# Counterfactual Explanations for Continuous Action Reinforcement Learning

**Shuyang Dong**, **Shangtong Zhang** and **Lu Feng**
University of Virginia
{sd3mn, shangtong, lu.feng}@virginia.edu

## Abstract

Reinforcement Learning (RL) has shown great promise in domains like healthcare and robotics but often struggles with adoption due to its lack of interpretability. Counterfactual explanations, which address "what if" scenarios, provide a promising avenue for understanding RL decisions but remain underexplored for continuous action spaces. We propose a novel approach for generating counterfactual explanations in continuous action RL by computing alternative action sequences that improve outcomes while minimizing deviations from the original sequence. Our approach leverages a distance metric for continuous actions and accounts for constraints such as adhering to predefined policies in specific states. Evaluations in two RL domains, Diabetes Control and Lunar Lander, demonstrate the effectiveness, efficiency, and generalization of our approach, enabling more interpretable and trustworthy RL applications.

## 1 Introduction

Reinforcement Learning (RL) has shown significant potential in tackling complex decision-making tasks across diverse fields, including healthcare [Yu *et al.*, 2021] and robotics [Tang *et al.*, 2024]. However, its adoption in high-stakes applications is often hindered by a critical barrier: the lack of interpretability. Understanding an RL agent's decision-making process is essential for fostering trust and enhancing performance. While there is a growing body of work on explainable RL [Milani *et al.*, 2023], most methods focus on summarizing policies or explaining individual actions using natural language or saliency maps. Counterfactual explanations, which address "what if" questions to improve interpretability, remain underexplored in RL.

In contrast, most research on counterfactual explanations has focused on supervised learning, aiming to identify minimal changes to input features that yield a desired classifier output [Verma *et al.*, 2024]. These approaches often incorporate constraints such as *validity* (ensuring counterfactuals belong to the target class), *proximity* (minimizing deviations from the original instance), *actionability* (restricting changes to modifiable features), *sparsity* (minimizing the number of changes), *data manifold closeness* (ensuring realism by adhering to the training data distribution), and *causality* (preserving known causal relationships). However, as noted in [Gajcin and Dusparic, 2024], these methods cannot be directly applied to RL due to unique challenges, such as the temporal dependencies in decision-making and outcomes.

Existing research on counterfactual explanations for RL remains limited and often targets discrete action spaces. These approaches typically rely on heuristics [Amitai *et al.*, 2024] or structural causal models [Tsirtsis *et al.*, 2021] to generate counterfactual actions, or they focus narrowly on generating alternative state features [Olson *et al.*, 2021]. A detailed discussion of these prior works is provided in Section 2.

To address these gaps, we propose a novel approach for generating counterfactual explanations in RL tailored for continuous action spaces. Our method computes an alternative sequence of actions for a given observed trajectory, aiming to achieve better outcomes (i.e., higher cumulative rewards) while minimizing deviations from the original actions. This is accomplished using a distance metric designed for continuous action sequences. Additionally, we extend the problem formulation to account for scenarios where actions in specific constrained states must adhere to a predefined policy.

A compelling example is the application of RL for managing blood glucose levels in diabetes patients [Tejedor *et al.*, 2020]. Counterfactual explanations in this scenario can answer questions like: "What alternative insulin dosages could have resulted in better glycemic control?" These explanations must ensure minimal deviation from the original treatment plan while adhering to constraints, such as following a doctor's prescription (e.g., administering specified insulin doses when glucose levels are within defined ranges). This example highlights the practical relevance of our approach in high-stakes decision-making tasks.

Instead of generating counterfactuals for individual observed trajectories one at a time, our approach computes a counterfactual policy that simultaneously generates desired counterfactuals for a set of observed trajectories (e.g., a patient's historical glycemic control trajectories over a specific period). To enhance interpretability, the counterfactual policy is deterministic (e.g., ensuring that a specific insulin dosage, rather than a probabilistic range, is recommended at a time).

