2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

Safe Explicable Policy Search

Akkamahadevi Hanni, Jonathan Montaño, and Yu Zhang

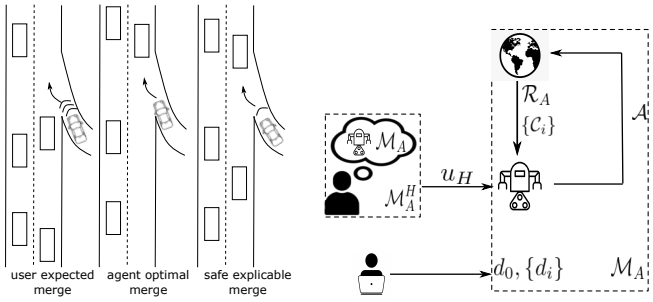
***Abstract*— When users work with AI agents, they form conscious or subconscious expectations of them. Meeting user expectations is crucial for such agents to engage in successful interactions and teaming. However, users may form expectations of an agent that differ from the agent's planned behaviors. These differences lead to the consideration of two separate decision models in the planning process to generate explicable behaviors. However, little has been done to incorporate safety considerations, especially in a learning setting. We present *Safe Explicable Policy Search (SEPS)*, which aims to provide a learning approach to explicable behavior generation while minimizing the safety risk, both during and after learning. We formulate SEPS as a constrained optimization problem where the agent aims to maximize an explicability score subject to constraints on safety and a suboptimality criterion based on the agent's model. SEPS innovatively combines the capabilities of Constrained Policy Optimization and Explicable Policy Search to introduce the capability of generating safe explicable behaviors to domains with continuous state and action spaces, which is critical for robotic applications. We evaluate SEPS in safety-gym environments and with a physical robot experiment to show its efficacy and relevance in human-AI teaming.**

I. Introduction

Recent developments in Artificial Intelligence (AI) have led to AI systems being widely deployed for user access. Users are increasingly involved in close interaction with these systems, forming a team-like relationship. In such cases, users form expectations of these "teammates". For successful teaming interactions, it is important that user expectations are met. Otherwise, it may lead to user confusion, poor teaming performance, and loss of trust [1].

More specifically, users may not always be experts in AI systems, which may lead to them forming expectations that are different from the system's planned behaviors. These differences may be due to, for instance, different understandings of the task objective and/or domain dynamics during decision-making. Explicable planning [2], [3] addresses the planning challenges where the AI agent aims to generate more expected plans or policies under such model differences. However, when the agent focuses on meeting user expectations, it may lead to safety and/or efficiency issues.

Consider a motivating example, shown in Fig. 1(a), where an autonomous driving car is to merge with highway traffic. The autonomous driving agent seeks to conserve energy so that it avoids hard acceleration and deceleration. However, such an objective can directly conflict with the user's expectation of minimizing the travel time. More specifically, while the optimal behavior of the agent is to slow down and wait for a large opening between the front and rear cars, the expected behavior is to merge as fast as it can, which can lead to safety risks. We can incorporate a safety criterion to maintain a minimum distance from the other cars to guarantee safety. However, an agent seeking explicable behaviors subject to the safety constraint may still opt for hard acceleration and deceleration with relatively small openings. A safe explicable behavior may involve alternating between soft deceleration and acceleration for an opportunity with a larger opening. The proposed Safe Explicable Policy Search (SEPS) aims to best match the user's expectation within the agent's safety and quality limits, thus avoiding unsafe and inefficient behaviors during and after learning.



(a) Motivating example

(b) Problem setting of SEP and the operation flow of SEPS

Fig. 1. (a) The car merging scenario showing the user expected behavior (left), the optimal behavior (middle), and the safe explicable behavior (right), (b) The agent learns a surrogate reward function u_H that captures the user's expectations that are generated from her belief of the agent's model $\mathcal{M}_A$, represented as $\mathcal{M}_A^H$. In a reinforcement learning setting, when the agent takes an action, it receives the agent's task reward $\mathcal{R}_A$, the safety costs associated with a set of cost functions, $\mathcal{C}_i$, and the assumed surrogate reward u_H as user feedback. The agent approximates the expected returns for these costs and rewards while applying the quality d_0 and safety limits $\{d_i\}$ to ensure that its policy is safe and efficient.

A previous study [4], introduced the problem of Safe Explicable Planning (SEP) that aims to find user expected policies that are safe by considering a safety bound (provided by an expert) in the agent's decision making (Fig. 1(b)). This work is limited to discrete states and actions, and to a planning setting where the models of the agent and the user (for generating expectations of the agent) are known. In reality, it is desirable for such methods to work in environments with continuous states and actions, and, more importantly, when the user's model is not directly available. Our work bridges the gap by proposing a method for finding safe and explicable policies in continuous stochastic environments in a reinforcement learning (RL) setting, which is especially valuable for robotic applications. It is worth noting that, in a

Akkamahadevi Hanni and Yu Zhang are with the School of Computing and Augmented Intelligence, Arizona State University, Tempe, USA ahanni@asu.edu and Yu.Zhang.442@asu.edu.

Jonathan Montaño is with the School of Mathematical and Statistical Sciences, Arizona State University, Tempe, USA montano@asu.edu.

planning setting, the solution is a final plan or policy that is safe. However, in a learning setting, the solution may require the agent to be safe even during the learning process.

Both safe behavior and explicable behavior generation, considered independently, can be addressed by the existing methods. For example, Constrained Markov Decision Process (CMDP) [5] and its solution techniques [6], [7], [8], etc., are effective in optimizing a task objective subject to safety constraints. The challenge in our problem is that the objective is defined over two different models with potentially different objectives and domain dynamics. This limits directly applying CMDP solutions. Similarly, a previous study on Explicable Policy Search (EPS) [3] proposes learning a surrogate reward function from user feedback on the agent's behavior to inform learning. It is more practical than learning both the reward and domain dynamics and hence can be applied when the user's model is unknown. However, EPS depends on the specification of a reconciliation parameter, and a weighting parameter for a few entropy terms, to combine an explicability score with the agent's objective. It requires careful fine-tuning via trial and error due to the difficulty in relating these hyper-parameters to the importance of the objectives reconciled since they are at different scales (i.e., reward vs. entropy). These issues in existing methods limit their applicability to the generation of safe and explicable behaviors.

