

Scientific Machine Learning Based Pursuit-Evasion Strategy in Unmanned Surface Vessel Defense Tactics

Ugurcan Celik, Mevlut Uzun, Gokhan Inalhan

Autonomy and AI, Cranfield University

Cranfield, United Kingdom

Email : {ugurcan.celik, mevlut.uzun, inalhan}@cranfield.ac.uk

Mike Woods

BAE Systems Maritime

Portsmouth, United Kingdom

mike.woods@baesystems.com

Abstract—In this work we develop an AI-aided tactics generator for uncrewed surface vessels (USVs) for protection of critical national infrastructure and maritime assets in face of surface vehicle attacks. Our scientific machine learning (SciML) based methodology incorporates physical principles into the learning process, enhancing the model's ability to generalize and perform accurately in scenarios not encountered during training. This innovation addresses a critical gap in existing AI applications for maritime defense: the ability to operate effectively in novel or changing conditions without the need for retraining.

Index Terms—autonomous decision making, maritime defence, pursuit evasion, scientific machine learning

I. INTRODUCTION

The need for advanced security measures to protect critical infrastructure and maritime assets has revealed the inadequacies of traditional defense approaches in the marine environment. Integrating artificial intelligence (AI) and autonomous systems into Unmanned Surface Vehicle (USV) operations is essential to enhance the effectiveness and adaptability of maritime security strategies. These technologies significantly improve the strategic capabilities of USVs. AI-aided tactics for infrastructure protection using USVs are a critical application of these technologies, effectively modeled as a pursuit-evasion game where the USV is the pursuer and an intruder vessel is the evader. Although pursuit-evasion strategies are still in their early stages of application for USVs, they have been extensively explored in other domains. Research in this area typically divides methods into two categories: conventional control techniques and machine learning approaches.

The conventional approach involves applying classical control and optimization theory. One such methodology is the use of the Model Predictive Control framework to design a controller [1]. A study in [2] utilized MPC for the USV pursuit and evasion game scenario and similar applications have been made in other robotic platforms [3], [4]. However, implementing the MPC methods in pursuit-evasion games necessitates numerous simplifications, resulting in diminished performance in real-world applications.

Another method in the first category applies game theory, where the problem is treated as a min-max optimization

problem and addressed based on cost definitions [5]. Pursuit-evasion, a subtopic of game theory, is extensively studied across various application areas. For maritime domain, the study in [6] analyzes how game theory is applied to various USV tasks. In [7], the authors investigated the pursuit-evasion game problem for USVs based on the threat potential field (TPF). Similarly, game theory methodology has been investigated and applied for other platforms like robotics, unmanned aerial vehicles and unmanned underwater vehicles. Rajan et. al. studied the pursuit-evasion of two aircraft in a horizontal planed and proposed a solution based on extremal trajectory maps [8]. Another study in [9] solved the problem through modelling the aircraft control input limits, creating the Hamiltonian equation and solving the optimization problem numerically. The authors of [10] presented a multiplayer pursuit-evasion game consisting of prey, predator, and protector for collision avoidance problem. Gong et. al. handled the pursuit-evasion problem for two satellites considering their continuous thrust reachable domains [11]. Another multiplayer differential game theory application is presented in [12] for enabling multiple UAVs to engage in multiple targets. However, analytical solutions in game theory are feasible for simplified cases but challenging to implement in real-time. Deriving these solutions requires many assumptions based on extensive calculus, leading to reduced effectiveness in real-world applications, similar to classical methods

In pursuit-evasion problems, our study's second category explores machine learning methodologies. Recent advancements in computational technology enable the training and deployment of machine learning models for real-world applications. Many studies use machine learning, particularly Reinforcement Learning (RL), for pursuit-evasion problems. RL is a trial-and-error-based learning approach where a machine autonomously learns optimal behavior or policies within a specific environment by maximizing or minimizing cumulative rewards or costs. An application of RL on maritime applications is to design a path following controller as the USV dynamics are usually not highly accurate and expose to high level of uncertainties [13], [14], [15], [16]. For the pursuit evasion problem, there have been several applications

for training a pursuer, training an evader, co-operative hunting [17], [18], [19], [20], [21]. RL for pursuit evasion game is also widely studied for robotics and UAVs for single/multi agent use cases [22], [23], [24], [25], [26], [27], [28], [29]. One advantage of RL methods is their suitability for fast real-time inference. However, a major limitation of RL methods is their training efficiency, which heavily depends on the design of reward functions, requiring careful reward engineering. Additionally, in continuous systems, the rapid expansion of the state space exploration complicates the search for global optimum solutions, representing a significant challenge in RL-based approaches.

Within the machine learning category, another approach under consideration is scientific machine learning (SciML) [30], [31]. Unlike traditional machine learning methodologies, SciML is a recently proposed approach that shares similarities with Reinforcement Learning but distinguishes itself by integrating governing physics into the learning processes. This incorporation of physical principles contributes to superior generalization compared to other approaches. Like Reinforcement Learning, SciML exhibits fast real-time inference capabilities. Notably, the impact of the expanding state space on the SciML learning process is significantly less pronounced than in Reinforcement Learning. While SciML has primarily been explored in the context of scientific model discovery problems, its application to pursuit-evasion problems remains unexplored. This study aims to pioneer the application of the SciML approach to pursuit-evasion scenarios, marking a novel exploration of its potential within this domain.

The remainder of this paper is organized as follows: Section II introduces the USV pursuit-evasion problem definition, objectives, and mathematical motion model. Section III details the SciML methodology and formulations used. Section IV presents simulations and discusses results for various scenarios. Finally, Section V summarizes the study's outcomes.

II. PROBLEM DEFINITION AND OBJECTIVE

In this study, we have addressed the problem of how to develop an AI-aided tactics generator for uncrewed surface vessels (USVs) for protection of critical national infrastructure and maritime assets in face of surface vehicle attacks as illustrated in Figure 1.

The general mission profile for the pursuer USV includes initially either waiting near the protected platform or patrolling. When an approaching evader is localised and identified, the pursuer USV initiates the pursuing policy to intercept and neutralize the evader.

To accomplish this mission, an AI-aided defense strategy must overcome several challenges. The pursuer policy must navigate the USV to neutralize the evader in minimum time for protection. The steering commands must adhere to the vehicle's dynamic limits. Given the safety-critical nature of the mission, a safe distance must be maintained while tracking to avoid collision with the evader. When facing a weaponized evader USV, it is necessary to approach from behind (astern approach) to avoid defeat. Additionally, the defense strategy

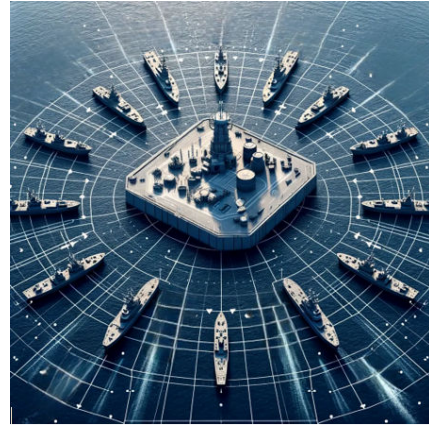


Figure 1. Scenario Illustration of Infrastructure Evasion

must be capable of tracking and countering intelligent maneuvers by the evader.

A. Unmanned Surface Vehicle Model

The USVs are assumed to exhibit point mass characteristics. The equations motion model for an unmanned surface vehicle relative to inertial reference frame centered at the sea surface, considering 2 degrees of freedom, are expressed as follows:

$$\begin{aligned}\dot{x} &= V \cos \theta \\ \dot{y} &= V \sin \theta \\ \dot{V} &= a_v \\ \dot{\theta} &= w\end{aligned}\tag{1}$$

In this motion model, x and y denote the vehicle's position coordinates, V represents velocity magnitude, and θ indicates the heading angle. The control inputs a_v and w correspond to longitudinal acceleration and angular velocity, respectively.

