Data-Driven Safe Policy Optimization for Black-Box Dynamical Systems With Temporal Logic Specifications

Chenlin Zhang, Shijun Lin, Hao Wang, Ziyang Chen, Shaochen Wang, *Graduate Student Member, IEEE*, and Zhen Kan, *Senior Member, IEEE*

***Abstract*— Learning-based policy optimization methods have shown great potential for building general-purpose control systems. However, existing methods still struggle to achieve complex task objectives while ensuring policy safety during learning and execution phases for black-box systems. To address these challenges, we develop data-driven safe policy optimization (D^2SPO), a novel reinforcement learning (RL)-based policy improvement method that jointly learns a control barrier function (CBF) for system safety and a linear temporal logic (LTL) guided RL algorithm for complex task objectives. Unlike many existing works that assume known system dynamics, by carefully constructing the data sets and redesigning the loss functions of D^2SPO, a provably safe CBF is learned for black-box dynamical systems, which continuously evolves for improved system safety as RL interacts with the environment. To deal with complex task objectives, we take advantage of the capability of LTL in representing the task progress and develop LTL-guided RL policy for efficient completion of various tasks with LTL objectives. Extensive numerical and experimental studies demonstrate that D^2SPO outperforms most state-of-the-art (SOTA) baselines and can achieve over 95% safety rate and nearly 100% task completion rates. The experiment video is available at https://youtu.be/2RgaH-zcmkY.**

***Index Terms*— Control barrier function (CBF), linear temporal logic (LTL), reinforcement learning (RL), safe policy optimization.**

I. Introduction

Designing control policies for complex robotic tasks with safety constraints is difficult and even more challenging if the system dynamics is unknown. A popular method to deal with black-box systems is reinforcement learning (RL), which is a sequential decision making process in which agent finds optimal policies by interacting with the environment to maximize reward collection, including Atari [1], robotic motion planning [2], and images segmentation [3]. However, unfettered exploration in RL can lead to unsafe or undesired behaviors. For instance, when training drones for cruising tasks, they may collide with obstacles during exploration, as well as struggle to navigate to specific locations. Finding an optimized policy that can perform tasks safely is often referred to safe policy optimization, where "safe" means that the agents have a very low probability of violating constraints (i.e., entering unsafe subsets) when operating in the environment, e.g., in a cruise mission, the drone will not collide with obstacles either during training or deployment. While recent works investigated this safety issue via safety certificates [4], [5] or based on accurate system models [6], [7], few of them consider both safety and task complexity at the same time. Therefore, this work is motivated to design a provable safe policy optimization method for black-box dynamical systems to perform complex tasks (i.e., tasks subject to temporal and logic constraints).

Optimization-based approaches show great promise in dealing with dynamical systems. For instance, a novel position-transitional particle swarm optimization algorithm was developed in [8]. In [9], an inner second-order solver that employs a Hessian-free method was developed to avoid the highly expensive manipulations of a Hessian matrix. In [10], a gradient-based differential neural-solution was developed for time-dependent nonlinear optimization problems subject to linear inequality and equality constraints. To achieve safe policy optimization, control barrier functions (CBFs) have been widely explored [6], [7], [11], [12], [13], which keep the system evolving inside a safe invariant set. However, designing a qualified CBF is challenging and hand-crafted certificates for particular systems are often required. To address these issues, data-driven methods that fit CBFs with neural networks have shown significant progress. For instance, the CBF was parameterized by the support vector machine in [14] where the state space was characterized as either safe or unsafe based on the collected sensor data via supervised learning. However, such an approach cannot guarantee in advance the existence of control actions such that the learned safe set is forward invariant. In [15] and [16], safe control policies and barrier certificates were jointly learned to achieve their goals while avoiding collisions with static obstacles and other agents. However, these works are only empirically validated, without establishing formal safety guarantees. As an exception, the work of [17] develops an optimization-based approach to learn the CBF from expert trajectories with provable safety guarantees. A key assumption behind these data-driven-based approaches is that an explicitly differentiable model of the system dynamics is required. If the dynamical systems are not accurately modeled or complete black-box, the aforementioned methods are no longer applicable. Recently, learning barrier certificates for black-box dynamical systems was investigated in [18]. The work of [19] extends the single-step invariant property of the barrier certificate to a multi-step version and then learns a neural barrier certificate under stepwise state constraint setting. However, the works of [18] and [19] still lack provable safety. Safe RL [20], [21], [22] is a popular method for safe policy optimization, which can handle black-box dynamical systems. Constrained Markov decision processes (CMDPs) is an alternative to address system safety. To solve CMDP problems, inspired by the trust domain, the constrained policy optimization (CPO) was developed in [23] and Lagrange multipliers-based approaches were developed in [24]. Based on CPO, the projection-based CPO (PCPO) [25] solves the problem of slow convergence after encountering infeasible solutions in CPO. In [26], the safe policy adaptation with constrained exploration (SPACE) uses a baseline policy to accelerate training. However, these methods depend on convex approximation for objective and constraints, which will cause residual error and lead to poor performance. As for Lagrange multipliers-based approaches, a novel multitimescale-constrained actor-critic approach, namely reward CPO (RCPO) was developed in [27], which leverages penalty signals to guide the policy learning to meet the constraints. However, the above

Manuscript received 13 February 2023; revised 7 November 2023; accepted 3 December 2023. Date of publication 18 December 2023; date of current version 6 February 2025. This work was supported in part by the National Natural Science Foundation of China under Grant U2013601 and Grant 62173314. *(Chenlin Zhang, Shijun Lin, and Hao Wang contributed equally to this work.) (Corresponding author: Zhen Kan.)*

The authors are with the Department of Automation, University of Science and Technology of China, Hefei, Anhui 230026, China (e-mail: zkan@ustc.edu.cn).

Digital Object Identifier 10.1109/TNNLS.2023.3339885

safe RL algorithms suffer from challenging reward design, lack of interpretable safety guarantees, low sampling efficiency, and complex constraints, making it difficult to yield satisfactory performance. A provable and efficient safe policy optimization method is needed when encountering black-box dynamical systems.

Besides safety concerns, another challenge is the synthesis of control policies for complex task constraints. Due to the rich expressivity in specifying complex logic and temporal constraints, linear temporal logics (LTLs) are used with RL to enable efficient learning [28], [29], [30], [31]. However, most of these methods struggle to guarantee the safety during exploration and execution. In contrast, the works of [32] and [33] ensure exploration safety and policy safety while satisfying complex task constraints by integrating CBF with temporal logic-guided RL (TLGRL). However, the CBF in [32] and [33] requires accurate dynamics model and has to be manually designed, which imposes considerable challenges in real-world applications.

In this work, we propose data-driven safe policy optimization (D^{2}SPO), a new RL-based policy improvement method for black-box dynamical systems that jointly learns a CBF for system safety and an LTL-guided RL algorithm for complex task objectives. Since synthesizing CBF by hand for dynamical systems is challenging, data-driven methods are exploited to learn CBF from collected demonstrations of safe and desirable behaviors. Unlike many existing works that assume known system dynamics, by carefully constructing the data sets and redesigning the loss functions of D^{2}SPO to enable the back propagation of the gradient to the control policy, a provably safe CBF is learned for black-box dynamical systems. During the learning process, the CBF continuously evolves using the online data collected by the RL for improved system safety. In addition, D^{2}SPO takes advantage of the capability of LTL in representing the task progress, and develops LTL-based reward functions to guide the agent learning RL policy. Thanks to the fact that LTL can decompose the training task at an abstract level and inform the agent about their current task progress to facilitate robot learning, D^{2}SPO can learn to accomplish tasks with complex logic and temporal constraints more efficiently.