To take advantage of both sides for safe explicable behavior generation, we first formulate our objective based on EPS since it allows us to convert the consideration of two models to the consideration of two linearly combined objectives. This is particularly beneficial since it enables us to focus policy search on a single model, reducing the new objective to a CMDP. Meanwhile, to relax the requirement of the hyper-parameters, we adapt and reformulate the objective by breaking it into an objective and a constraint, where the constraint can be dissolved into the CMDP formulation. We consider the popular Constrained Policy Optimization (CPO) [6] approach to solve the reduced CMDP problem. Our work considers a novel integration of EPS and CPO, while avoiding their limitations, to introduce safe explicable policy search. The remaining challenge lies in the derivation of an analytic solution for the resulting constrained optimization problem, for which we provide a solution for the most common case with two constraints. Furthermore, we evaluate SEPS in different safety-gym environments and with a physical robot experiment. The results show that our approach is effective in searching for an explicable policy within the safety limits. Our approach also demonstrates its applicability to real-life scenarios in the robot experiment.

II. Related Work

Recent advances in AI have made AI agents increasingly accessible to users, often requiring these agents to collaborate or interact directly with humans. This has motivated significant progress in Explainable AI (XAI), which aims to develop AI systems whose behavior can be understood by users [9], [10]. Our work is related to research focused on generating user-understandable agent behavior, studied under various terminology such as legibility [11], predictability [12], transparency [13], and explicability [2]. A comprehensive survey of these approaches can be found in [14].

Within the XAI literature, our work is closely related to explicable planning [2], [3], [15], [16], which seeks to generate agent behavior aligned with user expectations while balancing the costs incurred by the agent. Existing methods propose different metrics to quantify similarity to user expectations, but do not account for safety considerations. There exist methods that incorporate human feedback during the learning process, such as [17], [18], [19], which enable agents to learn user-preferred behaviors alongside an expert-designed RL objective. However, these methods assume that the user's belief of the domain dynamics matches that of the agent. When this assumption is invalid, it can lead to undesirable behaviors [20]. Furthermore, these methods also do not address safety issues.

Safe Explicable Planning (SEP) extends explicable planning to support the specification of a safety bound [4]. However, it requires searching through the policy space, which is intractable in large, continuous MDPs. Furthermore, SEP is limited to a planning setting in which the decision models are given. In reality, such models may be difficult to provide a priori. The models can be learned individually, for example, [21] learned the domain dynamics and [22] learned the reward function, or jointly [23], [20]. However, for real-world problems, this can be challenging due to its complexity and high dimensionality. In this work, we consider the problem of finding a safe, explicable policy in a reinforcement learning setting. A prior work [3] learns sufficient information about the user's model from user feedback, through a surrogate reward model. In this work, we assume that the surrogate reward model is given. This allows us to focus on developing the methodology for SEPS. Furthermore, unlike SEP, in our formulation, we consider the general case of continuous state and action spaces.

In our work, we define the agent's decision model as a Constrained Markov Decision Process (CMDP) [5], which allows us to specify the costs associated with safety separately from the agent's task objective. CMDP is a popularly used formulation in Safe RL (refer to [24] for a review of Safe RL methods). Another common way to handle safety in RL is to adopt a risk-aware formulation that includes a risk measure in the optimization criterion, which may take various forms, such as the exponential utility function [25], [26], a weighted sum of the return and risk functions [27], [28], and probabilistic conditions [29], [30]. In this work, we adopt the constrained formulation due to the ease of specification and its safety guarantees. Several techniques have been proposed to solve a CMDP [6], [31], [32], [33]. However, these methods cannot be directly applied to solve SEPS due to its problem setting (refer to Fig. 1(b)), where the optimization is under two different MDPs, specifically different transition models in MDPs. We consider the Constrained Policy Optimization (CPO) [6] approach to solve CMDP due to its convergence and safety guarantees. Other

recent studies [34], [35], [36] that build on top of CPO can be easily applied to solve SEPS.

III. Preliminaries

Our formulation is based on Markov Decision Processes (MDP). An MDP is represented by a tuple $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \rho, \gamma \rangle$, where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of actions, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto [0, 1]$ is the transition probability function, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto \mathbb{R}$ is the reward function, $\rho : \Delta(\mathcal{S})$ is the initial state distribution, and γ is the discount factor. A stationary policy $\pi : \mathcal{S} \mapsto \Delta(\mathcal{A})$ is a map from states to probability distribution over actions, with $\pi(a|s)$ denoting the probability of selecting action a in state s. We denote the set of all stationary policies by Π. The performance (expected discounted return) of a policy w.r.t. a measure $\mathcal{R}$ under the transition dynamics $\mathcal{T}$ is typically given by $J_{\mathcal{R},\mathcal{T}}(\pi) \doteq E_{\tau\sim\langle\pi,\mathcal{T}\rangle}[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}(s_t, a_t, s_{t+1})]$, where $\tau\sim\langle\pi,\mathcal{T}\rangle$ indicates that the distribution over trajectories depends on π and $\mathcal{T}$ i.e., $s_0\sim\rho, a_t\sim\pi(.|s_t), s_{t+1}\sim\mathcal{T}(.|s_t, a_t)$. The probability of realizing a trajectory τ with policy π is $p(\tau)=\rho(s_0)\prod_t \mathcal{T}(s_{t+1}|s_t, a_t)\pi(a_t|s_t)$. Reinforcement learning algorithms (refer to [37]) seek to learn a policy that maximize the performance, i.e., $\pi^* = \arg\max_{\pi\in\Pi} J_{\mathcal{R},\mathcal{T}}(\pi)$.

CMDP and Constrained Policy Optimization (CPO): A CMDP is an MDP augmented with constraints that restrict the permissible policies for the MDP. Specifically, $\mathcal{M}$ is augmented by auxiliary cost functions $\{\mathcal{C}_i\}_{i=1:k}$ and limits $\{d_i\}_{i=1:k}$ where each $\mathcal{C}_i : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto \mathbb{R}$ is a function, same as $\mathcal{R}$. The performance of a policy w.r.t. $\mathcal{C}_i$ is $J_{\mathcal{C}_i,\mathcal{T}}(\pi) = E_{\tau\sim\langle\pi,\mathcal{T}\rangle}[\sum_{t=0}^{\infty} \gamma^t \mathcal{C}_i(s_t, a_t, s_{t+1})]$. Then, the set of all feasible stationary policies is denoted by $\Pi_{\mathcal{C}} \doteq \{\pi\in\Pi : \forall i J_{\mathcal{C}_i,\mathcal{T}}(\pi) \leq d_i\}$. The objective of a CMDP is to learn a feasible policy that maximizes the performance $\pi^* = \arg\max_{\pi\in\Pi_{\mathcal{C}}} J_{\mathcal{R},\mathcal{T}}(\pi)$.

