A Robust Physics-Informed Machine Learning Framework for Safe and Optimal Control of Autonomous Systems

Manan Tayal*[1], Aditya Singh*[1,2], Ganga Nair B[1], Shishir Kolathaya[1], Somil Bansal[3]

***Abstract*—As autonomous systems become more ubiquitous in daily life, ensuring high performance with guaranteed safety is crucial. However, safety and performance could be competing objectives, which makes their co-optimization difficult. Learning-based methods, such as Constrained Reinforcement Learning (CRL), achieve strong performance but lack formal safety guarantees due to safety being enforced as soft constraints, limiting their use in safety-critical settings. Conversely, formal methods such as Hamilton-Jacobi (HJ) Reachability Analysis and Control Barrier Functions (CBFs) provide rigorous safety assurances but often neglect performance, resulting in overly conservative controllers. To bridge this gap, we formulate the co-optimization of safety and performance as a robust state-constrained optimal control problem, where performance objectives are encoded via a cost function and safety requirements are imposed as state constraints. We demonstrate that the resultant value function satisfies a Hamilton-Jacobi-Bellman (HJB) equation, which we approximate efficiently using a novel physics-informed machine learning framework. In addition, we introduce a conformal prediction-based verification strategy to quantify the learning errors, recovering a high-confidence safety value function, along with a probabilistic error bound on performance degradation. Through several case studies and hardware experiments, we demonstrate the efficacy of the proposed framework in enabling scalable learning of safe and performant controllers for complex, high-dimensional autonomous systems operating under uncertainty in real-world environments. The accompanying videos can be found on the project website: https://tayalmanan28.github.io/robust-piml-soc/.**

I. Introduction

Autonomous systems are becoming increasingly prevalent across various domains, from self-driving vehicles and robotic automation to aerospace and industrial applications. Designing control algorithms for these systems involves balancing two fundamental objectives: *performance* and *safety*. Ensuring high performance is essential for achieving efficiency and task objectives under practical constraints, such as fuel limitations or time restrictions. For instance, a warehouse humanoid robot navigating to a destination must optimize its route for efficiency. At the same time, safety remains paramount to prevent catastrophic accidents or system failures. These two objectives, however, often conflict, making it challenging to develop control strategies that achieve both effectively.

A variety of data-driven approaches have been explored to integrate safety considerations into control synthesis. Constrained Reinforcement Learning (CRL) methods [1, 2] employ constrained optimization techniques to co-optimize safety and performance where performance is encoded as a reward function and safety is formulated as a constraint. These methods often incorporate safety constraints into the objective function, leading to only a soft imposition of the safety constraints. Moreover, such formulations typically minimize cumulative constraint violations rather than enforcing strict safety at all times, which can result in unsafe behaviors.

Another class of methods involves *safety filtering* [3], which ensures constraint satisfaction by modifying control outputs in real-time. Methods such as Control Barrier Function (CBF)-based quadratic programs (QP) [4] and Hamilton-Jacobi (HJ) Reachability filters [5, 6] act as corrective layers on top of a (potentially unsafe) nominal controller, making minimal interventions to enforce safety constraints. However, because these safety filters operate independently of the underlying performance-driven controller, they often lead to myopic and suboptimal decisions. Alternatively, online optimization-based methods, such as Model Predictive Control (MPC) [7, 8] and Model Predictive Path Integral (MPPI) [9, 10], can naturally integrate safety constraints while optimizing for a performance objective. These methods approximate infinite-horizon optimal control problems (OCPs) with a receding-horizon framework, enabling dynamic re-planning. While effective, solving constrained OCPs online remains computationally expensive, limiting their applicability for high-frequency control applications. The challenge is further exacerbated when dealing with nonlinear dynamics and nonconvex (safety) constraints, limiting the feasibility of these methods for ensuring safety and optimality for real-world systems.

A more rigorous approach to addressing the trade-off between performance and safety is to formulate the problem as a *state-constrained optimal control problem (SC-OCP)*, where safety is explicitly encoded as a hard constraint, while performance is expressed through a reward (or cost) function. While theoretically sound, characterizing the solutions of SC-OCPs is challenging unless certain controllability conditions hold [11]. To address these challenges, [12] proposed an epigraph-based formulation, which characterizes the value function of an SC-OCP by computing its epigraph using dynamic programming, resulting in a Hamilton-Jacobi-Bellman Partial Differential Equation (HJB-PDE). The SC-OCP value function as well as the optimized policy are then recovered from this epigraph. However, dynamic programming suffers from the curse of dimensionality, making it impractical for high-dimensional systems with traditional numerical solvers [13, 14]. Furthermore, the epigraph formulation itself increases the problem's dimensionality, exacerbating computational complexity

* Denotes equal contribution.

[1] Authors are with the Center for Cyber-Physical Systems, Indian Institute of Science, Bangalore, India (email: {manantayal, ganganairb, shishirk}@iisc.ac.in)

[2]Author is with the Department of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia, USA (email:{singhadi@seas.upenn.edu})

[3]Author is with the Department of Aeronautics and Astronautics, Stanford University, USA (email:{somil@stanford.edu})

further. Many techniques for speeding up the computation of solutions to the HJB PDE put restrictions on the type of system allowed [15]. However, solving the HJB PDE for general nonlinear systems remains a key challenge.

Recent advances in Deep Learning have enabled the development of physics-informed machine learning approaches [16, 17] for solving partial differential equations (PDEs) with neural networks. These methods have demonstrated notable effectiveness in addressing high-dimensional PDEs while ensuring that the learned solutions adhere to the governing physical laws. In particular, DeepReach [18] proposes a framework for solving Hamilton–Jacobi–Bellman (HJB) PDEs in safety-critical settings using physics-informed machine learning. However, its exclusive focus on safety neglects performance considerations, resulting in overly conservative control strategies.

A. Contributions and Outline

In this work, we propose a novel algorithmic approach to *co-optimize safety and performance for high-dimensional, real-world autonomous systems*. Specifically, we formulate the problem as a Robust SC-OCP and leverage the epigraph formulation in [12]. To efficiently solve this epigraph formulation, we leverage physics-informed machine learning [19, 20] to learn a solution to the resultant HJB-PDE by minimizing PDE residuals. This enables us to efficiently scale epigraph computation for higher-dimensional autonomous systems, leading to safe and performant policies. To ensure reliable deployment, we introduce a conformal prediction-based safety verification strategy that provides high-confidence probabilistic safety guarantees for the learned policy, thereby mitigating the impact of learning errors on safety. In addition, we develop a performance quantification framework that leverages conformal prediction to compute high-confidence probabilistic bounds on performance degradation.

Section II presents the problem formulation for co-optimizing safety and performance. It then introduces the epigraph reformulation, which converts the original co-optimization problem into a more tractable two-stage optimization framework. Finally, the section derives the associated HJB-PDE conditions that must be satisfied to obtain a solution to the co-optimization problem.

Section III *details our preliminary work presented at International Conference on Machine Learning* [21], which examines the safety-performance co-optimization problem in an idealized setting where the system dynamics are assumed to be known exactly and no disturbances are present, including those arising from adversarial agents, sensor noise, or modeling errors. In this section, we also introduce a receding-horizon execution strategy, *not included in the preliminary work*, which enables the learned policy to be deployed over time horizons longer than those used during training. Through three case studies, we showcase the effectiveness of the proposed approach in co-optimizing safety and performance while remaining scalable to complex, high-dimensional systems.

In practice, robotic systems rarely operate in disturbance-free environments. Disturbances may arise from multiple sources, including modeling inaccuracies, adversarial agents, and sensor noise. Addressing this limitation, Section IV introduces a new extension to disturbance-aware settings, *which is not considered in the preliminary work and constitutes the primary novel contribution of this evolved paper*. Through two case studies involving disturbances generated by different sources, we demonstrate that the proposed framework can effectively co-optimize safety and performance under uncertainty. This highlights the practical applicability of the approach for real-world autonomous systems. To further validate this claim, we conduct hardware experiments, the results of which are presented in Section V. These experiments demonstrate that the proposed method transfers successfully to hardware and can be deployed on real-world autonomous platforms.

II. Problem Setup

Consider a nonlinear dynamical system characterized by the state $x \in \mathcal{X} \subseteq \mathbb{R}^n$, control input $u \in \mathcal{U} \subseteq \mathbb{R}^m$, and disturbance $d \in \mathcal{D} \subseteq \mathbb{R}^k$ (which may arise from multiple sources, including modeling inaccuracies, adversarial agents, and sensor noise) governed by the dynamics $\dot{x}(t) = f(x(t), u(t), d(t))$, where the function $f : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^k \to \mathbb{R}^n$ is locally Lipschitz continuous. In this work, we assume that the dynamics model f is known. However, an accurate model is not required; a nominal model, typically available in practice, is sufficient, with model inaccuracies accounted for as disturbances.

We are given a failure set $\mathcal{F} \subseteq \mathcal{X}$ that represents the set of unsafe states for the system (e.g., obstacles for an autonomous ground robot). The system's performance is quantified by the cost function $C(t, x, \mathbf{u}, \mathbf{d})$, given by:

$$C(t, x(t), \mathbf{u}, \mathbf{d}) = \int_{s=t}^{T} l(x(s))\, ds + \phi(x(T)), \tag{1}$$

where $l : \mathcal{X} \to \mathbb{R}_{\geq 0}$ and $\phi : \mathcal{X} \to \mathbb{R}_{\geq 0}$ are Lipschitz continuous and non-negative functions, representing the running cost over the time horizon $[t, T)$ and the terminal cost at time T, respectively. $\mathbf{u} : [t, T) \to \mathcal{U}$ is the control signal applied to the system and $\mathbf{d} : [t, T) \to \mathcal{D}$ is the disturbance of the system. Using this premise, we define the main objective of this paper:

> ***Objective*** 1. We aim to synthesize an optimal policy $\pi^* : [t, T) \times \mathcal{X} \to \mathcal{U}$ that minimizes the cost function C while ensuring that the system remains outside the failure set $\mathcal{F}$ at all times despite the worst-case disturbance.

A. Robust State-Constrained Optimal Control Problem

To achieve the stated objective, the first step is to encode the safety constraint via a function $g : \mathbb{R}^n \to \mathbb{R}$ such that, $\mathcal{F} := \{x \in \mathcal{X} \mid g(x) > 0\}$. Using these notations, the objective can be formulated as the following Robust State-Constrained Optimal Control Problem (RSC-OCP) to compute the value function V:

Problem 1 (Robust State-Constrained OCP).

$$V(t,x(t)) = \min_{\mathbf{u}} \max_{\mathbf{d}} \int_t^T l(x(s))ds + \phi(x(T)) \quad \text{s.t. } \dot{x} = f(x,u,d), \quad g(x(s)) \leq 0 \quad \forall s \in [t,T] \tag{2}$$

This RSC-OCP enhances the system's performance by minimizing the cost, while maintaining system safety through the state constraint, $g(x) \leq 0$, ensuring that the system avoids the failure set, $\mathcal{F}$. Thus, the policy, π^*, derived from the solution of this RSC-OCP co-optimizes safety and performance.

B. Epigraph Reformulation

Directly solving the RSC-OCP in (2) presents significant challenges due to the presence of (hard) state constraints. To address this issue, we reformulate the problem in its epigraph form [22], which transforms the constrained optimization into a more tractable two-stage optimization problem. This reformulation allows us to efficiently obtain a solution to the RSC-OCP in (2). The resulting formulation is given by:

$$V(t,x(t)) = \min_{z \in \mathbb{R}^+} z \quad \text{s.t. } \hat{V}(t,x,z) \leq 0, \tag{3}$$

where z is a non-negative auxiliary optimization variable, and $\hat{V}$ represents the auxiliary value function. Here, $\hat{V}$ is defined as [12]:

$$\hat{V}(t,x(t),z) = \min_{\mathbf{u}} \max_{\mathbf{d}} \max\{C(t,x(t),\mathbf{u},\mathbf{d}) - z, \max_{s\in[t,T]} g(x(s))\}. \tag{4}$$

Note that if $\hat{V}(t,x,z) < 0$, it implies that $g(x(s)) < 0$ for all $s \in [t,T]$. In other words, the system must be outside the failure set at all times; therefore, the system is guaranteed to be safe whenever $\hat{V}(t,x,z) < 0$.

In this reformulated problem, state constraints are effectively eliminated, enabling the use of dynamic programming to characterize the value function, as we explain later in this section. Intuitively, optimal z (z^*) can be thought of as the *minimum permissible cost* the policy can incur without compromising on safety. From Equation 3, it can be inferred that if $z > z^*$, the safety constraint dominates in the max term, resulting in a conservative policy. Conversely, if $z < z^*$, the performance objective takes precedence, leading to a potentially aggressive policy that might compromise safety.

Furthermore, to facilitate solving the epigraph reformulation, z can be treated as a state variable, with its dynamics given by $\dot{z}(t) = -l(x(t))$. This implies that as the trajectory progresses over time, the minimum permissible cost, z, decreases by the step cost $l(x)$ at each time step. This allows us to define an augmented system that evolves according to the following dynamics:

$$\dot{\hat{x}} = \hat{f}(t,\hat{x},u,d) := \begin{bmatrix} f(t,x,u,d) \\ -l(x) \end{bmatrix}, \tag{5}$$

where $\hat{x} := [x,z]^T$ represents the augmented state. With the augmented state representation and under assumptions A1–A4 of [12]:

$$(\mathcal{A}_1'): \begin{cases} (i)\ f(t,x,u,d) \in \mathbb{R}^d \text{ is continuous,} \\ \text{where } (t,x,u,d) \in [0,T]\times\mathbb{R}^d\times\mathcal{U}\times\mathcal{D}, \\ (ii)\ \exists L \geq 0,\ \forall x,y \in \mathbb{R}^d,\ \forall (u,d) \in \mathcal{U}\times\mathcal{D}, \\ \forall t,s \in [0,T],\ |f(t,x,u,d) - f(s,y,u,d)| \\ \leq L(|x-y| + |t-s|). \end{cases} \tag{6}$$

$$(\mathcal{A}_2'): \begin{cases} (i)\ \ell(t,x,u,d) \in \mathbb{R} \text{ is continuous,} \\ \text{where } (t,x,u,d) \in [0,T]\times\mathbb{R}^d\times\mathcal{U}\times\mathcal{D}, \\ (ii)\ \exists L \geq 0,\ \forall x,y \in \mathbb{R}^d,\ \forall (u,d) \in \mathcal{U}\times\mathcal{D}, \\ \forall t,s \in [0,T],\ |\ell(t,x,u,d) - \ell(s,y,u,d)| \\ \leq L(|x-y| + |t-s|). \end{cases} \tag{7}$$

$$(\mathcal{A}_3'): \varphi : \mathbb{R}^d \to \mathbb{R} \text{ is Lipschitz continuous.} \tag{8}$$

The following convexity assumption on $\hat{f}$ will be considered:

$$(\mathcal{A}_4'): \forall (t,x,d),\ \hat{f}(t,x,\mathcal{U},d) \text{ is a convex set.} \tag{9}$$

The auxiliary value function $\hat{V}(t,x(t),z(t))$ is a unique continuous viscosity solution satisfying the following Hamilton–Jacobi–Isaacs Variational Inequality (HJI-VI):

$$\min\Big(-\partial_t \hat{V} - \min_{\mathbf{u}} \max_{\mathbf{d}} \langle \nabla_{\hat{x}} \hat{V}(t,\hat{x}),\ \hat{f}(\hat{x},u,d)\rangle,\ \hat{V} - g(x)\Big) = 0, \tag{10}$$

$\forall t \in [0,T)$ and $\hat{x} \in \mathcal{X}\times\mathbb{R}$, where $\langle\cdot,\cdot\rangle$ denotes the dot product of vectors. The boundary condition for the PDE is given by:

$$\hat{V}(T,\hat{x}) = \max\left(\phi(x(T)) - z, g(x)\right), \quad \hat{x} \in \mathcal{X}\times\mathbb{R}. \tag{11}$$

Note that by a slight abuse of notations, we have replaced the arguments x, z for $\hat{V}$ with the augmented state $\hat{x}$.

III. Safe and Optimal Control Without Disturbances

When there are no disturbances, we need to solve the following SC-OCP:

Problem 2 (State-Constrained OCP).

$$V(t,x(t)) = \min_{\mathbf{u}} \int_t^T l(x(s))ds + \phi(x(T)) \quad \text{s.t. } \dot{x} = f(x,u), \quad g(x(s)) \leq 0 \quad \forall s \in [t,T] \tag{12}$$

To solve the SC-OCP in Equation (12), we aim to compute the optimal value function V, which minimizes the cost while ensuring system safety. In this section, we outline a structured approach: first, we learn the auxiliary value function $\hat{V}$ using a physics-informed machine learning framework. Then, we apply a conformal prediction-based method to verify safety

and correct for potential learning errors in $\hat{V}$. The final value function V is obtained from the safety-corrected $\hat{V}$ using the epigraph formulation in (3). Lastly, we assess the performance of V through a second conformal prediction procedure. The majority of this section is taken from [21]. Figure 1 gives an overview of the proposed approach. The following subsections provide a detailed explanation of each step, beginning with the methodology for learning $\hat{V}$.

A. Training the Auxiliary Value Function ($\hat{V}$)

The auxiliary value function, $\hat{V}$, satisfies the HJB-PDE in Equation (10), as discussed in Section II-B. Traditionally, numerical methods are used to solve the HJB-PDE over a grid representation of the state space [13, 26], where time and spatial derivatives are approximated numerically. While grid-based methods are accurate for low-dimensional problems, they struggle with the curse of dimensionality – their computational complexity increases exponentially with the number of states – limiting their use in high-dimensional systems. To address this, we adopt a physics-informed machine learning framework, inspired by [18], which has proven effective for high-dimensional reachability problems.