The main contributions are summarized as follows.

1) A novel D^{2}SPO is developed for black-box dynamics subject to safety constraints and complex LTL tasks constraints. A provably safe CBF is learned for black-box dynamical systems, which continuously evolves for improved system safety as RL interacts with the environment.
2) The developed D^{2}SPO takes advantage of RL algorithms in learning high-performance controllers while utilizing CBF as safety certificates to guide the exploration of polices. Hence, D^{2}SPO can greatly enhance the system safety not only by the learned policy but also during the exploration process of RL.
3) Extensive numerical and experimental studies demonstrate that D^{2}SPO outperforms most state-of-the-art (SOTA) baselines and can achieve over 95% safety rate and nearly 100% task completion rates.

II. Preliminaries

A. Reinforcement Learning

As a sequential decision-making process, RL is usually modeled as a Markov decision process (MDP) $\mathcal{M} = (S, A, \mathcal{T}, r, \gamma)$, where S is the state space, A is the action space, $\mathcal{T} : S \times A \mapsto S$ is the state-action transition function, $r : S \times A \times S \mapsto \mathbb{R}$ is the reward function, and γ is the discount factor. Given a policy π that maps a state to an action, the reward r is received after applying the action. The expected cumulative reward is defined as $Q_\pi = \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t r_t]$. A common goal is to learn an optimal policy $\pi^* = \text{argmax}_\pi Q_\pi$. Table I summarizes the symbols.

TABLE I
SYMBOL APPOINTMENT

Symbol	Description
S	state space
A	action space
T	$S \times A \mapsto S$, the state-action transition function
r	reward function
γ	discount factor
h	$\mathbb{R}^n \to \mathbb{R}$, a twice continuously differentiable function
ϕ	LTL formula
$s(t)$	system state, $s(t) \in \mathbb{R}^n$
$u(t)$	system control input, $u(t) \in \mathbb{R}^m$
f	system dynamics
$\triangle t$	an extremely small sampling rate
$s_n(t + \triangle t)$	the estimated state at $t + \triangle t$ using the nominal model f_n
$s_g\ (t + \triangle t)$	an error-free estimation of $s\ (t + \triangle t)$
z_i	current state $s(t)$
$z_i^{'}$	the state $s_g(t + \triangle t)$
O_s	$\{obs_i\}_{i=1}^{M^{'}}$, represent a set of obstacles within the sensing range of the agent at $s(t)$
$b(z_i, O_s)$	the distance from z_i to the closest obstacle in O_s
x_i and $x_i^{'}$	$b(z_i, O_s)$ and $b(z_i^{'}, O_s)$, respectively
$d \in \mathbb{R}^+$	a desired safety threshold
$\mathcal{B}_{\epsilon,p}(x_i)$	$\{ x \in \mathbb{R}^n \mid \| x - x_i \|_p \leq \epsilon \}$, the closed p-norm ball centered at $x_i \in \mathbb{R}^n$ with radius ϵ
$bd(D^{'})$	the boundary of $D^{'}$
$Lip(\cdot, \epsilon)$	the upper bound of the Lipschitz constant given its argument is in an ϵ-neighborhood
$\gamma_s, \gamma_u, \gamma_{exp}, L_h$ and L_e	positive constants determined by the ϵ and $\bar{\epsilon}$-nets using the data from the data-sets X_s and $X_{\mathcal{L}}$
q and q'	$L\left(z_{t-1}\right)$ and $L\left(z_t\right)$, respectively, denote the automaton states of B_ϕ as defined in Sec. II-B
L	maps the state z to an automaton state
k_1 and k_2	tuning parameters, $\in \mathbb{R}^+$
d_s	the distance to the destination
I_{ep}	the index of episode
$\lambda_s, \lambda_u, \lambda_e, \lambda_g$	all $\in \mathbb{R}^+$, tuning parameters indicating the relative importance to balance the goal-reaching and safety objectives

B. Linear Temporal Logic

LTL is a formal language capable of expressing rich task specifications. Given the atomic propositions AP, the syntax of the LTL formula is defined as follows:

$$\phi ::= \text{true} \mid \text{ap} \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \bigcirc\phi \mid \phi_1 \mathsf{U} \phi_2$$

where true is the Boolean value, ap $\in$ AP is an atomic proposition, ϕ, ϕ_1, ϕ_2 are LTL formulas, $\neg$ (negation) and $\wedge$ (conjunction) are standard Boolean operators, and $\bigcirc$ (next) and U (until) are temporal operators. The semantics of an LTL formulae is defined over an infinite sequence $\sigma = \sigma_0\sigma_1, \ldots$ with $\sigma_i \in 2^{\text{AP}}$, $i \in \mathbb{Z}_{\geq 0}$, where 2^{AP} represents the power set of AP. Denote by $\sigma \models \phi$ if the word σ satisfies the LTL formula ϕ. Detailed descriptions of the syntax and semantics of LTL can be found in [34].

An LTL formula can be translated to a nondeterministic Büchi automaton (NBA) $B = \{\mathcal{Q}, q_0, \Sigma, \Delta, F\}$, where $\mathcal{Q}$ is a finite set of automaton states, and $q_0 \in \mathcal{Q}$ is the initial state, $\Sigma = 2^{\text{AP}}$ is the input alphabet, $\Delta : S \times S \to 2^\Sigma$ is the transition function, and $F \in \mathcal{Q}$ is the set of accepting states. For any LTL formula ϕ, one can construct an NBA with input alphabet Σ accepting all and only words that satisfy ϕ [34]. NBA will be used in this work to indicate the task progress. To convert an LTL formula to an NBA, readers are referred to [35] for algorithms and implementations.

III. Problem Formulation

Consider a black-box dynamical system

$$\dot{s}(t) = f(s(t), u(t)), \quad s(0) \in \mathbb{R}^n \tag{1}$$

where $s(t) \in \mathbb{R}^n$ and $u(t) \in \mathbb{R}^m$ represent the system state and control input, respectively, and the system dynamics f is unknown.