The maneuver capabilities of both pursuer and evader unmanned surface vehicles, defined by their ability to change velocity and heading, are constrained by dynamic models that are functions of current velocity of the vehicle. Figure 2 features illustrative examples of these dynamic constraints alongside several USVs currently on the market, intended purely for illustrative purposes.

III. METHOD

In this section, we present the methodology and framework for developing AI-aided defense tactics for a pursuer USV against an evader. The baseline pursuing policy is formulated using scientific machine learning.

A. Scientific Machine Learning

The principle of integrating governing physics into the training processes of machine learning models is to enhance their generalization capabilities. In accordance with this approach, the concept of scientific machine learning, implemented using Universal Ordinary Differential Equations (UODEs), has been

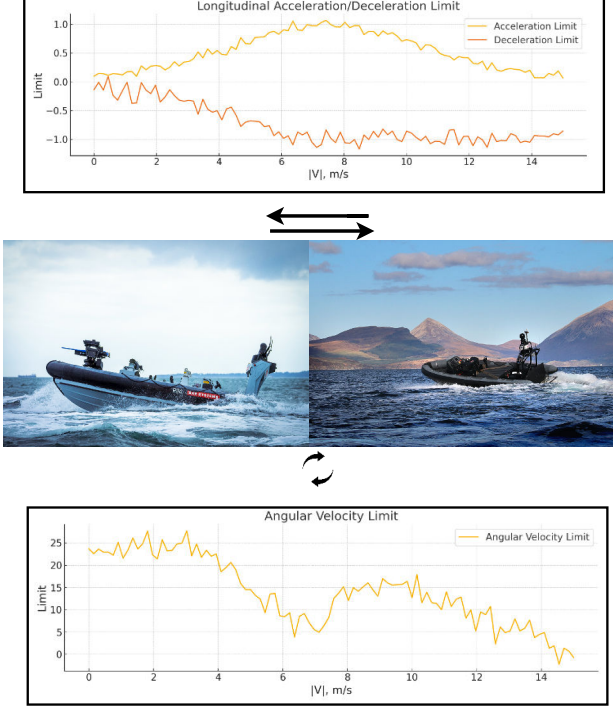


Figure 2. Illustration of Example USVs with Vehicle Dynamics Limit Trends

proposed and formalized in [31] for the purpose of scientific model discovery.

General training framework for SciML is illustrated in Figure 3 .

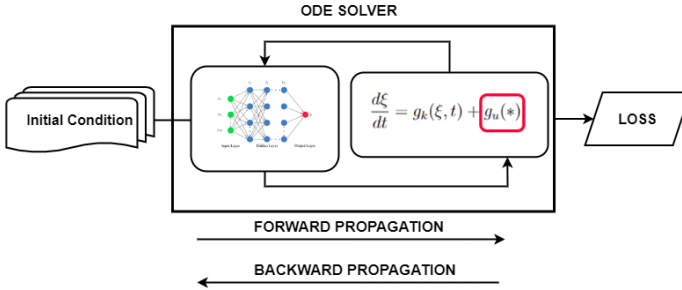


Figure 3. SciML flow

The main idea of the UODEs involves formulating an ordinary differential equation that incorporates a mechanistic model with a universal approximator to estimate unknown dynamics. Specifically, a simple first-order ODE defined by the ground truth model $\frac{d\xi}{dt} = f(\xi, t)$ can be represented in UODE form as $\frac{d\xi}{dt} = g_k(\xi, t) + g_u(*)$, where the known part of the ground truth system is denoted with the function of $g_k(\xi, t)$ and $g_u(*)$ represents the unknown dynamics that we want to discover. This unknown component is approximated with a universal approximator, specifically a neural network,

expressed as $g_u(*) \approx \mathcal{NN}(\mathcal{X}, W_{NN})$. During each forward propagation, the ODE system is simulated by solving it, concluding with the calculation of a scalar loss function, such as the mean square error between the ODE solution and real-world experimental data that represents the ground truth model. In backward propagation, continuous-time gradients of the network parameters with respect to the loss function are computed using adjoint sensitivity methods detailed in [31], [32], [33]. After gradient computation, network parameters are updated using gradient descent methods.

B. Pursuit Evasion Problem Formulation with SciML

In addressing the USV tactic generation problem, we have reformulated our pursuit-evasion problem and treating it as a scientific model discovery problem similar to framework described. The motion model presented earlier is deconstructed into components of known and unknown dynamics. Pursuer's defence tactic generation policy meaning the controls of the vehicle is conceptualized as a learnable, unknown dynamical model. It is modelled utilizing an artificial neural network parameterized with W_{NN} including the network weights and biases.

$$\pi(\mathcal{X}(t), W_{NN}) = \begin{bmatrix} a_{v_{REF}}(t) \\ w_{REF}(t) \end{bmatrix} \approx \mathcal{NN}(\mathcal{X}(t), W_{NN}) \quad (2)$$

In Equation 2, $\mathcal{X}(t)$ represents the feature vector of the neural network policy model, which will be designed accordingly to performance objectives described in subsequent sections.

Consistent with the scientific model discovery approach described, the mechanistic motion model in Equation is reformulated to include both known and unknown dynamics as follows:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{V} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} V \cos \theta \\ V \sin \theta \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mathcal{NN}^{(1)}(\mathcal{X}(t), W_{NN}) \\ \mathcal{NN}^{(2)}(\mathcal{X}(t), W_{NN}) \end{bmatrix} \quad (3)$$

Where, In $\mathcal{NN}^{(*)}()$, “*” shows the $*_{th}$ output of the neural network policy model.

C. Performance Index as Loss Functions

In the UODEs framework detailed in Equation 3, learning the decision-making policy model for the pursuer involves training the artificial neural network. To do so, we need to define a performance index to enforce the pursuer for following the evader while also meeting the mission objectives. Accordingly, we have developed the following loss function to serve this purpose.

$$\mathcal{L}_{total} = \lambda_F \mathcal{L}_F + \lambda_{CE} \mathcal{L}_{CE} + \lambda_{SC} \mathcal{L}_{SC} + \lambda_{CA} \mathcal{L}_{CA} + \lambda_{AA} \mathcal{L}_{AA} \quad (4)$$

In Equation 4, we have presented the total loss function that includes various components crafted for specific objectives and aggregated using a weighted sum approach. The component $\mathcal{L}_F$ is associated with the pursuer's objective to follow the evader. $\mathcal{L}_{CE}$ accounts for the control effort loss. $\mathcal{L}_{SC}$ addresses the pursuer's state constraints, including maximum and minimum state limits. $\mathcal{L}_{CA}$ focuses on avoiding collisions between the pursuer and the evader. Additionally, we have used $\mathcal{L}_{AA}$ component to guide the pursuer to execute the astern approach maneuver while intercepting the evader. The coefficients $\lambda_F, \lambda_{CE}, \lambda_{PC}, \lambda_{CA}$ and λ_{AA} scale the importance of these components within the optimization process.

The primary objective of the pursuer is to follow and intercept the evader in the shortest possible time. To align with this goal, we have considered that minimizing the distance between the pursuer and the evader is crucial. However, it is also crucial for the pursuer to maintain a safe tracking distance from the evader to avoid collisions. Consequently, the loss component $\mathcal{L}_F$ can be expanded as follows:

$$\mathcal{L}_F = \left| DTR - \left| \vec{P}_P(t) - \vec{P}_E(t) \right| \right| \quad (5)$$

Where, $\vec{P}_P(t) = \begin{bmatrix} x_P(t) \\ y_P(t) \end{bmatrix}$ and $\vec{P}_E(t) = \begin{bmatrix} x_E(t) \\ y_E(t) \end{bmatrix}$ represent the position vectors of the pursuer and evader, respectively. DTR stands for the "Desired Tracking Range".