CPO [6] is a popular approach to solving CMDP. It applies trust region policy optimization (TRPO) [38] to policy search updates in a CMDP. The policy update rule is

$$\begin{aligned} \pi^{n+1} &= \arg\max_{\pi\in\Pi_\theta} \mathbb{E}_{\pi^n,\mathcal{T}}[A^{\pi^n}_{\mathcal{R}}(s,a)] \\ s.t. J_{\mathcal{C}_i,\mathcal{T}}(\pi_n) &+ \frac{1}{1-\gamma}\mathbb{E}_{\pi^n,\mathcal{T}}[A^{\pi^n}_{\mathcal{C}_i}(s,a)] \leq d_i \quad i = 1, ..., k \\ &\bar{\mathcal{D}}_{KL}(\pi||\pi_n) \leq \delta \end{aligned} \tag{1}$$

where, $A^{\pi}_{\cdot}(s,a) \doteq Q^{\pi}_{\cdot}(s,a) - V^{\pi}_{\cdot}(s)$ is the advantage function given by the on-policy value function $V^{\pi}_{\cdot}(s)$ and the on-policy action-value function $Q^{\pi}_{\cdot}(s,a)$, $\Pi_\theta \subset \Pi$ is a set of parameterized policies with parameters θ, $\bar{\mathcal{D}}_{KL}(\pi||\pi_n) = \mathbb{E}_{s\sim\pi^n}[\mathcal{D}_{KL}(\pi||\pi_n)[s]]$ is a measure of KL Divergence between the current policy π_n and a potential new policy π, and $\delta>0$ is the step size. The set of policies that satisfy the third constraint is called the trust region, i.e. $\{\pi_\theta \in \Pi_\theta | \bar{\mathcal{D}}_{KL}(\pi||\pi_n) \leq \delta\}$. For performance guarantees and constraint satisfaction guarantees, see [38] and [6], respectively.

For continuous MDPs with high dimensional spaces, solving Eq. (1) exactly is intractable and requires an approximation. The CPO method proposes the following approximation

$$\begin{aligned} \theta_{n+1} = &\arg\max_\theta g^T(\theta - \theta_n) \\ &s.t.\ c_i + b_i^T(\theta - \theta_n) \leq 0 \quad i = 1, ..., k \\ &\frac{1}{2}(\theta - \theta_n)^T H(\theta - \theta_n) \leq \delta \end{aligned} \tag{2}$$

where, g is the gradient of the objective (explicability measure), b_i is the gradient of the constraint i, $c_i \doteq J_{\mathcal{C}_i,\mathcal{T}}(\pi_n) - d_i$, and H is the Hessian of the KL Divergence.

Explicable Policy Search (EPS): In EPS [3], two MDP models, the agent's model M_A and the user's belief of the agent's model M_A^H, are considered at play. The two MDPs share the same $\mathcal{S}$, $\mathcal{A}$, ρ, and γ but may differ in their domain dynamics ($\mathcal{T}_A$, $\mathcal{T}_A^H$) and their reward functions ($\mathcal{R}_A$, $\mathcal{R}_A^H$). In EPS, the aim is to search for a policy that maximizes a linear combination of two objectives, namely, the expected cumulative reward and a policy explicability score, weighted by a reconciliation factor (λ):

$$\pi^* = \arg\max_{\pi\in\Pi} J_{\mathcal{R}_A,\mathcal{T}_A}(\pi) + \lambda\, \mathcal{E}(\pi, M_A, \pi_A^H, M_A^H). \tag{3}$$

The explicability score $\mathcal{E}(.)$ is well motivated (refer to [3]) and is defined as the negative KL-divergence of the trajectory distributions in the agent's model and the human's model denoted by $p_A(\tau)$ and $p_A^H(\tau)$, respectively. Formally, $\mathcal{E}(.) = -\mathcal{D}_{KL}(p_A(\tau)||p_A^H(\tau)) = -\mathbb{E}_{p_A}[\sum_t \log \mathcal{T}_A(s_{t+1}|s_t, a_t) + \log \pi_A(a_t|s_t) - \log \mathcal{T}_A^H(s_{t+1}|s_t, a_t) - \log \pi_A^H(a_t|s_t)] + C$. Essentially, the agent that maximizes $\mathcal{E}(.)$ would learn to replicate (as closely as possible) the human's expected behavior of the agent.

Estimating the parameters in $p_A^H(\tau)$ (i.e. $\mathcal{T}_A^H$ and π_A^H), individually, is expensive. Instead, the authors propose to learn a surrogate reward function u_H via a preference-based learning framework [39], [40] by presenting pairs of trajectories and asking human subjects which trajectory is expected. The goal of learning u_H is to obtain feedback on the user's assessment of *explicability* of the agent's behavior, which is evaluated under the user's belief (i.e., $\mathcal{M}_A^H$). The learned reward function u_H correlates with the distribution of human expected trajectories i.e., $u_H(s_t, a_t) = \log \mathcal{T}_A^H(s_{t+1}|s_t, a_t) + \log \pi_A^H(a_t|s_t) + C_1$. Assuming a parametrized agent policy π_θ, the surrogate reward and other log terms in the explicability score effectively reshape the reward function $\mathcal{R}_A$, and Eq. (3) is approximated as follows

$$\begin{aligned} \theta^* \doteq \arg\max_{\pi_\theta} \mathbb{E}_{p_A}\Big[\sum_t \gamma^t (\mathcal{R}_A(s_t, a_t) \\ + \lambda(u_H(s_t, a_t) + \mathcal{H}_{\mathcal{T}_A}(s_{t+1}|s_t, a_t) + \mathcal{H}_{\pi_\theta}(a_t|s_t)))\Big], \end{aligned} \tag{4}$$

where $\mathcal{H}_{\pi_\theta}$ and $\mathcal{H}_{\mathcal{T}_A}$ are the entropies of the agent's policy and domain dynamics (i.e., the true environment dynamics), respectively.

IV. Problem Formulation

In safe explicable policy search (SEPS), similar to EPS, the user's belief of the agent's model is modeled as an MDP i.e., $\mathcal{M}_A^H = \langle \mathcal{S}, \mathcal{A}, \mathcal{T}_A^H, \mathcal{R}_A^H, \rho, \gamma \rangle$; furthermore, the agent's decision model is modeled as a CMDP i.e., $\mathcal{M}_A =$

$\langle \mathcal{S}, \mathcal{A}, \mathcal{T}_A, \mathcal{R}_A, \{\mathcal{C}_i\}_{i=1:k}, \{d_i\}_{i=0:k}, \rho, \gamma \rangle$. To focus on policy search, we assume access to the surrogate reward function u_H in EPS. Note that u_H can be learned from user feedback as described in the previous section, assuming the user is noisily rational. While learning u_H, one can also weigh the policy entropy term more to achieve a similar effect as in Maximum Entropy IRL [41] to increase its robustness.

