

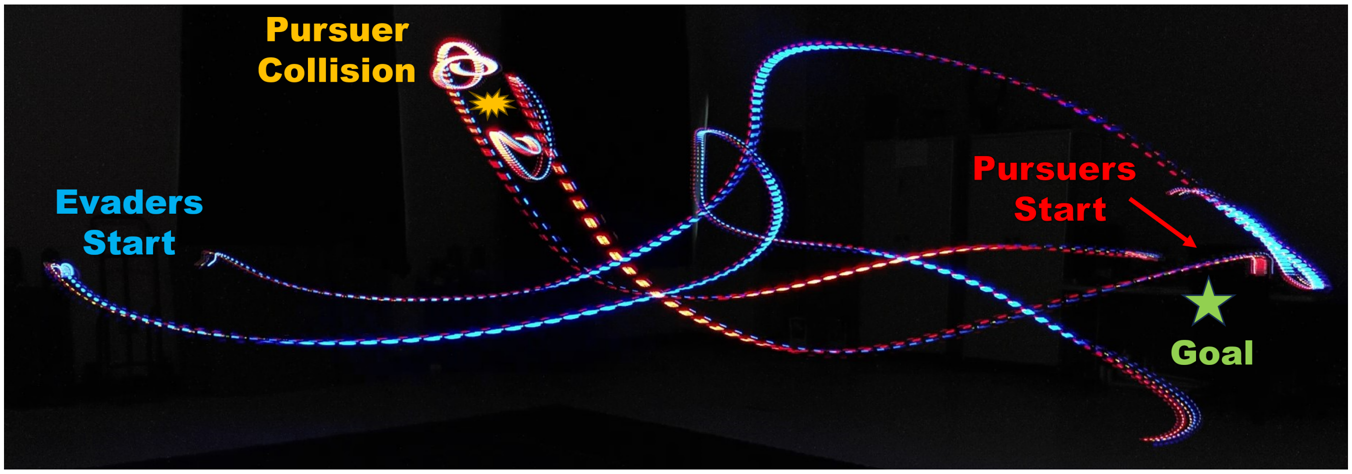


Fig. 1: A long exposure image of the hardware demonstration for a 2 vs. 2 pursuit-evasion problem. The pursuers (red) start at the goal on the right and the evaders (blue) start on the left. The pursuers attempt to intercept the evaders while the evaders use multi-agent reinforcement learning to disable pursuers before reaching the goal.

***Abstract*— In this work, we investigate cooperative strategies for an evader drone team of various sizes using multi-agent reinforcement learning in a multi-agent pursuit-evasion scenario. The objective of the evader team is to reach a goal with minimal velocity while not colliding with the pursuer team. The objective of the pursuer team is to defend the goal by catching evaders before they reach it. In this environment, we allow the pursuer to have superior control authority compared to the evader such that reaching the goal is challenging for the evader in a one-on-one scenario. The proposed strategy for an evader is to team up with an ally to lead pursuers into a collision with each other instead of intercepting the evader. We design policies using multi-agent proximal policy optimization, an actor-critic reinforcement learning method, and investigate how the learned strategy changes when we vary the size of the pursuer and evader teams. Finally, we demonstrate the learned policy's sim-to-real capabilities through a hardware demonstration.**

I. INTRODUCTION

Pursuit-evasion (PE) [1], [2] problems have been a topic of research for the past few decades. Variations of PE that have been studied include limited visibility [3], single evader and multi-pursuer with complex dynamic environment interactions [4], limited information transmission of the evader's position [5], obstacles in the environment [6], a faster evader [7], [8], a faster pursuer with limited visibility in the context of search complexity [9], and an evader using nonlinear model predictive tracking control [10]. Pursuit-evasion can be categorized into three types based on speed of pursuer and evader: a faster evader, an equally matched evader and pursuer, and a faster pursuer. The faster pursuer problem is generally limited in the literature to simple two-dimensional problems with constant velocity. However, there is a need for studying more complex multi-agent pursuit-evasion problems and especially when the pursuer is faster or has superior control capabilities compared to the evader. In the context of this paper, we refer to multi-agent as both sides in the PE problem each having more than one agent.

In this work, we set up a multi-agent pursuit-evasion problem (MAPE) where an evader team is tasked with reaching a target in an arena with a pursuer team defending the target. Additionally, each agent has six degrees of freedom. The pursuers are assumed to have a fixed strategy using the evader's position and velocity. In this work, we test two pursuer strategies – pure pursuit (PP) and augmented proportional navigation (APN). Additionally, pursuers have superior acceleration capabilities compared to the evaders. Therefore, in a one-on-one scenario it is challenging for the evaders to reach the target before being captured by pursuers. To increase evader success rate, evaders can cooperate to lead pursuers into collisions with each other before attempting to reach the target. Cooperation strategies for the evader are learned using multi-agent reinforcement learning (MARL) to lead pursuers into a collision before navigating to the goal.

We summarize our contributions as follows:

[1]School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA. Emails: {christian.llanes, sam.coogan}@gatech.edu
[2]Sandia National Laboratories, Albuquerque, NM 87123, USA. Emails: {swjense, kwilli3}@sandia.gov

1) We develop a multi-agent proximal policy optimization algorithm for the full six degree of freedom MAPE problem for drone swarms.
2) We propose a new augmented proportional navigation policy for drone swarm defense.
3) We investigate the adaptability of our algorithm to different team sizes by testing on 2 vs. 2 and 4 vs. 4 MAPE scenarios with different defensive policies: pure pursuit and augmented proportional navigation.
4) We demonstrate the usefulness of this algorithm by testing it on a 2 vs. 2 MAPE scenario on Crazyflie hardware and verify the sim-to-real capabilities of the trained actor network even with radio communication delay, radio bandwidth, sensor noise, and model uncertainties.