To align with the objective of minimizing capture time, we have included the minimization of angular velocity control effort in our loss definition. Minimizing control effort helps guide the pursuer to take the most efficient action, steering it toward the evader along the shortest possible path.

$$\begin{aligned} \mathcal{L}_{CE} &= \frac{1}{N} \sum_0^N w_{REF}^2 \\ &= \frac{1}{N} \sum_0^N \left(\mathcal{NN}^{(2)}(\mathcal{X}(t), W_{NN}) \right)^2 \end{aligned} \quad (6)$$

For state constraints, we have implemented continuous upper and lower bounds on the heading angle to prevent the pursuer from rotating around itself with divergent behavior during training iterations. This approach helps to ensure a more stable training process. Additionally, we have included endpoint constraints on velocity and its direction to further guide the pursuer's movement and enhance the effectiveness of the training.

$$\begin{aligned} \mathcal{L}_{SC} &= \lambda_{SC1} (||\text{relu}(\theta(t) - \theta_{upper})|| + ||\text{relu}(-\theta(t) + \theta_{lower})||) \\ &\quad + \lambda_{SC2} \left((V_x^P(t_f) - V_x^E(t_f))^2 + (V_y^P(t_f) - V_y^E(t_f))^2 \right) \end{aligned} \quad (7)$$

Where, $\text{relu}(\cdot)$ denotes the rectified linear unit (ReLU) function, which outputs the input value if the input is greater than zero; otherwise, it outputs zero. θ is the pursuer's heading angle and $\theta_{upper}, \theta_{lower}$ are the upper and lower limits of the heading angle. Furthermore, V_x^P, V_y^P and V_x^E, V_y^E are the

velocity components of the pursuer and evader along the x and y directions, respectively.

Considering safety requirements, we have implemented the collision avoidance loss function between the pursuer and the evader as follows:

$$\mathcal{L}_{CA} = \sum_1^N \left| DCAR - \left| \text{relu}(\vec{P}_P(t) - \vec{P}_E(t)) \right| \right| \quad (8)$$

Where, $DCAR$ stands for "Desired Collision Avoidance Range" which is determined by considering the size information of the example USVs involved.

To successfully execute an astern approach for intercepting the evader, we have designed a loss function that operates within the evader's body reference frame. This frame of reference provides key information about the relative positioning of the pursuer, specifically indicating whether the pursuer is ahead of or behind the evader. This orientation is essential for adjusting the pursuer's strategy to either close in from behind or adjust its path to move into an astern position effectively.

$$\mathcal{L}_{AA} = ||(\text{relu}(-\tilde{R}_x^{EBF}(t)))|| + M() * ||(\text{relu}(DSD - |\tilde{R}_y^{EBF}(t)|))|| \quad (9)$$

Where, $\tilde{R}_x^{EBF}(t)$ and $\tilde{R}_y^{EBF}(t)$ are the x and y components of the relative position vector between the pursuer and the evader, expressed in the evader's body frame (EBF). DSD stands for "Desired Separation Distance". Additionally, $M()$ is a mask function which assigns a value of 0 when the pursuer is positioned behind the evader, and a value of 1 when the pursuer is ahead of the evader.

$$M() = \begin{cases} 1, & \text{if } R_x^{EBF}(t) \leq 0 \\ 0, & \text{otherwise} \end{cases} \quad (10)$$

From Equation 9, astern approach loss function $\mathcal{L}_{AA}$ consists of two components. The first component aims to guide the pursuer to decrease the distance to the evader and follow from behind, penalizing the pursuer when it is positioned in front of the evader. The second component ensures that the pursuer maintains a safe separation (DSD) from the target line when it is in front of the evader, addressing safety requirements. Together, these components effectively constitute the astern approach maneuver.

D. Vehicle Dynamics Integration

To incorporate vehicle dynamics as control constraints, we have approximated the system dynamic limits using shallow neural networks (One Hidden Layer Networks) $\mathcal{NN} : R^1 \rightarrow R^1$. These networks take the vehicle velocity as input and output the corresponding vehicle dynamic limits. The training utilizes a dataset derived from real-world experiments involving an experimental USV. Furthermore, physical constraints are included as a Lagrangian penalty in the optimization objective, following the physics-informed neural networks framework, to ensure the limit functions behave well and generalize

effectively. An example objective function for training the longitudinal acceleration upper limit model is provided below.

$$J = MSE(y_{synthetic} - \hat{y}) + \lambda relu(-\hat{y}) \quad (11)$$

The equation utilizes $MSE()$ to denote the mean squared error between, $y_{synthetic}$ and $\hat{y}$ representing the synthetic data from the vehicle dynamics model and the network approximation value, respectively. The coefficient serves as a loss weight, balancing the importance within the objective function. The $relu()$ function is employed to penalize the model, preventing negative values for the acceleration upper limit.

After approximating the vehicle dynamics models with shallow neural networks, we integrated the system dynamics into the scientific machine learning training process by employing a neuron-wise adaptive tanh activation function in the final layer of the policy network.

$$Adaptive\ Tanh := \frac{e^{n\alpha x} - e^{-n\alpha x}}{e^{n\alpha x} + e^{-n\alpha x}} \quad (12)$$

The main idea of adaptive activation functions is to make them trainable to enhance their ability to model nonlinear behavior. In Equation 12, α is the trainable parameter used to adjust the slope of the tanh activation function. Moreover, Ameya et al. highlighted that adding a non-trainable scaling factor n , into the equation contributes to the convergence speed [34]. These adaptive activation functions can be applied either layer-wise, where a single trainable parameter is used uniformly across the layer, or neuron-wise, with each neuron having its own independent trainable parameter, matching the number of neurons in the layer [35]

This choice of activation function allows for dynamic scaling and adjustment of outputs, ensuring that the network's response remains within the physically plausible limits dictated by the vehicle's dynamics.

Employing the neuron-wise adaptive Tanh activation function in the final layer of the policy network effectively bounds the output variables within the range $[1, 1]$. Subsequently, a dynamic re-scaling operation maps this $[1, 1]$ interval to the current minimum and maximum limits as determined by the output of the trained vehicle dynamic's limit networks. This mechanism matches the real-world limit values, thus integrating the policy outputs directly with the system's dynamic constraints and preventing any violation of these limits. The re-mapping operation is given by the following formula.

$$a_{V_{REF}} = \mathcal{NN}^{(1)}(\mathcal{X}(t), W_{NN}) \frac{a_V^{max}(V(t)) - a_V^{min}(V(t))}{2} + \frac{a_V^{max}(V(t)) + a_V^{min}(V(t))}{2} \quad (13)$$

$$w_{REF} = \mathcal{NN}^{(1)}(\mathcal{X}(t), W_{NN}) \frac{w^{max}(V(t)) - w^{min}(V(t))}{2} + \frac{w^{max}(V(t)) + w^{min}(V(t))}{2} \quad (14)$$

Where, $a_V^{max}, a_V^{min}, w^{max}, w^{min}$ are the corresponding trained neural network models used for approximating the maximum and minimum longitudinal acceleration and angular velocity limits of the vehicle, respectively.

E. Simulation and Optimization

After establishing our performance index through specific loss functions and integrating vehicle dynamics, we conducted our scientific machine learning optimization processes using the Julia programming language [36]. This choice was made due to Julia's high performance in numerical and scientific computing, effectively facilitating complex computations and model training.