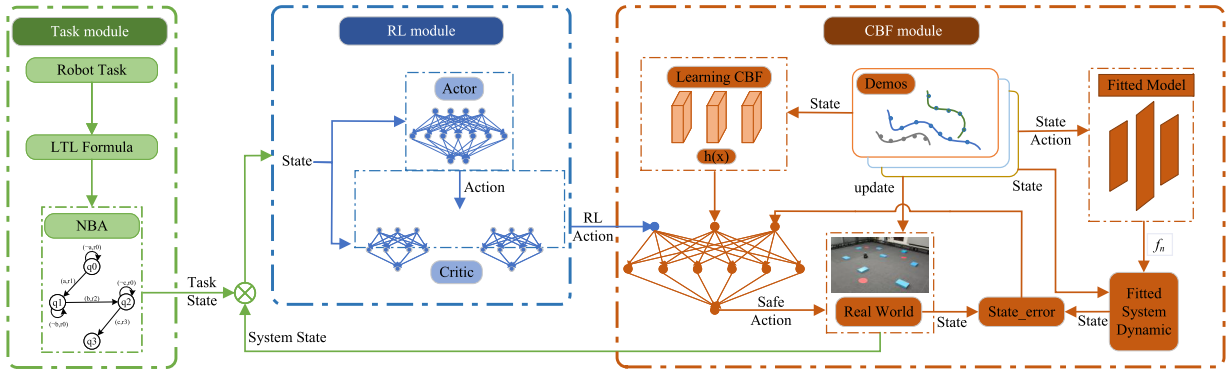


Fig. 1. Framework of D^{2}SPO consists of three main components: the robot task expressed by LTL specification, the learning of CBF for system safety, and the learning of policies for LTL tasks. Using online collected demonstrations of safe and desirable behaviors, the CBF is continuously learned and improved, which is then integrated with the LTL-guided RL policy to achieve safe exploration and robot control.

The system in (1) is considered as safe if the states are restricted within the set

$$\mathcal{C} \coloneqq \{s \in \mathbb{R}^n \mid h(s) \geq 0\} \tag{2}$$

where $h\colon \mathbb{R}^n \to \mathbb{R}$ is a twice continuously differentiable function. That is, the set $\mathcal{C}$ models system safety specifications.

Definition 1 [6]: Let $\mathcal{D}$ be an open set such that $\mathcal{D} \supset \mathcal{C}$. The function $h(s)$ is called a CBF on $\mathcal{D}$ if there exists a locally Lipschitz continuous extended class $\mathcal{K}$ function $\alpha\colon \mathbb{R} \to \mathbb{R}$ such that for all $s \in \mathcal{D}$

$$\sup_{u \in \mathcal{U}} \{\langle \nabla h(s), f(s,u)\rangle + \alpha(h(s))\} \geq 0 \tag{3}$$

where $\mathcal{U} \in \mathbb{R}^m$ is the control space. The set of inputs induced by the CBF $h(s)$ is defined as follows:

$$\mathcal{I}(s) \coloneqq \{u \in \mathcal{U} \mid \langle \nabla h(s), f(s,u)\rangle + \alpha(h(s)) \geq 0\}. \tag{4}$$

The following lemma indicates that for all $s \in \mathcal{D}$, the safety of system (1) can be guaranteed with appropriate control input $u(s) \in \mathcal{I}(s)$.

Lemma 1 [6]: Assume that $h(s)$ is a CBF on $\mathcal{D}$ and $u(s) \in \mathcal{I}(s)$ is locally Lipschitz continuous. Then, it holds that $s(t) \in \mathcal{C}$ for all $t \geq 0$ if $s(0) \in \mathcal{C}$.

Lemma 1 indicates, given a valid CBF, the system safety is guaranteed. However, the constructed CBF in (3) builds upon a key assumption that the system dynamics $f(s,u)$ is known. If the system dynamics is unknown, as in this work, significant challenges are imposed. Hence, given a black-box dynamical system and an LTL task ϕ, the goal of this work is to: 1) learn the CBF using data collected during run time without knowing system dynamics and 2) develop safe RL algorithms that synthesize the learned CBF to accomplish the LTL task ϕ.

Example 1: To elaborate this idea, we use a navigation task as a running example throughout this work. Consider a mobile robot navigating in an environment scattered with a set of M obstacles $\{obs_i\}_{i=1}^M$. The robot dynamics is unknown as in (1) and it can only detect the obstacles within its sensing range, e.g., a disk area with radius H. The agent is required to navigate among a set of destinations while avoiding collision with obstacles. Such a task is represented as an LTL formula ϕ. Let (x, y) and Θ denote the position and orientation of robot and let $\mathcal{G}$ denote the geometric safe set.[1] The control inputs are the angular velocity ω and linear velocity v.

[1]The geometric safe set $\mathcal{G}$ is a superset of $\mathcal{C}$, i.e., $\mathcal{C} \subset \mathcal{G}$, that can be directly specified on the state-space of the system (i.e., the free motion region without collisions with obstacles).

IV. Data-Driven Safe Policy Optimization

This section presents the D^{2}SPO for black-box dynamical systems with safety constraints and LTL task constraints. We first present how the dataset of expert demonstration is constructed in Section IV-A, based on which an optimization-based approach is developed to facilitate the learning of CBF in Section IV-B. Together with the LTL-guided RL policy developed in Section IV-C, the D^{2}SPO is finally presented in Section IV-D which generates safe policies for the completion of LTL tasks. The architecture of D^{2}SPO is illustrated in Fig. 1.

A. Construction of the Datasets of Safe Demonstration

Despite the unknown dynamics, suppose that the states $s(t)$ and $s(t+\triangle t)$ of the system (1) can be sampled. Unlike many existing works with known differentiable dynamics f, due to the considered black-box dynamics, the policy gradient cannot be back-propagated to update the controller in this work. To address this challenge, we consider a differentiable nominal model f_n, which can be established using many existing methods, e.g., fitting a neural network using sampled system states [36]. Note that f_n does not have to perfectly match the system (1).

Given the current control input $u(t)$ and the sampled system state $s(t)$, the estimated state at $t+\triangle t$ using the nominal model f_n is denoted by

$$s_n(t+\triangle t) = s(t) + f_n(s(t), u(t))\triangle t. \tag{5}$$

Based on (5) and the sampled system state $s(t+\triangle t)$, inspired by [18], we construct

$$s_g(t+\triangle t) = s_n(t+\triangle t) + g(s(t+\triangle t) - s_n(t+\triangle t)) \tag{6}$$

where $g(s) = s$ is an identity function but without gradient. That is, $s_g(t+\triangle t)$ is an error-free estimation of $s(t+\triangle t)$. The requirement of no gradient is to avoid the back-propagation of the its gradient to the argument s.

Based on (5) and (6), a set of N discretized data-point-pairs is constructed as $T \coloneqq \{(z_i, z_i', O_s)\}_{i=1}^N$. To facilitate the following development, we use z_i and z_i' to represent the current state $s(t)$ and the state $s_g(t+\triangle t)$, respectively, and use $O_s = \{obs_i\}_{i=1}^{M'}$ to represent a set of obstacles within the sensing range of the agent at $s(t)$. To learn CBF via collected data, a crucial step is to identify a set of safe states that can be exploited to learn the CBF. Hence, after obtaining the data-point-pairs T, the next is to construct the safe demonstration P_{exp} from T. Specifically, we propose $b(z_i, O_s)$ refer to I, based on which we construct the set of data-point-pairs $P_{\text{exp}} \coloneqq \{(x_i, x_i') \mid x_i \geq d\}_{i=1}^{N_1}$., where $x_i = b(z_i, O_s)$, $x_i' = b(z_i', O_s)$,

$\forall(z_i, z_i', O_s) \in T$ and $d \in \mathbb{R}^+$ is a desired safety threshold. Note that P_{exp} is defined based on the relative distance, which improves the response to the obstacles. In Sections IV-B–IV-D, we will show how P_{exp} can be exploited to learn the CBF. It is worth pointing out that P_{exp} is online updated during the runtime, which enables continuous improvement of CBF using the newly generated date by RL.