Our approach extends the Twin Delayed Deep Deterministic Policy Gradient (TD3) algorithm [Fujimoto *et al.*, 2018], introducing novel mechanisms for generating counterfactual trajectories in continuous action spaces and a sparse reward-shaping framework to balance reward gains with minimal action deviations.

Finally, we evaluated our approach in two RL domains: diabetes control using the FDA-approved UVA/PADOVA simulator [Man *et al.*, 2014] and Lunar Lander from OpenAI Gym [Brockman, 2016]. Experimental results demonstrate the effectiveness, efficiency, and generalization of our approach, paving the way for more interpretable and trustworthy RL applications in high-stakes settings.

## 2 Related Work

Existing work on counterfactual explanations has predominantly focused on supervised learning, with early efforts like [Wachter *et al.*, 2018] framing the problem as an optimization task to identify minimal changes to input features that result in a desired classifier output. Following this foundational work, extensive research has explored incorporating constraints to ensure counterfactuals are actionable and realistic, as surveyed in [Verma *et al.*, 2024; Guidotti, 2022]. [Chen *et al.*, 2024] employed RL-based methods to generate optimal counterfactual explanations for classifiers and subsequently distilled decision trees to explain the counterfactual generation process.

Research on counterfactual explanations for RL is relatively limited compared to supervised learning. [Olson *et al.*, 2021] investigated counterfactual state explanations for deep RL agents in visual environments like Atari, examining how minimal changes to game images could alter an agent's action. [Gajcin and Dusparic, 2024] extended counterfactual explanation concepts from classifiers to RL, defining various types to address questions such as "Had state features taken different values, action $a'$ (or policy $\pi'$) would be chosen instead of action $a$ (or policy $\pi$)" and "Had the agent followed a different goal (or preferred a different objective) in state $s$, action $a'$ (or policy $\pi'$) would be chosen instead of action $a$ (or policy $\pi$)." Conversely, our work focuses on computing counterfactual explanations to explore what alternative actions or policies the agent could take to achieve better outcomes.

Our definition of counterfactual explanations is inspired by [Tsirtsis *et al.*, 2021], which identifies alternative action sequences differing in at most $k$ actions to achieve better outcomes in finite-horizon MDPs. However, their approach is limited to discrete action spaces and relies on a Gumbel-Max structural causal model. In contrast, our work targets RL with continuous actions, introduces a novel distance measure for action sequences, and avoids dependence on causal models.

More recently, [Amitai *et al.*, 2024] proposed a method for visually comparing an agent's chosen action to counterfactual alternatives, conducting user studies in the highway environment with videos illustrating counterfactual outcomes. However, their approach is limited to discrete action spaces, with counterfactual actions selected heuristically. Instead, our work introduces an optimization-based method to compute counterfactual actions in continuous spaces, ensuring minimal distance from observed actions.

Additionally, other studies have explored counterfactual reasoning to enhance RL techniques. [Bica *et al.*, 2021] modeled expert decisions by defining reward functions based on preferences for "what if" outcomes and incorporating counterfactual reasoning into batch inverse RL. [Frost *et al.*, 2021] generated counterfactual trajectories by guiding agents to diverse, unseen states, enabling users to view rollouts of agent behavior and gain insights into its actions under test-time conditions. Our work has a different goal and formalizes counterfactual reasoning in RL through a novel problem formulation tailored for continuous action spaces.

## 3 Problem Formulation

We consider a problem setup of an RL agent interacting with an environment modeled as a Markov decision process (MDP), denoted as $\mathcal{M} = (S, A, P, R)$, where $S$ represents the state space, $A$ denotes the (continuous) action space, $P$ is the (unknown) probabilistic transition function, and $R$ is the reward function. At each time step $t$, the agent selects an action $a_t$ based on the current state $s_t$, receives a reward $R(s_t, a_t)$, and transitions to the next state $s_{t+1} \sim P(s_t, a_t)$ as governed by the environment dynamics. The agent's execution over a finite number of $n$ steps yields a trajectory $\tau = \{(s_{t+i}, a_{t+i})\}_{i=0}^{n}$ with cumulative reward $G(\tau) = \sum_{i=0}^{n} R(s_{t+i}, a_{t+i})$.