To perform the optimization loop, we defined optimization scenarios that establish the initial conditions and movement behaviors for both the pursuer and the evader.

For training purposes, we have considered to train our pursuer agent against a simple evader, which lacks intelligence or the capability to perform maneuvers during simulations. This simplification means that throughout a simulation, the evader maintains a constant speed and heading angle.

Specifically:

- The initial position of the evader is stochastically sampled from all quadrants within a 500-meter length square centered at the origin, denoted as $x_0^E, y_0^E \in \mathcal{U}(-500m, 500m)$.
- The initial velocity magnitude of the evader is sampled from a uniform distribution $V_0^E \in \mathcal{U}(1\frac{m}{s}, 17\frac{m}{s})$ and remains constant throughout a single simulation.
- The initial heading angle of the evader is sampled from a uniform distribution $\theta_0^E \in \mathcal{U}(-180^\circ, 180^\circ)$ and remains constant throughout a single simulation.

For the pursuer, we created a structured initial conditions grid:

- The initial position of the pursuer is distributed over 10 equally spaced intervals on circles centered at the origin with radius of 50 meters and 100 meters, resulting in a total of 20 distinct initial positions.
- The initial velocity magnitude of the pursuer is divided into 5 evenly spaced samples within the range $[1\frac{m}{s}, 17\frac{m}{s}]$
- The initial heading angle of the pursuer is divided into 10 evenly spaced samples spanning $[-180^\circ, 180^\circ]$

This configuration of initial conditions for the pursuer generates a grid by taking permutations of the states, resulting in 2000 distinct initial condition combinations.

The overall policy optimization framework, which utilizes SciML training for our pursuit-evasion formulation, is illustrated in Figure 3 and detailed in Algorithm 1.

The SciML training flowchart in Figure 4 and corresponding Algorithm 1 can be briefly explained as follows: The initial

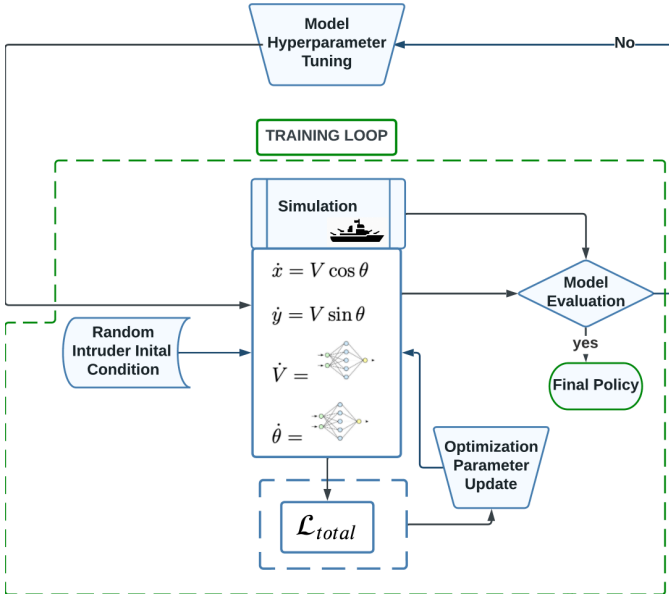


Figure 4. Policy development flowchart for scientific machine learning methodology

Algorithm 1 SciML minibatch training algorithm

```

1 Input: ODE Solver(ODE, IC's), Epoch Size: N, Mini-Batch Size: M
2 Data: Initial Conditions Grid for Pursuer
3 Initialization: Initialization of Network Weights,  $\theta_{NN}$ 
4 Optimization Variable:  $W_{NN}$ 
5 for  $epoch = 1, 2, \dots, N$  do
6   Generate Minibatches
7   for  $Minibatch = 1, 2, \dots$  do
8     for Pursuer IC's in Minibatch do
9        $x^E(t_0) \leftarrow$  Sample the Evader's Initial Condition
10       $x(t)_i, u(t)_i \leftarrow$  Run the  $simulation(IC_i, x^E(t_0), W_{NN})$ 
11      Compute Loss,  $\mathcal{L}_i \leftarrow x(t)_i$ 
12      Compute Gradient,  $\left(\frac{d\mathcal{L}_i}{dW_{NN}}\right)_i \leftarrow \mathcal{L}_i, x(t)_i, u(t)_i, W_{NN}$ 
13    end for
14    Compute Mean Loss,  $\mathcal{L}_{mean} = \frac{1}{M} \sum_{i=1}^M \mathcal{L}_i$ 
15    Compute Mean Gradient:
16       $\left(\frac{d\mathcal{L}}{dW_{NN}}\right)_{mean} = \frac{1}{M} \sum_{i=1}^M \left(\frac{d\mathcal{L}_i}{dW_{NN}}\right)_i$ 
17    Gradient Descent with Adam Optimizer:
18       $W_{updated} \leftarrow \mathcal{L}_{mean}, \left(\frac{d\mathcal{L}}{dW_{NN}}\right)_{mean}$ 
19    Update Network Weights,  $W_{NN} \leftarrow W_{updated}$ 
20  end for
21 end for

```

conditions' training grid for pursuer states is given as input to the algorithm. The first step of the algorithm involves initializing the weights of the neural network that models the decision-making policy. After successful initialization, mini batches are randomly formed from the initial condition grid created for the pursuer agent. Then, a simulation is run for each initial condition in the mini batches. For each mini-batch simulation, the initial condition states for the evader are sampled stochastically as explained. Subsequently, the defined loss functions are computed along with the policy gradient with respect to total loss given in Equation 4. After completing all the initial conditions in a mini batch, the average loss and gradient calculations are performed. This makes conducting gradient descent over the policy network possible. After updating the network weights for all mini

batches, a new epoch starts. At the beginning of a new epoch, mini batches are created randomly. The same steps are followed until convergence is achieved.

IV. RESULTS

In this section, we provide a brief overview of the final form of the policy architecture, hyperparameters used in the training processes and present the results from various simulation scenarios. Our simulations includes several cases: a simple evader heading directly towards infrastructure, a scenario incorporating actuator delay, and a scenario featuring a maneuvering evader. These settings help to evaluate the robustness and efficiency of our developed policy under different conditions.

A. Policy Architecture and Hyper-Parameters

1) *Policy Architecture:* As outlined in Equation 2, we have modeled pursuer agent's policy by using a neural network architecture. The loss components, defined in Equations 5-9 based on mission requirements, contribute to designing the feature vector $\mathcal{X}(t) \in \mathcal{R}^{11}$. This vector contains essential information for the agent's decision-making at time t .

$$\mathcal{X}(t) = \begin{bmatrix} \vec{R}_{P2E}^I(t) \\ \vec{R}_{P2E}^{EBF}(t) \\ \dot{R}_{P2E}(t) \\ ||\vec{V}_P(t)|| \\ \vec{V}_P(t) \\ \vec{V}_E(t) \\ \dot{\lambda}_{LOS_{P2E}}(t) \end{bmatrix} \quad (15)$$

The vectors $\vec{R}_{P2E}^I(t)$, $\vec{R}_{P2E}^{EBF}(t)$ denote the relative distance between the pursuer and the evader, expressed in the inertial reference frame and the evader body fixed reference frame, respectively. $\dot{R}_{P2E}(t)$ indicates the time derivative of the relative position vector, highlighting the dynamic distance changes between pursuer and evader over time. The velocity vectors of the pursuer and evader are denoted as $\vec{V}_P(t)$ and $\vec{V}_E(t)$, with $||\vec{V}_P(t)||$ specifying the magnitude of the pursuer's velocity. Lastly, $\dot{\lambda}_{LOS_{P2E}}(t)$ represents the rate of change of the line-of-sight angle between the pursuer and the evader, capturing the angular dynamics of their engagement.