The solution of the HJB-PDE inherently evolves backward in time, as the value function at time t is determined by its value at $t+\Delta t$. To facilitate neural network training, we use a curriculum learning strategy, progressively expanding the time sampling interval from the terminal time $[T,T]$ to the full time horizon $[0,T]$. This approach allows the neural network to first accurately learn the value function from the terminal boundary conditions, subsequently propagating the solution backward in time by leveraging the structure of the HJB-PDE.

Specifically, the auxiliary value function is approximated by a neural network, $\hat{V}_\theta$, where θ denotes the trainable parameters of the network. Training samples, $(t_k, x_k, z_k)_{k=1}^N$, are randomly drawn from the state space based on the curriculum training scheme. The proposed learning framework utilizes a loss function that enforces two primary objectives: (i) compliance with the PDE in (10), using the PDE residual error given by:

$$\mathcal{L}_{pde}\left(t_k, \hat{x}_k|\theta\right) = \| \min\Big\{-\partial_t \hat{V}_\theta\left(t_k, \hat{x}_k\right) - H(t_k, \hat{x}_k), \hat{V}_\theta\left(t_k, \hat{x}_k\right) - g\left(x_k\right)\Big\} \|, \tag{13}$$

where $H(t,\hat{x}) = \min_{u\in\mathcal{U}}\langle \nabla \hat{V}_\theta(t,\hat{x}), \hat{f}(\hat{x},u)\rangle$ and (ii) satisfaction of the boundary condition in (11), using boundary condition loss, given by:

$$\mathcal{L}_{bc}\left(t_k, \hat{x}_k|\theta\right) = \|\max\left(\phi(x_k) - z_k, g(x_k)\right) - \hat{V}_\theta\left(t_k, \hat{x}_k\right)\Big\| \mathbb{1}\left(t_k = T\right). \tag{14}$$

These terms are balanced by a trade-off parameter λ, leading to the overall loss function:

$$\mathcal{L}\left(t_k, \hat{x}_k|\theta\right) = \mathcal{L}_{pde}\left(t_k, \hat{x}_k|\theta\right) + \lambda \mathcal{L}_{bc}\left(t_k, \hat{x}_k|\theta\right) \tag{15}$$

Furthermore, we use the adaptive loss re-balancing scheme proposed in [27] to reduce the impact of λ on the learned value function. Minimizing the overall loss function provides a self-supervised learning mechanism to approximate the auxiliary value function.

Algorithm 1 Safety Verification using Conformal Prediction

Require: $\mathcal{S}$, N_s, β_s, ϵ_s, $\hat{V}_\theta(\hat{x},0)$, $\hat{V}_{\hat{\pi}_\theta}(\hat{x},0)$, M (number of levels to search for δ),

$D_0 \leftarrow$ Sample N_s IID states from $\mathcal{S}_{\delta=0}$
$\delta_0 \leftarrow \min_{\hat{x}_j\in D_0}\{\hat{V}_\theta(0,\hat{x}_j) : \hat{V}_{\hat{\pi}_\theta}(0,\hat{x}_j) \geq 0\}$
$\epsilon_0 \leftarrow$ (19) (using $\alpha_{\delta=0}$)
$\Delta \leftarrow$ Ordered list of M uniform samples from $[\delta_0, 0]$
for $i = 0, 1, \ldots, M-1$ **do**
  while $\epsilon_i \leq \epsilon_s$ **do**
    $\delta_i \leftarrow \Delta_i$
    Update α_{δ_i} from δ_i
    $\epsilon_i \leftarrow$ (19) (using α_{δ_i})
return $\delta \leftarrow \delta_i$

B. Safety Verification

The learned auxiliary value function, $\hat{V}_\theta$, induces a policy, $\hat{\pi}_\theta$, that minimizes the Hamiltonian term $H(t,\hat{x})$ in the HJB-PDE. The policy is given by:

$$\hat{\pi}_\theta(t,\hat{x}) = \arg\min_{u\in\mathcal{U}} \langle \nabla \hat{V}_\theta(t,\hat{x}), \hat{f}(\hat{x},u)\rangle. \tag{16}$$

The rollout cost corresponding to this policy is defined as:

$$\hat{V}_{\hat{\pi}_\theta}(t,\hat{x}) = \max\{C(t,x(t),\mathbf{u}) - z, \max_{s\in[t,T]} g(x(s))\}\Big|_{\mathbf{u}=\hat{\pi}_\theta} \tag{17}$$

Ideally, the rollout cost from a given state under $\hat{\pi}_\theta$ should match the value of the auxiliary value function at that state. However, due to learning inaccuracies, discrepancies can arise. This becomes critical when a state, $\hat{x}_i$, is deemed safe by the auxiliary value function ($\hat{V}_\theta(t,\hat{x}) \leq 0$) but is unsafe under the induced policy ($\hat{V}_{\hat{\pi}_\theta}(t,\hat{x}) > 0$). To address this, we introduce a uniform value function correction margin, δ, which guarantees that the sub-δ level set of the auxiliary value function remains safe under the induced policy. Mathematically, the optimal δ (δ^*) can be expressed as:

$$\delta^* := \min_{\hat{x}\in\mathcal{X}}\{\hat{V}_\theta(0,\hat{x}) : \hat{V}_{\hat{\pi}_\theta}(0,\hat{x}) \geq 0\} \tag{18}$$

Intuitively, δ^* identifies the tightest level of the value function that separates safe states under $\hat{\pi}_\theta$ from unsafe ones. Hence, any initial state within the sub-δ^* level set is guaranteed to be safe under the induced policy, $\hat{\pi}^*_\theta$. However, calculating δ^* exactly requires infinitely many state-space points. To overcome this, we adopt a conformal-prediction-based approach to approximate δ^* using a finite number of samples, providing a probabilistic safety guarantee. The following theorem formalizes our approach:

Theorem III.1 (Safety Verification Using Conformal Prediction)**.** *Let $\mathcal{S}_\delta$ be the set of states satisfying $\hat{V}_\theta(0,\hat{x}) \leq \delta$, and let $(0,\hat{x}_i)_{i=1,\ldots,N_s}$ be N_s i.i.d. samples from $\mathcal{S}_\delta$. Define α_δ as the safety error rate among these N_s samples for a given δ*

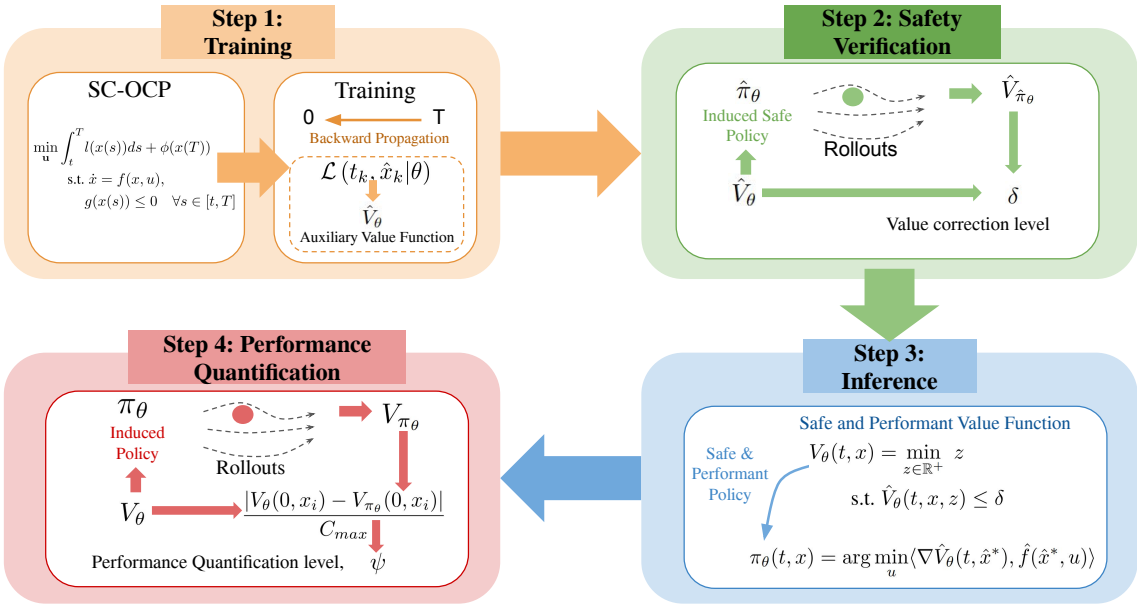

Fig. 1: Overview of the proposed approach: The methodology is organized into four steps. The **first step** involves training the auxiliary value function, $\hat{V}_\theta$, using a physics-informed machine learning framework. The **second step** applies a conformal prediction approach for safety verification of the learned $\hat{V}_\theta$. In the **third step**, the final value function V_θ and the optimal safe and performant policy π_θ are inferred. The **fourth step** quantifies the performance of V_θ through a second conformal prediction procedure.

level. Select a safety violation parameter $\epsilon_s \in (0,1)$ and a confidence parameter $\beta_s \in (0,1)$ such that:

$$\sum_{i=0}^{l-1} \binom{N_s}{i} \epsilon_s^i (1-\epsilon_s)^{N_s-i} \leq \beta_s, \tag{19}$$

where $l = \lfloor (N_s+1)\alpha_\delta \rfloor$. Then, with the probability of at least $1-\beta_s$, the following holds:

$$\mathbb{P}_{\hat{x}\in\mathcal{S}_\delta}\left(\hat{V}(0,\hat{x}_i) \leq 0\right) \geq 1-\epsilon_s. \tag{20}$$

The proof is available in Appendix VII-A. The safety error rate α_δ is defined as the fraction of samples satisfying $\hat{V}_\theta \leq \delta$ and $\hat{V}_{\hat{\pi}_\theta} \geq 0$ out of the total N_s samples.

Algorithm 1 presents the steps to calculate δ using the approach proposed in this theorem.

C. Obtaining Safe and Performant Value Function and Policy from $\hat{V}_\theta$

Using the δ-level estimate from Algorithm (1), we can finally obtain the safe and performant value function, $V_\theta(t,x)$, by solving the following epigraph optimization problem:

$$\begin{aligned} V_\theta(t,x) = \min_{z\in\mathbb{R}^+} \ & z \\ \text{s.t. } & \hat{V}_\theta(t,x,z) \leq \delta. \end{aligned} \tag{21}$$

Note that $V_\theta(t,x)$ is trivially ∞ for the states where $\hat{V}_\theta(t,x,z) > \delta$, since such states are unsafe and hence do not satisfy the safety constraint.

In practice, we solve this optimization problem by using a binary search approach on z. The resulting optimal state-feedback control policy, $\pi_\theta : \mathcal{X} \times [t,T) \to \mathcal{U}$, satisfying Objective (1), is given by:

$$\pi_\theta(t,x) = \arg\min_u \langle \nabla \hat{V}_\theta(t,\hat{x}^*), \hat{f}(\hat{x}^*,u)\rangle, \tag{22}$$

where $\hat{x}^*$ is the augmented state associated with the optimal z^* obtained by solving (21), i.e., $\hat{x}^* = [x, z^*]^T$. Intuitively, we can expect π_θ to learn behaviors that best tradeoff the safety and performance of the system.

Algorithm 2 Performance Quantification using Conformal Prediction

Require: $\mathcal{S}^*$, N_p, β_p, $V_\theta(x,0)$, $V_{\pi_\theta}(x,0)$

$D \leftarrow$ Sample N_p IID states from$\{x : x \in \mathcal{S}^*\}$
for $i = 0, 1, \ldots, N_p - 1$ **do**
  $P_i \leftarrow p_i(0, D)$
$P \leftarrow P$ sorted in decreasing order
$\alpha_p \leftarrow \frac{1}{N_p+1}$, $\psi_0 \leftarrow P_0$, $\epsilon_0 \leftarrow$ (23)
for $i = 0, 1, \ldots, N_p - 1$ **do**
  while $\epsilon_i \leq \epsilon_p$ **do**
    $\alpha_p \leftarrow \frac{i+1}{N_p+1}$, $\psi_i \leftarrow P_i$, $\epsilon_i \leftarrow$ (23)
return $\psi \leftarrow \psi_i$

D. Performance Quantification

In general, the learning inaccuracies in the auxiliary value function $\hat{V}_\theta$, may lead to errors in the value function V_θ. These errors, in turn, can lead to performance degradation under policy π_θ.

To quantify this degradation, we propose a conformal prediction-based performance quantification method that provides a probabilistic upper bound on the error between the

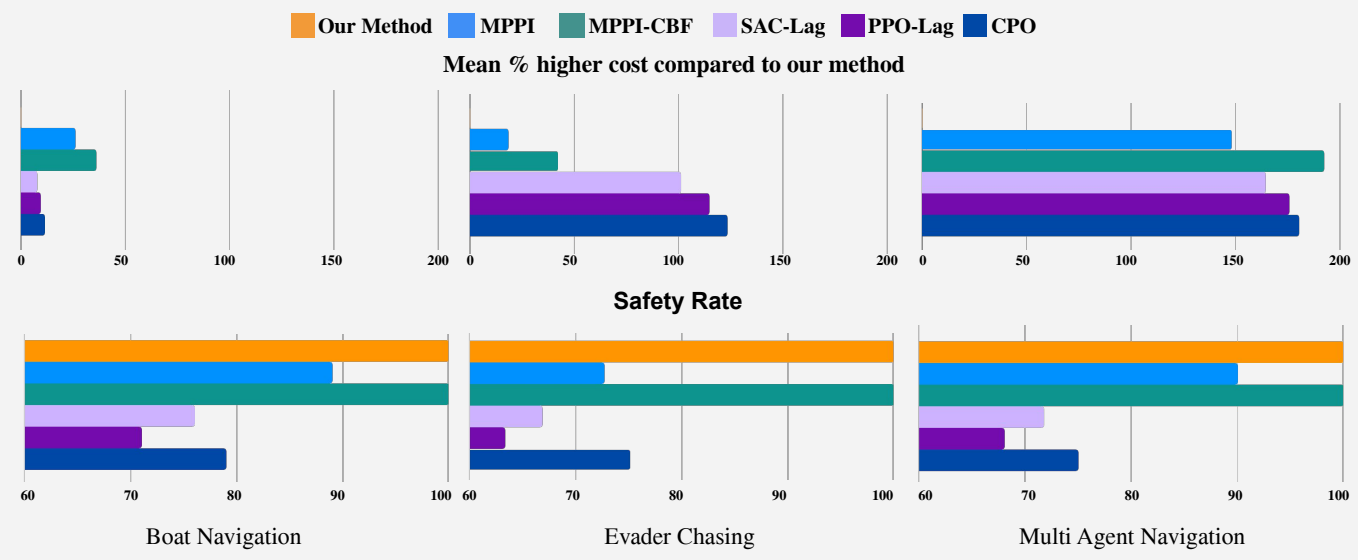

Fig. 2: This figure presents a comparative study between all the methods based on our evaluation metrics. The top plot illustrates the **mean percentage increase in cumulative cost** relative to our method for each baseline, demonstrating that our approach consistently incurs lower costs, with the gap widening as system complexity grows. The bottom plot depicts the **safety rates**, showing that our method maintains a 100% safety rate, while baselines that encourage safety rather than enforcing it (like MPPI and C-SAC) achieve lower rates. MPPI-CBF also attains 100% safety but at the expense of performance. Overall, our method uniquely **balances both safety and performance**, whereas the baselines compromise on at least one aspect.

value function and the value obtained from the induced policy. The following theorem formalizes our approach:

Theorem III.2 (Performance Quantification Using Conformal Prediction)**.** *Suppose $\mathcal{S}^*$ denotes the safe states satisfying $V_\theta(0,x) < \infty$ (or equivalently $\hat{V}_\theta(0,\hat{x}^*) < \delta$) and $(0,x_i)_{i=1,\dots,N_p}$ are N_p i.i.d. samples from $\mathcal{S}^*$. For a user-specified level α_p, let ψ be the $\frac{\lceil (N_p+1)(1-\alpha_p)\rceil}{N_p}$th quantile of the scores $(p_i := \frac{|V_\theta(0,x_i)-V_{\pi_\theta}(0,x_i)|}{C_{max}})_{i=1,\dots,N_p}$ on the N_p state samples. Select a violation parameter $\epsilon_p \in (0,1)$ and a confidence parameter $\beta_p \in (0,1)$ such that:*

$$\sum_{i=0}^{l-1} \binom{N_p}{i} \epsilon_p^i (1-\epsilon_p)^{N_p-i} \leq \beta_p \tag{23}$$

where, $l = \lfloor (N_p+1)\alpha_p \rfloor$. Then, the following holds, with probability $1-\beta_p$:

$$\mathbb{P}_{x\in\mathcal{S}^*}\left(\frac{|V_\theta(0,x_i) - V_{\pi_\theta}(0,x_i)|}{C_{max}} \leq \psi\right) \geq 1-\epsilon_p. \tag{24}$$

where C_{max} is a normalizing factor and denotes the maximum possible cost that could be incurred for any $x \in \mathcal{S}^*$.

The proof is available in Appendix VII-B. Note that C_{max} can be easily calculated by calculating the upper bound of the cost function $C(t,x(t),\mathbf{u})\forall x \in \mathcal{S}^*$.

Intuitively, the performance of the resultant policy is the best when the ψ value approaches 0, while the worst performance occurs at $\psi = 1$. Algorithm 2 presents the steps to calculate ψ using the approach proposed in this theorem.

E. Numerical Simulations

The objective of this paper is to demonstrate the co-optimization of performance and safety. To achieve this, we evaluate the proposed method and compare them with baselines using three metrics: (1) **Cumulative Cost:** This metric represents the total cost $\int_0^T l(x(s))ds + \phi(x(T))$, accumulated by a policy over the *safe* trajectories. (2) **Safety Rate:** This metric is defined as the percentage of trajectories that remain safe, i.e., never enter the failure region $\mathcal{F}$ at any point in time. (3) **Computation Time:** This metric compares the offline and online computation times of our method and the baselines.