Given an observed trajectory $\tau = \{(s_{t+i}, a_{t+i})\}_{i=0}^{n}$, we define a *(positive) counterfactual trajectory* $\tau'$ as one that originates from the same initial state $s_t$ but achieves a higher cumulative reward, i.e., $G(\tau') > G(\tau)$, by adopting an alternative sequence of actions $\alpha(\tau') = \{a'_{t+i}\}_{i=0}^{n}$. We define the distance between two action sequences $\alpha(\tau)$ and $\alpha(\tau')$ as:

$$D\big(\alpha(\tau), \alpha(\tau')\big) = \sum_{i=0}^{n} \frac{|a_{t+i} - a'_{t+i}|}{|a_{t+i}| + \delta} \tag{1}$$

where $\delta > 0$ is a parameter to ensure numerical stability. For multi-dimensional action spaces, this equation is generalized using $\ell_p$ norms.

We formulate an optimization problem to compute a counterfactual trajectory with minimal action distance.

**Problem 1.** *Given a trajectory $\tau = \{(s_{t+i}, a_{t+i})\}_{i=0}^{n}$, find a counterfactual trajectory $\tau'$ starting from $s_t$ that satisfies:*

$$\arg\min_{\tau'} D\big(\alpha(\tau), \alpha(\tau')\big), \quad \textit{s.t. } G(\tau') > G(\tau).$$

Next, we consider a variant where actions in certain constrained states (denoted by $S^c \subseteq S$) must follow a predefined policy $\pi^c$, while actions in unconstrained states can be freely optimized to minimize the action distance and satisfy the reward constraint.

**Problem 2.** *Given a trajectory $\tau = \{(s_{t+i}, a_{t+i})\}_{i=0}^{n}$, find a counterfactual trajectory $\tau'$ starting from $s_t$ that satisfies:*

$$\begin{aligned} &\arg\min_{\tau'} D\big(\alpha(\tau), \alpha(\tau')\big), \\ \textit{s.t.} \quad & G(\tau') > G(\tau), \\ & \forall s'_{t+i} \in S^c,\ a'_{t+i} \sim \pi^c(s'_{t+i}). \end{aligned}$$

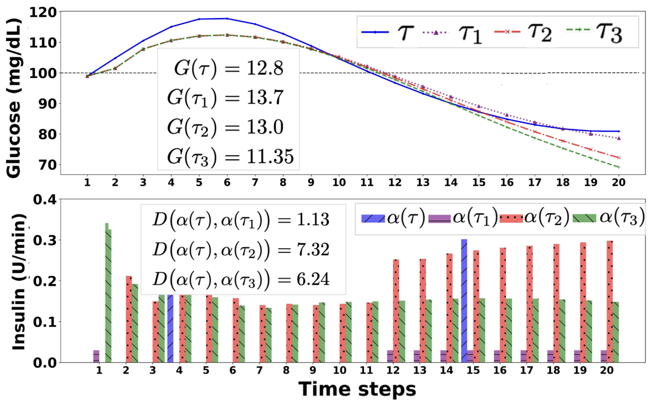


Figure 1: Observed ($\tau$) and counterfactual trajectories ($\tau_1, \tau_2, \tau_3$) of glucose levels (states) and insulin dosages (actions) over one hour.

**Motivating example.** We illustrate these problems using the example of blood glucose (BG) control for Type 1 Diabetes patients. The state space represents BG levels, influenced by factors such as patient physiology and behaviors (e.g., eating, exercise). The action space consists of insulin doses. At each time step (e.g., every three minutes), the agent computes an insulin dosage to regulate BG levels. The reward function assigns positive rewards for BG levels within the target range and penalties for hypoglycemia (BG $< 70$ mg/dL) or hyperglycemia (BG $> 180$ mg/dL). Figure 1 shows an observed trajectory of a patient's BG level and insulin dosage over an hour, with a cumulative reward $G(\tau) = 12.8$.