The specific details of the policy architecture, including layer configurations, activation functions, and other number of parameters, are summarized in Table I

Table I
NEURAL NETWORK POLIC ARCHITECTURE

Layer	Type	# units	Activation
Input	Scaling	11	-
Layer 1	Dense	$11 * n_1$	Tanh
Layer 2	Dense	$n_1 * n_2$	Tanh
Layer 3	Dense	$n_2 * 2$	NA-Tanh
Output	Re-Scaling	2	-

Table I shows that the neural network architecture consists of three layers, each with 16 neurons ($n_1 = n_2 = 16$),

and includes Scaling and Re-Scaling layers at the input and output stages, respectively. Min-max scaling is applied for the scaling operation, and re-scaling follows the method described in Section III-D at Equation 14. The activation function for each layer is the $\tanh$ function, except for the last layer, which uses the $NA - \tanh$ function—a neuronwise adaptive tanh function.

2) *Hyper-Parameters*: SciML’s policy training process involves tuning hyper-parameters such as cost coefficients, mini-batch size, simulation time, and learning rate. We have used the grid search algorithm for this purpose. The complete training setup and the optimized hyper-parameter values are summarized in Table II.

Table II
TRAINING SETUP AND HYPERPARAMETERS

Object	Component/Description
ODE Solver	Tsit5 (Tsitouras 5/4 Runge-Kutta method)
	Adjoint Method $\rightarrow$ Interpolating Adjoint
	$\Delta t \rightarrow$ Variable
	Time Span=[0, 80], s
Cost Coefficients	Absolute/Relative Tolerances = 10^{-5}
	$\lambda_F = \frac{t}{8}$
	$\lambda_{CE} = 0.05, \lambda_{CA} = 1$
	$\lambda_{SC1} = 100, \lambda_{SC2} = 10$
Cost Parameters	$\lambda_{AA} = 0.3$
	$DTR = 20, m, DCAR = 10, m$
	$DSD = 20, m$
	ADAM
Optimizer	Mini-Batch Size, $M = 1$
	Learning Rate, $\zeta = 0.0003$
	Momentum decay $\beta = (0.9, 0.999)$

B. Simulation Results

1) *Simple Evader Case-1*: In this section, we present the simulation results of a pursuer using the SciML policy against a simple evader, which directly approaches the infrastructure at a constant speed and heading without any strategic maneuvering. The initial conditions for both the pursuer and evader are summarized in Table III.

Table III
SCENARIO-1 INITIAL CONDITIONS SETUP

Vehicle	Initial Conditions
Pursuer	$x_0 = y_0 = 0$
	$V_0 = 0$
	$\theta_0 = 90^\circ$
Evader	$x_0 = -700, m \quad y_0 = 700, m$
	$V_0 = 17, \frac{m}{s}$
	$\theta_0 = -45^\circ$

Figure 5 displays the time-elapsed trajectories of both the pursuer and evader, with the pursuer’s path in blue and the evader’s in orange, as the evader heads straight for the infrastructure. Figure 6 illustrates the changing distance between the pursuer and evader over time. The simulation results show that the pursuer, guided by our trained policy, successfully maneuvers towards the capture zone and maintains tracking of the evader. A key aspect of performance highlighted here is that the pursuer executes an astern approach to maintain

safety. Additionally, Figure 6 indicates that the pursuer keeps a safe distance, minimizing collision risks. These findings confirm that the trained policy meets the defined performance requirements effectively.

Figure 7 shows the SciML policy’s commanded reference longitudinal acceleration and angular velocity over time. The blue lines represent the control action values generated by the policy network, while the red dashed lines show the system dynamics’ upper and lower boundaries. The data are normalized to the range of $[-1, 1]$, with +1 and -1 representing the upper and lower limits, respectively. This normalization is due to confidentiality reasons. The results demonstrate that the trained policy operates within the vehicle’s dynamic limits, with no commands exceeding these boundaries, which is a direct outcome of the vehicle dynamics integration strategy employed. Additionally, the action values from our policy exhibit a bang-bang control type, typical of minimum time control laws, indicating that the policy functions as an optimal controller.

2) *Simple Evader Scenario-2*: In this scenario, we followed the same setup as in Scenario-1, with the initial conditions of a simple evader summarized in Table IV. However, to assess the generalization capability of our approach, we introduced actuator delay into the system dynamics as a form of uncertainty. It is important to note that the policy optimization process did not originally account for actuator delay.

Table IV
SCENARIO-1 INITIAL CONDITIONS SETUP

Vehicle	Initial Conditions
Pursuer	$x_0 = y_0 = 0$
	$V_0 = 0$
	$\theta_0 = 90^\circ$
Evader	$x_0 = 500, m \quad y_0 = 600, m$
	$V_0 = 14, \frac{m}{s}$
	$\theta_0 = -130^\circ$

The reference longitudinal acceleration and angular velocity commanded by the SciML policy are subject to a first-order dynamic actuator delay, modeled as follows:

$$\begin{aligned}
 a_{XCOM} &= \mathcal{NN}^{(1)}(\mathcal{X}(t), W_{NN}) \\
 w_{COM} &= \mathcal{NN}^{(2)}(\mathcal{X}(t), W_{NN}) \\
 \dot{a}_x &= \frac{(a_{XCOM} - a_x)}{\tau} \\
 \dot{w} &= \frac{(w_{COM} - w)}{\tau}
 \end{aligned} \tag{16}$$

In Equation 16, a_{XCOM} and w_{COM} represent the commanded control actions from the SciML policy. The variables a_x and w denote the actual realized control actions, which are affected by actuator delay characterized by a time constant $\tau = 1$ seconds.

Figure 8 displays the time-elapsed trajectories of the pursuer and evader in Scenario-2, with the pursuer’s path marked in blue and the evader’s in orange. Figure 9 tracks the relative distance between the pursuer and evader over time. Additionally, Figure 10 illustrates the time evolution of the