A. Rationale

Explicability and safety are orthogonal issues: an explicable policy may place too much focus on user understandability, resulting in unsafe behaviors when the user is simply unaware of the safety risks; on the other hand, an explicable policy may align well with safety if the user understands the risks but is risk-averse, resulting in overly conservative behaviors that negatively impact task efficiency. Hence, it is imperative for explicable behavior generation to be guarded by both safety constraints and efficiency criteria. Hence, it is imperative for explicable behavior generation to be guarded by safety constraints. However, we will explain next that a few alternatives would not work well.

1) Multi-objective optimization with linear combination: One idea is to optimize an objective that balances safety costs with rewards and explicability. The corresponding optimization problem can then be defined as maximizing $\mathcal{R}' = \mathcal{R} - \omega_1 \mathcal{C} + \omega_2 \mathcal{E}$, where ω_1 and ω_2 are the respective scaling parameters.

2) Constrained EPS: Alternatively, we can consider EPS subject to safety constraints as follows:

$$\pi^* = \arg\max_{\pi \in \Pi_d} J_{\mathcal{R}_A, \mathcal{T}_A}(\pi) + \lambda\, \mathcal{E}(\pi, \mathcal{M}_A, \pi_A^H, \mathcal{M}_A^H)$$
$$s.t.\ J_{\mathcal{C}_i, \mathcal{T}_A}(\pi) \leq d_i \quad i = 1, ..., k. \tag{5}$$

The above two alternative formulations considered share two related limitations: 1) determining an appropriate reconciliation/weight factors can be difficult. Designing a single reward that simultaneously drives task completion, meets user expectations, and discourages unsafe behavior is challenging. It requires carefully scaling each component to provide a meaningful learning signal at each step, preventing issues like reward hacking. For example, in CPO [6], the results of optimizing $\mathcal{R} - \omega_1 \mathcal{C}$ demonstrate a high sensitivity to the cost coefficient ω_1, highlighting the difficulty of crafting an effective overloaded reward function. Prior research [29], also emphasizes the importance of separating the task objective from safety constraints, as in CMDPs [5], [6], [7], [31], to enable the agent to learn meaningful behavior. 2) since the task objective is maximized in combination with explicability, for $\lambda > 0$, there is no guarantee of task performance or even achievement, since the agent may be able to increase the explicability score to "compensate" for the reduction in task performance.

B. Safe Explicable Policy Search (SEPS)

In SEPS, we observe that the two components of EPS can be separated to remove the reconciliation factor and let the agent reward be absorbed into the safe constraints in CMDP, with the limit d_0. Even though this would introduce a constraint parameter, it has an intuitive appeal: the suboptimality level of the behavior, which can be used to ensure the task performance/achievement as desired. Hence, such a parameter can be set more straightforwardly by the engineer than the semantically confusing reconciliation factor.

A remaining challenge is the additional entropy terms in $\mathcal{E}$. Since the surrogate reward u_H in EPS is learned to reflect $\mathcal{H}_{\mathcal{T}_A^H}$ and $\mathcal{H}_{\pi_A^H}$, combining it with the remaining entropy terms ($\mathcal{H}_{\mathcal{T}_A}$ and $\mathcal{H}_{\pi_\theta}$) requires additional scaling parameters. In SEPS, we simplify by removing these terms. This means that instead of maximizing the negative KL divergence between the trajectory distributions, we maximize the probability of the most expected trajectory that the agent can generate. This leads to an approximation of the explicability objective that encourages a narrower distribution than the original objective.

Definition 1: Safe Explicable Policy Search (SEPS) is the problem of searching for a policy that maximizes the probability of the user's expected trajectory, captured by u_H, subject to the constraints on the task objective $\mathcal{R}_A$ and the costs $\mathcal{C}_i$ associated with safety in $\mathcal{M}_A$, or formally,

$$\pi^* \doteq \arg\max_{\pi \in \Pi_d} J_{u_H, \mathcal{T}_A}(\pi)$$
$$s.t.\ C_0 :\ J_{\mathcal{R}_A, \mathcal{T}_A}(\pi) \geq d_0$$
$$C_i :\ J_{\mathcal{C}_i, \mathcal{T}_A}(\pi) \leq d_i \quad i = 1, ..., k. \tag{6}$$

The main objective here is to maximize u_H under the true domain dynamics $\mathcal{T}_A$, which in turn maximizes the approximate explicability score. Although the SEPS problem in Eq. (6) appears similar to the SEP [4] problem below

$$\pi^* = \arg\max_{\pi \in \Pi_d} J_{\mathcal{R}_A^H, \mathcal{T}_A^H}(\pi)$$
$$s.t.\ J_{\mathcal{R}_A, \mathcal{T}_A}(\pi) \geq d_0, \tag{7}$$

there are a few critical differences: 1) in SEPS, the constraint on the agent's reward or task performance, whereas it was associated with the safety constraint in [4]. It requires the safety constraint to be correlated with task performance, which is not always feasible; 2) in [4], $\mathcal{M}_A^H$ is assumed to be given whereas it was replaced here by the surrogate reward function that can be learned, which is evaluated under the true domain dynamics. Such a subtle but important difference allows us to apply learning methods to solve the problem instead of replying on planning; 3) formulating SEPS as a CMDP allows us to leverage CMDP solutions to handle multiple constraints in continuous domains, instead of as a complex search problem under two different models in [4].

V. Search Method

When the MDP models are known, as in Eq. (7), SEP [4] proposes searching through the policy space while evaluating policies in the agent's model for feasibility and in the user's model for explicablity. The SEP algorithms iterate over all unique deterministic policies (formed by iterating over all states and actions) to find an optimal safe, explicable policy, which is impractical in continuous MDPs. The SEPS problem in Eq. (6) reduces to a CMDP, allowing us to use its policy search techniques.