II. Related Works

A popular PE game originally proposed by Isaacs [2] is the homicidal chauffeur game where a faster pursuer with limited turn radius is pursuing a slower evader that can make sharp turns. When the distance between the pursuer and evader is within some capture radius then the evader is said to be captured. A modified version of the homicidal chauffeur with multiple pursuers is proposed by Bopardikar et al. [11], where the authors propose a cooperative strategy for pursuers to confine an evader within a bounded region. Cooperative PE strategies have demonstrated interesting results in prior literature [12], [13], [14], [15], [16], [17]. Cooperative engagements may require increasing algorithm complexity and scale poorly with the number of agents. Kumar et al. [17] propose a cooperative guidance strategy for survival of a cooperating team of unmanned aerial vehicles (UAVs) against two attackers. However, the proposed method is limited to 2-dimensional planar 2 vs. 2 engagements and to pursuers that use proportional navigation or APN guidance strategies.

Multi-agent reinforcement learning (MARL) has become a topic of interest in the reinforcement learning community stemming from game theory. MARL scenarios are typically posed as team problems where each team has a specific goal and typically requires cooperation amongst agents in the same team. MARL has been previously used [14], [18], [19] to solve PE and predator prey problems with cooperation in complex environments for drone swarms. However, [18] doesn't consider obstacles in the environment and [14] assumes obstacles are fixed. Similarly, both works assume a 2-dimensional environment by fixing the altitude.

Makkapati et al. [20] discuss two scenarios for a two pursuer and a single evader problem where both pursuers are faster than the evader. In the first scenario, the pursuers have knowledge of the evader's position and velocity and therefore use a constant-bearing strategy which guarantees capture of the evader. In the second scenario, the pursuers only have position information of the evader and therefore use a pure-pursuit strategy. Since capture is guaranteed the objective of the evader is to delay capture for as long as possible. The proposed methodology is to first determine which pursuers are most important and which do not affect the outcome of the game. Then, the game is either a one-pursuer one-evader game in the degenerate case, or in the non-degenerate case, the optimal strategy of the evader involves simultaneous capture by the two pursuers. Pachter et al. [21] further characterize the geometric solution for the same scenario. Similarly, Sun and Tsiotras [22] propose a two-pursuer one-evader evader scenario with faster pursuers, but only one pursuer is active at any time. The objective of the evader is to also prolong capture as much as possible. They determine an evasion strategy using optimal control and a suboptimal strategy is proposed for generalized multi pursuer problems where the evasion from the active pursuer is done through line-of-sight until it reaches and follows a Voronoi cell boundary formed by other inactive pursuers.

III. Problem Formulation

A. Pursuit-Evasion Problem

In this section, we define the pursuit-evasion problem for two multirotor drones such that one is the pursuer and the other is the evader. The objective of the evader is to reach an a priori defined goal location. The evader is said to have completed the goal when velocity $||\mathbf{v}_e|| < \delta_v$ and relative position between evader and goal are within a ball such that $||\mathbf{p}_e - \mathbf{p}_{\text{goal}}|| < \delta_{\text{goal}}$. The objective of the pursuer is to catch the evader before it reaches the goal. The pursuer is said to have caught the evader when their relative position is within a ball such that $||\mathbf{p}_e - \mathbf{p}_p|| < \delta_c$.

The dynamics for the pursuer and evader are taken to be a simplified form of multirotor drone dynamics given by

$$\begin{aligned}
\dot{p}_x &= v_x, \quad \dot{p}_y = v_y, \quad \dot{p}_z = v_z \\
\dot{v}_x &= [\cos\phi\cos\psi\sin\theta + \sin\phi\sin\psi]u_T \\
\dot{v}_y &= [\cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi]u_T \\
\dot{v}_z &= [\cos\phi\cos\theta]u_T - g \\
\dot{\phi} &= \tau_\phi^{-1}(\phi_{\text{des}} - \phi) \\
\dot{\theta} &= \tau_\theta^{-1}(\theta_{\text{des}} - \theta) \\
\dot{\psi} &= \tau_\psi^{-1}(\psi_{\text{des}} - \psi)
\end{aligned} \tag{1}$$

with position $\mathbf{p} = [p_x, p_y, p_z]^\top$ and velocity $\mathbf{v} = [v_x, v_y, v_z]^\top$ with gravitational acceleration g. The orientation is represented using Euler angles for roll $\phi \in [-\pi, \pi]$, pitch $\theta \in [-\pi/2, \pi/2]$, and yaw $\psi \in [-\pi, \pi]$. The control inputs are the desired attitude angles ϕ_{des}, θ_{des}, and ψ_{des} and mass-normalized collective thrust u_T. The simplified model assumes an inner-loop PID controller tracks desired attitude commands so that the attitude dynamics can be approximated as a first-order linear system with time constants τ_ϕ, τ_θ, and τ_ψ. This simplified model has proven successful in model predictive control [23] for Crazyflie hardware.

The control strategy for the evader team is provided state information of itself and the position $\mathbf{p}_e$, and velocity $\mathbf{v}_e$ of its evader target to define a desired acceleration command using one of two proposed methods; pure pursuit (PP) and augmented proportional navigation (APN).

The PP control law is defined as

$$\mathbf{a}_{des} = \mathbf{K}_p(\mathbf{p}_e - \mathbf{p}_p) + \mathbf{K}_v(\mathbf{v}_e - \mathbf{v}_p) + g\mathbf{z}_W \quad (2)$$

for positive definite gain matrices $\mathbf{K}_p$ and $\mathbf{K}_v$ and world frame z axis $\mathbf{z}_W = [0,0,1]^\top$.

The APN desired acceleration vector uses a true ProNav term with closing velocity and rotation vector as well as an additional term for feedforward target acceleration perpendicular to the line of sight. We first define the position of evader relative to pursuer as $\mathbf{R} = \mathbf{p}_e - \mathbf{p}_p$ and velocity of evader relative to pursuer $\mathbf{V}_r = \mathbf{v}_e - \mathbf{v}_p$. Then we define the line-of-sight (LOS) rotation rate as

$$\dot{\Omega} = \frac{\mathbf{R} \times \mathbf{V}_r}{\mathbf{R} \cdot \mathbf{R}}. \quad (3)$$

The projected acceleration of the evader in the line-of-sight direction is defined as

$$\mathbf{a}_{e_{\text{LOS}}} = \frac{\mathbf{R}}{||\mathbf{R}||}\left(\mathbf{a}_e \cdot \frac{\mathbf{R}}{||\mathbf{R}||}\right) \quad (4)$$

for evader acceleration $\mathbf{a}_e$. The evader acceleration orthogonal to the LOS is therefore

$$\mathbf{a}_{e_{\text{ORTHO}}} = \mathbf{a}_e - \mathbf{a}_{e_{\text{LOS}}}. \quad (5)$$

We define the APN pursuer desired acceleration vector as

$$\mathbf{a}_{des} = N_{\text{FB}}(\mathbf{V}_r \times \dot{\Omega}) + N_{\text{FF}}\mathbf{a}_{e_{\text{ORTHO}}} + g\mathbf{z}_W \quad (6)$$

with constants N_{FF} and N_{FB} for scaling feedforward and feedback terms.