The goal is to find a counterfactual explanation addressing the question: "What alternative treatment plan (i.e., insulin dosage sequence) could lead to better glycemic control outcomes (i.e., higher cumulative reward)?" The counterfactual explanation should involve minimal changes from the original treatment plan (i.e., minimal distance of actions) and may include constraints (e.g., following clinical prescriptions).

Figure 1 shows three counterfactual trajectories $\tau_1$, $\tau_2$ and $\tau_3$. Among them, trajectory $\tau_1$ achieves the highest cumulative reward $G(\tau_1) = 13.7$ with the smallest action distance $D\big(\alpha(\tau), \alpha(\tau_1)\big) = 1.13$. We note that trajectory $\tau_1$ also adheres to the constraint of "injecting 0.03 units of insulin when BG levels fall below 100 mg/dL".

## 4 Approach

We propose a novel approach to solve the problems described in Section 3. Our approach extends the Twin Delayed Deep Deterministic Policy Gradient (TD3) algorithm [Fujimoto *et al.*, 2018], incorporating mechanisms for counterfactual generation and sparse reward shaping to account for trajectory action distance.

### 4.1 Solving Problem 1

We first relax the hard constraint $G(\tau') > G(\tau)$ in Problem 1 into a soft penalty term, yielding the following optimization problem:

$$\min_{\tau'} D(\alpha(\tau), \alpha(\tau')) + \lambda^{-1}(G(\tau) - G(\tau')) \tag{2}$$

where $\lambda$ is a hyperparameter reflecting the user's preferred weight for the penalty term. This is equivalent to

$$\max_{\tau'} G(\tau') - \lambda D(\alpha(\tau), \alpha(\tau')) \tag{3}$$

Directly searching over all possible trajectories $\tau'$ is challenging without knowledge of the transition function. Observing that trajectories are generated by policies, we approximate the problem by:

$$\max_{\mu} \quad \text{ess sup}\left[G(\tau'(\mu)) - \lambda D(\alpha(\tau), \alpha(\tau'(\mu)))\right] \tag{4}$$

Here $\mu : S \to A$ denotes a deterministic policy, and $\tau'(\mu)$ represents the trajectory generated by following $\mu$ starting from $s_t$, the first state of $\tau$. Specifically, $\tau'(\mu) \doteq \big\{(s'_{t+i}, a'_{t+i})\big\}_{i=0}^{n}$, where $s'_t \doteq s_t, a'_{t+i} \doteq \mu(s'_{t+i}), s'_{t+i+1} \sim P(s'_{t+i}, a'_{t+i})$. Notably, while $\mu$ is deterministic, $\tau'(\mu)$ remains a random variable due to the stochasticity in sampling $s'_{t+i+1}$ from the transition function. The essential supremum (ess sup) is taken over all possible realizations of $\tau'(\mu)$. We use the essential supremum here because Equation 3 requires finding the single best trajectory generated by $\mu$.

Optimizing the essential supremum directly is difficult, as no existing methods are designed for this in sequential decision-making. Instead, leveraging concentration inequalities (e.g., Hoeffding's inequality) and large deviation theory, which indicate that deviations from the expectation are typically well-bounded, we optimize the expectation:

$$\max_{\mu} \mathbb{E}\left[G(\tau'(\mu)) - \lambda D(\alpha(\tau), \alpha(\tau'(\mu)))\right] \tag{5}$$

A policy $\mu$ that maximizes this expectation will, with high probability, achieve a large essential supremum. Here, $\mathbb{E}\left[G(\tau'(\mu))\right]$ corresponds to the expected total rewards of the deterministic policy $\mu$. Thus, Equation 5 can be framed as a standard RL problem, where the goal is to maximize both the total rewards $G(\tau'(\mu))$ and an additional sparse reward $-\lambda D(\alpha(\tau), \alpha(\tau'(\mu)))$, provided only at the end of each episode.