B. Learning of Control Barrier Function

In this section, we present how the CBF is learned from P_{exp} without knowing the system dynamics. We first define regions $D' := \bigcup_{i=1}^{N_1} B_{\epsilon,p}(x_i)$, $\forall x_i \in P_{\text{exp}}$, and $D := D' \setminus bd(D')$, where $\mathcal{B}_{\epsilon,p}(x_i) := \{x \in \mathbb{R}^n | \parallel x - x_i \parallel_p \leq \epsilon\}$ denotes the closed p-norm ball centered at $x_i \in \mathbb{R}^n$ with radius ϵ and $bd(D')$ denotes the boundary of D'. By adjusting ϵ or ignoring the sampled data near the boundary of D, $D \subseteq \mathcal{G}$ can be ensured. Then, we construct the set[2]

$$\mathcal{L} \triangleq \left\{bd(D) \oplus B_{\sigma,p}(0)\right\} \setminus D$$

which can be considered as a ring with width σ around D. The idea behind is to define regions such that $h(x_i) \geq 0$ for $x_i \in D$ and $h(x_i) < 0$ for $x_i \in \mathcal{L}$ so that CBF can be learned. Based on D, the safe dataset is denoted as $X_s = \{x_i | (x_i, x_i') \in P_{\text{exp}}\}$. Similarly, to get unsafe dataset, $X_{\mathcal{L}} = \{x_i\}_{i=0}^{N_2}$ is constructed by sampling from $\mathcal{L}$. The set X_s and $X_{\mathcal{L}}$ form the ϵ-net and $\bar{\epsilon}$-nets on D and $\mathcal{L}$, respectively, where the ϵ-net indicates that for any $x \in D$, there exists $x_i \in X_s$ such that $\|x - x_i\|_p \leq \epsilon$.

Consider a twice continuously differentiable function $h : \mathbb{R}^n \to \mathbb{R}$ with the local Lipschitz bound

$$L_h(x) \triangleq \sup_{x_1, x_2 \in \mathcal{B}_{\epsilon,p}(x)} \frac{| h(x_1) - h(x_2) |}{\| x_1 - x_2 \|_p}.$$

To ensure that the learned CBF is valid with respect to Definition 1, we have the following constraints.

Proposition 1: Suppose $h(x)$ is Lipchitz continuous with constant $L_h(x)$. It holds that $h(x) < 0$ for all $x \in \mathcal{L}$, if

$$h(x_i) \leq -\gamma_u \quad \forall x_i \in X_{\mathcal{L}}$$

where $X_{\mathcal{L}}$ is an $\bar{\epsilon}$-net of $\mathcal{L}$ with $\bar{\epsilon} < \gamma_u / L_h(x_i)$.

The proof is given in appendix. After finding the constraint on $\bar{\epsilon}$, we consider a function that finds the condition for ϵ.

Proposition 2: Denote by $e(x) := \dot{h}(x) + \alpha(h(x))$ and suppose $e(x)$ is Lipchitz continuous with constant $L_e(x)$. It holds that $e(x) \geq 0$ for all $x \in D$, if

$$e(x_i) \geq \gamma_{\text{exp}} \quad \forall x_i \in X_s$$

where X_s is an ϵ-net of D with $\epsilon \leq \gamma_{\text{exp}} / L_e(x_i)$.

The proof is given in appendix. Based on the Propositions 1 and 2, the CBF can be learned from data by solving the following optimization problem:

$$\begin{aligned} &\min \ \| h \| \\ &\text{s.t. } h(x_i) \geq \gamma_s \quad \forall x_i \in X_s \\ &\quad h(x_i) \leq -\gamma_u \quad \forall x_i \in X_{\mathcal{L}} \\ &\quad \text{Lip}(h(x_i), \bar{\epsilon}) \leq L_h \quad \forall x_i \in X_{\mathcal{L}} \\ &\quad e(x_i) \geq \gamma_{\text{exp}} \quad \forall x_i \in X_s \\ &\quad \text{Lip}(e(x_i), \epsilon) \leq L_e \quad \forall \left(x_i, x_i'\right) \in \mathcal{P}_{\text{exp}}. \end{aligned} \tag{7}$$

In (7), $\text{Lip}(\cdot, \epsilon)$ indicates the upper bound of the Lipschitz constant given its argument is in an ϵ-neighborhood, and the hyperparameters γ_s, γ_u, γ_{exp}, L_h, and L_e are positive constants determined by the ϵ and $\bar{\epsilon}$-nets using the data from the datasets X_s and $X_{\mathcal{L}}$. The first and second constraints ensure that $h(x) > 0$ if x is in the safe set X_s and $h(x) < 0$ if x is in the unsafe set $X_{\mathcal{L}}$. The fourth constraint ensures the derivative condition in (3). Although the learning of CBF can be well formulated as an optimization problem as in (7), it cannot be solved as $e(x)$ contains $\dot{h}(x)$, which is not available due to black-box dynamics. We will show in Section IV-D how this issued can be addressed.

C. Temporal Logic-Guided RL

Due to the consideration of black-box dynamical systems, model-free RL is employed to search for a policy π_{rl} to complete the LTL task ϕ. TLGRL is developed in this work, which takes advantage of the capability of LTL in representing the task progress to design the reward function of RL so that π_{rl} can be efficiently learned. Specifically, given the state z_{t-1} at time $t-1$ and the state z_t after applying the action a_t, let $q = L(z_{t-1})$ and $q' = L(z_t)$ denote the automaton states of B_ϕ as defined in Section II-B, where L maps the state z to an automaton state. Since B_ϕ is a graph, for any given $q \in Q$, graph search methods can be employed to determine a node path that starts from q and ends at the accepting set. We denote by $\text{Prog}(q)$ by such a node sequence that excludes q. The reward function is designed as follows:

$$r(z_{t-1}, a_1, z_t) = \begin{cases} -c_n, & \text{if } q' = q \\ c_p, & \text{if } q' \in \text{Prog}(q) \\ r_{\text{goal}}, & \text{if } q' \in F \end{cases} \tag{8}$$

where $c_n, r_{\text{goal}}, c_p \in \mathbb{R}^+$. If $q = q'$, i.e., no mission progress occurs, a negative reward $-c_n$ will be given to motivate transitions to the next state in B_ϕ, as less reward is obtained if more steps are spent in the current state. If the state transits toward the accepting set F, i.e., $q' \in Prog(q)$, a positive reward c_p will be given. Once the task completes, i.e., $q' \in F$, a large reward r_{goal} will be received.

During training, the completion of a subtask (e.g., visiting a destination) is evaluated in the form of $k_1 d_s \cdot e^{k_2 I_{ep}}$. Given a predetermined threshold $\bar{d}$, the subtask is considered as completed if $k_1 d_s \cdot e^{k_2 I_{ep}} \leq \bar{d}$ (i.e., the robot reaches the proximity of the destination). In the early stage of training where I_{ep} is small, a large d_s is allowed to relax the mission completion requirement. As training continues where I_{ep} becomes larger, smaller d_s is enforced to meet the mission completion requirement (i.e., the robot has to be closer to the destination to be considered as mission completion). Such a design, together with the progressive rewards, can improve training efficiency, as the robot can complete the task quickly in the early training to reducing unnecessary exploration.