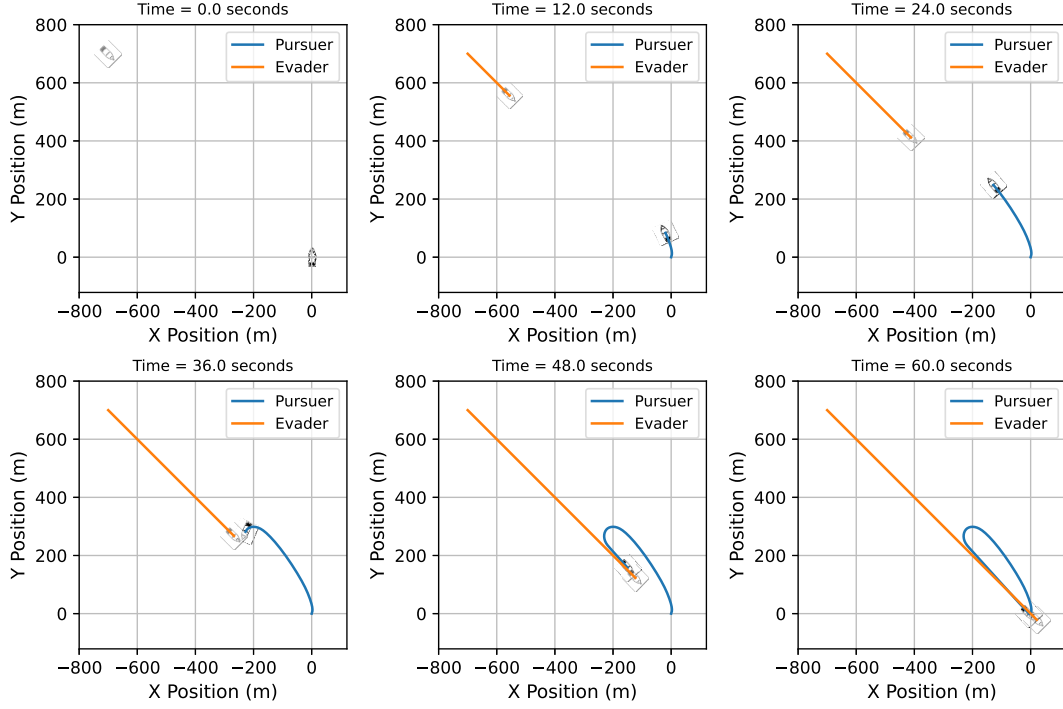


Figure 5. Pursuer and simple evader elapsed time trajectories for scenario-1

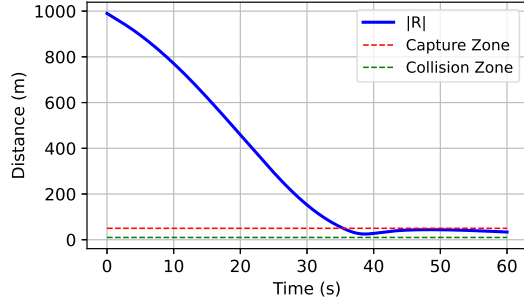


Figure 6. Relative distance between pursuer and evader over time for scenario-1

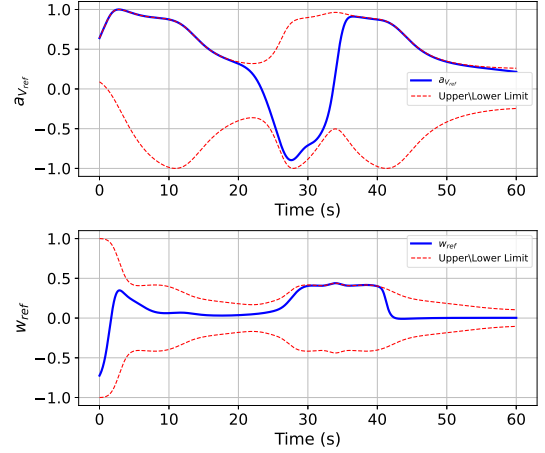


Figure 7. SciML Policy control actions over time for scenario-1

policy's commanded reference control actions and the realized actions in Scenario-2. The blue lines represent the reference control action values generated by the policy network, while the orange lines show the realized control actions affected by actuator delay dynamics. Despite the the presence of significant actuator delay, which was not accounted for during the policy training, the simulation results show that the pursuer, guided by our trained policy, effectively approaches the evader and skillfully executes the astern approach maneuver. This demonstrates the generalization capability and the robustness of the policy in handling dynamics not explicitly included in its training.

Although there is a performance degradation compared to the Scenario-1 results, it remains within acceptable levels, as no unstable behavior is observed. The pursuer effectively

tracks the evader and manages to avoid collision risks. However, unlike in Scenario-1, the pursuer does not maintain a consistent distance within the capture zone after executing an astern approach, due to the actuator delay impacting its ability to accelerate promptly. This performance loss could be eliminated by incorporating delay dynamics into the training processes, which would allow the policy to better handle delays.

3) *Manoeuvring Evader: Scenario-3:* To further assess the generalization capability of our trained SciML policy, we conducted tests against an evader executing an S-curve-like

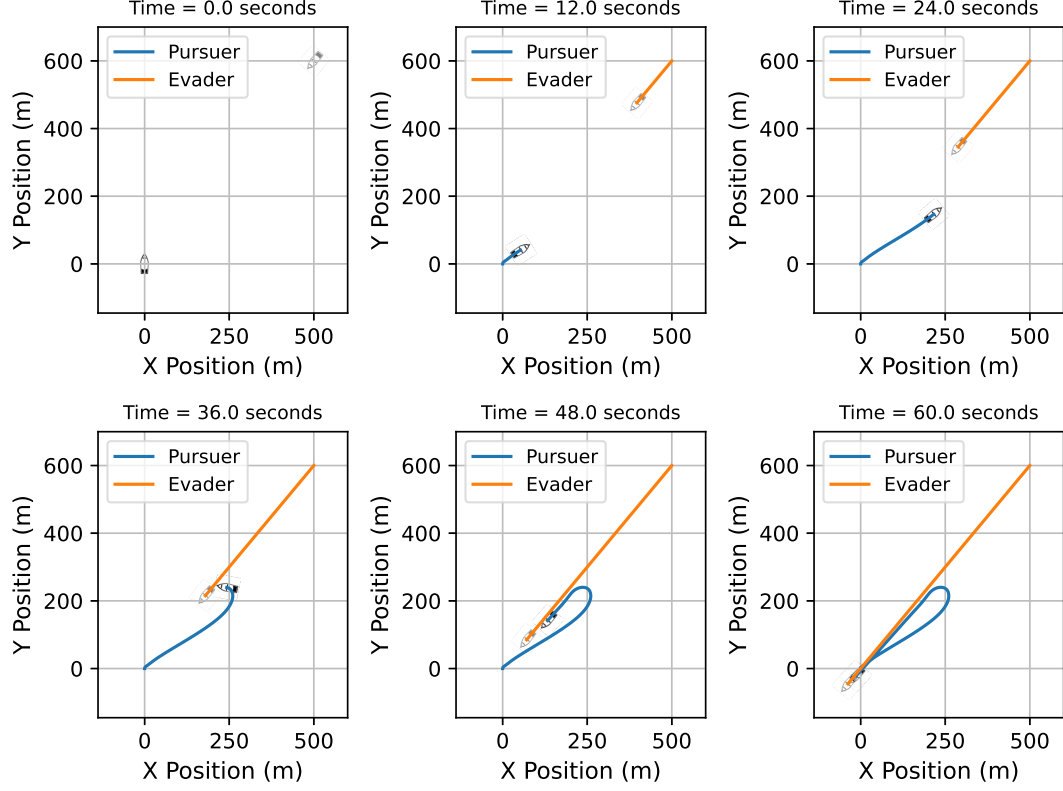


Figure 8. Pursuer and simple evader elapsed time trajectories for scenario-2

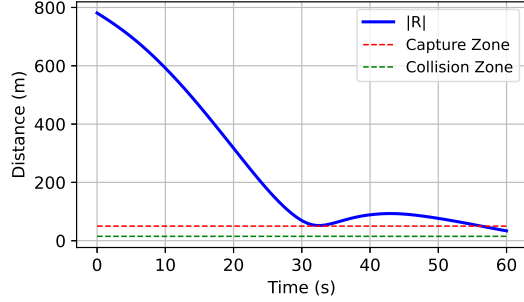


Figure 9. Relative distance between pursuer and evader over time for scenario-2

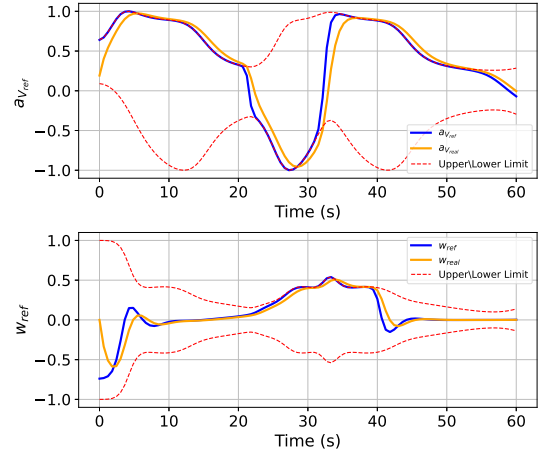


Figure 10. SciML Policy control actions over time for scenario-2

trajectory, representing an out-of-distribution scenario. Since, this type of evader trajectory was not encountered during the training phase; instead, as previously mentioned, the policy was trained against simpler evaders maintaining constant velocity and heading throughout a single simulation. This testing scenario helps evaluate how well the policy adapts to complex and unexpected maneuvers from the evader.