Since the drones are initialized with a zero velocity $\mathbf{V}_r$ this results in a small feedback term which means the pursuers don't chase the evaders until they pick up more speed. Therefore, we use the pure pursuit strategy until the closing velocity $V_c \in \mathbb{R}$ is within some threshold. The closing velocity is defined as

$$V_c = -\mathbf{V}_r \cdot \frac{\mathbf{R}}{||\mathbf{R}||}. \quad (7)$$

Agents with a closing velocity $V_c < V_{c_{\min}}$ use the pure pursuit strategy until the conditional is satisfied which then switches the policy to APN.

Then a standard Mellinger-type [24] conversion converts the desired acceleration into a desired rotation matrix and collective thrust due to its robustness on the Crazyflie hardware platform. The desired roll $\phi_{p_{\text{des}}}$, pitch $\theta_{p_{\text{des}}}$, and yaw $\psi_{p_{\text{des}}}$ commands are extracted from the rotation matrix assuming a ZYX sequence.

B. *Multi-Agent Pursuit-Evasion*

The superior pursuer assumption makes it difficult for the evader to reach the goal within a radius of δ_{goal} and velocity δ_v before being captured by the pursuer. In this section, we redefine the problem statement for a multi-agent pursuit-evasion (MAPE) problem that is related to the problem defined in the prior section. In our approach, we are interested in emergent behavior of cooperative evader teams to improve goal success for the superior pursuer problem.

The objective of the MAPE problem is to maximize the number of evaders that reach the goal within position radius δ_{goal} and velocity δ_v. Specifically, reaching the goal is defined as $||\mathbf{p}_{e_i} - \mathbf{p}_{\text{goal}}|| < \delta_{\text{goal}}$ and $||\mathbf{v}_{e_i}|| < \delta_v$ for evader $e_i \in [0,..,N_e]$ in a N_e-evader team. The objective of the pursuer is to capture the assigned evader where the condition for capture is defined as $||\mathbf{p}_{e_i} - \mathbf{p}_{p_i}|| < \delta_c$ for evader e_i and pursuer $p_i \in [0,..,N_p]$ in a N_p-pursuer team. With MAPE, we allow collision between team members such that for evaders the team collision is defined by $||\mathbf{p}_{e_i} - \mathbf{p}_{e_j}|| < \delta_{e_{\text{coll}}}$ with collision radius $\delta_{e_{\text{coll}}}$ and $e_j \in [0,..,N_e], e_i \neq e_j$. Pursuer team collision is similarly defined with collision radius $\delta_{p_{\text{coll}}}$. An evader is defined to be deactivated if it is caught by the pursuers, reaches the goal, or crashes into another evader. A pursuer is said to be deactivated if it catches an evader or crashes into another pursuer. We assume that evaders have knowledge of the active status of the other evaders and the pursuers.

Another challenge in MAPE is target selection for the pursuers. We take a simplified approach. At initialization, evader targets are randomly assigned to each pursuer. If an evader is deactivated then pursuers are reassigned to target the closest evader. The evaders are given explicit knowledge of which pursuer is targeting which evader. Pursuers are only given information about its state and the position and velocity of its current evader target. They are not given knowledge of other pursuers' or non-targeted evaders' states. In order for pursuer collisions to occur, evaders need to successfully cooperate with each other.

IV. Methodology

Synthesizing optimal cooperative strategies for the MAPE problem is challenging and becomes exponentially complex with increasing team sizes. Therefore, we propose using multi-agent reinforcement learning (MARL) to synthesize pseudo-optimal cooperative strategies for the evaders with varying team sizes.

A. *Multi-Agent Reinforcement Learning*

Reinforcement learning (RL) is a method for solving Markov Decision Processes (MDPs) composed of state observations, actions, and rewards. There are various RL algorithms, but in this work we focus on Proximal Policy Optimization (PPO) [25]. PPO is a constrained, on-policy, reinforcement learning algorithm that uses an actor-critic structure. The actor-critic method combines both value-based and policy-based methods. That is, the critic provides the value function and the actor provides the optimal policy which are actions that yield the highest expected sum of future rewards based on the current state of the environment. Typically, the actor and critic are approximations using neural networks as universal function approximators. For on-policy methods, the neural networks are trained using exploration and gradient-based algorithms. In PPO, an advantage estimate from the critic is used in the policy update. There are various approaches to estimating the advantage function and in our work we used the generalized advantage estimate (GAE) [26].

There are two common strategies for decentralized PPO in multi-agent MDPs. One is independent PPO (IPPO), where the problem is defined as solving N single agent problems with other agents assumed to be part of the environment, and the other is multi-agent PPO (MAPPO) [27]. MAPPO is similar to IPPO but is differentiated by the use of a centralized critic V_ϕ parameterized by ϕ. For homogeneous agents, as in this work, the decentralized actor network π_θ shares parameters θ between all agents. In this work, we use MAPPO with some slight adjustments as outlined. During training, individual state-action trajectories are recorded for each agent, along with the reward trajectory experience of the whole multi-agent team. The termination criteria of state-action trajectories is met when all evaders have succeeded, collided, or timed-out. In the following, we summarize the loss functions used to train the actor and critic networks where we assume all the agent actors share the same parameters and the critic is a centralized critic using the global state of the environment. The actor loss is defined as

$$L_a(\theta) = \frac{1}{BN_e}\sum_{b=1}^{B}\sum_{e=1}^{N_e}\left[\min\left(r_b^e(\theta)\hat{A}_b, L_{clip}\hat{A}_b\right)\right] \tag{8}$$