With the designed reward function r, RL algorithm is used to solve the proposed problem. Due to the consideration of continuous state and action spaces, the twin delayed deep deterministic policy gradient algorithm (TD3) from [37] is adapted in this work. The TD3 algorithm can mitigate the issue of overestimation of Q values by double networks and reduce the correlation of preceding and following actions using the experience buffer. In TD3, there are two critic networks (Twin) representing Q_1 and Q_2 values, which are parameterized by θ_1 and θ_2, respectively. The target networks are parameterized by θ_1' and θ_2', respectively, with the relatively smaller one serving as the target Q network, i.e.,

$$y = r + \gamma \min_{i=1,2} Q_{\theta_i'}\left(z', \tilde{a}\right)$$

where r is the reward designed in (8). The actor network is parameterized by ϕ and the control action a is selected according to the policy network π_ϕ. After applying a, the reward r and new state z' are obtained. Then, at time step t, the agent observes the current

[2]For two sets $\mathcal{D}_1$ and $\mathcal{D}_2$, the Minkowski sum is defined as $\mathcal{D}_1 \oplus \mathcal{D}_2 := \{x_1 + x_2 \in \mathbb{R}^n \mid x_1 \in \mathcal{D}_1, x_2 \in \mathcal{D}_2\}$.

state z_t and takes action a_t. The transition tuple (z, a, r, z') is stored in the replay buffer. To enable adequate exploration and smoothing regularization, clipped Gaussian noise is added to the action as follows:

$$\tilde{a} = \pi_{\phi'}(z') + \xi, \quad \xi \sim \text{clip}\big(\mathcal{N}(0, \tilde{\sigma}), -c, c\big).$$

Randomly sampling mini-batch of T transitions from the replay buffer, the critics are updated by following:

$$\theta_i = \underset{\theta_i}{\text{argmin}} \frac{1}{T} \sum \big(y - Q_{\theta_i}(z, a)\big)^2, \quad i = 1, 2.$$

The policy is optimized by following the gradient

$$\nabla_a Q_{\theta_1}(z, a) \mid_{a=\pi_\phi(z)} \nabla_\phi \pi_\phi(z).$$

D. Data-Driven Safe Policy Optimization

An unconstrained relaxation of problem (7) is proposed in this section which can be solved efficiently in practice by first-order gradient-based methods. The idea is outlined in Algorithm 1, which consists of an outer loop and an inner loop. Since each outer loop has at most 200 inner loop steps, N outer loops has at most $200N$ loop steps, and thus the time complexity of Algorithm 1 is $O(N)$.

Specifically, we model joint control policy π and CBF h as neural networks with parameters Ω and Θ, respectively. The unconstrained optimization is then formulated as follows:

$$T_L = \min T(\Theta, \Omega) \tag{9}$$

where

$$\begin{aligned}
T_s^\Theta &= \sum_{x_i \in X_s} \max\{\gamma_s - h_\Theta(x_i), 0\} \\
T_u^\Theta &= \sum_{x_i \in X_\mathcal{L}} \max\{\gamma_u + h_\Theta(x_i), 0\} \\
T_e^{\Theta,\Omega} &= \sum_{(x_i, x_i') \in P_{\text{exp}}} \max\left\{\gamma_{\text{exp}} - \frac{h_\Theta(x_i') - h_\Theta(x_i)}{\triangle t} \right. \\
&\qquad \left. - \alpha(h_\Theta(x_i)), 0\right\} \\
T_g^\Omega &= \sum_{x_i \in X_s, z_i \in Z_{\text{exp}}} \| \pi_\Omega(x_i) - \pi_{\text{rl}}(z_i) \|_2^2
\end{aligned}$$

where $T(\Theta, \Omega) \triangleq \| \Theta \|^2 + \lambda_s T_s^\Theta + \lambda_u T_u^\Theta + \lambda_e T_e^{\Theta,\Omega} + \lambda_g T_g^\Omega$. and $\lambda_s, \lambda_u, \lambda_e, \lambda_g \in \mathbb{R}^+$ are tuning parameters indicating the relative importance to balance the goal-reaching and safety objectives. In (9), T_s^Θ and T_u^Θ are consistent with the first two constraints in (7). T_g indicates that the joint control policy π_Ω should be close to the RL policy π_{rl} generated in Section IV-C to maximally complete the LTL task ϕ. Since f is unknown in (1), the gradient cannot be back-propagated to Ω. Instead, we approximate $\dot{h}(x)$ by $(h(x') - h(x))/\triangle t$, where x and x' are defined in Section IV-A. It is worth pointing out that the datasets are not fixed during training and will be periodically updated with new samples during the runtime by the current controller.

The following theorem shows that $T_e^{\Theta,\Omega}$ is a valid candidate to learn the CBF.

Theorem 1: $T_e^{\Theta,\Omega}$ is differentiable with respect to the parameter Ω, and when $\triangle t \to 0$, this is an error-free approximation.

Proof: When the fit model is differentiable, s_n and s_g in (6) are differentiable w.s.t. π, T_e is also differentiable w.s.t. π and its parameter Ω. Thus, $\nabla_\Omega T_e$ exists. Since $s_g(t + \triangle t)$ is an error-free estimation of $s(t + \triangle t)$, $\dot{h}(x)$ is an error-free approximation when $\triangle t \to 0$, thus T_e is an error-free approximation when $\triangle t \to 0$. ■

Algorithm 1 Learning Controller for Black-Box Systems

Input: State sets X_s, $X_\mathcal{L}$ and set P_{exp}, loss function $T_L(\Omega, \Theta)$, training iterations N and episode steps E.
Initial: controller π with initial random parameters Ω, CBF h with parameters Θ. Initialize critic networks Q_{θ_1}, Q_{θ_2}, and actor network with random parameters θ_1, θ_2, ϕ. Initialize target networks $\theta_1' \leftarrow \theta_1$, $\theta_2' \leftarrow \theta_2$, $\phi' \leftarrow \phi$. Initialize replay buffer.
for i=0, ... N **do**
  Initialize Trajectory data-set $\mathcal{D} \leftarrow \emptyset$
  GET($z(0)$)
  for j=0, ... E **do**
    $\pi_{rl}(z) \leftarrow \pi(z; \phi)$
    $\mathcal{D}_j \leftarrow$TRAJECTORY($\pi(\cdot; \Omega)$)
    Store transition tuple $(z, \pi(\cdot; \Omega), r, z')$
    $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_j$
    $X_\mathcal{L}, X_s, P_{exp} \leftarrow$UPDATE($\mathcal{D}$)
    if i mod d **then**
      Update θ_i
      Update ϕ, θ_i', ϕ'
    end if
  end for
  $\Omega, \Theta \leftarrow$UPDATE($\Omega, \Theta, X_s, X_\mathcal{L}, P_{exp}, T_L$)
end for
return parameter Ω of π and parameter Θ of h

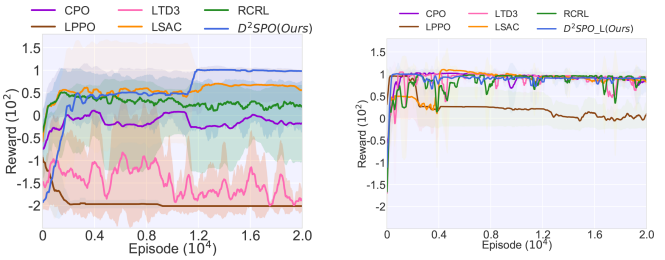


Fig. 2. (a) Evolution of reward function for ϕ_1. (b) Evolution of reward for the relaxed ϕ_1 with fewer obstacles and destinations.