Figure 11 illustrates the elapsed time trajectories of the pursuer and evader, showing the paths from the initial to the final time. The pursuer's path is depicted by the blue line, while the evader's path, following a predefined S-curve-like trajectory, is shown in orange. Despite the pursuer's policy

not being specifically trained to handle maneuvering evaders, the simulation results demonstrate the policy's exceptional generalization ability. It successfully intercepts the evader and maintains effective tracking throughout the scenario.

V. CONCLUSION

In this research, we addressed the USV tactic generation problem for protection of critical national infrastructure and

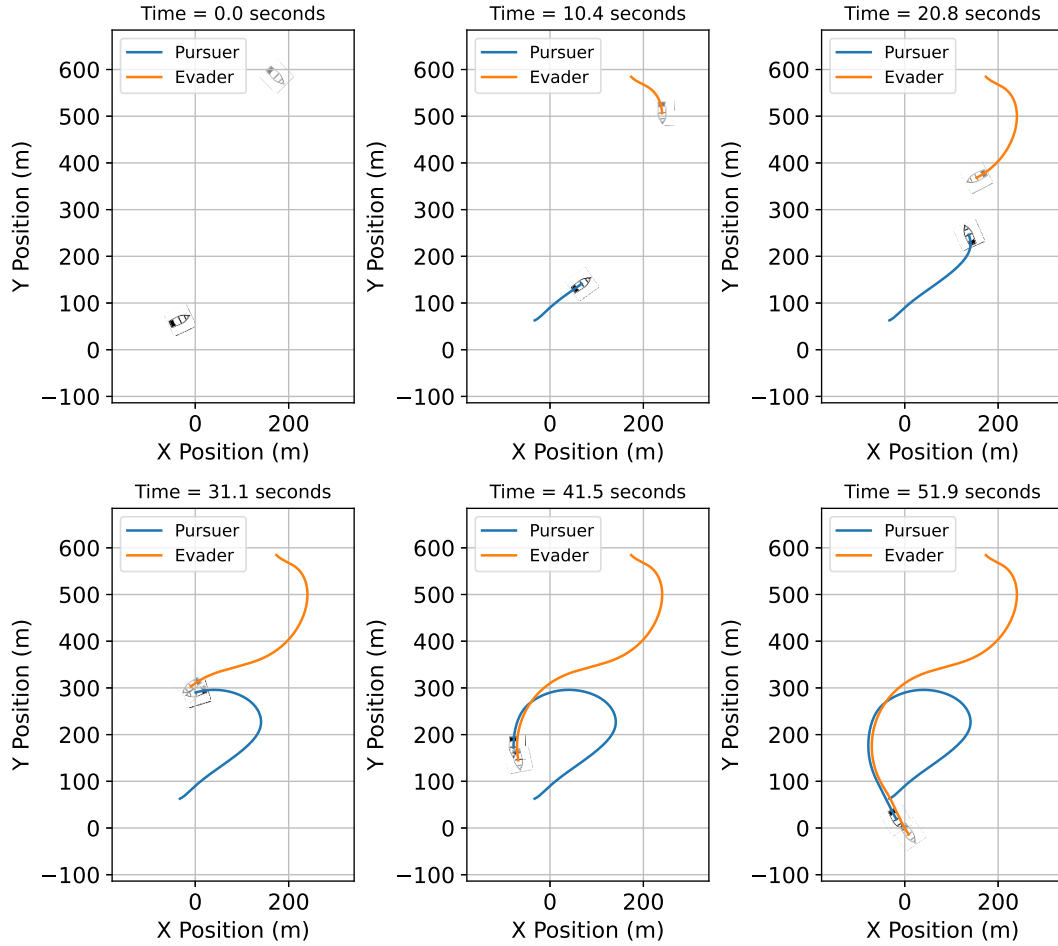


Figure 11. Pursuer and maneuvering evader elapsed time trajectories for scenario-3

maritime assets in face of surface vehicle attacks. We have conceptualized the pursuit-evasion problem as a scientific model discovery problem, wherein the tactic generation policy is treated as a learnable, unknown dynamical model that governs the relative system state.

Our approach has shown outstanding performance through extensive simulations, even in scenarios that significantly deviate from the training domain. Moreover, our approach effectively mitigates various challenges, including achieving near-optimal minimum time-to-capture, diminishing the reliance on extensive training datasets, enhancing convergence speed, and lowering computational demands. We illustrate the results across a wide range of applicable scenarios.

These results underscore the potential of scientific machine learning to revolutionize the field of pursuit-evasion and, by extension, the broader domain of security and defense. The methodology holds promise for various security and defense mechanisms, offering a scalable solution to many similar pursuit-evasion problems including one to many and many to many scenarios in both defensive and offensive setting. Currently, we are focused on implementation of this tactic generator on actual USV platforms to demonstrate the real-

life performance of the proposed approach.

ACKNOWLEDGMENTS

Ugurcan Celik is co-funded by the EPSRC and the BAE Systems under 220124 numbered Industrial CASE award :Towards AI-driven Intelligent Decision Making in Warfare.

REFERENCES

- [1] K. Holkar, K. K. Wagh, and L. M. Waghmare, "An overview of model predictive control," 2010. [Online]. Available: <https://api.semanticscholar.org/CorpusID:5659744>
- [2] Y. Peng, T. Mo, D. Zheng, Q. Deng, J. Wang, D. Qu, and Y. Xie, "Model predictive control-based pursuit-evasion games for unmanned surface vessel," in *Intelligent Robotics and Applications*, H. Yang, H. Liu, J. Zou, Z. Yin, L. Liu, G. Yang, X. Ouyang, and Z. Wang, Eds. Singapore: Springer Nature Singapore, 2023, pp. 267–278.
- [3] M. Sani, B. Robu, and A. Hably, "Limited information model predictive control for pursuit-evasion games," in *2021 60th IEEE Conference on Decision and Control (CDC)*, 2021, pp. 265–270.
- [4] —, "Pursuit-evasion games based on game-theoretic and model predictive control algorithms," in *2021 International Conference on Control, Automation and Diagnosis (ICCAD)*, 2021, pp. 1–6.
- [5] T. Başar and G. Olsder, "Dynamic non-cooperative game theory," vol. 160, 01 1999.