where

$$L_{clip} = \text{clip}(r_b^e(\theta), 1-\varepsilon, 1+\varepsilon) \tag{9}$$

ε is a clipping term from PPO, and $\hat{A}_b$ is the advantage estimate in the batch which is calculated using the generalized advantage estimate (GAE) [26] where γ_{GAE} is the reward discount factor and λ_{GAE} is the smoothness factor. Additionally, $r_b^e(\theta) = \frac{\pi_\theta(a_b^e|o_b^e)}{\pi_{\theta_{\text{old}}}(a_b^e|o_b^e)}$, for evader observation o_b^e and evader action a_b^e in the trajectory batch of size B. $V(s_b)$ is the output of the central value function given the global state s_b. The GAE is used to estimate the reward-to-go $\hat{R}_b = \hat{A}_b + V_{\phi_{old}}(s_b)$. Therefore, the critic loss can be defined as

$$L_c(\phi) = \frac{\lambda_{\text{critic}}}{B}\sum_{b=1}^{B}[V_{\phi_{new}}(s_b) - (\hat{A}_b + V_{\phi_{old}}(s_b))]^2 \tag{10}$$

where λ_{critic} is a hyperparameter to tune the effective learning rate of the critic with respect to the actor network.

B. Observations

For this multi-agent Markov Decision Process formulation, each evader observes its own state, *i.e.*, position, velocity, and attitude, as well as the state of all other evaders and pursuers and a flag $\lambda \in \{0,1\}$ that indicates whether that agent is active. Additionally, we implement a one-hot encoding vector for pursuer-evader targets. We define the one-hot evader target vector for pursuer i as a bit array ζ_{p_k} of length N_e. We define $\bar{e}_i := [p_x, p_y, p_z, v_x, v_y, v_z, \phi, \theta, \psi]$ as the full state of evader i. Likewise, we define $\bar{p}_i$ as the full state of pursuer i. The global state s_t used by the central value function is a concatenation of the full state of each agent in the MAPE problem. Specifically, we define s_t as,

$$s_t := [\bar{e}_j \times \lambda_{e_j}, \lambda_{e_j}, \bar{p}_k \times \lambda_{p_k}, \lambda_{p_k}, \zeta_{p_k} \times \lambda_{p_k}],$$
$$\forall j \in [0,...,N_e],\ \forall k \in [0,...,N_p].$$

The observation for an evader i is defined as $o_{e_i} = [\bar{e}_i, s_t]$. Notice in the observation the evader state $\bar{e}_i$ contains a copy of itself in s_t. This approach allows a consistent ordering of other evader agents in the team. This allows the neural network to correlate the pursuer one-hot targeting vector with indices of the evader observation. We mask the pursuer states, evader states, and one-hot vector target using the active flag λ. This avoids learning from these state elements when they are no longer active in the problem while also providing the flag to differentiate when the agent is active versus deactivated at the origin. Putting $\bar{e}_i$ first in each o_{e_i} vector ensures the generic multi-agent evader policy can correctly map evader state i to optimal action for evader i.

C. Actions

We use a Tanh activation function on the output of the actor and apply the following gain and bias to actor policy network output $\pi_\theta(a_t^e|o_t^e)$ such that

$$u_e = \pi_\theta \cdot [\sigma_{e_\phi}, \sigma_{e_\theta}, \sigma_{e_\psi}, \sigma_{e_T}] + [0,0,0,g]$$

is the control action. The control action applied to evader i is therefore $u_{e_i} = [\phi_{\text{des}}, \theta_{\text{des}}, \psi_{\text{des}}, u_T]$. We clamp the desired control commands for the evader and pursuer within an interval such that

$$\begin{aligned} \phi_{p_{\text{des}}} &\in [-\sigma_{p_\phi}, \sigma_{p_\phi}] & \phi_{e_{\text{des}}} &\in [-\sigma_{e_\phi}, \sigma_{e_\phi}] \\ \theta_{p_{\text{des}}} &\in [-\sigma_{p_\theta}, \sigma_{p_\theta}] & \theta_{e_{\text{des}}} &\in [-\sigma_{e_\theta}, \sigma_{e_\theta}] \\ \psi_{p_{\text{des}}} &\in [-\sigma_{p_\psi}, \sigma_{p_\psi}] & \psi_{e_{\text{des}}} &\in [-\sigma_{e_\psi}, \sigma_{e_\psi}] \\ u_{p_T} - g &\in [-\sigma_{p_T}, \sigma_{p_T}] & u_{e_T} - g &\in [-\sigma_{e_T}, \sigma_{e_T}]. \end{aligned}$$

As stated before, we are interested in the PE variation where the pursuer has stronger control authority over the evader which implies that $\sigma_{p_\phi} > \sigma_{e_\phi}$, $\sigma_{p_\theta} > \sigma_{e_\theta}$, $\sigma_{p_\psi} > \sigma_{e_\psi}$, and $\sigma_{p_T} > \sigma_{e_T}$.

D. Rewards

While observations and actions are unique to each evader, rewards are given to the team. Rewards are based on discrete binary events as defined in Section III-B. In this section, we summarize these events with discrete variables and define our reward function using these discrete variables. For an active evader e_i *i.e.*, $\lambda_{e_i} = 1$, we define goal success as

$$\text{success}_{e_i} = \begin{cases} \text{true} & \text{if } \|\mathbf{v}_{e_i}\| < \delta_v \text{ and } \|\mathbf{p}_{e_i} - \mathbf{p}_{\text{goal}}\| < \delta_{\text{goal}} \\ \text{false} & \text{otherwise} \end{cases} \tag{11}$$

for goal radius δ_{goal} and velocity residual δ_v. Next, we define evader team collisions for an active evader e_i,

$$\text{collision}_{e_i} = \begin{cases} \text{true} & \text{if } \|\mathbf{p}_{e_i} - \mathbf{p}_{e_j}\| < \delta_{e_{\text{coll}}} \\ & \exists j \in [0,...,N_e], i \neq j, \lambda_{e_j} = 1, \\ \text{false} & \text{otherwise} \end{cases} \tag{12}$$

for evader team collision radius $\delta_{e_{\text{coll}}}$. For active pursuer p_i such that $\lambda_{p_i} = 1$, we define pursuer team collisions as

$$\text{collision}_{p_i} = \begin{cases} \text{true} & \text{if } \|\mathbf{p}_{p_i} - \mathbf{p}_{p_j}\| < \delta_{p_{\text{coll}}} \\ & \exists j \in [0, ..., N_p], i \neq j, \lambda_{p_j} = 1, \\ \text{false} & \text{otherwise} \end{cases} \quad (13)$$

for pursuer team collision radius $\delta_{p_{\text{coll}}}$. Define an active pursuer p_j capturing an active evader e_i event as,

$$\text{capture}_{e_i} = \begin{cases} \text{true} & \text{if } \|\mathbf{p}_{e_i} - \mathbf{p}_{p_j}\| < \delta_c \\ & \exists j \in [0, ..., N_p], \lambda_{p_j} = 1. \\ \text{false} & \text{otherwise} \end{cases}$$