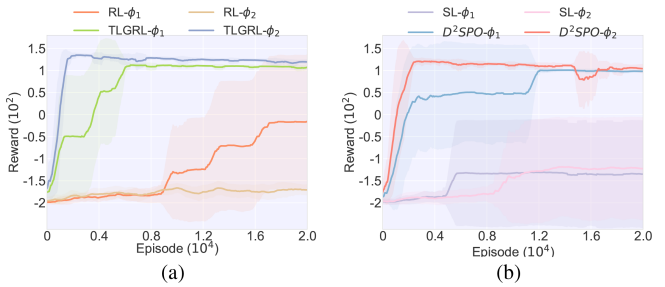


Fig. 3. Performance of TLGRL when completing different level tasks. (a) Is the reward curves for each method on complex cruising missions. (b) Display the performance of high-freedom cruising mission.

V. Simulations and Experiment

Numerical simulation and physical experiments are carried out to evaluate the performance of D^{2}SPO in terms of: 1) the effectiveness of the learned CBF; 2) the learning efficiency of TLGRL in completing desired tasks; and 3) the performance in physical experiment.

A. Experiment Setup

1) Environments: Consider a workspace consisting of four areas of interest and eight obstacles scattered in the environment. The agent

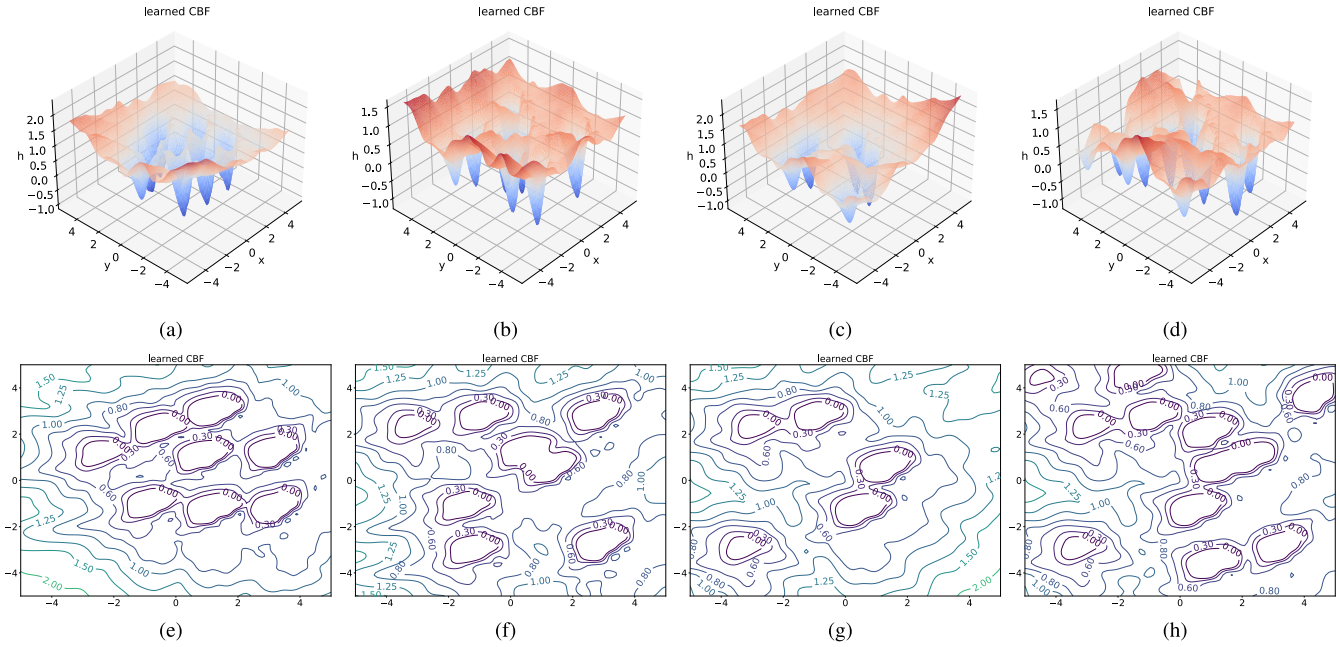


Fig. 4. (a)–(d) Are the surface plots of the learned CBF under different quantities and location and (e)–(h) Are the corresponding level set plots of the learned CBF.

TABLE II
PERFORMANCE OF DIFFERENT METHODS FOR ϕ_1

Method	Safety Rate	Task Completion Rate
CPO [23]	90.71%	19.44%
LPPO [27]	96.84%	0%
LTD3	83.08%	0%
LSAC	77.76%	96.74%
RCRL [38]	74.47%	77.52%
D^{2}SPO (Ours)	**95.23%**	**99.31%**

TABLE III
PERFORMANCE OF DIFFERENT METHODS FOR RELAXED ϕ_1

Method	Safety Rate	Task Completion Rate
CPO [23]	94.49%	98.02%
LPPO [27]	91.59%	63.50%
LTD3	95.10%	97.11%
LSAC	93.94%	98.75%
RCRL [38]	94.19%	94.82%
D^{2}SPO (Ours)	**97.94%**	**99.48%**

is required to monitor the workspace by sequentially visiting the areas of interest while avoiding obstacles. We consider two tasks of different complexities

$$\phi_1 = ((\neg A_3)\mathsf{U}A_2) \wedge ((\neg A_2)\mathsf{U}A_1) \wedge (\Diamond A_1)$$
$$\phi_2 = ((A_1 \wedge (\bigcirc A_3)) \vee (A_3 \wedge (\bigcirc A_1))) \wedge (\bigcirc(\Diamond(A_4 \wedge (\bigcirc(\Diamond A_2)))))$$

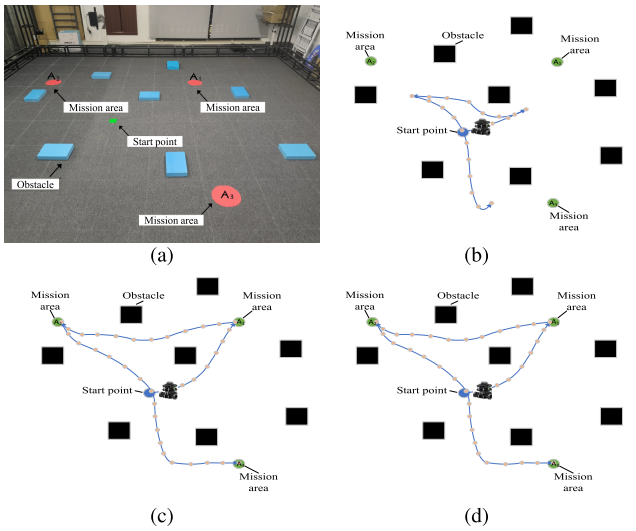
where A_i, $i = 1, \ldots 4$, represents the ith area of interest. The task ϕ_1 requires the agent to visit the areas of interest in a given order (i.e., $A_1A_2A_3$). Such an order is relaxed in task ϕ_2 to give the agent more freedom to complete the task (i.e., $A_1A_3A_4A_2$ or $A_3A_1A_4A_2$). In the following analysis, ϕ_1 will be used to evaluate the performance of our approach against the SOTA methods, while both ϕ_1 and ϕ_2 will be used to validate the effectiveness of TLGRL.