We adopt CPO [6] to solve SEPS as it ensures safe behavior even during the learning process. Applying the CPO update rule to SEPS in Eq. (6) will yield an expression similar to that in Eq. (1). An approximation to that update rule (as in CPO, Eq. (2)) is given by the primal

$$\begin{aligned} \theta_{n+1} = \arg\max_\theta \, & g^T(\theta - \theta_n) \\ s.t. \; & c_0 - b_0^T(\theta - \theta_n) \leq 0 \\ & c_1 + b_1^T(\theta - \theta_n) \leq 0 \\ & \frac{1}{2}(\theta - \theta_n)^T H(\theta - \theta_n) \leq \delta \end{aligned} \tag{8}$$

where, b_0 is the gradient of the constraint on agent's task objective with $c_0 \doteq -J_{\mathcal{R}_A,\mathcal{T}_A}(\pi_n) + d_0$, b_1 and c_1 are as defined in Eq. (2). The optimization problem above is convex. It is worth noting that for policies with high-dimensional parameter spaces like neural networks, the primal problem (even with few constraints) is impractical to solve using a convex program solver due to the computation complexity of the Hessian matrix, which motivates the need for an efficient analytical solution. However, the CPO algorithm is limited to solving a CMDP with a single constraint. The SEPS problem in Eq. (8) includes at least two constraints. Deriving feasible bounds for the solution becomes challenging with an increase in the number of constraints, as it requires analyzing the pairwise intersection of feasible regions of constraints.

Feasible case: We consider the feasible case when the current policy π_{θ_n} satisfies all constraints. In this case, it is easier to solve the dual of Eq. (8), given by

$$\max_{\substack{\lambda \geq 0 \\ \nu_0 \geq 0 \\ \nu_1 \geq 0}} g^T x + \lambda(\frac{1}{2}x^T H x - \delta) + \nu_0(b_0^T x + c_0) + \nu_1(b_1^T x + c_1) \tag{9}$$

where λ, ν_0, and ν_1 are Lagrangian variables. The solution to the feasible case is given by Theorem 1 in Appendix A of the extended version of this paper [42]. To determine the optimal values of the Lagrangian variables, we need to consider the intersection of the trust region with the two linear constraints. Specifically, the optimal value of λ depends on the following four cases, viz., 1) both linear constraints are active ($\nu_0 > 0$ and $\nu_1 > 0$), 2) only the first linear constraint is active ($\nu_0 > 0$ and $\nu_1 = 0$): which means that the trust region lies well within the feasible region of the second constraint and can be safely ignored, 3) only the second linear constraint is active ($\nu_0 = 0$ and $\nu_1 > 0$), and 4) both linear constraints are inactive ($\nu_0 = 0$ and $\nu_1 = 0$): which means that the optimization depends solely on the objective ($g^T x$). In brief, if λ^*, ν_0^*, and ν_1^* are optimal solutions to the dual above, then, the update rule for the feasible case is

$$\theta_{n+1}^* = \theta_n + \frac{1}{\lambda^*} H^{-1} \left(g + \nu_0^* b_0 - \nu_1^* b_1\right). \tag{10}$$

Infeasible case: We consider the infeasible case when the current policy π_{θ_n} violates one or both linear constraints. This could be the result of a bad step taken previously (occurs due to errors in approximation). The system can recover by updating the policy parameters to strictly reduce the constraint violation. In other words, the system must focus on satisfying the constraint(s) and ignore optimizing the objective ($g^T x$), temporarily. In this case, depending on the linear constraint that is violated in Eq. (8), the update rule is found by solving the primal below

$$\begin{aligned} \theta_{n+1} = \arg\min_\theta \; & c_m + b_m^T(\theta - \theta_n) \qquad \text{where, } m \in \{0, 1\} \\ s.t. \; & \frac{1}{2}(\theta - \theta_n)^T H(\theta - \theta_n) \leq \delta, \end{aligned} \tag{11}$$

and its corresponding dual is given by

$$\max_{\phi \geq 0} -\frac{b_m^T H^{-1} b_m}{2\phi} + c_m - \phi\delta. \tag{12}$$

where, ϕ is a Lagrangian variable. The solution to the feasible case is given by Theorem 2 in Appendix A of the extended version of this paper [42].

In brief, if ϕ^* is the solution to the dual, then the update rule for the infeasible case is

$$\theta_{n+1}^* = \theta_n - \frac{1}{\phi^*} H^{-1} b_m. \tag{13}$$

When multiple constraints are violated, we solve Eq. (11) for each constraint that is violated and linearly combine the solutions in Eq. (13). This adjusts the policy towards the feasible region w.r.t. all constraints in a single step. Alternatively, one could update the policy with respect to a single constraint at a time.

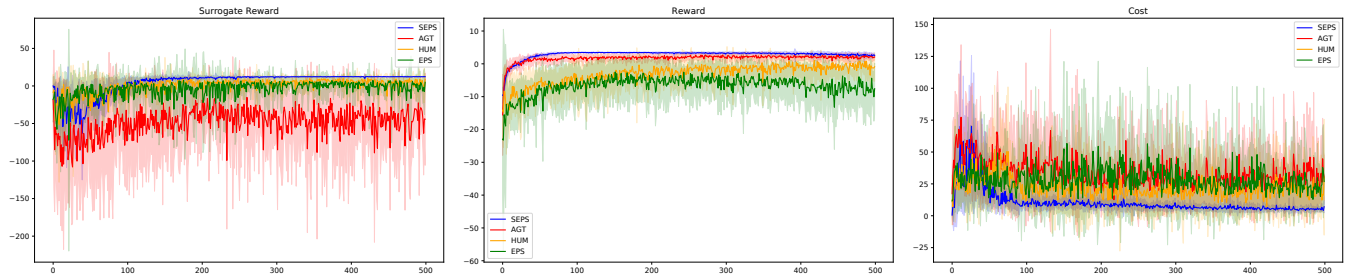
VI. Evaluation

We evaluated our method in continuous Safety-Gym environments [43] and through a physical robot experiment, designing scenarios that included explicit risk elements. Across all evaluations, we assumed a learned surrogate reward function that captures sufficient information about user expectations in $\mathcal{M}_A^H$. Our goal is to 1) validate the ability of our method to find safe, efficient, and explicable policies that meet user expectations, 2) analyze the behaviors generated by our method relative to state-of-the-art and baseline approaches, and 3) demonstrate the applicability of SEPS in real-world scenarios.

A. Safety

In our first evaluation, our goal is to answer 'What are the benefits of safety constraints in explicable planning?' and 'Can SEPS identify an explicable policy under safety constraints?'. We use the Safety-Gym PointGoal task (shown in Fig. 3(a)) where we assume user expectations conflict with the agent's safety requirements. In $\mathcal{M}_A$, the agent must reach the goal (green) without entering hazardous regions (dark blue). The reward $\mathcal{R}_A$ is positive for progress toward the goal and negative for moving away, and $\mathcal{C}_1$ penalizes entry into hazardous regions. In contrast, according to user expectations (u_H), the agent can traverse hazardous regions as the user is unaware of their risks but must avoid the light blue boxes as the user regards them fragile.