Finally, we define an evader arena boundary collision for an active evader e_i as,

$$\text{geofence}_{e_i,b} = \begin{cases} \text{true} & \text{if } \|\mathbf{p}_{e_i}\| > \delta_b, \\ \text{false} & \text{otherwise} \end{cases} \quad (14)$$

where δ_b is the radius of the arena boundary. The reward function for the multi-agent pursuit evasion problem is defined as the sum of multiple discrete events

$$\begin{aligned} \text{reward} = & \sum_{i=1}^{N_e} \big(K_s \cdot \text{success}_{e_i} - K_{ee} \cdot \text{collision}_{e_i} \\ & - K_c \cdot \text{capture}_{e_i} - K_b \cdot \text{geofence}_{e_i}\big) + \sum_{i=1}^{N_p} K_{pp} \cdot \text{collision}_{p_i} \end{aligned} \quad (15)$$

where $K_s, K_{ee}, K_{pp}, K_c, K_b$ are tunable reward constants. Additionally, we normalize the reward by the number of evader and pursuer pairs. For this work, initially, there is always an equal number of evaders and pursuers.

V. SIMULATIONS

In this section we present simulation results from our trained models for the MAPE drone problem. We begin with a discussion of our ablation study that compares our trained MAPE models to other trained models with no active pursuers and verify that the trained MAPE models successfully increase the average number of evader success through simulation results. We then discuss the model and training configurations, simulation results, and compare our method to a cooperative guidance strategy proposed in [17].

A. Ablation Study

We compare our trained models to various ablation baselines to verify that our trained models are successfully improving evader success for the MAPE problem. In the following discussion, we describe each ablation baseline for the 2 vs. 2 pursuit-evasion problem, but each baseline is also provided for the 4 vs. 4 problem. In the first baseline, BL2v0, we train two evaders to reach the goal without any pursuers. This provides a comparison for how good the model can do to reach the goal if no evaders are present with the initialization given. Then we test the trained BL2v0 with two pursuers. This provides a baseline for how well the pursuers can capture the evaders. Then, from the BL2v0 model we train a second baseline, BL1v1, with one random pursuer and a random evader at each initialization. To test this baseline we place the trained model in a 2 vs. 2 scenario. This provides a naive approach without any cooperative training scenarios. This combination of baselines gives enough information to conclude whether our trained MARL for evader teams model has learned cooperative strategies to combat the MAPE problem. Additionally, we implement the baselines for both the PP and APN pursuer strategies.

B. Model and Training Configuration

The actor policy is represented by a fully connected neural network architecture comprising two hidden layers, each with 512 neurons. Both layers use ReLU activation functions in between and a Tanh activation function on the output as stated in Section IV-C. The critic is also a fully connected neural network with two hidden layers of width 512 with ReLu activation functions in between. The output of the critic network is a scalar. To train the MAPE problem for various team sizes, we construct a custom gym environment and a MAPPO training interface using PyTorch [28]. The gym environment is designed to run any N_e vs. N_p MAPE problem. We perform initialization of the pursuers within a bounded box with side length L_p centered at the goal. The evaders are initialized within a volumetric segment in spherical coordinates. The sampled parameters for each evader are the initial radial distance $\rho_0 \in [\underline{\rho}_0, \overline{\rho}_0]$, inclination angle $\alpha_0 \in [\underline{\alpha}_0, \overline{\alpha}_0]$, azimuth angle $\beta_0 \in [\underline{\beta}_0, \overline{\beta}_0]$, and a centering offset for the azimuth angle that is sampled in $[0, 2\pi]$. The initialization and environment parameters for each team scenario are tabulated in Table Ia. The evader PP strategy diagonal gain matrices used in the training and simulation are $\mathbf{K}_p = \text{diag}(6.3, 6.3, 20)$ and $\mathbf{K}_v = \text{diag}(2.4, 2.4, 10)$. Additionally, for the APN strategy we use the constants $N_{\text{FB}} = 5$ for the feedback gain term and $N_{\text{FF}} = 1$ for the feedforward gain term. Coefficients are tuned to provide good pursuer tracking performance.

We use curriculum learning [29] to stage the problem into different difficulty levels. This is especially useful for the 4 vs. 4 scenario where we stage the problem into a subset of scenarios of 2 vs. 2. That way the MAPPO algorithm experiences different scenarios in training uniformly to learn the switching behaviors it may require when the environment team size changes in a test. We tabulate the hyperparameters for MAPPO used in training in Table Ib. We tuned the reward coefficients to maximize the number of successful events per episode. The success event is the overall goal which is why we significantly reward the agents and in order to reach this success event the evaders need to make pursuers collide into each other. There, we also prioritize rewarding pursuer-on-pursuer collisions. We also noticed that penalizing capture events in the 4 vs. 4 scenario leads to faster training. Additional parameters used for the MAPE simulation are $\delta_{\text{goal}} = 0.2\text{m}$, $\delta_v = 0.1\text{m}$, $\delta_{e_{\text{coll}}} = 0.2\text{m}$, $\delta_{p_{\text{coll}}} = 0.2\text{m}$, and $\delta_c = 0.2\text{m}$. For the arena boundary we use $\delta_b = 15\text{m}$.

TABLE I: a) Initialization and environment parameters for training different team sizes. b) Multi-agent proximal policy optimization hyperparameters.