Baselines: We consider two categories of baselines: the first set of methods aims to enhance the system performance (i.e., minimize the cumulative cost) while encouraging safety, encompassing methods such as Lagrangian-based CRL algorithms like SAC-Lagrangian (SAC-Lag), PPO-Lagrangian (PPO-Lag) [28, 29] and Model Predictive Path Integral (MPPI) [9] algorithms. The second category prioritizes safety, potentially at the cost of performance. This includes Constrained Policy Optimization (CPO) [2] and safety filtering techniques such as Control Barrier Function (CBF)-based quadratic programs (QP) [4] that modify a nominal, potentially unsafe controller to satisfy the safety constraint.

F. Efficient and Safe Boat Navigation

In our first experiment, we consider a 2D autonomous boat navigation problem, where a boat with coordinates (x_b, y_b) navigates a river with state-dependent drift to reach an island. The boat must avoid two circular boulders (obstacles) of different radii, which corresponds to the safety constraint in the system (see Fig. 3). The cost function penalizes the distance

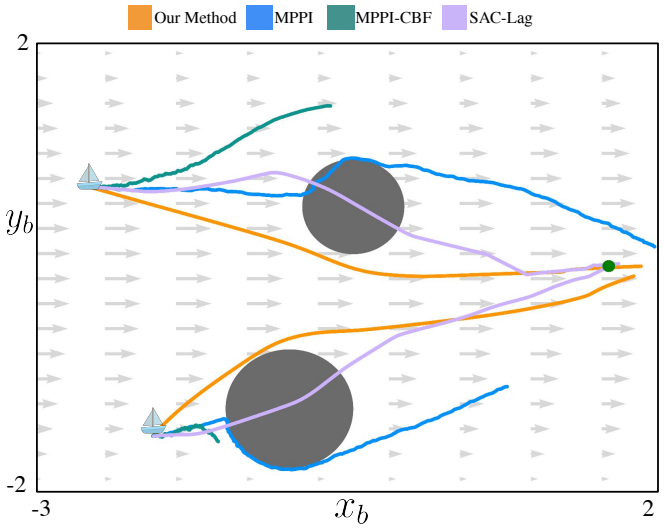

Fig. 3: Trajectories from two distinct initial states are shown, with dark grey circles representing obstacles and the green dot indicating the goal at $[1.5, 0]^T$. Notably, our method is the **only one** that **successfully approaches the goal** while **adhering to safety constraints**.

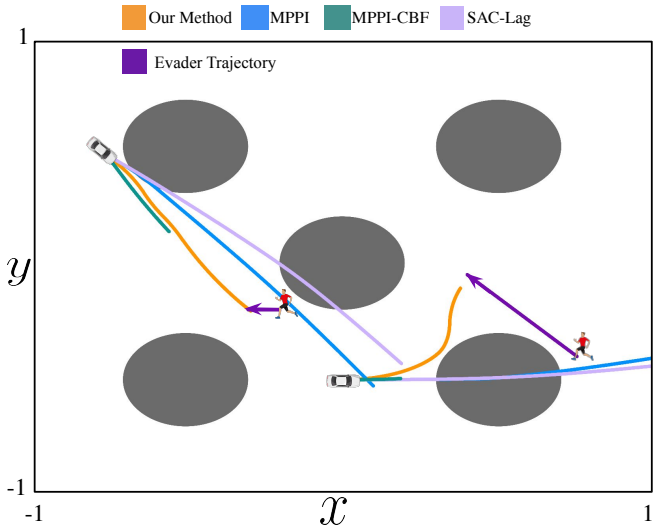

Fig. 4: Trajectories from two distinct initial states are depicted, with dark grey circles representing obstacles and purple trajectories indicating the evader's path, with arrows showing its direction of motion. Our method successfully **tracks the evader** while **avoiding collisions**, whereas all other methods either fail to maintain safety, struggle to track the evader or both

to the goal. The system state, x, evolves according to the dynamics:

$$x = [x_b, y_b], \quad \dot{x} = [u_1 + 2 - 0.5y_b^2, u_2] \tag{25}$$

where $[u_1, u_2]$ are the bounded control inputs in the x_b and y_b directions, constrained by the control space $\mathcal{U} = \{[u_1, u_2] \in \mathbb{R}^2 \mid ||[u_1, u_2]|| \leq 1\}$. The term $2 - 0.5y_b^2$ introduces a state-dependent drift, complicating the control task as the actions must counteract the drift while ensuring safety, which is challenging under bounded control inputs. The rest of the details about the experiment setup can be found in the Appendix VIII-A.

Safety Guarantees and Performance Quantification: We use $N_s = 300K$ and $N_p = 300K$ samples for thorough verification, ensuring dense state space sampling. For this experiment, we set $\epsilon_s = 0.001$ and $\beta_s = 10^{-10}$, resulting in a δ-level of 0. This implies that, with $1 - 10^{-10}$ confidence, any state with $\hat{V}_\theta(t, x, z) \leq 0$, is safe with at least 99.9% probability. For performance quantification, we set $\epsilon_p = 0.01$ and $\beta_p = 10^{-10}$, leading to a ψ-level of 0.136. This ensures, with $1-10^{-10}$ confidence, that any state in $\mathcal{S}^*$ has a normalized error between the predicted value and the policy value of less than 0.136 with 99% probability. Low δ and ψ values with high confidence indicate that the learned policy closely approximates the optimal policy and successfully co-optimizes safety and performance.

Baselines: This being a 2-dimensional system, we compare our method with the ground truth value function computed by solving the HJB-PDE numerically using the Level Set Toolbox [13] (results in Appendix VIII-A1). Additional baselines include: (1) MPPI, a sample-based path-planning algorithm with safety as soft constraints, (2) MPPI-NCBF, where safety is enforced using a Neural CBF-based QP with MPPI as the nominal controller [30, 31], and (3) Constrained RL methods like SAC-Lag, PPO-Lag, and CPO.

Comparative Analysis: Figure 3 shows that our method effectively reaches the goal while avoiding obstacles, even when starting close to them. In contrast, MPPI and CRL-based policies fail to maintain safety, while MPPI-NCBF ensures safety but performs poorly (leading to very slow trajectories). Figure 2 highlights that our method outperforms all others. SAC-Lag attains a mean cost that is 7.5% higher than ours, while exhibiting the lowest safety rate at 76%. The remaining CRL methods display comparable trends, highlighting their inability to jointly optimize for safety and performance. MPPI, with a more competitive safety rate of 89%, performs poorly with a 32.67% higher mean cost. MPPI-NCBF achieves 100% safety but performs significantly worse, with a 50.72% higher mean cost. Additionally, CBF-based controllers sometimes violate control bounds, limiting their applicability. This demonstrates that our method balances safety and performance, unlike others that compromise on one aspect. Moreover, the 100% safety rate of our method aligns closely with at least 99.9% safety level that we expect using our proposed verification strategy, providing empirical validation of the safety assurances.

G. Pursuer Vehicle tracking a moving Evader

In our second experiment, we consider an acceleration-driven pursuer vehicle, tracking a moving evader while avoiding five circular obstacles (see Fig. 4). This experiment involves an 8-dimensional system, with the state x defined as $x = [x_p, y_p, v, \Theta, x_e, y_e, v_{xe}, v_{ye}]^T$, where x_p, y_p, v, Θ represent the coordinates, linear velocity, and orientation of the pursuer vehicle, respectively, and x_e, y_e, v_{xe}, v_{ye} represent the coordinates and linear velocities of the evader vehicle. The pursuer vehicle

is controlled by linear acceleration (u_1) and angular velocity (u_2). The control space is $\mathcal{U} = \{[u_1, u_2] \in [-2, 2]^2\}$. The complexity of this system stems from the dynamic nature of the goal, along with the challenge of ensuring safety in a cluttered environment, which in itself is a difficult safety problem. More details about the experiment setup are in Appendix VIII-B.

Safety Guarantees and Performance Quantification: Similar to the previous experiment, we set $N_s = N_p = 300k$. We choose $\epsilon_s = 0.01$ and $\beta_s = 10^{-10}$, yielding a δ-level of -0.04 and a safety level of 99% on the auxiliary value function. For performance, we set $\epsilon_p = 0.01$ and $\beta_p = 10^{-10}$, leading to a ψ-level of 0.137. These values indicate the learned policy maintains high safety with low-performance degradation in this cluttered environment.

Baselines: As in the previous experiment, we employ MPPI and CRL methods (SAC-Lag, PPO-Lag, and CPO). For safety filtering, we utilize a QP based on the collision cone CBF (C3BF) [32], chosen for its effectiveness in managing acceleration-driven systems.

Comparative Analysis: Figure 4 shows that our method effectively tracks the moving evader while avoiding obstacles, even when starting close to them. In contrast, other methods have limitations: MPPI and CRL methods attempt to follow the evader but fail to maintain their pace, violating safety constraints, while MPPI-C3BF sacrifices performance to maintain safety. Figure 2 highlights our method's superior performance in balancing safety and performance. MPPI achieves the best performance among the baselines but with an 18% higher mean cost and only a 72% safety rate. MPPI-NCBF ensures 100% safety but has a 42% higher mean cost. SAC-Lag underperforms both in safety (66% safety rate) and performance (101% higher mean cost). A similar trend is evident across all other CRL methods, indicating their difficulty in co-optimizing safety and performance in high-dimensional, complex systems.

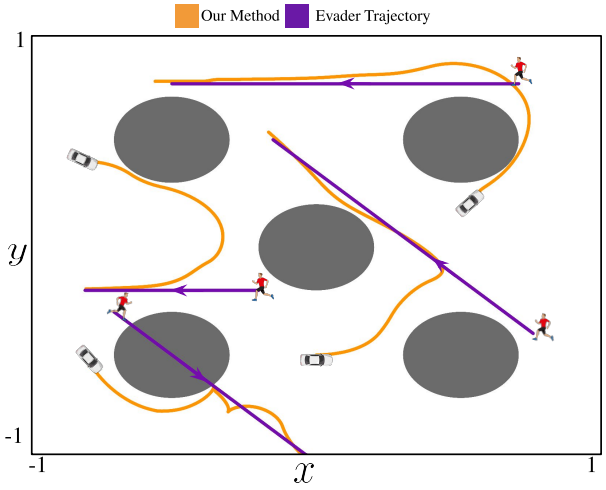

Fig. 5: Trajectories of the **receding horizon policy** for the pursuer tracking an evader over a 6-second horizon, while the value function is trained over a 1-second horizon. The results demonstrate that the pursuer **successfully tracks** the evader while ensuring safety, even when initialized near the obstacle. This highlights the effectiveness of the proposed approach in **jointly optimizing safety and performance for long-horizon tasks.**

Receding Horizon Control: An interesting application of the synthesized policy is its deployment in a receding horizon fashion over a time horizon longer than that used for training the value function, as illustrated in Fig. 5. The results indicate that the learned policy successfully maintains safety while effectively tracking the evader over a 6-second horizon, despite being trained over a horizon of 1 second. This suggests that the proposed approach can be extended to effectively co-optimize safety and performance for long-horizon tasks by solving the SC-OCP over a shorter time horizon. This approach therefore enables practical deployment in real-world autonomous systems that require long-horizon safety and performance guarantees, as validated by the hardware experiments in Section V.

H. Multi-Agent Navigation

In our third experiment, we consider a multi-agent setting where each of the 5 agents, represented by $x_i = [x_{a_i}, y_{a_i}, x_{g_i}, y_{g_i}]$, tries to reach its goal while avoiding collisions with others. (x_{a_i}, y_{a_i}) denote the position of the ith agent, while (x_{g_i}, y_{g_i}) represent the goal locations for that agent. The system is 20-dimensional, with each agent controlled by its x and y velocities. The control space for each agent is $\mathcal{U}_i = \{[v_{x_i}, v_{y_i}] \mid ||[v_{x_i}, v_{y_i}]|| \leq 1\}$. The complexity of this system stems from the interactions and potential conflicts between agents as they attempt to reach their goals while avoiding collisions. The rest of the details about the experiment setup can be found in Appendix VIII-C.

Safety Guarantees and Performance Quantification: We set $N_s = N_p = 300k$, $\epsilon_s = 0.001$, and $\beta_s = 10^{-10}$, resulting in a δ-level of -0.09 with safety assurance of 99.9% for the auxiliary value function. For performance quantification, we set $\epsilon_p = 0.01$ and $\beta_p = 10^{-10}$, leading to a ψ-level of 0.068. It is evident that the δ and ψ values remain very low with high confidence, highlighting the effectiveness of our method in co-optimizing safety and performance for high-dimensional, multi-agent systems.

Baselines: Similar to previous experiments, we have used MPPI, SAC-Lag, PPO-Lag, CPO, and MPPI-NCBF as our baselines for this experiment too.

Comparative Analysis: Figure 6 shows that our method ensures long-horizon safety while enabling all agents to reach their goals without collisions. In contrast, the baseline methods either exhibit overly conservative behavior or fail to maintain safety, leading to collisions, as detailed in Appendix VIII-C1. Figure 2 demonstrates the superior performance of our approach, with MPPI, MPPI-NCBF, and SAC-Lag showing mean percentage cost increases of 148%, 192%, and 164%, respectively. Although MPPI and MPPI-NCBF achieve competitive safety rates of 90% and 100%, their significant performance degradation highlights their inability to balance safety and performance in complex systems. MPPI's subpar performance stems from its reliance on locally optimal solutions in a finite data regime, leading to several deadlocks along the way and overall suboptimal trajectories over a long horizon. Furthermore, CRL methods struggle with both safety and performance, further demonstrating their limitations in

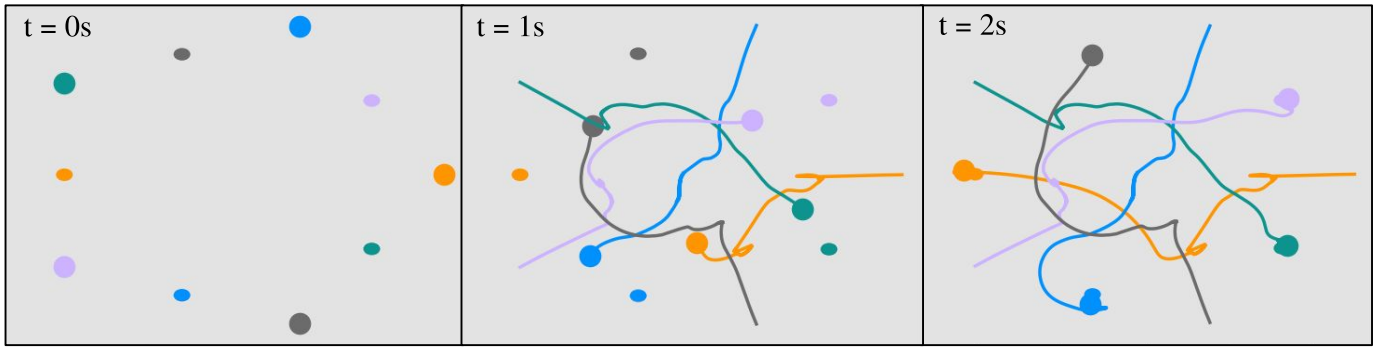

Fig. 6: Snapshots of multi-agent navigation trajectories at different times using the proposed method. Agents are represented as circles with radius R, indicating the minimum safe distance they must maintain from each other. Smaller dots mark their respective goals. The trajectories show that agents proactively **maintain long-horizon safety** by adjusting their paths to avoid close encounters, rather than enforcing safety reactively, which could lead to suboptimal behaviors. Finally, the agents **reach their respective goals within the specified time horizon**.

handling increasing system complexity and dimensionality. These results confirm our method's ability to co-optimize safety and performance in high-dimensional systems, demonstrating its scalability. Additionally, the safety guarantees hold in the test samples, validating the scalability of our safety verification framework for multi-agent systems.

I. Computation time Analysis

Figure 7 presents a comparative analysis of the offline and online computation times for our method against the baselines. While traditional grid-based methods suffer from an exponentially scaling computational complexity (and are completely intractable for the 8D Evader Chasing and 20D Multi-Agent case studies), the proposed method scales much better with the system dimensionality. For example, the computation time increases only minimally from the 2D system to the 8D system, thanks to neural network parallelization. Similarly, the computation time increases sublinearly from 8D to 20D system. This scalability is a key advantage of the proposed approach. We finally note that while offline training requires time, our method achieves real-time inference speeds, with optimal policy computed in just **2ms** across all systems, making the approach highly suitable for real robotic systems.

IV. Safe and Optimal Control With Disturbances

While the preceding framework offers a principled means to co-optimize safety and performance in deterministic settings, real-world autonomous systems seldom operate without uncertainty. Such uncertainties may arise from imperfect state information, external disturbances, or other unmodeled effects, and their presence substantially complicates the co-optimization process by necessitating explicit reasoning about uncertainty. To address this challenge, we extend our analysis to the RSC-OCP introduced in Problem 1, which incorporates disturbances directly into the problem structure. This transition naturally guides us toward the primary objective of this work, as articulated in Objective 1. A tractable solution to the RSC-OCP equips our methodology with robustness to uncertainty, thereby enhancing its suitability for real-world deployment.

To solve the RSC-OCP in Equation (2), our goal is to compute the optimal robust value function V_r, which minimizes the cost while ensuring system safety, despite the worst-case uncertainty. The theoretical background required for this formulation, together with its epigraph reformulation, is provided in Section II. Building on this foundation, we now present a systematic procedure for computing V_r.

In the subsections that follow, we proceed in a structured manner. We first construct the robust auxiliary value function $\hat{V}_r$ using a physics-informed machine learning framework. We then apply a disturbance-aware conformal prediction procedure to certify safety and mitigate approximation errors in the learned auxiliary value function $\hat{V}_r$. The resulting safety-refined approximation is subsequently mapped to the final value function V_r via the epigraph formulation in (3). Finally, we assess the performance of the resulting value function using a modified conformal prediction–based evaluation method that accounts for disturbances.