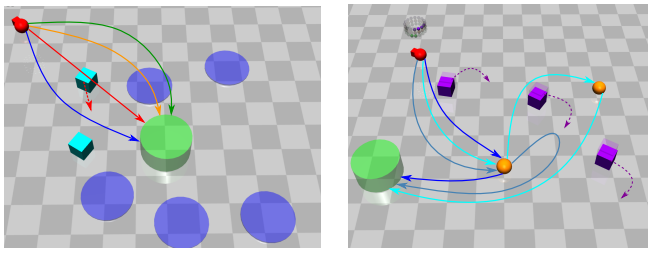
We compared SEPS with three baselines: AGT (optimal agent behavior), HUM (human-expected behavior), and EPS. AGT optimizes $\mathcal{R}_A$, HUM optimizes u_H, and EPS optimizes $\mathcal{R}_A + \lambda u_H$ (see the Preliminary Section). All baselines were



(a) Returns under u_H (b) Returns under $\mathcal{R}_A$ (c) Returns under $\mathcal{C}_1$

Fig. 2. Average performance over eight runs for SEPS and baselines in PointGoal; the x-axis is training epoch.

trained using PPO [44]. Since these methods do not incorporate safety constraints, the comparison highlights the need for safety in explicable planning. We evaluated the resulting behaviors and returns under u_H, $\mathcal{R}_A$, and $\mathcal{C}_1$.



(a) PointGoal (b) PointButton

Fig. 3. Trajectories of different methods in the simulation domains; (a) SEPS - blue, AGT - red, HUM - orange, EPS - green, (b) SEPS - blue, SEPS$-C_0$ - light blue, SEPS$+\mathcal{R}_A-C_0$ - medium blue.

Results and Discussion. Figure 3(a) illustrates the behaviors generated. AGT reached the goal via the shortest path by moving one of the boxes, contrary to user expectations. HUM and EPS avoided the boxes, but entered hazardous regions due to a lack of safety constraints. In contrast, SEPS efficiently reached the goal while avoiding both boxes and hazardous regions.

These behaviors aligned with the quantitative results in Fig. 2. The primary finding is that all baselines violated safety constraints, as reflected by returns in $\mathcal{C}_1$ (Fig. 2(c)), where values farther from zero indicate greater violations. AGT, HUM, and EPS showed higher violations as their objectives ignore safety specifications. AGT also performed poorly on user expectations (Fig. 2(a)), prioritizing the shortest path even if it involves moving boxes. HUM achieved high returns in $\mathcal{R}_A$ (Fig. 2(b)) as it does not deviate much from the goal. EPS, despite using the best reconciliation factor ($\lambda = 2$), showed instability in learning $\mathcal{R}_A$ (Fig. 2(b)) due to its linear combination objective. SEPS outperformed the baselines on the agent's task (Fig. 2(a)) and user expectations (Fig. 2(b)) while satisfying safety requirements (Fig. 2(c)).

B. Ablation Study

In our second evaluation, our goal is to answer 'Can safe explicable policy search be formulated differently?' or, equivalently, 'What is the benefit of the SEPS formulation?'. We used the Safety-Gym PointButton task (Fig. 3(b)), where the user's model aligns with the agent's safety requirements but differs from its task objective.

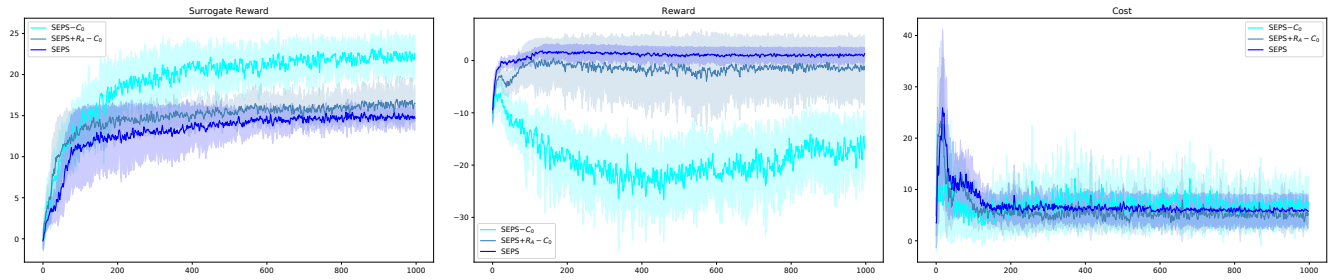
In $\mathcal{M}_A$, the agent must reach the goal (green) without colliding with gremlins (purple) that move in fixed patterns, and may optionally press buttons (orange). $\mathcal{R}_A$ assigns positive values for reaching the goal and pressing buttons and negative values for moving away from the goal, while $\mathcal{C}_1$ penalizes collisions with gremlins. According to user expectations (u_H), the agent must press both buttons before reaching the goal. Thus, u_H mirrors $\mathcal{R}_A$ except that deviations from the goal do not incur a negative value.

We compared the formulation of SEPS with two ablation baselines: SEPS$-C_0$, which removes the constraint C_0 (refer to Eq. (6)), and SEPS$+\mathcal{R}_A-C_0$, which linearly combines the SEPS objective with $\mathcal{R}_A$ while omitting C_0. Both baselines were trained using CPO [6], as they involve a single constraint. The comparison with these baselines also represents a direct comparison with the most relevant safe-RL method (i.e., CPO). We also implemented a baseline that used CVPX [45], [46] in place of the analytical solution proposed in SEPS, which however was observed to be significantly slower and failed to produce solutions within a reasonable time limit. Consequently, we did not report the result here. We analyzed the generated behaviors and returns under u_H, $\mathcal{R}_A$, and $\mathcal{C}_1$.

Results and Discussion. Figure 3(b) shows the generated behaviors. All baselines avoided gremlins due to safety constraints. SEPS$-C_0$ optimized user expectations, pressing both buttons before reaching the goal. SEPS pressed only the closer button to avoid large penalties for deviations, thereby satisfying the suboptimality criterion. SEPS$+\mathcal{R}_A-C_0$ also pressed the closer button but exhibited random deviations (Fig. 3(b)) due to optimizing linearly combined objectives.

These behaviors align with the quantitative results in Fig. 4. SEPS$-C_0$ performed poorly under $\mathcal{R}_A$ as it focuses solely on u_H. SEPS$+\mathcal{R}_A-C_0$, despite optimizing both u_H and $\mathcal{R}_A$ (with λ=3), also underperformed in $\mathcal{R}_A$ because it lacked constraints to enforce efficiency. In contrast, SEPS enforces the suboptimality criterion and as a result it guarantees efficiency in the agent's task objective.