Given the deterministic nature of the policy $\mu$ and the proven success of TD3 in optimizing deterministic policies, we extend TD3 to solve this problem, as outlined in Algorithm 1. The algorithm starts with initializing the critic networks $Q_{\theta_1}$ and $Q_{\theta_2}$, the actor network $\pi_\phi$ with random parameters, and target networks as copies of the original networks to ensure training stability. A replay buffer $\mathcal{B}$ is also initialized to store counterfactual trajectories. Given a set of observed trajectories $\mathcal{T}$, the algorithm samples a trajectory $\tau = (s_{t+i}, a_{t+i})_{i=0}^{n}$ from $\mathcal{T}$ and generates $N_c$ counterfactual trajectories. Each counterfactual trajectory $\tau_k$ starts from the same initial state as $\tau$ and is rolled out by selecting actions through the actor network $\pi_\phi$, with exploration noise added. To incorporate the action distance objective in Problem 1, the distance $D\big(\alpha(\tau), \alpha(\tau_k)\big)$ is computed using Equation 1, and the reward is adjusted with a weighted function (line 12) following Equation 5. Each generated counterfactual transition is stored in the replay buffer $\mathcal{B}$. The algorithm then samples a batch of transitions from $\mathcal{B}$ to compute the target action $\tilde{a}$ and target value $y$. The critic networks are updated by minimizing the mean squared error between the predicted Q-values

**Algorithm 1:** Counterfactual Generation

```
1  Initialize critic networks $Q_{\theta_1}$, $Q_{\theta_2}$, and actor network $\pi_\phi$ with random parameters $\theta_1$, $\theta_2$, $\phi$
2  Initialize target networks: $\theta'_1 \leftarrow \theta_1$, $\theta'_2 \leftarrow \theta_2$, $\phi' \leftarrow \phi$
3  Initialize replay buffer $\mathcal{B}$
4  for $e = 1$ to $N_o$ do
5      Sample a trajectory $\tau = \{(s_{t+i}, a_{t+i})\}_{i=0}^{n}$ from $\mathcal{T}$
6      for $k = 1$ to $N_c$ do
7          Set initial state $s'_t \leftarrow s_t$
8          for $i = 0$ to $n$ do
9              Select action with exploration noise:
                 $a'_{t+i} \sim \pi_\phi(s'_{t+i}) + \epsilon$, $\epsilon \sim \mathcal{N}(0, \sigma)$
10             Observe reward $r'_{t+i}$ and next state $s'_{t+i+1}$
11             if $i = n$ then
12                 Adjust reward:
                     $r'_{t+i} \leftarrow r'_{t+i} - \lambda \cdot D\big(\alpha(\tau), \alpha(\tau_k)\big)$
13             Store $(s'_{t+i}, a'_{t+i}, r'_{t+i}, s'_{t+i+1})$ in $\mathcal{B}$
14     Sample mini-batch of $N$ transitions $(s, a, r, s')$ from replay buffer $\mathcal{B}$
15     $\tilde{a} \leftarrow \pi_{\phi'}(s') + \epsilon$, $\epsilon \sim \text{clip}(\mathcal{N}(0, \tilde{\sigma}), -c, c)$
16     $y \leftarrow r + \gamma \min_{j=1,2} Q_{\theta'_j}(s', \tilde{a})$
17     Update critics:
         $\theta_j \leftarrow \arg\min_{\theta_j} N^{-1} \sum \big(y - Q_{\theta_j}(s, a)\big)^2$
18     if $e \bmod q = 0$ then
19         Update actor $\phi$ using deterministic policy gradient: $\nabla_\phi J(\phi) =$
             $N^{-1} \sum \nabla_a Q_{\theta_1}(s, a)\big|_{a=\pi_\phi(s)} \nabla_\phi \pi_\phi(s)$
20         Update target networks:
             $\theta'_j \leftarrow \eta\theta_j + (1-\eta)\theta'_j$,   $\phi' \leftarrow \eta\phi + (1-\eta)\phi'$
```

and the target value. To stabilize training, the actor network $\pi_\phi$ is updated less frequently, every $q$ iterations, using deterministic policy gradients. Finally, target networks are softly updated with a weighted average of the current and target parameters, further enhancing training stability. The algorithm terminates after sampling and processing $N_o$ observed trajectories.