A. Training the Robust Auxiliary Value Function ($\hat{V}_r$)

The robust auxiliary value function, $\hat{V}_r$, is learned using the same curriculum training scheme as described in the previous section. The DNN architectures and training details are the same as in the previous section. The proposed learning framework utilizes a loss function that enforces two primary objectives: (i) compliance with the PDE in (10), using the PDE residual error given by:

$$\begin{aligned}\mathcal{L}_{pde}\left(t_k, \hat{x}_k|\theta\right) = \| \min \Big\{ &-\partial_t \hat{V}_\theta\left(t_k, \hat{x}_k\right) - H(t_k, \hat{x}_k), \\ &\hat{V}_\theta\left(t_k, \hat{x}_k\right) - g\left(x_k\right) \Big\} \|,\end{aligned} \tag{26}$$

where $H(t, \hat{x}) = \min_{u\in\mathcal{U}} \max_{d\in\mathcal{D}} \langle \nabla \hat{V}_\theta(t, \hat{x}), \hat{f}(\hat{x}, u, d)\rangle$ and (ii) satisfaction of the boundary condition in (11), using boundary condition loss, given by:

$$\begin{aligned}\mathcal{L}_{bc}\left(t_k, \hat{x}_k|\theta\right) = \|\max\left(\phi(x_k) - z_k, g(x_k)\right) - \\ \hat{V}_\theta\left(t_k, \hat{x}_k\right)\Big\| \mathbb{1}\left(t_k = T\right).\end{aligned} \tag{27}$$

The key difference compared to no-uncertainty case is that the Hamiltonian now involves solving a min-max problem,

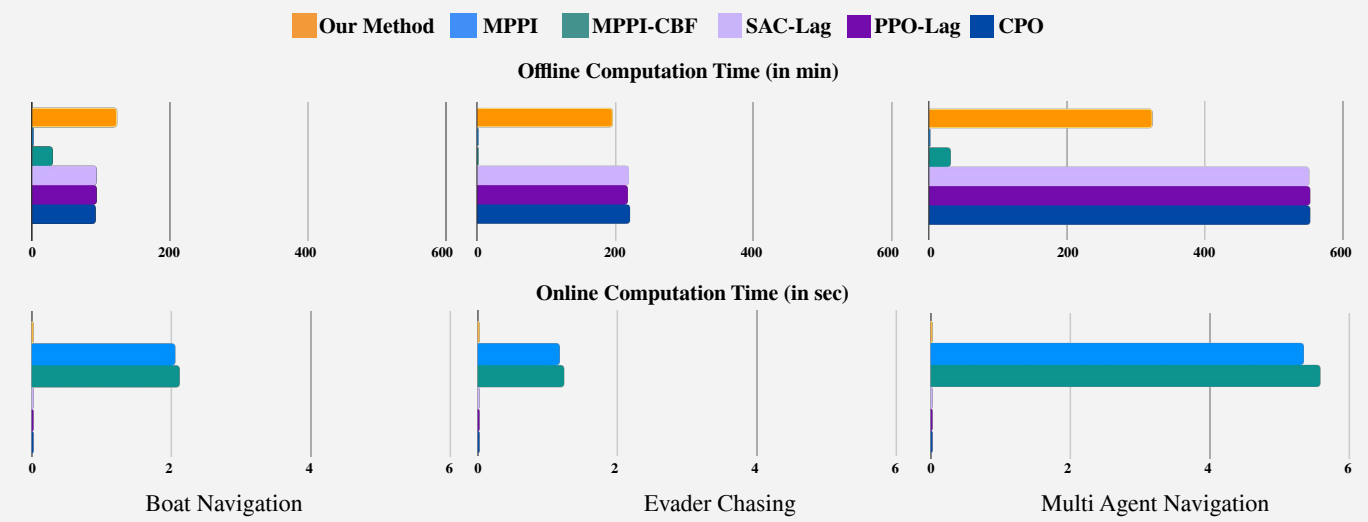

Fig. 7: This figure presents a comparative analysis of all methods based on online and offline computation time evaluated on the same computing machine. The top plot illustrates the **offline computation time** for our method and the baselines. Since our method and SAC-Lag involve training value functions, they incur higher offline computation costs, whereas MPPI-based methods require no offline training. The bottom plot depicts the **online computation time**, demonstrating that our method and SAC-Lag have minimal online computation requirements, whereas MPPI-based methods exhibit significantly higher online computational costs.

resulting in a robust value function. The two loss terms are balanced by a trade-off parameter λ, leading to the overall loss function:

$$\mathcal{L}\left(t_k, \hat{x}_k|\theta\right) = \mathcal{L}_{pde}\left(t_k, \hat{x}_k|\theta\right) + \lambda \mathcal{L}_{bc}\left(t_k, \hat{x}_k|\theta\right) \tag{28}$$

Furthermore, we use the adaptive loss re-balancing scheme proposed in [27] to reduce the impact of λ on the learned value function. Minimizing the overall loss function provides a self-supervised learning mechanism to approximate the auxiliary value function.

B. Safety Verification

The learned auxiliary value function, $\hat{V}_\theta$, induces a policy, $\hat{\pi}_\theta$, that minimizes the Hamiltonian term $H(t, \hat{x})$ in the HJB-PDE. The policy is given by:

$$\hat{\pi}_\theta(t, \hat{x}) = \arg\min_{u \in \mathcal{U}} \max_{d \in \mathcal{D}} \langle \nabla \hat{V}_\theta(t, \hat{x}), \hat{f}(\hat{x}, u, d) \rangle. \tag{29}$$

The disturbance policy induced by the learned value function is given by:

$$\hat{\pi}_{\mathbf{d}}(t, \hat{x}) = \arg\max_{d \in \mathcal{D}} \min_{u \in \mathcal{U}} \langle \nabla \hat{V}_\theta(t, \hat{x}), \hat{f}(\hat{x}, u, d) \rangle. \tag{30}$$

The rollout cost corresponding to this policy is defined as:

$$\hat{V}_{\hat{\pi}_\theta}(t, \hat{x}) = \max\left\{C(t, x(t), \mathbf{u}, \mathbf{d}) - z, \max_{s \in [t,T]} g(x(s))\right\}\Big|_{\substack{\mathbf{u}=\hat{\pi}_\theta, \\ \mathbf{d}=\hat{\pi}_{\mathbf{d}}}} \tag{31}$$

As in the no-disturbance case, discrepancies between the auxiliary value function and rollout cost can lead to states that are incorrectly classified as safe. To account for this, we introduce a uniform correction margin δ such that the sub-δ level set of $\hat{V}_\theta$ remains safe under the induced policy. In the disturbance setting, the rollout cost is evaluated under the learned control policy together with the induced disturbance policy. Since the exact margin is intractable to compute, we approximate it using conformal prediction, yielding probabilistic safety guarantees. The following theorem formalizes our approach:

Algorithm 3 Safety Verification using Conformal Prediction

Require: $\mathcal{S}$, N_s, β_s, ϵ_s, $\hat{V}_\theta(\hat{x}, 0)$, $\hat{V}_{\hat{\pi}_\theta}(\hat{x}, 0)$, M

$D_0 \leftarrow$ Sample N_s IID states from $\mathcal{S}_{\delta=0}$

$\delta_0 \leftarrow \min_{\hat{x}_j \in D_0} \left\{\hat{V}_\theta(0, \hat{x}_j) : \hat{V}_{\hat{\pi}_\theta}(0, \hat{x}_j) \geq 0\right\}$

$\epsilon_0 \leftarrow$ (32) (using $\alpha_{\delta=0}$)

$\Delta \leftarrow$ Ordered list of M uniform samples from $[\delta_0, 0]$

for $i = 0, 1, \ldots, M-1$ **do**

  while $\epsilon_i \leq \epsilon_s$ **do**

    $\delta_i \leftarrow \Delta_i$

    Update α_{δ_i} from δ_i

    $\epsilon_i \leftarrow$ (32) (using α_{δ_i})

return $\delta \leftarrow \delta_i$

Theorem IV.1 (Safety Verification Under Induced Disturbances)**.** *Consider a system with the induced rollout cost $\hat{V}_{\hat{\pi}_\theta}$ evaluated under both the learned control policy $\hat{\pi}_\theta$ and the induced disturbance policy $\hat{\pi}_{\mathbf{d}}$, as defined in (31). Let $\mathcal{S}_\delta$ be the set of states satisfying $\hat{V}_\theta(0, \hat{x}) \leq \delta$, and let $(0, \hat{x}_i)_{i=1,\ldots,N_s}$ be N_s i.i.d. samples from $\mathcal{S}_\delta$. Define α_δ as the safety error rate among these N_s samples for a given δ level. Select a safety violation parameter $\epsilon_s \in (0, 1)$ and a confidence parameter $\beta_s \in (0, 1)$ such that:*

$$\sum_{i=0}^{l-1} \binom{N_s}{i} \epsilon_s^i (1 - \epsilon_s)^{N_s - i} \leq \beta_s, \tag{32}$$

where $l = \lfloor (N_s + 1)\alpha_\delta \rfloor$. Then, with the probability of at least

Algorithm 4 Performance Quantification using Conformal Prediction

Require: $\mathcal{S}^*$, N_p, β_p, $V_\theta(x,0)$, $V_{\pi_\theta}(x,0)$
$D \leftarrow$ Sample N_p IID states from $\{x : x \in \mathcal{S}^*\}$
for $i = 0, 1, \ldots, N_p - 1$ **do**
  $P_i \leftarrow p_i(0, D)$
$P \leftarrow P$ sorted in decreasing order
$\alpha_p \leftarrow \frac{1}{N_p+1}$, $\psi_0 \leftarrow P_0$, $\epsilon_0 \leftarrow$ (36)
for $i = 0, 1, \ldots, N_p - 1$ **do**
  while $\epsilon_i \leq \epsilon_p$ **do**
    $\alpha_p \leftarrow \frac{i+1}{N_p+1}$, $\psi_i \leftarrow P_i$, $\epsilon_i \leftarrow$ (36)
return $\psi \leftarrow \psi_i$

$1 - \beta_s$, *the following holds:*

$$\mathbb{P}_{\hat{x} \in \mathcal{S}_\delta} \left(\hat{V}_{\hat{\pi}_\theta}(0, \hat{x}_i) \leq 0 \right) \geq 1 - \epsilon_s. \tag{33}$$

The proof is available in Appendix VII-C. The safety error rate α_δ is defined as the fraction of samples satisfying $\hat{V}_\theta \leq \delta$ and $\hat{V}_{\hat{\pi}_\theta} \geq 0$ out of the total N_s samples. Algorithm 3 presents the steps to calculate δ using the approach proposed in this theorem.

Nature of the robust guarantee: Unlike the deterministic setting where the conformal guarantee holds against the true system dynamics, the robust guarantee is with respect to the disturbance policy induced by the learned value function, $\hat{\pi}_{\mathbf{d}}$. While this yields a weaker theoretical guarantee, in practice, the learned disturbance closely approximates the true worst case due to the min-max optimization in training. This result generalizes Theorem III.1 to the robust setting; when $\mathcal{D} = \{0\}$, the disturbance vanishes and the guarantee reduces to Theorem III.1.

C. Obtaining Safe and Performant Value Function and Policy from $\hat{V}_\theta$

Following the no-disturbance case, we obtain the value function $V_\theta(t,x)$ via the epigraph optimization problem described in Equation (21). States that violate this constraint are unsafe and assigned infinite cost. As before, we solve this using a binary search over z. The resulting policy is:

$$\pi_\theta(t,x) = \arg\min_{u \in \mathcal{U}} \max_{d \in \mathcal{D}} \langle \nabla \hat{V}_\theta(t, \hat{x}^*), \hat{f}(\hat{x}^*, u, d) \rangle, \tag{34}$$

where $\hat{x}^* = [x, z^*]^T$, yielding a policy that balances safety and performance. Similarly, the resulting induced disturbance policy is given by:

$$\pi_{\mathbf{d}}(t,x) = \arg\max_{d \in \mathcal{D}} \min_{u \in \mathcal{U}} \langle \nabla \hat{V}_\theta(t, \hat{x}^*), \hat{f}(\hat{x}^*, u, d) \rangle, \tag{35}$$

D. Performance Quantification

As in the no disturbance case, we conduct a conformal prediction-based performance quantification step that provides a probabilistic upper bound on the error between the value function and the value obtained from the induced policy. The following theorem formalizes our approach:

Theorem IV.2 (Performance Quantification Under Induced Disturbances)**.** *Consider a system where the induced value function $V_{\pi_\theta}(0,x)$ is obtained by rolling out the learned safe and performant control policy π_θ (from (34)) against its induced disturbance policy $\pi_{\mathbf{d}}$ (from (35)). Suppose $\mathcal{S}^*$ denotes the safe states satisfying $V_\theta(0,x) < \infty$ (or equivalently $\hat{V}_\theta(0,\hat{x}^*) < \delta$) and $(0,x_i)_{i=1,\ldots,N_p}$ are N_p i.i.d. samples from $\mathcal{S}^*$. For a user-specified level α_p, let ψ be the $\frac{\lceil (N_p+1)(1-\alpha_p) \rceil}{N_p}$th quantile of the scores $(p_i := \frac{|V_\theta(0,x_i) - V_{\pi_\theta}(0,x_i)|}{C_{max}})_{i=1,\ldots,N_p}$ on the N_p state samples. Select a violation parameter $\epsilon_p \in (0,1)$ and a confidence parameter $\beta_p \in (0,1)$ such that:*

$$\sum_{i=0}^{l-1} \binom{N_p}{i} \epsilon_p^i (1-\epsilon_p)^{N_p - i} \leq \beta_p \tag{36}$$

where, $l = \lfloor (N_p+1)\alpha_p \rfloor$. Then, the following holds, with probability $1 - \beta_p$:

$$\mathbb{P}_{x \in \mathcal{S}^*} \left(\frac{|V_\theta(0,x_i) - V_{\pi_\theta}(0,x_i)|}{C_{max}} \leq \psi \right) \geq 1 - \epsilon_p. \tag{37}$$

where C_{max} is a normalizing factor and denotes the maximum possible cost that could be incurred for any $x \in \mathcal{S}^*$.

The proof is available in Appendix VII-D. Algorithm 4 presents the steps to calculate ψ using the approach proposed in this theorem.

Interpretation of the robust performance bound: The performance bound ψ quantifies the discrepancy between the learned value function and the rollout cost under the learned disturbance policy. Since $\hat{\pi}_{\mathbf{d}}$ is derived from the trained value function (rather than being the true worst-case adversary), the bound characterizes performance degradation against the *learned adversary*. This result generalizes Theorem III.2 to the robust setting; when $\mathcal{D} = \{0\}$, it reduces to Theorem III.2.

E. Numerical Simulations

We evaluate the proposed framework under disturbances and partial model uncertainty using representative numerical simulation studies.

F. Efficient and Safe Boat Navigation under Model Uncertainty

We revisit the boat navigation task and extend it to explicitly account for model uncertainty. The system dynamics are given by

$$x = \begin{bmatrix} x_b \\ y_b \end{bmatrix}, \quad \dot{x} = \begin{bmatrix} u_1 + 2 - 0.5y_b^2 + d_1 \\ u_2 + d_2 \end{bmatrix} \tag{38}$$

where $[u_1, u_2]$ denote the control inputs and $[d_1, d_2]$ represent exogenous disturbances acting along the x_b and y_b directions, respectively. The control inputs are constrained to the admissible set $\mathcal{U} = \{[u_1, u_2] \in \mathbb{R}^2 \mid ||[u_1, u_2]|| \leq 1\}$, while the disturbances are assumed to be bounded such that $\mathcal{D} = \{[d_1, d_2] \in \mathbb{R}^2 \mid |d_i| \leq 0.1, 0.2\}, i \in \{1, 2\}$.

To study the impact of disturbance modeling on safety and performance, we train three separate value functions: one assuming nominal dynamics without disturbances, and two

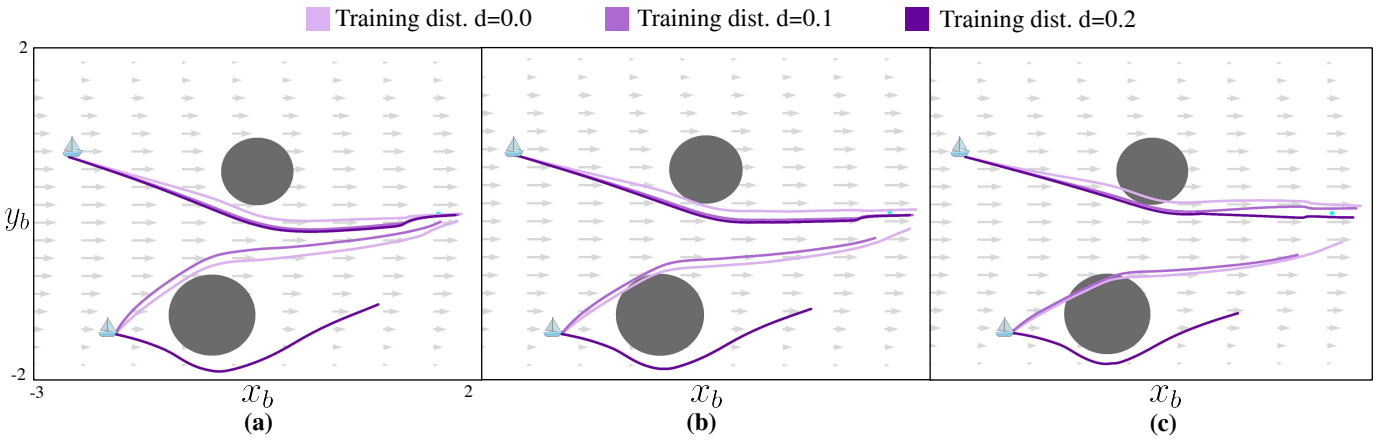

Fig. 8: Trajectories from two different initial conditions are shown for value functions trained under varying disturbance levels and evaluated in simulation with different disturbance magnitudes. (a) Evaluation disturbance $d = 0.0$. (b) Evaluation disturbance $d = 0.1$. (c) Evaluation disturbance $d = 0.2$. The results indicate that policies trained with disturbance levels greater than or equal to those encountered during evaluation exhibit improved safety, highlighting the importance of learning robust value functions for maintaining safety under uncertainty. Additionally, policies trained with disturbance levels exceeding those present in the environment exhibit increased conservatism, although the performance degradation remains modest. These results demonstrate that the proposed framework can synthesize safe policies for uncertain environments without substantial performance loss.

corresponding to different disturbance magnitudes. Each value function is then evaluated across all three disturbance settings, allowing us to assess both safety rates and task performance under matched and mismatched disturbance assumptions.