In SEPS, we set $d_0 = 0.0$ and $d_1 = 2.5$. These limits have intuitive semantic interpretations. For example, the



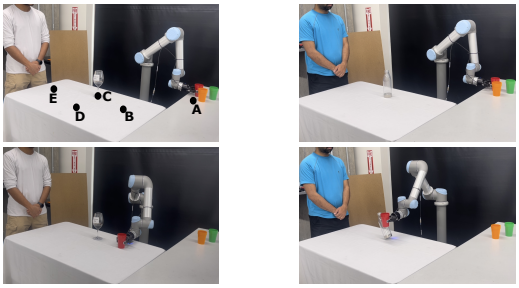
(a) Returns under u_H (b) Returns under $\mathcal{R}_A$ (c) Returns under $\mathcal{C}_1$

Fig. 4. Average performance over eight runs for SEPS and baselines in PointButton task over random seeds; the x-axis is training epoch.

agent can deviate from the goal provided that its expected return in $\mathcal{R}_A$ remains above 0.0, making them easier to specify than the hyperparameters in EPS. Unlike SEPS, all baselines impose only a single safety constraint, whereas handling multiple constraints can be more challenging due to potential conflicts. Even under such conditions, SEPS remained close to $\mathcal{M}_A$ in both $\mathcal{R}_A$ and $\mathcal{C}_1$, demonstrating that it can remain explicable while meeting both performance and safety requirements.

C. Physical Robot Experiment

In this experiment, we analyzed the behaviors generated when a physical UR5 robot assisted a user to set a dining table, to demonstrate that SEPS also applies to task planning domains. The experiment (Fig. 5) was designed in a discrete setting with preset object locations, and robot motions calibrated using the UR5 inverse kinematics library. The states were defined by the locations of objects (A, B, C, D, E, shown in Fig. 5(a)) and their status (steady or slipped).

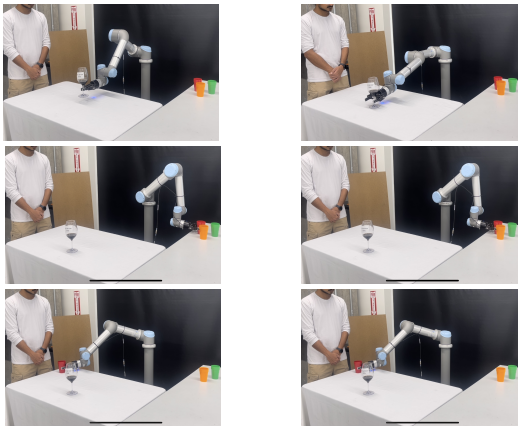


(a) τ_{AGT} (b) τ_{HUM}

Fig. 5. Robot optimal behavior and explicable behavior.

We assumed a mismatch between robot functionality and user expectations. In $\mathcal{M}_A$, the robot is tasked with passing cups to the user by placing them anywhere on the dining table. In contrast, the user expects the cups to be placed near him. However, it risks tipping the glass in front of the robot and incurs a high cost. We evaluated SEPS against AGT, HUM, and SEPS+$\mathcal{R}_A-C_0$.

Results and Discussion. In AGT, the robot picked up the cup and placed it on the table next to the glass (Fig. 5(a)). In HUM, the robot tipped the glass (replaced with an empty plastic bottle for safety), which in reality would have spilled liquid due to obstruction of the manipulator (Fig. 5(b)). In SEPS+$\mathcal{R}_A-C_0$ (Fig. 6(a)), the robot first picked up and placed the glass out of the way before placing the cup near the user, which met the user's expectations but was inefficient due to extra movements. In SEPS (Fig. 6(b)), the robot gently pushed the glass aside and then placed the cup near the user, achieving both explicability and efficiency. Thus, while both SEPS+$\mathcal{R}_A-C_0$ and SEPS produced explicable behaviors, SEPS was more efficient due to the suboptimality constraint.



(a) $\tau_{\text{SEPS}+\mathcal{R}_A \quad C_0}$ (b) τ_{SEPS}

Fig. 6. Safe explicable behaviors generated by SEPS+$\mathcal{R}_A-C_0$ and SEPS.

VII. Conclusion

In this paper, we introduced Safe Explicable Policy Search (SEPS) for learning explicable policies when both the environment model and user expectations are unknown, while ensuring safety and efficiency. SEPS adopts the surrogate reward model from EPS as the explicability metric, reducing optimization over two models to a single model and enabling formulation as a CMDP. This formulation simplifies safety specification through expected return thresholds, avoiding the need to tune weights or sensitivity parameters used in risk-aware RL methods. Unlike EPS, SEPS also reduces hyperparameter dependence by treating the agent's task objective as a constraint. The CPO method elegantly solves CMDP problems with convergence and safety guarantees, but is limited to a single constraint. To address this limitation, we derive an analytical solution for the common case with

two constraints. Experiments in Safety-Gym environments demonstrate that SEPS learns safe, explicable, and efficient policies, and we further demonstrate its applicability through a physical robot experiment.

SEPS has several limitations that motivate future work. Explicability is evaluated using a user model. We defer the human subjects study to future work. The complexity of the analytical solution increases with the number of constraints which remains to be studied. Additionally, enforcing both safety and suboptimality constraints may lead to infeasible solutions, requiring relaxation of the suboptimality bound, which is challenging. Finally, since CPO guarantees safety only in expectation over state visitations, violations may still occur under stricter criteria such as trajectory-level constraints. Addressing this limitation through techniques such as shielding [47] or alternative risk measures [48] remains an important direction for future research.

ACKNOWLEDGMENT

We thank the anonymous reviewers for their valuable comments and suggestions. We are grateful to Taha Shaheen for his help in recording the robotic experiments. This research is supported in part by the NSF grant 2047186.