- [6] M. Cococcioni, L. Fiaschi, and P. F. J. Lermusiaux, "Game theory for unmanned vehicle path planning in the marine domain: State of the art and new possibilities," *Journal of Marine Science and Engineering*, vol. 9, no. 11, 2021. [Online]. Available: <https://www.mdpi.com/2077-1312/9/11/1175>
- [7] Y. Wang, X. Wang, W. Zhou, H. Yan, and S. Xie, "Threat potential field based pursuit-evasion games for underactuated unmanned surface vehicles," *Ocean Engineering*, vol. 285, p. 115381, 2023. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0029801823017651>
- [8] N. Rajan, U. Prasad, and N. Rao, "Pursuit-evasion of two aircraft in a horizontal plane," *Journal of Guidance and Control*, vol. 3, no. 3, pp. 261–267, 1980. [Online]. Available: <https://doi.org/10.2514/3.55982>
- [9] B. Jarmark and C. Hillberg, "Pursuit-evasion between two realistic aircraft," *Journal of Guidance, Control, and Dynamics*, vol. 7, no. 6, pp. 690–694, 1984. [Online]. Available: <https://doi.org/10.2514/3.19914>
- [10] D. W. Oyler, P. T. Kabamba, and A. R. Girard, "Pursuit-evasion games in the presence of obstacles," *Automatica*, vol. 65, pp. 1–11, 2016. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0005109815004823>
- [11] H. Gong, S. Gong, and J. Li, "Pursuit-evasion game for satellites based on continuous thrust reachable domain," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 56, no. 6, pp. 4626–4637, 2020.
- [12] J. M. Reimann, "Using multiplayer differential game theory to derive efficient pursuit-evasion strategies for unmanned aerial vehicles," PhD thesis, Georgia Institute of Technology, May 2007, available at <https://repository.gatech.edu/entities/publication/0a078dbb-c05a-42e1-b0ec-57efb140a8f0>.
- [13] N. Wang, Y. Gao, H. Zhao, and C. K. Ahn, "Reinforcement learning-based optimal tracking control of an unknown unmanned surface vehicle," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 7, pp. 3034–3045, 2021.
- [14] J. Woo, C. Yu, and N. Kim, "Deep reinforcement learning-based controller for path following of an unmanned surface vehicle," *Ocean Engineering*, vol. 183, pp. 155–166, 2019. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0029801819302203>
- [15] A. Gonzalez-Garcia, D. Barragan-Alcantar, I. Collado-Gonzalez, and L. Garrido, "Adaptive dynamic programming and deep reinforcement learning for the control of an unmanned surface vehicle: Experimental results," *Control Engineering Practice*, vol. 111, p. 104807, 2021. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0967066121000848>
- [16] N. Wang, Y. Gao, and X. Zhang, "Data-driven performance-prescribed reinforcement learning control of an unmanned surface vehicle," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 12, pp. 5456–5467, 2021.
- [17] J. Xia, Y. Luo, Z. Liu, Y. Zhang, H. Shi, and Z. Liu, "Cooperative multi-target hunting by unmanned surface vehicles based on multi-agent reinforcement learning," *Defence Technology*, vol. 29, pp. 80–94, 2023. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S221491472200215X>
- [18] X. Qu, W. Gan, D. Song, and L. Zhou, "Pursuit-evasion game strategy of usv based on deep reinforcement learning in complex multi-obstacle environment," *Ocean Engineering*, vol. 273, p. 114016, 2023. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0029801823004006>
- [19] W. Gan, X. Qu, D. Song, and P. Yao, "Multi-usv cooperative chasing strategy based on obstacles assistance and deep reinforcement learning," *IEEE Transactions on Automation Science and Engineering*, pp. 1–16, 2023.
- [20] X. Liang, Y. Yang, Q. Wang, W. Wang, and W. Han, "Research on anti-pursuit evasion strategy of unmanned surface vehicle based on t-d3qn," in *2022 5th International Conference on Intelligent Autonomous Systems (ICoIAS)*, 2022, pp. 172–178.
- [21] X. Wang, Y. Wang, W. Zhou, and J. Zhang, "Pursuit-evasion game of unmanned surface vehicles based on deep reinforcement learning," in *2023 4th International Conference on Electronic Communication and Artificial Intelligence (ICECAI)*, 2023, pp. 358–363.
- [22] Y. Wang, L. Dong, and C. Sun, "Cooperative control for multi-player pursuit-evasion games with reinforcement learning," *Neurocomputing*, vol. 412, pp. 101–114, 2020. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S092523122031002X>
- [23] A. T. Bilgin and E. Kadioglu-Urtis, "An approach to multi-agent pursuit evasion games using reinforcement learning," in *2015 International Conference on Advanced Robotics (ICAR)*, 2015, pp. 164–169.
- [24] B. Vlahov, E. Squires, L. Strickland, and C. Pippin, "On developing a uav pursuit-evasion policy using reinforcement learning," in *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2018, pp. 859–864.
- [25] 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.
- [26] J. Liu, S. Liu, H. Wu, and Y. Zhang, "A pursuit-evasion algorithm based on hierarchical reinforcement learning," in *2009 International Conference on Measuring Technology and Mechatronics Automation*, vol. 2, 2009, pp. 482–486.
- [27] Y. Kartal, K. Subbarao, A. Dogan, and F. Lewis, "Optimal game theoretic solution of the pursuit-evasion intercept problem using on-policy reinforcement learning," *International Journal of Robust and Nonlinear Control*, vol. 31, no. 16, pp. 7886–7903, 2021. [Online]. Available: <https://onlinelibrary.wiley.com/doi/abs/10.1002/rnc.5719>
- [28] K. Wan, D. Wu, Y. Zhai, B. Li, X. Gao, and Z. Hu, "An improved approach towards multi-agent pursuit-evasion game decision-making using deep reinforcement learning," *Entropy*, vol. 23, no. 11, 2021. [Online]. Available: <https://www.mdpi.com/1099-4300/23/11/1433>
- [29] Z. Zhou and H. Xu, "Decentralized optimal large scale multi-player pursuit-evasion strategies: A mean field game approach with reinforcement learning," *Neurocomputing*, vol. 484, pp. 46–58, 2022. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0925231221015769>
- [30] S. Cuomo, V. S. Di Cola, F. Giampaolo, G. Rozza, M. Raissi, and F. Piccialli, "Scientific machine learning through physics-informed neural networks: Where we are and what's next," *J. Sci. Comput.*, vol. 92, no. 3, sep 2022. [Online]. Available: <https://doi.org/10.1007/s10915-022-01939-z>
- [31] C. Rackauckas, Y. Ma, J. Martensen, C. Warner, K. Zubov, R. Supekar, D. Skinner, and A. J. Ramadhan, "Universal differential equations for scientific machine learning," *CoRR*, vol. abs/2001.04385, 2020. [Online]. Available: <https://arxiv.org/abs/2001.04385>
- [32] M. Innes, A. Edelman, K. Fischer, C. Rackauckas, E. Saba, V. B. Shah, and W. Tebbutt, "A differentiable programming system to bridge machine learning and scientific computing," *CoRR*, vol. abs/1907.07587, 2019. [Online]. Available: <http://arxiv.org/abs/1907.07587>
- [33] S. Omi, H.-S. Shin, N. Cho, and A. Tsourdos, "Dynamic deep-reinforcement-learning algorithm in partially observed markov decision processes," 2023.
- [34] A. D. Jagtap, K. Kawaguchi, and G. E. Karniadakis, "Adaptive activation functions accelerate convergence in deep and physics-informed neural networks," *Journal of Computational Physics*, vol. 404, p. 109136, 2020. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0021999119308411>
- [35] U. Çelik and M. U. Demirezen, "Optimal reusable rocket landing guidance: A cutting-edge approach integrating scientific machine learning and enhanced neural networks," *IEEE Access*, vol. 12, pp. 16805–16829, 2024.
- [36] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah, "Julia: A fresh approach to numerical computing," *SIAM Review*, vol. 59, no. 1, pp. 65–98, 2017. [Online]. Available: <https://doi.org/10.1137/141000671>



DSPACE

<https://dspace.org/>

Scientific machine learning based pursuit-evasion strategy in unmanned surface vessel defense tactics

Çelik, Ugurcan; Uzun, Mevlut; Inalhan, Gokhan; Woods, Mike

2024-09-29

Institute of Electrical and Electronics Engineers (IEEE)

<https://doi.org/10.1109/dasc62030.2024.10749622>

Celik U, Uzun M, Inalhan G, Woods M. (2024) Scientific machine learning based pursuit-evasion strategy in unmanned surface vessel defense tactics. In: 2024 AIAA DATC/IEEE 43rd Digital Avionics Systems Conference (DASC), 29 September 2024 - 3 October 2024, San Diego, CA, USA