Tables I summarize the safety and performance of the learned value functions under varying levels of disturbance. When trained under nominal dynamics ($d = 0$), the value function achieves perfect safety in the disturbance-free setting but exhibits a progressive degradation in safety as the evaluation disturbance increases, indicating limited robustness to unmodeled uncertainty.

In contrast, value functions trained with nonzero disturbance bounds demonstrate improved generalization. Training with $d = 0.1$ preserves safety under matched and lower disturbance levels and partially mitigates safety degradation under higher disturbances. Notably, the value function trained with the largest disturbance bound ($d = 0.2$) maintains 100% safety across all evaluation settings, suggesting that incorporating worst-case disturbance assumptions during training leads to more robust safety preservation.

This increased robustness is accompanied by a modest increase in mean task cost, reflecting more conservative behavior. However, the cost increase remains gradual across disturbance levels, indicating that robustness is achieved without a substantial loss in efficiency. This trend is clearly illustrated in Figure 8. Overall, these results highlight a clear trade-off between conservatism and performance, and demonstrate that disturbance-aware value synthesis improves safety generalization under uncertainty.

Safety Guarantees and Performance Quantification: To evaluate the validity of the safety guarantees in the robust setting, we conduct an ablation study comparing safety rates with and without adjusting the value function using the δ-levels obtained from the CP-based safety verification procedure. Consistent with the previous disturbance-free boat navigation study, we set $N_s = N_p = 300k$, $\epsilon_s = 0.001$, and $\beta_s = 10^{-10}$. The verification procedure is performed under the same disturbance bounds used during training. As shown in Table II, adjusting the value function using the computed δ-level improves the observed safety rates relative to the case without safety verification. While the guarantees in the robust setting apply only to the policy-induced value of states, unlike the stronger guarantees on the true state values obtained in the non-robust setting, the results demonstrate that the verification step still provides meaningful improvements in safety rates.

We also quantify performance on the robust value functions trained under the three disturbance scenarios, obtaining ψ-levels of 0.137, 0.139, and 0.143, respectively. These results indicate that disturbances do not substantially increase training difficulty and that the proposed approach learns value functions that maintain strong performance under disturbances.

G. Pursuer Vehicle tracking an Uncertain moving Evader

Next, we consider a tracking problem in which a pursuer aims to track an uncertain moving evader. The pursuer dynamics are given by

$$\dot{x}_p = v\cos\Theta, \quad \dot{y}_p = v\sin\Theta, \quad \dot{v} = u_1, \quad \dot{\Theta} = u_2, \tag{39}$$

where (x_p, y_p) denote the pursuer position, v its speed, Θ its heading, and u_1, u_2 are bounded control inputs. The evader dynamics are modeled as

$$\begin{aligned} \dot{x}_e = v_{xe} + d_1, \quad \dot{y}_e = v_{ye} + d_2, \\ \dot{v}_{xe} = d_{v1}, \quad \dot{v}_{ye} = d_{v2}, \end{aligned} \tag{40}$$

where (x_e, y_e) denote the evader position, (v_{xe}, v_{ye}) its velocity components, and d_i and d_{vi} represent bounded disturbances

Training Disturbance	δ-Level	Evaluation Disturbance					
		$d = 0$		$d = 0.1$		$d = 0.2$	
		Safety Rate ↑	Mean Cost ↓	Safety Rate ↑	Mean Cost ↓	Safety Rate ↑	Mean Cost ↓
$d = 0.0$	0.000	100 %	2.88	89 %	2.99	80 %	3.15
$d = 0.1$	-0.018	100 %	2.97	100 %	3.10	88 %	3.19
$d = 0.2$	-0.028	100 %	2.96	100 %	3.14	100 %	3.28

TABLE I: Safety rate (%) and mean task cost for the boat navigation case study under varying disturbance levels. A notable trend is that the safety rate remains at 100% when the evaluation disturbance is equal to or smaller than the disturbance used during training, but decreases when the evaluation disturbance exceeds the training disturbance. Additionally, increasing the training disturbance does not lead to a substantial increase in the mean task cost, suggesting that the proposed robust training framework improves safety without significantly degrading task performance.

Training Dist.	w/o CP	w/ CP
$d = 0.0$	100%	100%
$d = 0.1$	98%	100%
$d = 0.2$	95%	100%

TABLE II: Safety rate (%) with and without conformal prediction (CP) correction. Each row reports the safety rate evaluated at its respective training disturbance level.

capturing uncertainty in the evader model. The disturbances are assumed to satisfy $|d_1|, |d_2| \leq 0.18, |d_{v1}|, |d_{v2}| \leq 0.08$, corresponding to approximately 20% of the respective state bounds.

We train two value functions: one assuming nominal evader dynamics without disturbances, and one incorporating the disturbance bounds above. For evaluation, we simulate environments under three disturbance settings: zero disturbance ($d = 0$), the nominal disturbance bounds (denoted dist), and twice the nominal disturbance magnitude ($2\,\mathrm{dist}$, corresponding to 40% of the state bounds). Each value function is evaluated over trajectories initialized from 1000 randomly sampled initial conditions.

Figure 9 illustrates that the robust policy adapts more effectively to uncertainty in the evader's behavior compared to the non-robust policy. This improved adaptability results in more consistent tracking performance, thereby demonstrating the effectiveness of the robust policy in uncertain environments. Table III summarizes the safety and performance of the learned value functions under increasing levels of evader model uncertainty. When trained without disturbances, the value function achieves perfect safety under nominal conditions but exhibits a progressive degradation in safety as the disturbance magnitude increases, indicating limited robustness to modeling errors in the evader dynamics. In contrast, incorporating disturbance bounds during training substantially improves safety generalization, maintaining near-perfect safety under matched disturbance conditions and significantly higher safety under amplified disturbances. This robustness is achieved with a modest increase in mean task cost, reflecting more conservative tracking behavior. Overall, these results demonstrate that explicitly modeling evader uncertainty during value

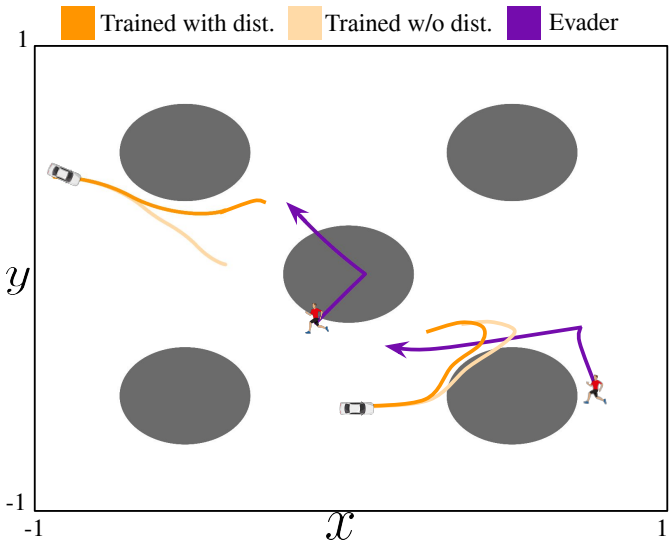

Fig. 9: Trajectories from two distinct initial conditions are shown for value functions trained with and without disturbance in the evader model. The policy derived from the robust value function effectively adapts to the evader's erratic behavior arising from model uncertainty and successfully maintains tracking. In contrast, the non-robust policy exhibits limited adaptation, resulting in less effective tracking and consequently higher tracking costs.

synthesis improves safety preservation under both expected and unforeseen disturbance levels.

Safety Guarantees and Performance Quantification: Similar to the robust boat navigation case study, we perform an ablation study to examine the effectiveness of the proposed safety verification scheme. As in the disturbance-free evader tracking experiment, we set $N_s = N_p = 300k$, $\epsilon_s = 0.01$, and $\beta_s = 10^{-10}$.The results in Table IV show a trend consistent with the boat navigation case study: applying safety verification leads to higher safety rates. This observation further demonstrates the effectiveness of the proposed CP-based verification scheme, even in the presence of disturbances.

In addition, for performance evaluation, we obtain ψ-levels of 0.136 and 0.141 for the disturbance and disturbance-free settings, respectively. These results indicate that the proposed robust training strategy, when coupled with the CP-based safety

Training Setting	δ-Level	Evaluation Disturbance					
		$d = 0$		$d = \text{dist}$		$d = 2 \times \text{dist}$	
		Safety Rate ↑	Mean Cost ↓	Safety Rate ↑	Mean Cost ↓	Safety Rate ↑	Mean Cost ↓
No disturbance	-0.04	100.0 %	1.65	88.6 %	1.70	83.8 %	1.78
With disturbance	-0.06	100.0 %	1.81	99.2 %	1.81	92.4 %	1.86

TABLE III: Safety rate (%) and mean task cost for the evader tracking case study with and without evader model uncertainty. A notable trend is that the safety rate remains high when the evaluation disturbance is equal to or smaller than the training disturbance, but decreases when the evaluation disturbance exceeds it. Additionally, increasing the training disturbance does not lead to a substantial increase in the mean task cost, suggesting that the proposed robust training framework, when trained with high disturbance values, improves safety without significantly degrading task performance.

Training Dist.	w/o CP	w/ CP
No disturbance	97.6%	100.0%
With disturbance	94.8%	99.2%

TABLE IV: Safety rate (%) with and without conformal prediction correction for the pursuer-evader tracking case study. Each row reports the safety rate evaluated at its respective training disturbance level.

verification scheme, enables the synthesis of value functions that achieve both strong safety rates and high performance.

V. Hardware Experiments

To validate the proposed robust framework's applicability for real-world deployment, we apply it to a real hardware testbed. Specifically, we deploy the pursuer-evader tracking task, where a pursuer must track a moving evader while avoiding circular obstacles, mirroring the simulation setup in Sections III and IV. The experimental setup is shown in Fig. 10.

We use two customized TurtleBot3 Burger platforms ($138 \times 178 \times 192$ mm, 1 kg each), fitted with mecanum wheels to enable omnidirectional motion (Fig. 10). Each robot is identified by a colored top: the blue robot serves as the pursuer, and the yellow robot serves as the evader. The pursuer operates under unicycle dynamics with state $[x_p, y_p, v, \Theta]^T$, controlled by linear acceleration (u_1) and angular velocity (u_2), consistent with the model described in Section III. The maximum linear speed is 0.22 m/s and the maximum angular velocity is 2.84 rad/s. The evader operates under point-mass dynamics with state $[x_e, y_e, v_{xe}, v_{ye}]^T$, controlled by velocity inputs (v_x, v_y) with a maximum speed of 0.22 m/s; the mecanum wheels allow it to realize point-mass dynamics directly. The global positions and orientations of both robots are measured using a PhaseSpace™ motion capture system at a tracking frequency of 960 Hz, with two LED markers placed on the front and rear of each robot for state estimation. Velocities are obtained via finite differencing of the position measurements, filtered using a low-pass filter to reduce noise.

The robust auxiliary value function $\hat{V}_\theta$ is trained offline using the evader disturbance bounds described in Section IV, together with additional disturbance bounds on the pursuer states, $|d_{x_p}|, |d_{y_p}|, |d_v| \leq 0.2$ and $|d_\Theta| \leq \pi/5$. These limits correspond to approximately 20% of the respective state bounds and are introduced to account for potential modeling inaccuracies in the pursuer dynamics. The resulting value function is subsequently used for online policy inference, with the policy executed in a receding-horizon manner, as described in Section III-G, enabling effective tracking of the evader over time horizons longer than those used during training. At each control step, the current state estimate is transmitted to an offboard PC, which evaluates the learned policy π_θ via (34). The resulting control commands are sent to the robots at a frequency of 100 Hz, with policy inference taking approximately 2 ms per step, well within the 10 ms control period. The experiments are conducted in a cluttered environment with obstacles of average size 0.25 m. The evader follows pre-planned trajectories that pass near obstacles, creating challenging tracking scenarios that stress both the safety and performance of the deployed policy. We evaluate the policy from 10 distinct initial conditions to assess repeatability.

Across 10 experimental trials, the pursuer consistently tracks the evader while satisfying all safety constraints, resulting in a 100% safety rate and a mean tracking error of 3.67 m. Figure 11 presents two representative trials. As illustrated, the deployed robust policy produces smooth, non-conservative trajectories that closely follow the evader, even in the presence of nearby obstacles, which is consistent with the behavior observed in simulation. We attribute the effective sim-to-real transfer to two factors: (i) the use of a disturbance-aware value function, which provides inherent robustness to modeling inaccuracies, sensor noise, and actuation delays not explicitly modeled during training; and (ii) the high control frequency (100 Hz), which mitigates discretization effects. These results demonstrate that the proposed framework can be deployed on physical robotic systems in real time, and that the robust formulation introduced in Section IV is instrumental in bridging the sim-to-real gap for safety-critical autonomy.

VI. Conclusion and Future Work

In this work, we introduced a physics-informed machine learning framework for co-optimizing safety and performance in autonomous systems. By formulating the problem as a state-constrained optimal control problem (SC-OCP) and leveraging an epigraph-based approach, we enabled scalable computation of safety-aware policies. Our method integrates conformal

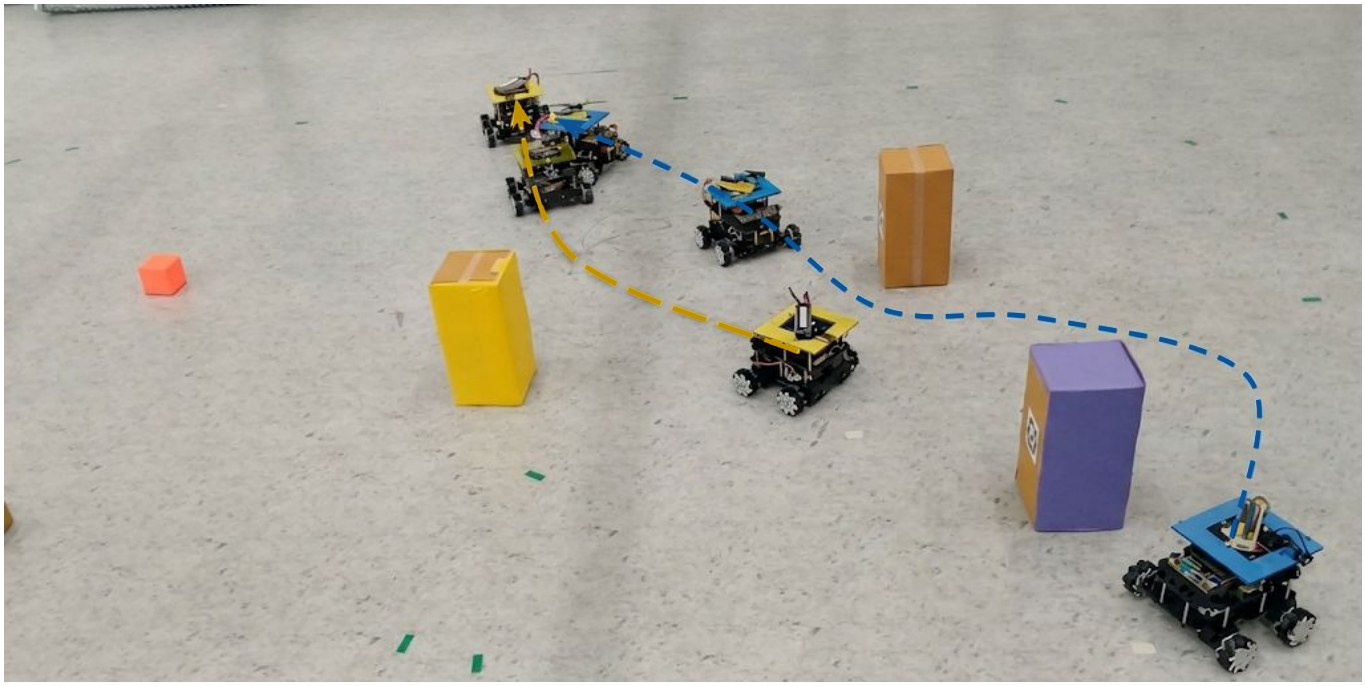

Fig. 10: Hardware experiment setup for the pursuer-evader tracking task. Two customized TurtleBot3 Burger platforms fitted with mecanum wheels are used: the blue-topped robot (pursuer, unicycle dynamics) tracks the yellow-topped robot (evader, point-mass dynamics) while avoiding box-shaped obstacles. The dashed lines indicate representative trajectories of the pursuer (blue) and evader (yellow). Global localization is provided by the PhaseSpace™ motion capture system, and policy inference is performed on an offboard PC that sends control commands to both robots in real time.