(a)

		2 vs. 2	4 vs. 4
$\tau_\phi, \tau_\theta, \tau_\psi$	[s]	0.1	0.1
dt	[s]	0.05	0.05
$\underline{\alpha}_0$	[deg]	85	85
$\overline{\alpha}_0$	[deg]	95	95
$\underline{\beta}_0$	[deg]	0	0
$\overline{\beta}_0$	[deg]	70	70
$\underline{\rho}_0$	[m]	3.5	5.5
$\overline{\rho}_0$	[m]	4	6.0
L_p	[m]	0.5	1.0
K_s		30	30
K_{ee}		2.5	5
K_{pp}		10	20
K_c		0	30
K_b		5	5
$V_{c_{\min}}$	[m/s]	2	2

(b)

parallel envs	128
steps per env	256
learning rate	1e-4
actor clip	0.05
network	[512, 512]
activation	ReLu
epochs	5
γ_{GAE}	0.99
λ_{GAE}	0.95
λ_{critic}	0.5

C. Simulation Results

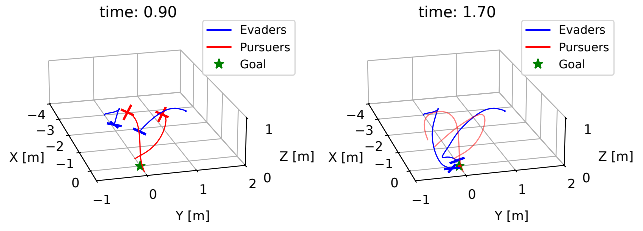
After training we deploy the trained actors into the test environment and provide trajectory plots in Figure 3 at different simulation times. The first plot is the initial setup of the environment with some time passed to indicate the initial evasion maneuvers taken by evaders. The second plot is just before the majority of pursuer collisions occur and the final plot is after all pursuers are deactivated and the evaders perform the final approach towards the goal. We tabulate the results of the test environment with average goal success and average steps taken in the simulation for each team size.

We observe and comment on the different strategy of differing team sizes. First, in the 2 vs. 2 scenario the evaders have only one team member to cooperate with and goal success is difficult if that team member is caught. Therefore, we end up with trajectories that try to focus on fast collision of the two pursuers. In the 4 vs. 4 scenario the evaders have three other team members to cooperate with. The strategy becomes challenging because there are more agents to cooperate within a team. Therefore, we end up observing a grouping strategy and encircling the pursuers to get them to stack up and collide with each other. A big part of the evader strategy is to make abrupt maneuvers in small clusters because the policy can avoid collision with each other while causing clustering and collision of the pursuers.

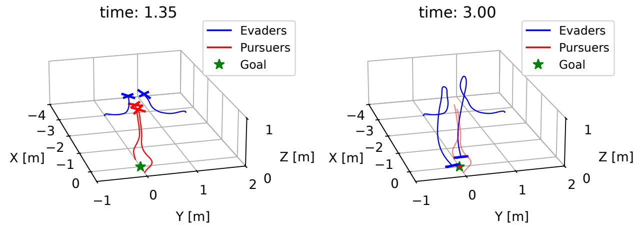
In Table II we tabulate the average success rate for evaders to reach the goal and the average time for an episode in seconds. The results provided compares our method to the ablation study baselines and the cooperative guidance strategy proposed in [17]. The method proposed by [17] is only for proportional navigation-based pursuers in 2 vs. 2. Therefore, we only tabulate the results for the APN pursuer strategy. Our proposed method provides the best average success rate when pursuers are active in the environment.

TABLE II: Simulation results for 2 vs. 2 and 4 vs. 4 pure pursuit (PP) and augmented proportional navigation (APN) pursuer strategies. The results compare our method to the ablation study results and the guidance strategy in [17]. The first column indicates how many evaders and how many pursuers are in the simulation. Average success rate (SR) indicates how many evaders have successfully reached the goal on average for an episode with a percent provided over the total number of evaders. The average episode length (EL) indicates how long an episode takes on average in seconds.

Evaders vs. Pursuers	Method	**Pursuer Strategies** *Pure Pursuit* SR(%)	EL[s]	*Augmented ProNav* SR(%)	EL[s]
2 vs. 2	**Ours**	**1.93 (96.5)**	**2.30**	**1.87 (94)**	**2.10**
2 vs. 2	[17]	-	-	1.58 (79)	8.10
2 vs. 0	BL2v0	1.97 (99)	1.65	1.97 (99)	1.65
2 vs. 2	BL2v0	0.27 (14)	2.05	0.22 (11)	1.55
2 vs. 2	BL1v1	0.45 (23)	5.45	0.19 (9.5)	1.35
4 vs. 4	**Ours**	**3.44 (86)**	**3.35**	**2.15 (54)**	**2.85**
4 vs. 0	BL4v0	3.78 (95)	2.35	3.78 (95)	2.35
4 vs. 4	BL4v0	0.24 (6.0)	2.40	0.32 (8.0)	1.70
4 vs. 4	BL1v1	0.11 (2.8)	4.50	0.3 (7.5)	7.35



(a) Pure Pursuit



(b) Augmented Proportional Navigation

Fig. 3: Pursuer and evader simulation trajectories for the 2 vs. 2 multi-agent pursuit evasion problem with both pure pursuit and augmented proportional navigation pursuer strategies. Pursuer trajectories are in red and evader trajectories are in blue. Trajectories with lower opacity represent agents that have become deactivated. For both pursuer strategies we provide two snapshots of an episode at different times.

VI. Experiments

In this section we discuss hardware deployment of our MARL models for the MAPE problem on real drones. In our experiments we use four Bitcraze Crazyflie 2.1 nano quadrotors for the 2 vs. 2 pursuit-evasion problem. We use Crazyswarm2 [30] to parse and send motion capture position data from unlabeled markers on the drones, receive state information at 50 Hz from each drone, and send attitude and thrust commands to each drone at 50 Hz. To setup the MAPE problem we developed a ROS 2 server node that subscribes to state information for each drone, monitors the MAPE problem for collisions and successes, publishes a status message with the active flag for each drone as well as all the state information for the pursuer and evader teams, and contains a service command for starting the experiment. A separate pursuer and evader ROS 2 node is set up for controlling the drones while subscribing to the MAPE server status messages. This allows us to create separate threads for monitoring the MAPE problem and controlling the drone.