2) Baseline: For comparisons, we consider the following SOTA methods: 1) CPO [23]; 2) reaching constrained RL (RCRL) [38], and 3) a set of variants of Lagrangian methods [27] with mainstream RL algorithm, i.e., PPO-Lagrangian (LPPO), TD3-Lagrangian (LTD3), and SAC-Lagrangian (LSAC).

3) Indicator Definitions: These methods, as well as our approach, are trained for 20 000 episodes with a maximum of 200 steps per episode under the same task and environment. The safety rate is defined as the ratio of the number of episodes that the agent does not hit the obstacle to the total episode number. The task completion rate is defined similarly as the ratio of the number of episodes that the agent completes the task to the total episode number.

B. Main Results

The simulation results are shown in Table II and Fig. 2(a), where the safety rate and the task completion rate for the last 2000 episodes are reported. The results indicate that the SOTA methods either fail to accomplish the specified task or violate the safety constraints. This is due to the difficulty of CPO in guaranteeing safety during the training process. The Lagrangian-based safe RL methods (LPPO, LTD3, and LSAC) aim to find a balance between accomplishing tasks and satisfying safety constraints, and it is found that the Lagrangian-based methods are very sensitive to the initial values of the Lagrange multipliers. Since the parameters in the multipliers and RL are updated alternately and can converge to a saddle point, its performance varies considerably with different RL algorithms. In our experiments, PPO and TD3 are more inclined to satisfy the safety constraints under the same parameter conditions, but this also greatly limits their exploration performance, making it difficult to complete the task. It can be seen that SAC has a high task success rate, which is due to the fact that SAC obtains superior exploration performance by maximizing the entropy of the action distribution in each state compared to PPO and TD3. In contrast, our method guarantees training and deployment safety through the learned CBF, while guaranteeing task completion through TLGRL algorithm, and



(a) (b) (c) (d)

Fig. 5. (a) Physical environment. (b)–(d) Corresponding environment in simulation, where black squares indicate the obstacles, the blue and green dots indicate the initial position of the robot and the areas of interest. The performance after 2000 episodes, 10 000 episodes, and 20 000 episodes of training are shown in (b)–(d), respectively.

thus outperforms these baselines in terms of task completion and safety guarantees.

C. Ablation Study

1) Safety in Simple Tasks: To show the performance on simple tasks, we consider a relaxed task that simplifies ϕ_1 by considering the same navigation task but only with one obstacle and two areas of interest. As indicated in Table III and Fig. 2(b), for relatively simple tasks, these SOTA methods can yield satisfying performance in terms of mission completion and safety guarantees, although our approach still outperforms them. However, if more complex tasks are considered as in Table II, their performance degrades significantly while our approach still shows promising performance. In summary, the SOTA methods might work for relatively simple tasks, but in general cannot yield satisfactory performance for complex tasks. In contrast, our approach works for both simple and complex tasks and outperforms SOTA methods in terms of mission completion and safety guarantees.

2) Role of LTL: To show the learning efficiency of TLGRL, we first consider the environment without obstacles. The ϕ_1 and ϕ_2 are both present to show the benefit of our method. RL-ϕ_1 and RL-ϕ_2 mean that the agent performs ϕ_1 and ϕ_2 using the method in [37], respectively. The performance of RL-ϕ_1, RL-ϕ_2, TLGRL-ϕ_1, and TLGRL-ϕ_2 are shown in Fig. 3(a). Clearly, TLGRL significantly improves the speed of convergence and the value of the reward compared to basic RL, which means temporal logic actually improves the learning efficiency in RL. Then, the safety-concerned environment is incorporated to show whether the approach in the learning process is still effective or not. Let SL denote the variant of D^{2}SPO without LTL-guided progressive reward. The performance of SL-ϕ_1, SL-ϕ_2, D^{2}SPO-ϕ_1, and D^{2}SPO-ϕ_2 are presented in Fig. 3(b). D^{2}SPO obtains far more return than SL, which is mainly due to the progression reward in (8) that guides the learning toward mission completion.

D. Physical Experiments

Physical experiments are carried out in this section, where the turtlebot with limited sensing capability, e.g., only detect obstacles within a certain radius, is used. To demonstrate the capability of learning a valid CBF via interactions with the environment, we consider four cases with different deployment of obstacles and destinations. TD3 is employed for the turtlebot to explore the environment, which can also be replaced by other RL algorithms. Fig. 4(a)–(d) shows the learned CBF, respectively, while Fig. 4(e)–(h) shows the corresponding contours. Note that the areas below 0 may be connected if the randomly generated obstacles are too close.

One of the four physical environments is shown in Fig. 5(a), where the blue squares indicate the obstacles, the green square is the initial position of the robot, and red disk are the areas of interest. The turtlebot is tasked to access the given area in the order of $A_1A_2A_3$. In the early training phase, although the mission of visiting areas of interest is not completed yet, the robot can successfully avoid obstacles, which demonstrates the safety during exploring. As more training is carried out, the robot finally completes the task with collision avoidance. The experiment video with more explanations is provided.[3]

VI. Conclusion

The D^{2}SPO for black-box dynamical systems is developed in this work, which jointly learns a CBF for system safety and a linear TLGRL algorithm for complex task objectives. Extensive numerical simulation shows that D^{2}SPO outperforms SOTA baselines. The effectiveness of D^{2}SPO is also verified in physical environments.

Appendix

A. Proof of Proposition 3

Proof: For any $x \in \mathcal{L}$, there exists $x_i \in X_{\mathcal{L}}$ satisfying $\|x - x_i\| \leq \bar{\epsilon} < \gamma_u / L_h(x_i)$, we have

$$\begin{aligned} h(x) &= h(x) - h(x_i) + h(x_i) \leq | h(x) - h(x_i) | - \gamma_u \\ &\leq L_h(x_i) \| x - x_i \|_p - \gamma_u \leq L_h(x_i)\bar{\epsilon} - \gamma_u < 0. \end{aligned}$$

■

B. Proof of Proposition 4

Proof: For any $x \in D$, there exists $x_i \in X_s$ satisfying $\|x - x_i\| \leq \epsilon \leq \gamma_{\exp} / L_e(x_i)$, we have

$$\begin{aligned} e(x) &= e(x) - e(x_i) + e(x_i) \geq \gamma_{\exp} - | e(x) - e(x_i) | \\ &\geq \gamma_{\exp} - L_e(x_i) \| x - x_i \|_p \geq \gamma_{\exp} - L_e(x_i)\epsilon \geq 0. \end{aligned}$$

■

References

[1] J. Schrittwieser, T. Hubert, A. Mandhane, M. Barekatain, I. Antonoglou, and D. Silver, "Online and offline reinforcement learning by planning with a learned model," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 27580–27591.

[2] M. Cai, S. Xiao, Z. Li, and Z. Kan, "Optimal probabilistic motion planning with potential infeasible LTL constraints," *IEEE Trans. Autom. Control*, vol. 68, no. 1, pp. 301–316, Jan. 2023.