### 4.2 Solving Problem 2

To address Problem 2, we construct an augmented MDP, transforming Problem 2 into an instance of Problem 1. Specifically, the augmented MDP has a state space defined as $\hat{S} \doteq S \setminus S^c$. In this augmented MDP, given a state-action pair $(s, a)$, the transition proceeds as follows. The successor state is first determined using the original transition function $P$. If the successor state does not belong to the constrained set $S^c$, it is presented to the agent immediately; otherwise, the predefined policy $\pi^c$ is followed until the agent reaches a state outside $S^c$.

By constructing this augmented MDP, Problem 2 in the original MDP is reduced to Problem 1 in the augmented MDP. Consequently, we can directly apply Algorithm 1 in this augmented MDP to solve Problem 2.

## 5 Experiments

We implemented the proposed approach and evaluated it in two RL domains: (i) diabetes control using the FDA-approved UVA/PADOVA simulator [Man *et al.*, 2014], and (ii) Lunar Lander from OpenAI Gym [Brockman, 2016]. Our implementation[1] is based on Stable-Baselines3 [Raffin *et al.*, 2021].

**Data Generation.** We trained a baseline policy using the Proximal Policy Optimization (PPO) algorithm [Schulman *et al.*, 2017] to generate trajectories for training and test datasets (details specific to each domain are provided later). Our approach is RL-method agnostic and can generate counterfactuals from trajectories produced by any RL algorithm.

**Baseline Method.** As no existing methods generate counterfactuals for continuous action RL, we compare our approach to a naive baseline that generates counterfactuals by rolling out the trained baseline policy in evaluation mode without state constraints or additional training.

**Metrics.** Our evaluation focuses on the effectiveness, efficiency, and generalization of the proposed approach. We define two metrics to assess the effectiveness.

- *Positive Counterfactual Percentage* ($\rho_+$): This metric measures the percentage of test set trajectories for which at least one positive counterfactual trajectory (i.e., a trajectory with a higher cumulative reward than the observed trajectory) is identified among a fixed number of generated counterfactuals. This metric applies to both the baseline and the proposed approaches.

- *Advantage Counterfactual Percentage* ($\rho_{\mathsf{adv}}$): This metric quantifies the percentage of test set trajectories for which the proposed approach generates a more advantageous counterfactual than the baseline. For an observed trajectory $\tau$, let $\tau_p^*$ and $\tau_b^*$ denote the best positive counterfactual trajectories with minimal action distance produced by the proposed and baseline approaches, respectively, satisfying $G(\tau_p^*) > G(\tau)$ and $G(\tau_b^*) > G(\tau)$. The proposed approach is considered advantageous if $\phi_G > \phi_D$, where:

$$\phi_G = \frac{G(\tau_p^*) - G(\tau)}{G(\tau_b^*) - G(\tau)}, \quad \phi_D = \frac{D\big(\alpha(\tau), \alpha(\tau_p^*)\big)}{D\big(\alpha(\tau), \alpha(\tau_b^*)\big)},$$

  indicating that reward gains outweigh additional action distance. $\rho_{\mathsf{adv}}$ is computed as the percentage of advantageous counterfactuals among all test trajectories with valid positive counterfactuals.

Efficiency is assessed through learning curves of $\rho_+$ and $\rho_{\mathsf{adv}}$, illustrating performance improvements during the training of the proposed approach. Generalization is evaluated by testing the approach on unseen trajectory datasets across diverse single- and multi-environment settings.

[1]Code is available at: https://github.com/safe-autonomy-lab/CounterfactualRL

## 5.1 Diabetes Control

**MDP Environment.** The FDA-approved UVA/PADOVA Simulator [Man *et al.*, 2014] was used to model glucose-insulin dynamics and simulate the effects of insulin delivery, carbohydrate intake, and other factors on blood glucose levels in Type 1 Diabetes patients. Virtual patient profiles were generated with varying parameters (e.g., age, weight, insulin sensitivity). The MDP state space includes glucose reading, glucose rate of change, and carbohydrate intake, while the action space consists of the insulin dosage at each time step. A reward function from [Zhu *et al.*, 2020] was used, providing the highest rewards for maintaining glucose levels within 90–140 mg/dL, smaller rewards for near-target ranges, and penalties for hypo- or hyperglycemic values, with increasing penalties for larger deviations.