Much of the sim-to-real transfer challenges we faced are related to the hardware limitations of the drone (*e.g.* radio bandwidth, latency, brushed motors leading to varying motor dynamics, motor saturation limiting maximum flight speeds to less than 1 m/s). To address this we changed the attitude and thrust bounds of the evader and pursuer for the MAPE problem to $\sigma_{p_\phi} = \sigma_{p_\theta} = 0.15$, $\sigma_{e_\phi} = \sigma_{e_\theta} = 0.2$, $\sigma_{p_\psi} = \sigma_{e_\psi} = 0.1$, $\sigma_{e_T} = 0.1g$, and for the pursuer we opted for a higher upper bound of $\overline{\sigma}_{p_T} = 1g$ with a lower bound of $\underline{\sigma}_{p_T} = 0.5g$ for a standard gravitational constant of $g = 9.80665\,\mathrm{m/s^2}$. Additionally, we change the minimum closure velocity to activate APN to $V_{c_{\min}} = 0.5\,\mathrm{m/s}$. Then, we retrained the model in our Python gym environment with the updated control saturation.

In the following sections we discuss our results for a 2 vs. 2 pursuit-evasion problem. We opted to use APN for the pursuer strategy as it was the more difficult strategy for the evader to overcome in simulation. For all experiments the goal is located at $(0,0,1)$m, pursuer initial position at $(0.2,0.4,1)$m for pursuer 1 and $(-0.25,-0.25,1)$m for pursuer 2, evader initial position at $(3,-2.5,1)$m for evader 1 and $(2,-4,1)$m for evader 2. The initial targets are pursuer 1 to evader 1 and pursuer 2 to evader 2. The parameters used for the hardware MAPE $\delta_{\text{goal}} = 0.3\,\mathrm{m}$, $\delta_v = 0.6\,\mathrm{m}$, $\delta_{e_{\text{coll}}} = 0.2\,\mathrm{m}$, $\delta_{p_{\text{coll}}} = 0.5\,\mathrm{m}$, and $\delta_c = 0.2\,\mathrm{m}$. Note that the collision parameters can be stricter if we train on a gym environment that is closer to the real Crazyflie dynamics, although we noticed that radio bandwidth limitations are the major limiting factor.

A. Hardware Results

We then conducted a hardware test using real Crazyflies for the 2 vs. 2 MAPE problem with pursuers using the APN strategy. Due to radio bandwidth limitations we noticed our experiment ran best with two radios — one for the pursuer team and one for the evader team.

We conduct the experiment and plot the trajectory in Figure 4 and illustrate a long exposure image of the flight in Figure 1. The initial and final hovering position for each agent on both evader and pursuers teams is slightly offset from each other to avoid an agent hovering in the downwash of another agent. This is why we see an offset in the evaders when they reach the goal. A major difference in the hardware flight is that the evaders initially pass the pursuers and then perform their collaborative maneuver to lead the pursuers into a collision event before going to the goal, otherwise the pursuers would turn around and catch the evaders before they reach the goal. The evaders also seem to have learned to take advantage of the fact that the pursuers observe the acceleration of the evaders and use it in the APN strategy. This can be observed through the use of the z-axis from the evaders to lead the pursuers because only the acceleration perpendicular to the line-of-sight between the pursuer and its evader target is used in the feedforward term of APN.

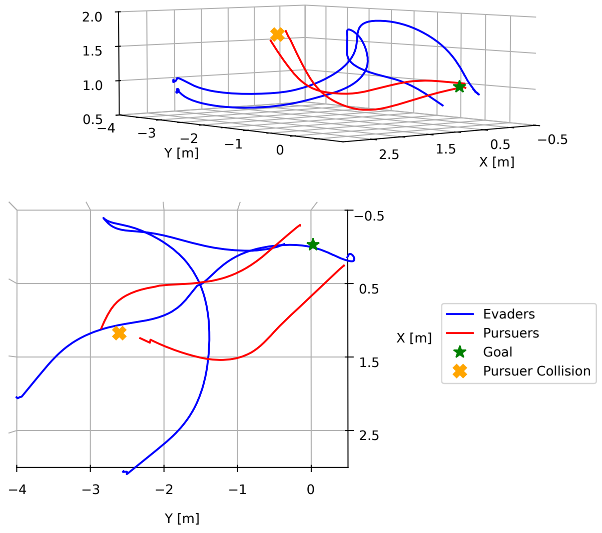


Fig. 4: A plot of the trajectory data for the hardware demonstration for the 2 vs. 2 pursuit-evasion problem. The pursuers start at the goal and attempt to defend it by chasing the evaders while the evaders use multi-agent reinforcement learning to cause the pursuers to become disabled and eventually to reach the goal.

VII. CONCLUSIONS & LIMITATIONS

In this work, we make several assumptions about the model's dynamics. Specifically, we assume the rotational dynamics behave as a first order system. This assumption allows the control variables to appear as a commanded Euler angle, which is what is used to command the Crazyflie drones. Otherwise, a more sophisticated model would be required that matches the timing of the inner loop control structure in the Crazyflie and the full dynamics of the drone. Another major limitation in the hardware results comes from using a radio that introduces latency into the control system and is limited in bandwidth, which can bottleneck information transfer back and forth between the drone. We noticed limitations in the rate the control and state information can be transferred to and from the drone. These two limitations are the largest source of model mismatch noticed in our experimental demonstration. Our hardware results demonstrate that the model successfully overcomes these limitations, yielding performance comparable to our simulations. Another limitation in our results may come from the use of a high precision localization motion capture system for the full state information required in the policy. In the real world, this may not be available and additional noise from using GPS and perception-based systems may require further research of this approach. In our future work, we plan to improve the model mismatch from the first two limitations by using a more sophisticated drone model with drag, rotor dynamics, and rotor gyroscopic effects through system identification for the Crazyflie.