[3] N. Zeng et al., "Deep-reinforcement-learning-based images segmentation for quantitative analysis of gold immunochromatographic strip," *Neurocomputing*, vol. 425, pp. 173–180, Feb. 2021.

[4] Z. Liu et al., "Constrained variational policy optimization for safe reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2022, pp. 13644–13668.

[5] M. Klissarov, P.-L. Bacon, J. Harb, and D. Precup, "Learnings options end-to-end for continuous action tasks," 2017, *arXiv:1712.00004*.

[6] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs for safety critical systems," *IEEE Trans. Autom. Control*, vol. 62, no. 8, pp. 3861–3876, Aug. 2017.

[3] https://youtu.be/2RgaH-zcmkY

[7] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications," in *Proc. 18th Eur. Control Conf. (ECC)*, Jun. 2019, pp. 3420–3431.

[8] X. Luo, Y. Yuan, S. Chen, N. Zeng, and Z. Wang, "Position-transitional particle swarm optimization-incorporated latent factor analysis," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 8, pp. 3958–3970, Aug. 2022.

[9] W. Li, Q. He, X. Luo, and Z. Wang, "Assimilating second-order information for building non-negative latent factor analysis-based recommenders," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 52, no. 1, pp. 485–497, Jan. 2022.

[10] L. Jin, L. Wei, and S. Li, "Gradient-based differential neural-solution to time-dependent nonlinear optimization," *IEEE Trans. Autom. Control*, vol. 68, no. 1, pp. 620–627, Jan. 2023.

[11] R. Cheng, G. Orosz, R. M. Murray, and J. W. Burdick, "End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 3387–3395.

[12] Y. Yang, K. G. Vamvoudakis, H. Modares, Y. Yin, and D. C. Wunsch, "Safe intermittent reinforcement learning with static and dynamic event generators," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 12, pp. 5441–5455, Dec. 2020.

[13] H. Zhang, Z. Li, and A. Clark, "Model-based reinforcement learning with provable safety guarantees via control barrier functions," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 792–798.

[14] M. Srinivasan, A. Dabholkar, S. Coogan, and P. A. Vela, "Synthesis of control barrier functions using a supervised machine learning approach," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 7139–7145.

[15] S. Yaghoubi, G. Fainekos, and S. Sankaranarayanan, "Training neural network controllers using control barrier functions in the presence of disturbances," in *Proc. IEEE 23rd Int. Conf. Intell. Transp. Syst. (ITSC)*, Sep. 2020, pp. 1–6.

[16] Z. Qin, K. Zhang, Y. Chen, J. Chen, and C. Fan, "Learning safe multi-agent control with decentralized neural barrier certificates," 2021, *arXiv:2101.05436*.

[17] A. Robey et al., "Learning control barrier functions from expert demonstrations," in *Proc. 59th IEEE Conf. Decis. Control (CDC)*, Dec. 2020, pp. 3717–3724.

[18] Z. Qin, D. Sun, and C. Fan, "Sablas: Learning safe control for black-box dynamical systems," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 1928–1935, Apr. 2022.

[19] Y. Yang, Y. Jiang, Y. Liu, J. Chen, and S. E. Li, "Model-free safe reinforcement learning through neural barrier certificate," *IEEE Robot. Autom. Lett.*, vol. 8, no. 3, pp. 1295–1302, Mar. 2023.

[20] T. Mannucci, E.-J. van Kampen, C. de Visser, and Q. Chu, "Safe exploration algorithms for reinforcement learning controllers," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 4, pp. 1069–1081, Apr. 2018.

[21] L. Zhang, R. Zhang, T. Wu, R. Weng, M. Han, and Y. Zhao, "Safe reinforcement learning with stability guarantee for motion planning of autonomous vehicles," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 12, pp. 5435–5444, Dec. 2021.

[22] M. Cai, S. Xiao, J. Li, and Z. Kan, "Safe reinforcement learning under temporal logic with reward design and quantum action selection," *Sci. Rep.*, vol. 13, no. 1, p. 1925, Feb. 2023.

[23] J. Achiam, D. Held, A. Tamar, and P. Abbeel, "Constrained policy optimization," in *Proc. Int. Conf. Mach. Learn*, 2017, pp. 22–31.

[24] S. Bhatnagar and K. Lakshmanan, "An online actor–critic algorithm with function approximation for constraint-mismatched Markov decision processes," *J. Optim. Theory Appl.*, vol. 153, no. 3, pp. 688–708, Jun. 2012.

[25] T.-Y. Yang, J. Rosca, K. Narasimhan, and P. J. Ramadge, "Projection-based constrained policy optimization," in *Proc. Int. Conf. Learn. Represent.*, 2019, pp. 1–24.

[26] T.-Y. Yang, J. Rosca, K. Narasimhan, and P. J. Ramadge, "Accelerating safe reinforcement learning with constraint-mismatched baseline policies," in *Proc. Mach. Learn. Res.*, 2021, pp. 11795–11807.

[27] C. Tessler, D. J. Mankowitz, and S. Mannor, "Reward constrained policy optimization," 2018, *arXiv:1805.11074*.

[28] M. Cai, H. Peng, Z. Li, and Z. Kan, "Learning-based probabilistic LTL motion planning with environment and motion uncertainties," *IEEE Trans. Autom. Control*, vol. 66, no. 5, pp. 2386–2392, May 2021.

[29] R. Toro Icarte, T. Q. Klassen, R. Valenzano, and S. A. McIlraith, "Teaching multiple tasks to an RL agent using LTL," in *Proc. Int. Joint Conf. Auton. Agents Multiagent Syst.*, 2018, pp. 452–461.

[30] A. K. Bozkurt, Y. Wang, M. M. Zavlanos, and M. Pajic, "Control synthesis from linear temporal logic specifications using model-free reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 10349–10355.

[31] P. Vaezipoor, A. C. Li, R. A. T. Icarte, and S. A. Mcilraith, "LTL2Action: Generalizing LTL instructions for multi-task RL," in *Proc. Int. Conf. Machin. Learn.*, 2021, pp. 10497–10508.

[32] X. Li, Z. Serlin, G. Yang, and C. Belta, "A formal methods approach to interpretable reinforcement learning for robotic planning," *Sci. Robot.*, vol. 4, no. 37, Dec. 2019, Art. no. eaay6276.

[33] M. Cai and C.-I. Vasile, "Safety-critical learning of robot control with temporal logic specifications," 2021, *arXiv:2109.02791*.

[34] C. Baier and J.-P. Katoen, *Principles of Model Checking*. Cambridge, MA, USA: MIT Press, 2008.

[35] P. Gastin and D. Oddoux, "Fast LTL to Bùchi automata translation," in *Proc. Int. Conf. Comput. Aided Verif.* Cham, Switzerland: Springer, 2001, pp. 53–65.

[36] M. Forgione and D. Piga, "Continuous-time system identification with neural networks: Model structures and fitting criteria," *Eur. J. Control*, vol. 59, pp. 69–81, May 2021.

[37] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1587–1596.

[38] D. Yu, H. Ma, S. Li, and J. Chen, "Reachability constrained reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2022, pp. 25636–25655.