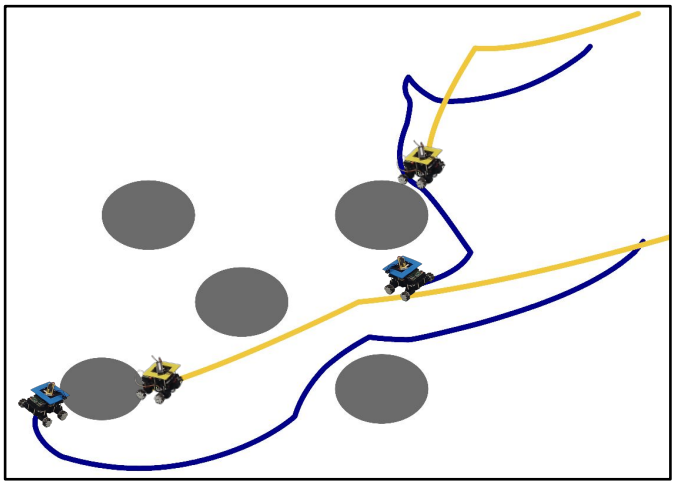

Fig. 11: This figure presents two representative trajectories obtained from the hardware experiments. The pursuer adapts to the evader's erratic motion and maintains successful tracking, demonstrating the robustness of the proposed approach to disturbances encountered during sim-to-real transfer and its potential for real-world deployment.

prediction-based safety verification to ensure high-confidence safety guarantees while maintaining optimal performance. Through multiple case studies, we demonstrated the effectiveness and scalability of our approach in high-dimensional systems. In future, we will explore methods for rapid adaptation of the learned policies in light of new information about the system dynamics, environments, or safety constraints. We will also apply our method to other high-dimensional autonomous systems and systems with unknown dynamics.

ACKNOWLEDGEMENTS

Manan is supported by the Prime Minister's Research Fellowship (PMRF), Government of India. This work is partially supported by the AI & Robotics Technology Park (ARTPARK) at IISc, the DARPA Assured Neuro Symbolic Learning and Reasoning (ANSR) program, and the NSF CAREER award (2240163).

REFERENCES

[1] E. Altman, *Constrained Markov Decision Processes*, ser. Stochastic Modeling Series. Taylor & Francis, 1999. [Online]. Available: https://books.google.co.in/books?id=3X9S1NM2iOgC

[2] J. Achiam, D. Held, A. Tamar, and P. Abbeel, "Constrained policy optimization," in *International conference on machine learning*. PMLR, 2017, pp. 22–31.

[3] K.-C. Hsu, H. Hu, and J. F. Fisac, "The safety filter: A unified view of safety-critical control in autonomous systems," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 7, no. Volume 7, 2024, pp. 47–72, 2024. [Online]. Available: https://www.annualreviews.org/content/journals/10.1146/annurev-control-071723-102940

[4] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs for

safety critical systems," *IEEE Transactions on Automatic Control*, vol. 62, no. 8, pp. 3861–3876, 2017.

[5] J. Borquez, K. Chakraborty, H. Wang, and S. Bansal, "On safety and liveness filtering using hamilton–jacobi reachability analysis," *IEEE Transactions on Robotics*, vol. 40, pp. 4235–4251, 2024.

[6] K. P. Wabersich, A. J. Taylor, J. J. Choi, K. Sreenath, C. J. Tomlin, A. D. Ames, and M. N. Zeilinger, "Data-driven safety filters: Hamilton-jacobi reachability, control barrier functions, and predictive methods for uncertain systems," *IEEE Control Systems Magazine*, vol. 43, no. 5, pp. 137–177, 2023.

[7] C. E. García, D. M. Prett, and M. Morari, "Model predictive control: Theory and practice—a survey," *Automatica*, vol. 25, no. 3, pp. 335–348, 1989. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0005109889900022

[8] L. Grüne, J. Pannek, L. Grüne, and J. Pannek, *Nonlinear model predictive control*. Springer, 2017.

[9] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou, "Information-theoretic model predictive control: Theory and applications to autonomous driving," *IEEE Transactions on Robotics*, vol. 34, no. 6, pp. 1603–1622, 2018.

[10] L. Streichenberg, E. Trevisan, J. J. Chung, R. Siegwart, and J. Alonso-Mora, "Multi-agent path integral control for interaction-aware motion planning in urban canals," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 1379–1385.

[11] H. M. Soner, "Optimal control with state-space constraint i," *SIAM Journal on Control and Optimization*, vol. 24, no. 3, pp. 552–561, 1986. [Online]. Available: https://doi.org/10.1137/0324032

[12] A. Altarovici, O. Bokanowski, and H. Zidani, "A general hamilton-jacobi framework for non-linear state-constrained control problems," *ESAIM: Control, Optimisation and Calculus of Variations*, vol. 19, no. 2, pp. 337–357, 2013.

[13] I. Mitchell, "A toolbox of level set methods," *http://www.cs. ubc. ca/mitchell/ToolboxLS/toolboxLS.pdf*, 2004.

[14] H. Wang, A. Dhande, and S. Bansal, "Cooptimizing safety and performance with a control-constrained formulation," *IEEE Control Systems Letters*, vol. 8, pp. 2739–2744, 2024.

[15] Y. T. Chow, J. Darbon, S. Osher, and W. Yin, "Algorithm for overcoming the curse of dimensionality for time-dependent non-convex hamilton–jacobi equations arising from optimal control and differential games problems," *Journal of Scientific Computing*, vol. 73, pp. 617–643, 2017.

[16] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics informed deep learning (part i & ii): Data-driven solutions of nonlinear partial differential equations," 2017.

[17] ——, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *Journal of Computational physics*, vol. 378, pp. 686–707, 2019.

[18] S. Bansal and C. J. Tomlin, "Deepreach: A deep learning approach to high-dimensional reachability," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 1817–1824.

[19] M. Raissi, P. Perdikaris, and G. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *Journal of Computational Physics*, vol. 378, pp. 686–707, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0021999118307125

[20] Z. Li, H. Zheng, N. B. Kovachki, D. Jin, H. Chen, B. Liu, A. Stuart, K. Azizzadenesheli, and A. Anandkumar, "Physics-informed neural operator for learning partial differential equations," 2022. [Online]. Available: https://openreview.net/forum?id=dtYnHcmQKeM

[21] M. Tayal, A. Singh, S. Kolathaya, and S. Bansal, "A physics-informed machine learning framework for safe and optimal control of autonomous systems," in *Proceedings of the 42nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, A. Singh, M. Fazel, D. Hsu, S. Lacoste-Julien, F. Berkenkamp, T. Maharaj, K. Wagstaff, and J. Zhu, Eds., vol. 267. PMLR, 13–19 Jul 2025, pp. 59 237–59 258. [Online]. Available: https://proceedings.mlr.press/v267/tayal25a.html

[22] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[23] A. Singh, Z. Feng, and S. Bansal, "Exact imposition of safety boundary conditions in neural reachable tubes," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*, 2025, pp. 5489–5495.

[24] A. Lin and S. Bansal, "Verification of neural reachable tubes via scenario optimization and conformal prediction," in *Proceedings of the 6th Annual Learning for Dynamics & Control Conference*, ser. Proceedings of Machine Learning Research, A. Abate, M. Cannon, K. Margellos, and A. Papachristodoulou, Eds., vol. 242. PMLR, 15–17 Jul 2024, pp. 719–731. [Online]. Available: https://proceedings.mlr.press/v242/lin24a.html

[25] M. Tayal, A. Singh, P. Jagtap, and S. Kolathaya, "Cp-ncbf: A conformal prediction-based approach to synthesize verified neural control barrier functions," 2025. [Online]. Available: https://arxiv.org/abs/2503.17395

[26] E. Schmerling, "hj_reachability: Hamilton-Jacobi reachability analysis in JAX," *https://github.com/StanfordASL/hj_reachability*, 2021.

[27] S. Wang, Y. Teng, and P. Perdikaris, "Understanding and mitigating gradient flow pathologies in physics-informed neural networks," *SIAM Journal on Scientific Computing*, vol. 43, no. 5, pp. A3055–A3081, 2021.

[28] A. Ray, J. Achiam, and D. Amodei, "Benchmarking safe exploration in deep reinforcement learning," 2019. [Online]. Available: https://cdn.openai.com/safexp-short.pdf

[29] J. Ji, J. Zhou, B. Zhang, J. Dai, X. Pan, R. Sun, W. Huang, Y. Geng, M. Liu, and Y. Yang, "Omnisafe: An infrastructure for accelerating safe reinforcement learning research," *Journal of Machine Learning Research*, vol. 25, no. 285, pp. 1–6, 2024. [Online]. Available: http://jmlr.org/papers/v25/23-0681.html

[30] C. Dawson, Z. Qin, S. Gao, and C. Fan, "Safe nonlinear control using robust neural lyapunov-barrier functions," in *Conference on Robot Learning*. PMLR, 2022, pp. 1724–1735.

[31] M. Tayal, H. Zhang, P. Jagtap, A. Clark, and S. Kolathaya, "Learning a formally verified control barrier function in stochastic environment," in *2024 IEEE 63rd Conference on Decision and Control (CDC)*, 2024, pp. 4098–4104.

[32] B. G. Goswami, M. Tayal, K. Rajgopal, P. Jagtap, and S. Kolathaya, "Collision cone control barrier functions: Experimental validation on ugvs for kinematic obstacle avoidance," in *2024 American Control Conference (ACC)*, 2024, pp. 325–331.

[33] A. N. Angelopoulos and S. Bates, "A gentle introduction to conformal prediction and distribution-free uncertainty quantification," 2022. [Online]. Available: https://arxiv.org/abs/2107.07511

[34] V. Vovk, "Conditional validity of inductive conformal predictors," 2012. [Online]. Available: https://arxiv.org/abs/1209.2673

[35] F. W. J. Olver, A. B. O. Daalhuis, D. W. Lozier, B. I. Schneider, R. F. Boisvert, C. W. Clark, B. R. Miller, B. V. Saunders, H. S. Cohl, and e. M. A. McClain, *NIST Digital Library of Mathematical Functions*. National Institute of Standards and Technology, 2023, release 1.1.11 of 2023-09-15. [Online]. Available: https://dlmf.nist.gov/

[36] M. Tayal, M. Tayal, A. Singh, S. Kolathaya, and R. Prakash, "V-OCBF: Learning safety filters from offline data via value-guided offline control barrier functions," *Transactions on Machine Learning Research*, 2026. [Online]. Available: https://openreview.net/forum?id=PGO9mpIyyb

[37] J. Chen, R. Du, and K. Wu, "A comparison study of deep galerkin method and deep ritz method for elliptic problems with different boundary conditions," *arXiv preprint arXiv:2005.04554*, 2020.

APPENDIX
VII. PROOFS

A. *Proof of Theorem III.1*

Theorem III.1 (Safety Verification Using Conformal Prediction) Let $\mathcal{S}_\delta$ be the set of states satisfying $\hat{V}_\theta(0,\hat{x}) \leq \delta$, and let $(0,\hat{x}_i)_{i=1,\dots,N_s}$ be N_s i.i.d. samples from $\mathcal{S}_\delta$. Define α_δ as the safety error rate among these N_s samples for a given δ level. Select a safety violation parameter $\epsilon_s \in (0,1)$ and a confidence parameter $\beta_s \in (0,1)$ such that:

$$\sum_{i=0}^{l-1} \binom{N_s}{i} \epsilon_s^i (1-\epsilon_s)^{N_s-i} \leq \beta_s,$$

where $l = \lfloor (N_s+1)\alpha_\delta \rfloor$. Then, with the probability of at least $1-\beta_s$, the following holds:

$$\mathop{\mathbb{P}}_{\hat{x}\in\mathcal{S}_\delta} \left(\hat{V}(0,\hat{x}_i) \leq 0 \right) \geq 1-\epsilon_s.$$

Proof. Before we proceed with the proof of the Theorem (III.1), let us look at the following lemma which describes split conformal prediction:

Lemma 1 (Split Conformal Prediction [33]). Consider a set of independent and identically distributed (i.i.d.) calibration data, denoted as $\{(X_i,Y_i)\}_{i=1}^n$, along with a new test point $(X_{\text{test}},Y_{\text{test}})$ sampled independently from the same distribution. Define a score function $s(x,y) \in \mathbb{R}$, where higher scores indicate poorer alignment between x and y. Compute the calibration scores $s_1 = s(X_1,Y_1),\dots,s_n = s(X_n,Y_n)$. For a user-defined confidence level $1-\alpha$, let $\hat{q}$ represent the $\lceil (n+1)(1-\alpha) \rceil / n$ quantile of these scores. Construct the prediction set for the test input X_{test} as:

$$\mathcal{C}(X_{\text{test}}) = \{y : s(X_{\text{test}},y) \leq \hat{q}\}.$$

Assuming exchangeability, the prediction set $\mathcal{C}(X_{\text{test}})$ guarantees the marginal coverage property:

$$\mathbb{P}(Y_{\text{test}} \in \mathcal{C}(X_{\text{test}})) \geq 1-\alpha.$$

Following the Lemma 1, we employ a conformal scoring function for safety verification, defined as:

$$s(X) = \max_{i\in\{1,\cdots,n_R\}} \hat{V}_{\hat{\pi}_\theta}(0,\hat{x}), \forall \hat{x} \in \mathcal{S}_{\tilde{\delta}},$$

where $\mathcal{S}_\delta$ denotes the set of states satisfying $\hat{V}_\theta(0,\hat{x}) \leq \delta$ and the score function robustly measures the alignment between the induced safe policy and the auxiliary value function.

Next, we sample N_s states from the safe set $\mathcal{S}_\delta$ and compute conformal scores for all sampled states. For a user-defined error rate $\alpha \in [0,1]$, let $\hat{q}$ denote the $\frac{(N_s+1)\alpha}{N_s}$th quantile of the conformal scores. According to [34], the following property holds:

$$\mathop{\mathbb{P}}_{\hat{x}\in\mathcal{S}_{\tilde{\delta}}} \left(\hat{V}_{\hat{\pi}_\theta}(\hat{x}_i,0) \leq \hat{q} \right) \sim \text{Beta}(N_s-l+1,l), \tag{41}$$

where $l = \lfloor (N_s+1)\alpha \rfloor$.

Define E_s as:

$$E_s := \mathop{\mathbb{P}}_{\hat{x}\in\mathcal{S}_\delta} \left(\hat{V}_{\hat{\pi}_\theta}(\hat{x}_i,0) \leq \hat{q} \right).$$

Here, E_s is a Beta-distributed random variable. Using properties of cumulative distribution functions (CDF), we assert that $E_s \geq 1-\epsilon_s$ with confidence $1-\beta_s$ if the following condition is satisfied:

$$I_{1-\epsilon_s}(N-l+1,l) \leq \beta_s, \tag{42}$$

where $I_x(a,b)$ is the regularized incomplete Beta function and also serves as the CDF of the Beta distribution. It is defined as:

$$I_x(a,b) = \frac{1}{B(a,b)} \int_0^x t^{a-1}(1-t)^{b-1}\,dt,$$

where $B(a,b)$ is the Beta function. From [35](8.17.5), it can be shown that $I_x(n-k,k+1) = \sum_{i=1}^k \binom{n}{i} x^i (1-x)^{n-i}$.

Then (42) can be rewritten as:

$$\sum_{i=1}^{l-1} \binom{N_s}{i} \epsilon_s^i (1-\epsilon_s)^{N_s-i} \leq \beta_s, \tag{43}$$

Thus, if Equation (43) holds, we can say with probability $1-\beta_s$ that:

$$\mathop{\mathbb{P}}_{\hat{x}\in\mathcal{S}_\delta}\left(\hat{V}_{\hat{\pi}_\theta}(\hat{x}_i,0)\leq\hat{q}\right)\geq 1-\epsilon_s. \tag{44}$$

Now, let k denote the number of allowable safety violations. Thus, the safety error rate is given by $\alpha_\delta=\frac{k+1}{N_s+1}$. Let $\hat{q}$ represent the $\frac{(N_s+1)\alpha_\delta}{N_s}$th quantile of the conformal scores. Since k denotes the number of samples for which the conformal score is positive, the $\frac{(N_s+1)\alpha_\delta}{N_s}$th quantile of scores corresponds to the maximum *negative score* amongst the sampled states. This implies that $\hat{q}\leq 0$. From this and Equation (44), we can conclude with probability $1-\beta_s$ that:

$$\mathop{\mathbb{P}}_{\hat{x}\in\mathcal{S}_\delta}\left(\hat{V}_{\hat{\pi}_\theta}(0,\hat{x}_i)\leq 0\right)\geq 1-\epsilon_s.$$

From Equation (4), it can be inferred that $\forall\ (t,\hat{x})$, $\hat{V}(0,\hat{x}_i)\leq\hat{V}_{\hat{\pi}_\theta}(\hat{x}_i,0)$. Hence, with probability $1-\beta_s$, the following holds:

$$\mathop{\mathbb{P}}_{\hat{x}\in\mathcal{S}_\delta}\left(\hat{V}(0,\hat{x}_i)\leq 0\right)\geq 1-\epsilon_s.$$

□

B. *Proof of Theorem III.2*

Theorem III.2 (Performance Quantification Using Conformal Prediction) Suppose $\mathcal{S}^*$ denotes the safe states satisfying $V_\theta(0,x)<\infty$ (or equivalently $\hat{V}_\theta(0,\hat{x}^*)<\delta$) and $(0,x_i)_{i=1,\dots,N_p}$ are N_p i.i.d. samples from $\mathcal{S}^*$. For a user-specified level α_p, let ψ be the $\frac{\lceil(N_p+1)(1-\alpha_p)\rceil}{N_p}th$ quantile of the scores $(p_i:=\frac{|V_\theta(0,x_i)-V_{\pi_\theta}(0,x_i)|}{C_{max}})_{i=1,\dots,N_p}$ on the N_p state samples. Select a violation parameter $\epsilon_p\in(0,1)$ and a confidence parameter $\beta_p\in(0,1)$ such that:

$$\sum_{i=0}^{l-1}\binom{N_p}{i}\epsilon_p^i(1-\epsilon_p)^{N_p-i}\leq\beta_p$$

where, $l=\lfloor(N_p+1)\alpha_p\rfloor$. Then, the following holds, with probability $1-\beta_p$:

$$\mathop{\mathbb{P}}_{x\in\mathcal{S}^*}\left(\frac{|V_\theta(0,x_i)-V_{\pi_\theta}(0,x_i)|}{C_{max}}\leq\psi\right)\geq 1-\epsilon_p.$$

where C_{max} is a normalizing factor and denotes the maximum possible cost that could be incurred for any $x\in\mathcal{S}^*$.