ACKNOWLEDGMENT

This work was supported by the Laboratory Directed Research and Development program at Sandia National Laboratories, a multi-mission laboratory managed and operated by the National Technology and Engineering Solutions of Sandia LLC, a wholly owned subsidiary of Honeywell International Inc. for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525. This letter describes objective technical results and analysis. Any subjective views or opinions that might be expressed in the letter do not necessarily represent the views of the U.S. Department of Energy or the United States Government.

REFERENCES

[1] R. Isaacs, *Differential games: a mathematical theory with applications to warfare and pursuit, control and optimization*. John Wiley and Sons, 1965.

[2] ——, "Games of pursuit," RAND Corporation, Santa Monica, CA, Tech. Rep. P-257, 1951. [Online]. Available: https://www.rand.org/pubs/papers/P257.html

[3] S. Bhattacharya and S. Hutchinson, "On the existence of nash equilibrium for a two-player pursuit—evasion game with visibility constraints," *The International Journal of Robotics Research*, vol. 29, no. 7, pp. 831–839, 2010.

[4] W. Sun, P. Tsiotras, T. Lolla, D. N. Subramani, and P. F. J. Lermusiaux, "Multiple-pursuer/one-evader pursuit–evasion game in dynamic flowfields," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 7, pp. 1627–1637, 2017.

[5] D. Maity, "Efficient communication for pursuit-evasion games with asymmetric information," in *2023 62nd IEEE Conference on Decision and Control (CDC)*, 2023, pp. 2104–2109.

[6] D. Oyler, P. Kabamba, and A. Girard, "Pursuit–evasion games in the presence of obstacles," *Automatica*, vol. 65, pp. 1–11, 03 2016.

[7] J. Szőts and I. Harmati, "Optimal strategies of a pursuit-evasion game with three pursuers and one superior evader," *Robotics and Autonomous Systems*, vol. 161, p. 104360, 2023.

[8] X. Fang, C. Cheng, and L. Xie, "3-d multi-player pursuit-evasion game with a faster evader," in *2020 39th Chinese Control Conference (CCC)*, 2020, pp. 118–123.

[9] F. Shkurti and G. Dudek, "On the complexity of searching for an evader with a faster pursuer," in *2013 IEEE International Conference on Robotics and Automation*, 2013, pp. 4062–4067.

[10] J. Sprinkle, J. Eklund, H. Kim, and S. Sastry, "Encoding aerial pursuit/evasion games with fixed wing aircraft into a nonlinear model predictive tracking controller," in *2004 43rd IEEE Conference on Decision and Control (CDC) (IEEE Cat. No.04CH37601)*, vol. 3, 2004, pp. 2609–2614 Vol.3.

[11] S. D. Bopardikar, F. Bullo, and J. P. Hespanha, "A cooperative homicidal chauffeur game," *Automatica*, vol. 45, no. 7, pp. 1771–1777, 2009.

[12] Q. Sun, N. Qi, Z. Xu, Y. Liu, and Y. Zhang, "An optimal one-way cooperative strategy for two defenders against an attacking missile," *Chinese Journal of Aeronautics*, vol. 30, no. 4, pp. 1506–1518, 2017.

[13] X. Wei and J. Yang, "Optimal strategies for multiple unmanned aerial vehicles in a pursuit/evasion differential game," *Journal of Guidance, Control, and Dynamics*, vol. 41, no. 8, pp. 1799–1806, 2018.

[14] R. Zhang, Q. Zong, X. Zhang, L. Dou, and B. Tian, "Game of drones: Multi-uav pursuit-evasion game with online motion planning by deep reinforcement learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 10, pp. 7900–7909, 2023.

[15] D. Li and J. B. Cruz, "Defending an asset: A linear quadratic game approach," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 47, no. 2, pp. 1026–1044, 2011.

[16] A. Ratnoo and T. Shima, "Guidance strategies against defended aerial targets," *Journal of Guidance, Control, and Dynamics*, vol. 35, no. 4, pp. 1059–1068, 2012.

[17] N. S. Kumar, R. V. Nanavati, and S. R. Kumar, "Robust nonlinear guidance strategies for survival of cooperating unmanned aerial vehicles against pursuing attackers," *Proceedings of the Institution of Mechanical Engineers, Part G*, vol. 237, no. 5, pp. 1025–1040, 2023. [Online]. Available: https://doi.org/10.1177/09544100221115238

[18] C. de Souza, R. Newbury, A. Cosgun, P. Castillo, B. Vidolov, and D. Kulić, "Decentralized multi-agent pursuit using deep reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4552–4559, 2021.

[19] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," 2020.

[20] V. R. Makkapati, W. Sun, and P. Tsiotras, "Optimal evading strategies for two-pursuer/one-evader problems," *Journal of Guidance, Control, and Dynamics*, vol. 41, no. 4, pp. 851–862, 2018.

[21] M. Pachter, A. Von Moll, E. Garcia, D. W. Casbeer, and D. Milutinović, "Two-on-one pursuit," *Journal of Guidance, Control, and Dynamics*, vol. 42, no. 7, pp. 1638–1644, 2019.

[22] W. Sun and P. Tsiotras, "An optimal evader strategy in a two-pursuer one-evader problem," in *53rd IEEE Conference on Decision and Control*, 2014, pp. 4266–4271.

[23] C. Llanes, Z. Kakish, K. Williams, and S. Coogan, "Crazysim: A software-in-the-loop simulator for the crazyflie nano quadrotor," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024, pp. 12 248–12 254.

[24] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 2520–2525.

[25] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017.

[26] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," 2018.

[27] C. Yu, A. Velu, E. Vinitsky, J. Gao, Y. Wang, A. Bayen, and Y. Wu, "The surprising effectiveness of ppo in cooperative multi-agent games," in *Proceedings of the 36th International Conference on Neural Information Processing Systems*, ser. NIPS '22. Red Hook, NY, USA: Curran Associates Inc., 2022.

[28] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, *PyTorch: an imperative style, high-performance deep learning library*. Red Hook, NY, USA: Curran Associates Inc., 2019.

[29] P. Soviany, R. T. Ionescu, P. Rota, and N. Sebe, "Curriculum learning: A survey," 2022.

[30] J. A. Preiss, W. Honig, G. S. Sukhatme, and N. Ayanian, "Crazyswarm: A large nano-quadcopter swarm," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 3299–3304.