REFERENCES

[1] D. Gunning and D. Aha, "Explainable artificial intelligence (xai)," *DARPA*, 2019.
[2] Y. Zhang, S. Sreedharan, A. Kulkarni, T. Chakraborti, H. Zhuo, and S. Kambhampati, "Plan explicability and predictability for robot task planning," in *ICRA*, 2017.
[3] Z. Gong and Y. Zhang, "Explicable policy search," in *NeurIPS*, 2022.
[4] A. Hanni, A. Boateng, and Y. Zhang, "Safe explicable planning," in *ICAPS*, 2024.
[5] E. Altman, *Constrained Markov decision processes*. Routledge, 2021.
[6] J. Achiam, D. Held, A. Tamar, and P. Abbeel, "Constrained policy optimization," in *ICML*, 2017.
[7] A. Wachi and Y. Sui, "Safe reinforcement learning in constrained Markov decision processes," in *ICML*, 2020.
[8] Y. Chow, O. Nachum, E. Duenez-Guzman, and M. Ghavamzadeh, "A lyapunov-based approach to safe reinforcement learning," in *NeurIPS*, 2018.
[9] M. Fox, D. Long, and D. Magazzeni, "Explainable planning," *arXiv*, 2017.
[10] T. Chakraborti, S. Sreedharan, and S. Kambhampati, "The emerging landscape of explainable ai planning and decision making," *arXiv*, 2020.
[11] A. D. Dragan and S. S. Srinivasa, "Generating legible motion," in *RSS*, 2013.
[12] A. D. Dragan, K. C. Lee, and S. S. Srinivasa, "Legibility and predictability of robot motion," in *HRI*, 2013.
[13] A. M. MacNally, N. Lipovetzky, M. Ramirez, and A. R. Pearce, "Action selection for transparent planning," in *AAMAS*, 2018.
[14] T. Chakraborti, A. Kulkarni, S. Sreedharan, D. E. Smith, and S. Kambhampati, "Explicability? legibility? predictability? transparency? privacy? security? the emerging landscape of interpretable agent behavior," in *ICAPS*, 2019.
[15] A. Kulkarni, T. Chakraborti, Y. Zha, S. G. Vadlamudi, Y. Zhang, and S. Kambhampati, "Explicable robot planning as minimizing distance from expected behavior," *arXiv*, 2016.
[16] A. Hanni and Y. Zhang, "Generating active explicable plans in human-robot teaming," in *IROS*, 2021.
[17] S. Griffith, K. Subramanian, J. Scholz, C. L. Isbell, and A. L. Thomaz, "Policy shaping: Integrating human feedback with reinforcement learning," *NeurIPS*, 2013.
[18] W. B. Knox and P. Stone, "Interactively shaping agents via human reinforcement: The tamer framework," in *International Conference on Knowledge Capture*, 2009, pp. 9–16.
[19] B. Knox and P. Stone, "Reinforcement learning from simultaneous human and mdp reward." in *AAMAS*, 2012.
[20] Z. Gong and Y. Zhang, "What is it you really want of me? generalized reward learning with biased beliefs about domain dynamics," in *AAAI*, vol. 34, no. 03, 2020, pp. 2485–2492.
[21] S. Reddy, A. Dragan, and S. Levine, "Where do you think you're going?: Inferring beliefs about dynamics from behavior," *NeurIPS*, vol. 31, 2018.
[22] C. Daniel, M. Viering, J. Metz, O. Kroemer, and J. Peters, "Active reward learning." in *RSS*, 2014.
[23] M. Herman, T. Gindele, J. Wagner, F. Schmitt, and W. Burgard, "Inverse reinforcement learning with simultaneous estimation of rewards and dynamics," in *AAAI*. PMLR, 2016.
[24] J. Garcıa and F. Fernández, "A comprehensive survey on safe reinforcement learning," *JMLR*, 2015.
[25] R. A. Howard and J. E. Matheson, "Risk-sensitive markov decision processes," *Management science*, 1972.
[26] S. D. Patek, "On terminating markov decision processes with a risk-averse objective function," *Automatica*, 2001.
[27] A. Gosavi, "Reinforcement learning for model building and variance-penalized control," in *Winter Simulation Conference*. IEEE, 2009, pp. 373–379.
[28] P. Geibel and F. Wysotzki, "Risk-sensitive reinforcement learning applied to control under constraints," *JAIR*, 2005.
[29] H. Kashima, "Risk-sensitive learning via minimization of empirical conditional value-at-risk," *IEICE - Trans. Inf. Syst.*, 2007.
[30] T. Morimura, M. Sugiyama, H. Kashima, H. Hachiya, and T. Tanaka, "Nonparametric return distribution approximation for reinforcement learning," in *ICML*, 2010, pp. 799–806.
[31] Y. Chow, O. Nachum, E. Duenez-Guzman, and M. Ghavamzadeh, "A lyapunov-based approach to safe reinforcement learning," *NeurIPS*, vol. 31, 2018.
[32] S. Huang, A. Abdolmaleki, G. Vezzani, P. Brakel, D. J. Mankowitz, M. Neunert, S. Bohez, Y. Tassa, N. Heess, M. Riedmiller *et al.*, "A constrained multi-objective reinforcement learning framework," in *CoRL*. PMLR, 2022.
[33] A. Stooke, J. Achiam, and P. Abbeel, "Responsive safety in reinforcement learning by pid lagrangian methods," in *ICML*, 2020.
[34] T.-Y. Yang, J. Rosca, K. Narasimhan, and P. J. Ramadge, "Projection-based constrained policy optimization," *arXiv*, 2020.
[35] C. Tessler, D. J. Mankowitz, and S. Mannor, "Reward constrained policy optimization," *arXiv*, 2018.
[36] Y. As, I. Usmanova, S. Curi, and A. Krause, "Constrained policy optimization via bayesian world models," *arXiv*, 2022.
[37] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. The MIT Press, 2018.
[38] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *ICML*, 2015.
[39] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei, "Deep reinforcement learning from human preferences," in *NeurIPS*, 2017.
[40] W. B. Knox, S. Hatgis-Kessell, S. Booth, S. Niekum, P. Stone, and A. Allievi, "Models of human preference for learning reward functions," *arXiv*, 2022.
[41] B. D. Ziebart, A. L. Maas, J. A. Bagnell, A. K. Dey *et al.*, "Maximum entropy inverse reinforcement learning." in *AAAI*, 2008.
[42] A. Hanni, J. Montaño, and Y. Zhang, "Safe explicable policy search," *arXiv*, 2025.
[43] J. Ji, B. Zhang, J. Zhou, X. Pan, W. Huang, R. Sun, Y. Geng, Y. Zhong, J. Dai, and Y. Yang, "Safety gymnasium: A unified safe reinforcement learning benchmark," in *NeurIPS*, 2023.
[44] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv*, 2017.
[45] S. Diamond and S. Boyd, "CVXPY: A Python-embedded modeling language for convex optimization," *JMLR*, 2016.
[46] A. Agrawal, R. Verschueren, S. Diamond, and S. Boyd, "A rewriting system for convex optimization problems," *Journal of Control and Decision*, 2018.
[47] M. Alshiekh, R. Bloem, R. Ehlers, B. Könighofer, S. Niekum, and U. Topcu, "Safe reinforcement learning via shielding," in *AAAI*, vol. 32, no. 1, 2018.
[48] H. Kashima, "Risk-sensitive learning via minimization of empirical conditional value-at-risk," *IEICE - Trans. Inf. Syst.*, 2007.