Proof. To quantify the performance loss, we employ a conformal scoring function defined as:

$$p(x):=\frac{|V_\theta(0,x_i)-V_{\pi_\theta}(0,x_i)|}{C_{max}},\forall x\in\mathcal{S}^*$$

where the score function measures the alignment between the induced optimal policy and the value function.

Next, we sample N_p states from the state space $\mathcal{S}^*$ and compute conformal scores for all sampled states. For a user-defined error rate $\alpha_p\in[0,1]$, let ψ denote the $\frac{(N_p+1)\alpha_p}{N_p}$ quantile of the conformal scores. According to [34], the following property holds:

$$\mathop{\mathbb{P}}_{x\in\mathcal{S}^*}\left(\frac{|V_\theta(0,x_i)-V_{\pi_\theta}(0,x_i)|}{C_{max}}\leq\psi\right)\sim\text{Beta}(N_p-l+1,l),$$

where $l=\lfloor(N_p+1)\alpha_p\rfloor$.

Define E_p as:

$$E_p:=\mathop{\mathbb{P}}_{x\in\mathcal{S}^*}\left(\frac{|V_\theta(0,x_i)-V_{\pi_\theta}(0,x_i)|}{C_{max}}\leq\psi\right).$$

Here, E_p is a Beta-distributed random variable. Using properties of CDF, we assert that $E_p\geq 1-\epsilon_p$ with confidence $1-\beta_p$ if the following condition is satisfied:

$$I_{1-\epsilon_p}(N_p-l+1,l)\leq\beta_p, \tag{45}$$

where $I_x(a,b)$ is the regularized incomplete Beta function. From [35](8.17.5), it can be shown that $I_x(n-k,k+1)=\sum_{i=1}^{k}\binom{n}{i}x^i(1-x)^{n-i}$. Hence, Equation (45) can be equivalently stated as:

$$\sum_{i=1}^{l-1}\binom{N_p}{i}\epsilon_p^i(1-\epsilon_p)^{N_p-i}\leq\beta_p \tag{46}$$

Thus, if Equation (46) holds, we can conclude with probability $1-\beta_p$ that:

$$\mathop{\mathbb{P}}_{x\in\mathcal{S}^*}\left(\frac{|V_\theta(0,x_i)-V_{\pi_\theta}(0,x_i)|}{C_{max}}\leq\psi\right)\geq 1-\epsilon_p.$$

□

C. Proof of Theorem IV.1

Theorem IV.1 (Safety Verification Under Induced Disturbances) Consider a system with the induced rollout cost $\hat{V}_{\hat{\pi}_\theta}$ evaluated under both the learned control policy $\hat{\pi}_\theta$ and the induced disturbance policy $\hat{\pi}_{\mathbf{d}}$, as defined in (31). Let $\mathcal{S}_\delta$ be the set of states satisfying $\hat{V}_\theta(0,\hat{x})\leq\delta$, and let $(0,\hat{x}_i)_{i=1,\dots,N_s}$ be N_s i.i.d. samples from $\mathcal{S}_\delta$. Define α_δ as the safety error rate among these N_s samples for a given δ level. Select a safety violation parameter $\epsilon_s\in(0,1)$ and a confidence parameter $\beta_s\in(0,1)$ such that:

$$\sum_{i=0}^{l-1}\binom{N_s}{i}\epsilon_s^i(1-\epsilon_s)^{N_s-i}\leq\beta_s, \tag{47}$$

where $l=\lfloor(N_s+1)\alpha_\delta\rfloor$. Then, with the probability of at least $1-\beta_s$, the following holds:

$$\mathop{\mathbb{P}}_{\hat{x}\in\mathcal{S}_\delta}\left(\hat{V}_{\hat{\pi}_\theta}(0,\hat{x}_i)\leq 0\right)\geq 1-\epsilon_s. \tag{48}$$

Proof. The proof follows the same conformal prediction framework as the proof of Theorem III.1, but now operates under dynamics subject to bounded disturbances $d\in\mathcal{D}$.

In the robust setting, the learned auxiliary value function $\hat{V}_\theta$ induces both a control policy $\hat{\pi}_\theta$ and a disturbance policy $\hat{\pi}_{\mathbf{d}}$ via Equations (29) and (30), respectively. The rollout cost is computed under both these policies simultaneously, as defined in Equation (31):

$$\hat{V}_{\hat{\pi}_\theta}(t,\hat{x})=\max\Big\{C(t,x(t),\mathbf{u},\mathbf{d})-z,\ \max_{s\in[t,T]}g(x(s))\Big\}\Big|_{\substack{\mathbf{u}=\hat{\pi}_\theta,\\ \mathbf{d}=\hat{\pi}_{\mathbf{d}}}}.$$

We employ the disturbance policy $\hat{\pi}_{\mathbf{d}}$ derived from the learned value function as the adversarial disturbance during rollouts. We note that $\hat{\pi}_{\mathbf{d}}$ is the worst-case disturbance with respect to the *learned* value function, and may not coincide with the true worst-case disturbance. Exhaustively searching over all possible disturbance sequences (*e.g.*, $|\mathcal{D}|^T$ sequences for T time steps) is computationally infeasible. Therefore, the conformal guarantee provided here is with respect to the *learned policy's value function* $\hat{V}_{\hat{\pi}_\theta}$ rather than the true value function.

Following Lemma 1, we define the conformal scoring function for safety verification as:

$$s(\hat{x})=\hat{V}_{\hat{\pi}_\theta}(0,\hat{x}),\quad\forall\hat{x}\in\mathcal{S}_\delta.$$

We sample N_s i.i.d. states from $\mathcal{S}_\delta$ and compute conformal scores for all sampled states. For a user-defined error rate $\alpha\in[0,1]$, let $\hat{q}$ denote the $\frac{(N_s+1)\alpha}{N_s}$th quantile of the conformal scores. By [34], the following holds:

$$\mathop{\mathbb{P}}_{\hat{x}\in\mathcal{S}_\delta}\left(\hat{V}_{\hat{\pi}_\theta}(0,\hat{x}_i)\leq\hat{q}\right)\sim\text{Beta}(N_s-l+1,l),$$

where $l=\lfloor(N_s+1)\alpha\rfloor$.

Defining $E_s:=\mathop{\mathbb{P}}_{\hat{x}\in\mathcal{S}_\delta}\left(\hat{V}_{\hat{\pi}_\theta}(0,\hat{x}_i)\leq\hat{q}\right)$ and proceeding identically to the proof of Theorem III.1 (via the regularized incomplete Beta function and its binomial representation), we obtain:

$$\sum_{i=0}^{l-1}\binom{N_s}{i}\epsilon_s^i(1-\epsilon_s)^{N_s-i}\leq\beta_s\implies\mathop{\mathbb{P}}_{\hat{x}\in\mathcal{S}_\delta}\left(\hat{V}_{\hat{\pi}_\theta}(0,\hat{x}_i)\leq\hat{q}\right)\geq 1-\epsilon_s,$$

with probability at least $1-\beta_s$.

Setting $\alpha_\delta=\frac{k+1}{N_s+1}$, where k is the number of states for which the conformal score is positive, ensures $\hat{q}\leq 0$ (as in the non-robust case). Consequently:

$$\mathop{\mathbb{P}}_{\hat{x}\in\mathcal{S}_\delta}\left(\hat{V}_{\hat{\pi}_\theta}(0,\hat{x}_i)\leq 0\right)\geq 1-\epsilon_s.$$

□

D. Proof of Theorem IV.2

Theorem IV.2 (Performance Quantification Under Induced Disturbances) Consider a system where the induced value function $V_{\pi_\theta}(0,x)$ is obtained by rolling out the learned safe and performant control policy π_θ (from (34)) against its induced disturbance policy $\pi_{\mathbf{d}}$ (from (35)). Suppose $\mathcal{S}^*$ denotes the safe states satisfying $V_\theta(0,x) < \infty$ (or equivalently $\hat{V}_\theta(0,\hat{x}^*) < \delta$) and $(0,x_i)_{i=1,\dots,N_p}$ are N_p i.i.d. samples from $\mathcal{S}^*$. For a user-specified level α_p, let ψ be the $\frac{\lceil (N_p+1)(1-\alpha_p)\rceil}{N_p}$th quantile of the scores $(p_i := \frac{|V_\theta(0,x_i)-V_{\pi_\theta}(0,x_i)|}{C_{max}})_{i=1,\dots,N_p}$ on the N_p state samples. Select a violation parameter $\epsilon_p \in (0,1)$ and a confidence parameter $\beta_p \in (0,1)$ such that:

$$\sum_{i=0}^{l-1} \binom{N_p}{i} \epsilon_p^i (1-\epsilon_p)^{N_p-i} \leq \beta_p \tag{49}$$

where, $l = \lfloor (N_p+1)\alpha_p \rfloor$. Then, the following holds, with probability $1-\beta_p$:

$$\mathop{\mathbb{P}}_{x\in\mathcal{S}^*}\left(\frac{|V_\theta(0,x_i)-V_{\pi_\theta}(0,x_i)|}{C_{max}} \leq \psi\right) \geq 1-\epsilon_p. \tag{50}$$

where C_{max} is a normalizing factor and denotes the maximum possible cost that could be incurred for any $x \in \mathcal{S}^*$.

Proof. The proof mirrors that of Theorem III.2, adapted to account for the presence of disturbances.

In the robust setting, the induced value function $V_{\pi_\theta}(0,x)$ is obtained by rolling out the system under the learned policy π_θ (derived from the robust auxiliary value function via Equation (34)) against the learned worst-case disturbance policy $\hat{\pi}_{\mathbf{d}}$ from Equation (30).

We define the conformal scoring function for performance quantification as:

$$p(x) := \frac{|V_\theta(0,x_i)-V_{\pi_\theta}(0,x_i)|}{C_{max}}, \quad \forall x \in \mathcal{S}^*,$$

where the score function measures the discrepancy between the learned value function and the value realized under the induced policy against the learned disturbance.

We sample N_p i.i.d. states from $\mathcal{S}^*$ and compute conformal scores. For a user-defined error rate $\alpha_p \in [0,1]$, let ψ denote the $\frac{(N_p+1)\alpha_p}{N_p}$th quantile. By [34]:

$$\mathop{\mathbb{P}}_{x\in\mathcal{S}^*}\left(\frac{|V_\theta(0,x_i)-V_{\pi_\theta}(0,x_i)|}{C_{max}} \leq \psi\right) \sim \text{Beta}(N_p-l+1, l),$$

where $l = \lfloor (N_p+1)\alpha_p \rfloor$.

Defining $E_p := \mathop{\mathbb{P}}_{x\in\mathcal{S}^*}\left(\frac{|V_\theta(0,x_i)-V_{\pi_\theta}(0,x_i)|}{C_{max}} \leq \psi\right)$ and proceeding identically to the proof of Theorem III.2 via the regularized incomplete Beta function:

$$I_{1-\epsilon_p}(N_p-l+1, l) \leq \beta_p,$$

which, using the binomial representation from [35](8.17.5), gives:

$$\sum_{i=0}^{l-1} \binom{N_p}{i} \epsilon_p^i (1-\epsilon_p)^{N_p-i} \leq \beta_p.$$

Thus, with probability $1-\beta_p$:

$$\mathop{\mathbb{P}}_{x\in\mathcal{S}^*}\left(\frac{|V_\theta(0,x_i)-V_{\pi_\theta}(0,x_i)|}{C_{max}} \leq \psi\right) \geq 1-\epsilon_p.$$

□

E. Relationship between α, β, and ϵ

The work [34] states that a smaller number of samples leads to greater fluctuations in the conformal prediction calibration, meaning that if we redraw N samples and repeat the conformal prediction process, we might get a different calibration result. This variance decreases as N increases.Similarly, in our work, a small N means that the value correction term δ might fluctuate each time the verification algorithm is executed. Therefore, to ensure a stable estimate of δ, it is desirable to select a sufficiently large value of N.

Figure 12 presents the $\alpha-\epsilon$ plots for varying numbers of verification samples N and different values of β. From the figure, we observe that as N increases, the effect of β diminishes, and the curve approaches the $\alpha=\epsilon$ line. Ideally, the user-specified

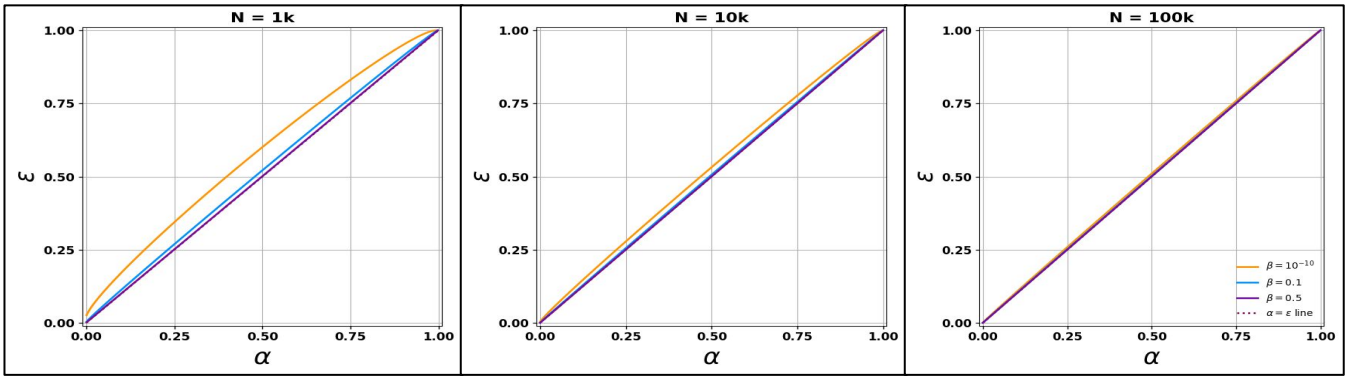

Fig. 12: This figure shows the α-ϵ plots for different numbers of verification samples, N, and different values of β.

safety error rate (α) should closely match the safety violation parameter (ϵ) while maintaining high confidence ($1-\beta$ close to 1). Thus, selecting a larger N enables a smaller β while ensuring the alignment of α and ϵ. Conversely, if N is small, one must either compromise on the confidence parameter β or accept that α will be lower than ϵ, resulting in a more conservative upper bound on the safety rate.

VIII. Additional Details the Systems in the Experiments

In this section, we will provide more details about the systems we have used in the experiments section III.

A. *Efficient and Safe Boat Navigation*

The states, x of the 2D Boat system are $x = [x_1, x_2]^T$, where, x_1, x_2 are the x and y coordinates of the boat respectively. We define the step cost at each step, $l(t, x)$, as the distance from the goal, given by:

$$l(t, x) := \|x - (1.5, 0)^T\|$$

The cost function $C(t, x(t))$ is defined as:

$$C(t, x(t), \mathbf{u}) = \int_t^T l(t, x(t))\, dt \;\; + \phi(x(T)) \tag{51}$$

where T is the time horizon ($2s$ in our experiment), $l(t, x(t)) = ||x(t) - (1.5, 0)^T||$ represents the running cost, and $\phi(x(T)) = ||x(T) - (1.5, 0)^T||$ is the terminal cost. Minimizing this cost drives the boat toward the island.

Consequently, the (augmented) dynamics of the 2D Boat system are:

$$\begin{aligned} \dot{x_1} &= u_1 + 2 - 0.5x_2^2 \\ \dot{x_2} &= u_2 \\ \dot{z} &= -l(t, x) \end{aligned}$$

where u_1, u_2 represents the velocity control in x_1 and x_2 directions respectively, with $u_1^2 + u_2^2 \leq 1$ and $2 - 0.5x_2^2$ specifies the current drift along the x_1-axis.

The safety constraints are formulated as:

$$\begin{aligned} g(x) := \max \big(&0.4 - \|x - (-0.5, 0.5)^T\|, \\ &0.5 - \|x - (-1.0, -1.2)^T\|\big) \end{aligned} \tag{52}$$

where $g(x) > 0$ indicates that the boat is inside a boulder, thereby ensuring that the super-level set of $g(x)$ defines the failure region.

1) Ground Truth Comparison: We compute the Ground Truth value function using the Level-Set Toolbox [13] and use it as a benchmark in our comparative analysis. To facilitate demonstration, unsafe states are assigned a high value of 20 instead of ∞. The value function in this problem ranges from 0 to 14.76.

As illustrated in Figure 13, the value function obtained using our method closely approximates the ground truth value function. Notably, the unsafe region (highlighted in yellow) remains identical in both cases, confirming the safety of the learned value function. Furthermore, the mean squared error (MSE) between the two value functions is 0.36, which is relatively low given the broad range of possible values.

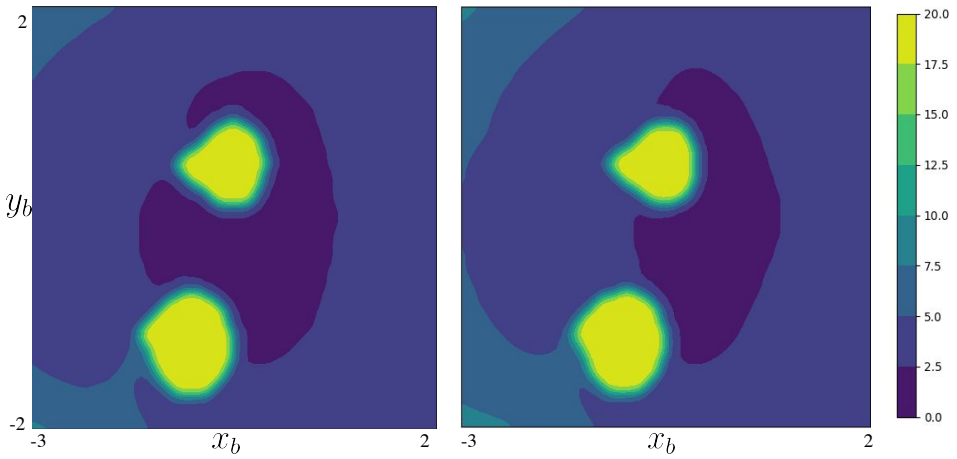

Fig. 13: Heatmap of the value function for the ground truth (left) and our method (right). The yellow region represents the unsafe area. Our method successfully captures most of the safe set, indicating that it is not overly conservative while completely recovering the unsafe regions.