**Data Generation.** The proposed approach was evaluated in two settings: single-environment (single patient) and multi-environment (population model with multiple patients). In the single-environment setting, a baseline policy was trained on a chosen patient profile for 100,000 steps, with a learning rate of 0.0001 and a gradient step size of 50. Observed trajectories were generated using a sliding window method, with each trajectory representing a 20-step segment (corresponding to a one-hour patient execution history). In the multi-environment setting, the baseline policy was trained on three patient profiles, with 3,000 steps per patient per round, continuing until stable performance was achieved. Other parameters were consistent with the single-environment setting. Both settings included 18 unique trajectories in each training and test set.

**Experimental Setup.** We evaluated three variants of the proposed approach: (P1) directly applying Algorithm 1 to solve Problem 1; (P2-base) adapting Algorithm 1 for Problem 2 with constrained states $S^c$ (glucose levels below 100 mg/dL) and the baseline policy as $\pi^c$; and (P2-fixed) solving Problem 2 with the same $S^c$ but using a fixed policy $\pi^c$ that injects 0.03 units of insulin per step. For each variant, Algorithm 1 trained an RL model to generate counterfactual trajectories with a distance reward weight of $\lambda = 1$. Counterfactual trajectories were computed for each training set trajectory and stored in a buffer. After a warm-up phase, batches of 256 trajectories were sampled for model updates using a learning rate of 0.0001 and 50 gradient steps. The RL agent was tested every 400 interaction steps, generating 10 counterfactual trajectories per test trajectory via policy rollout for evaluation. All three variants were compared to the same baseline method (rolling out the baseline policy without constrained states).

**Results Analysis.** We conducted seven independent trials for each method, and the results are analyzed below.

*Effectiveness:* Tables 1 and 2 present the $\rho_+$ and $\rho_{\mathsf{adv}}$ metrics evaluated on test performance across seven trials after training. All three methods outperform the baseline in Positive Counterfactual Percentage ($\rho_+$) across single- and multi-environment settings. In the single-environment setting, P1 and P2-fixed achieve the highest $\rho_+$, with P2-fixed slightly surpassing P1 in the multi-environment setting. For Advantage Counterfactual Percentage ($\rho_{\mathsf{adv}}$), P1 consistently demonstrates the best performance in both settings. P2-fixed also performs reliably, while P2-base moderately improves over the baseline in the multi-environment setting but fails to generate advantageous counterfactuals in the single-environment setting.

| Domain | Method | Single-Env | Multi-Env |
|---|---|---|---|
| Diabetes | P1 | **$0.53 \pm 0.01$** | $0.44 \pm 0.0$ |
| | P2-base | $0.47 \pm 0.01$ | $0.40 \pm 0.02$ |
| | P2-fixed | **$0.53 \pm 0.01$** | **$0.48 \pm 0.03$** |
| | Baseline | $0.44 \pm 0.0$ | $0.39 \pm 0.0$ |
| Lunar Lander | P1 | **$0.81 \pm 0.03$** | **$0.82 \pm 0.02$** |
| | P2-base | $0.65 \pm 0.08$ | $0.54 \pm 0.02$ |
| | P2-fixed | $0.31 \pm 0.06$ | $0.42 \pm 0.03$ |
| | Baseline | $0.61 \pm 0.02$ | $0.50 \pm 0.0$ |

Table 1: Mean and standard error of Positive Counterfactual Percentage ($\rho_+$) in single- and multi-environment settings.