It is also worth mentioning that computing a high-fidelity ground truth value function on a $210 \times 210 \times 210$ grid using the Level Set Toolbox requires approximately 390 minutes. In contrast, our proposed approach learns the value function in 122 minutes, achieving a substantial speedup. This demonstrates that even for systems with a relatively low-dimensional state space, our method efficiently recovers an accurate value function significantly faster than grid-based solvers.

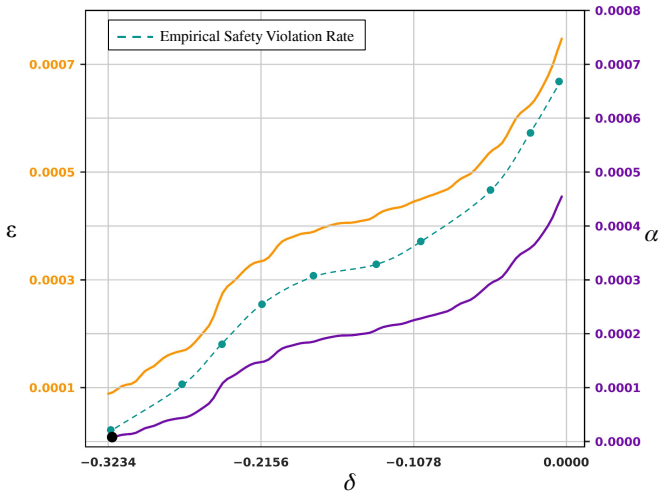

Fig. 14: This figure presents a comparative analysis of the relationships between ϵ–δ, α–δ, and empirical safety violation rate–δ. As observed, the empirical safety violation consistently remains below the theoretical bound, thereby supporting our theoretical guarantees. Furthermore, as ϵ decreases, the corresponding δ approaches zero, indicating that the learned value function incurs negligible safety violations.

2) Empirical Validation of the α-β-ϵ relationship and calculation of safety levels: We conducted an experiment to empirically validate the theoretical relationship between α, β, and ϵ. The results are presented in Figure 14. The figure visualizes the relationship between theoretical and empirical safety metrics across varying levels of δ, and includes the following elements:

- *Safety error rate* (α, purple line) as a function of different δ levels. Computed on the calibration dataset as $\alpha = \frac{k+1}{N_s+1}$, where k is the number of allowable safety violations and N_s is the number of calibration samples.
- *Theoretical safety violation probability* (ϵ, orange line) as a function of δ. Derived using the theoretical relation in Equation (14).
- *Empirical safety violation probability* (green points) as a function of δ. Computed by sampling 3M initial states from the δ-sublevel set of the learned value function, simulating rollouts, and measuring the observed safety violation rate. This serves as a practical estimate of system safety.

For this experiment, we set $N = 300k$ and $\beta = 10^{-10}$. As shown in the figure, the empirical violation rate remains consistently below the theoretical bound (ϵ) across all values of δ. This demonstrates that our method provides conservative and valid safety guarantees, confirming the soundness of the theoretical relationship in practice.

Additionally, from the δ vs ϵ plot, we can observe that the δ level approaches 0 as the ϵ values approach the chosen safety level of 0.001. Hence, we say that the sub-level set of the auxiliary value function, $\hat{V}(t, \hat{x})$ is safe with a probability of $1 - 0.001 = 0.999$.

B. Pursuer vehicle tracking an evader

The state, x of a ground vehicle (pursuer) tracking a moving evader is $x = [x_p, y_p, v, \Theta, x_e, y_e, v_{xe}, v_{ye}]^T$, where, x_e, y_e, v, Θ are position, linear velocity and orientation of the pursuer respectively, x_e, y_e, v_{xe}, v_{ye} are the position and the linear velocities of the evader respectively. We define the step cost at each step, $l(t, x)$, as the distance from the goal, given by:

$$l(t, x) := \|(x_p(t), y_p(t))^T - (x_e(t), y_e(t))^T\|$$

and the terminal cost is $\phi(x(T)) = ||(x_p(T), y_p(T))^T - (x_e(T), y_e(T))^T||$. The cost function $C(t, x(t))$ is defined as:

$$C(t, x(t), \mathbf{u}) = \int_t^T l(t, x(t))\, dt \ + \phi(x(T)) \tag{53}$$

where T is the time horizon ($1s$ in this experiment). Minimizing this cost aims to drive the pursuer toward the evader. Consequently, the (augmented) dynamics of the system is as follows:

$$\begin{aligned} \dot{x_p} &= v\cos(\Theta), \quad \dot{y_p} = v\sin(\Theta), \quad \dot{v} = u_1, \quad \dot{\Theta} = u_2, \\ \dot{x_e} &= v_{xe}, \quad \dot{y_e} = v_{ye}, \quad \dot{v_{xe}} = 0, \quad \dot{v_{ye}} = 0, \quad \dot{z} = -l(t, x), \end{aligned}$$

where u_1 represents the linear acceleration control and u_2 represents angular velocity control.

The safety constraints are defined as:

$$\begin{aligned} g(x) := \max \big(0.2 - \|x - (0.5, 0.5)^T\|, 0.2 - \|x - (-0.5, 0.5)^T\|, 0.2 - \|x - (-0.5, -0.5)^T\|, \\ 0.2 - \|x - (0.5, -0.5)^T\|, 0.2 - \|x - (0.0, 0.0)^T\|\big) \end{aligned}$$

which represents 5 obstacles of radius 0.2 units each.

C. Multi-Agent Navigation

A multi-agent setting with 5 agents. The state of each agent i is represented by $x_i = [x_{a_i}, y_{a_i}, x_{g_i}, y_{g_i}]$, tries to reach its goal while avoiding collisions with others. (x_{a_i}, y_{a_i}) denote the position of the ith agent, while (x_{g_i}, y_{g_i}) represent the goal locations for that agent. We define the step cost at each step, $l(t, x(t))$, as the mean distance of each agent from its respective goal, given by:

$$l(t, x(t)) := \frac{\sum_{i=1}^{5} \|(x_{a_i}(t), y_{a_i}(t)^T - (x_{g_i}(t), y_{g_i}(t))^T\|}{5}$$

The cost function $C(t, x(t), \mathbf{u})$ is defined as:

$$C(t, x(t), \mathbf{u}) := \int_t^T l(t, x(t))\, dt \ + \phi(x(T)) \tag{54}$$

where T is the time horizon ($2s$ in this experiment). Minimizing this cost aims to drive each agent towards its goal. Consequently, the (augmented) dynamics of the system is as follows:

$$\begin{aligned} \dot{x}_{a_i} = u_{1i}, \forall i \in \{1,2,3,4,5\}; \quad \dot{y}_{a_i} = u_{2i}, \forall i \in \{1,2,3,4,5\}; \quad \dot{x}_{g_i} = 0, \forall i \in \{1,2,3,4,5\} \\ \dot{y}_{g_i} = 0, \forall i \in \{1,2,3,4,5\}; \quad \dot{z} = -l(t, x) \end{aligned}$$

where u_{1i}, u_{2i} represents the linear velocity control of each agent i. The safety constraints are defined as:

$$g(x(t)) := \max_{\substack{i,j \in \{1,\ldots,5\}, \\ i \neq j}} \big(R - \|(x_{a_i}, y_{a_i})^T - (x_{a_j}, y_{a_j})^T\|\big) \tag{55}$$

1) Comparison of Multi-Agent Navigation with baselines: Figures 15, 16, and 17 illustrate the trajectories obtained by the baseline methods for the Multi-Agent Navigation problem. It can be observed that the trajectories obtained by MPPI and MPPI-SF are highly conservative, implying that these methods prioritize safety to mitigate potential conflicts among agents. In contrast, the policy derived from SAC-Lag fails to maintain safety, resulting in agent collisions. This indicates that as system complexity increases, the baseline methods tend to prioritize either safety or performance, leading to suboptimal behavior and safety violations. Conversely, the proposed approach effectively co-optimizes safety and performance, even in complex high-dimensional settings, achieving superior performance while ensuring safety. The visualization of the trajectories can be found on the project website: https://tayalmanan28.github.io/robust-piml-soc/.

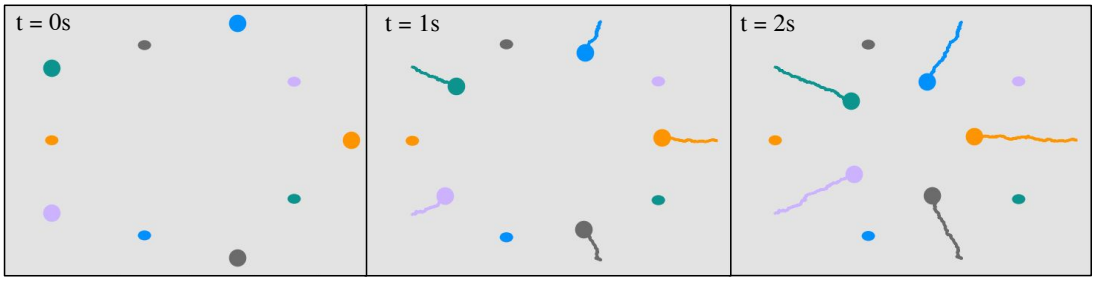

Fig. 15: Snapshots of multi-agent navigation trajectories at different time instances using **MPPI**. The trajectories indicate that the agents adopt a **highly conservative strategy** to prevent collisions. Consequently, this leads to a **reduction in performance**, as the agents **end up very far from their respective goals**.

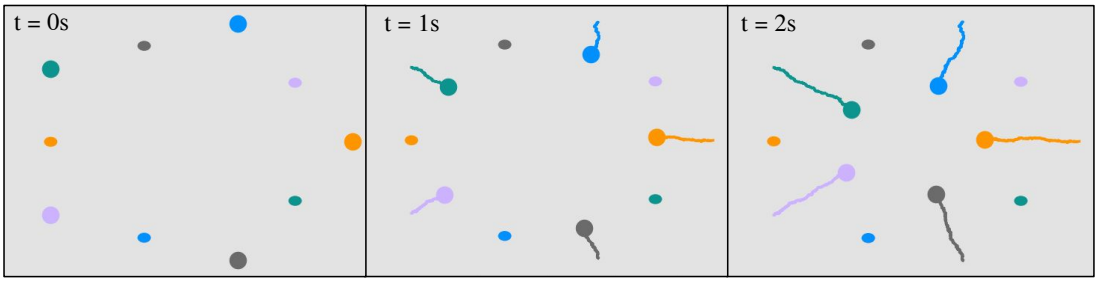

Fig. 16: Snapshots of multi-agent navigation trajectories at different time instances using **MPPI-NCBF**. The observed trajectories demonstrate **suboptimal behavior similar to that of the MPPI policy**. Consequently, this results in high-performance costs, indicating its **inability to effectively co-optimize safety and performance.**

IX. Implementation Details of the Algorithms

This section provides an in-depth overview of our algorithm and baseline implementations, including hyperparameter configurations and the cost/reward functions used in the baselines across all experiments.

A. Experimentation Hardware

All experiments were conducted on a system equipped with an 11th Gen Intel Core i9-11900K @ 3.50GHz × 16 CPU, 128GB RAM, and an NVIDIA GeForce RTX 4090 GPU for training.

Hyperparameter	**Value**
Trade-off parameter (λ)	100
Planning Horizon	20
Softmax Lambda	200
No. of Rollouts	8000

TABLE V: Hyperparameters for MPPI Baselines

Hyperparameter	**Value**
Network Architecture	Multi-Layer Perceptron (MLP)
Number of Hidden Layers	3
Activation Function	Sine function
Hidden Layer Size	256 neurons per layer
Optimizer	Adam optimizer
Learning Rate	2×10^{-5}
Boat Navigation	.
Number of Training Points	65000
Number of Pre Training Epochs	50K
No. of Training Epochs	200K
Pursuer Vehicle Tracking Evader	.
Number of Training Points	65000
Number of Pre Training Epochs	60K
No. of Training Epochs	300K
Multi Agent Navigation	.
Number of Training Points	65000
Number of Pre Training Epochs	60K
No. of Training Epochs	400K

TABLE VI: Hyperparameters for the proposed algorithm

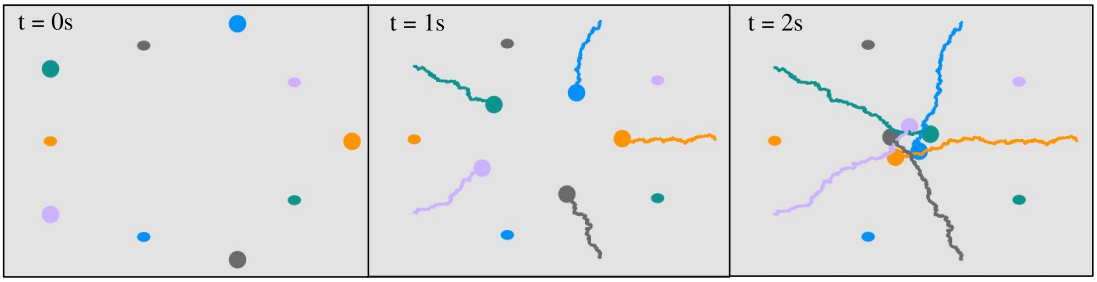

Fig. 17: Snapshots of multi-agent navigation trajectories at different time instances using SAC-Lag. The trajectories indicate that agents **demonstrate less conservative behavior compared to MPPI and MPPI-NCBF, but they lead to collisions**. These **safety violations are critical** and cannot be disregarded, further **highlighting the limitations of the baseline methods in simultaneously optimizing safety and performance.**

Parameter	**Value**
Policy Architecture	Multi-Layer Perceptron (MLP)
learning rate	3×10^{-4}
buffer size	$1,000,000$
learning starts	$10,000$
batch size	256
Target network update rate (τ)	0.005
Discount factor (γ)	0.99
Boat Navigation	.
Number of Training Steps	1,000,000
Pursuer Vehicle Tracking Evader	.
Number of Training Steps	2,500,000
Multi Agent Navigation	.
Number of Training Steps	1,000,000

TABLE VII: General Hyperparameters of SAC in our experiments

B. Hyperparameters for the Proposed Algorithm

We maintained training settings across all experiments, as detailed below:

C. MPPI based baselines

For all the experiments we consider the MPPI cost term as follows:

$$C_{MPPI} = C(t, x(t), \mathbf{u}) + \lambda \max(g(x), 0) \tag{56}$$

where, λ is the trade-off parameter, $C(t, x(t), \mathbf{u})$, $g(x)$ are the cost functions and safety functions as defined in Appendix VIII. Table V lists the hyperparameters we have used for MPPI experiments in all the cases:

D. SAC-Lag hyperparameters

For all the experiments, we consider the reward term as follows:

$$\begin{aligned} R_{SAC\text{-}Lag} = &- C(t, x(t), \mathbf{u}) - \mathbb{I}_{g(x)>0} \times (100) \\ &+ \mathbb{I}_{l(t,x(t))<0.1} \times (100) \end{aligned} \tag{57}$$

where, $C(t, x(t), \mathbf{u})$, $g(x)$ are the cost functions and safety functions as defined in Appendix VIII. Table VII provides the list of hyperparameters we have used for SAC experiments in all the cases.

E. PPO-Lag hyperparameters

For all the experiments, we consider the reward term as follows:

$$\begin{aligned} R_{PPO\text{-}Lag} = &- C(t, x(t), \mathbf{u}) - \mathbb{I}_{g(x)>0} \times (100) \\ &+ \mathbb{I}_{l(t,x(t))<0.1} \times (100) \end{aligned} \tag{58}$$

where, $C(t, x(t), \mathbf{u})$, $g(x)$ are the cost functions and safety functions as defined in Appendix VIII. Table VIII provides the list of hyperparameters we have used for PPO experiments in all the cases.

Parameter	Value
Policy Architecture	Multi-Layer Perceptron (MLP)
learning rate	3×10^{-4}
buffer size	$1,000,000$
learning starts	$10,000$
batch size	256
Target network update rate (τ)	0.005
Discount factor (γ)	0.99
Boat Navigation	.
Number of Training Steps	1,000,000
Pursuer Vehicle Tracking Evader	.
Number of Training Steps	2,500,000
Multi Agent Navigation	.
Number of Training Steps	1,000,000

TABLE VIII: General Hyperparameters of PPO in our experiments

Parameter	Value
Policy Architecture	Multi-Layer Perceptron (MLP)
Batch Size	128
Target KL Divergence	0.01
Entropy Coefficient	0.0
Reward Discount Factor (γ)	0.99
Cost Discount Factor (γ_c)	0.99
GAE Lambda (λ)	0.95
Cost GAE Lambda (λ_c)	0.95
Critic Norm Coefficient	0.001
Penalty Coefficient	0.0
Conjugate Gradient Damping	0.1
Conjugate Gradient Iterations	15
Actor Hidden Sizes	$[256, 256]$
Critic Hidden Sizes	$[256, 256]$
Critic Learning Rate	0.001
Boat Navigation	.
Number of Training Steps	10,000,000
Pursuer Vehicle Tracking Evader	.
Number of Training Steps	25,000,000
Multi Agent Navigation	.
Number of Training Steps	10,000,000

TABLE IX: CPO Hyperparameters from OmniSafe Configuration used for our experiments

F. CPO hyperparameters

For all the experiments, we consider the reward term as follows:

$$R_{CPO} = -C(t, x(t), \mathbf{u}) + \mathbb{I}_{l(t,x(t))<0.1} \times (100) \tag{59}$$

where $C(t, x(t), \mathbf{u})$, $g(x)$ are the cost functions and safety functions as defined in Appendix VIII. For the CPO implementation, we have used the training settings used in [37]. Table IX provides the list of hyperparameters we have used for CPO experiments in all the cases.