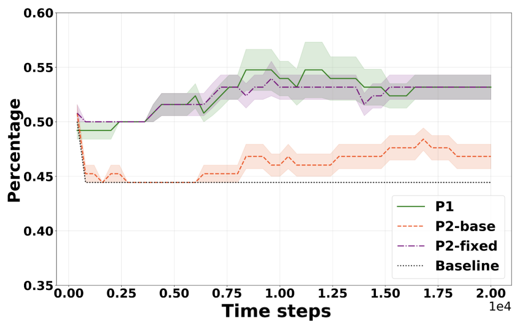
| Domain | Method | Single-Env | Multi-Env |
|---|---|---|---|
| Diabetes | P1 | **$0.67 \pm 0.0$** | **$0.86 \pm 0.0$** |
| | P2-base | $0.0 \pm 0.0$ | $0.41 \pm 0.02$ |
| | P2-fixed | $0.52 \pm 0.03$ | $0.65 \pm 0.06$ |
| Lunar Lander | P1 | **$0.38 \pm 0.03$** | **$0.94 \pm 0.04$** |
| | P2-base | $0.36 \pm 0.04$ | $0.72 \pm 0.05$ |
| | P2-fixed | $0.32 \pm 0.09$ | $0.63 \pm 0.04$ |

Table 2: Mean and standard error of Advantage Counterfactual Percentage ($\rho_{\mathsf{adv}}$) in single- and multi-environment settings.

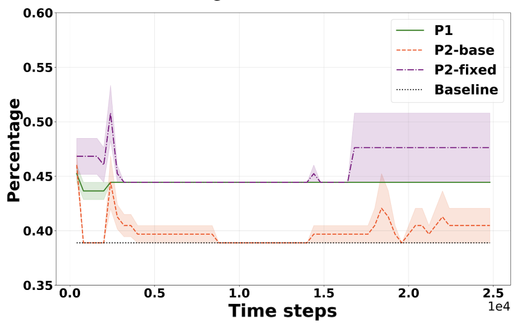
*Efficiency:* Figures 2 and 3 show the learning curves of $\rho_+$ and $\rho_{\mathsf{adv}}$ for each method in single- and multi-environment settings, with solid lines representing the mean and shaded areas denoting the standard error across seven trials. The baseline curve in Figure 2 represents counterfactuals generated by rolling out the baseline policy at each evaluation point, without additional training. In the single-environment setting (Figures 2a and 3a), P1 converges quickly and achieves the highest $\rho_+$ and $\rho_{\mathsf{adv}}$. P2-fixed steadily improves, matching P1 in $\rho_+$ and delivering strong $\rho_{\mathsf{adv}}$ performance. P2-base, however, converges more slowly and inconsistently, underperforming relative to P1 and P2-fixed. In the multi-environment setting (Figures 2b and 3b), P2-fixed achieves the highest $\rho_+$, while all three methods converge rapidly and maintain stable $\rho_{\mathsf{adv}}$ performance. Among them, P1 achieves the highest $\rho_{\mathsf{adv}}$, followed closely by P2-fixed, with P2-base showing slightly weaker but reliable results.

*Generalization:* Performance across settings highlights each method's generalization capabilities. P1 generalizes well, maintaining strong and consistent performance in both settings. P2-fixed also generalizes effectively, excelling in the multi-environment setting. P2-base shows limited generalization, struggling in the single-environment but improving slightly in the multi-environment, with notable variability.

*Summary:* In the diabetes control experiments, the proposed methods consistently outperform the baseline in effectiveness, efficiency, and generalization. P1 is the most effec-

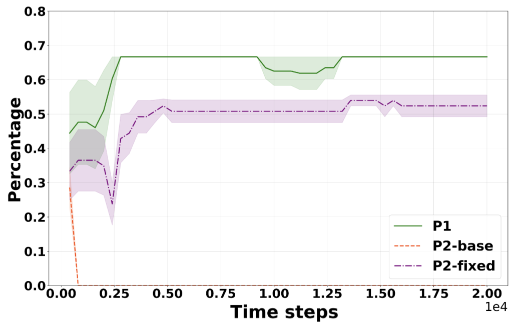


(a) Single-Environment

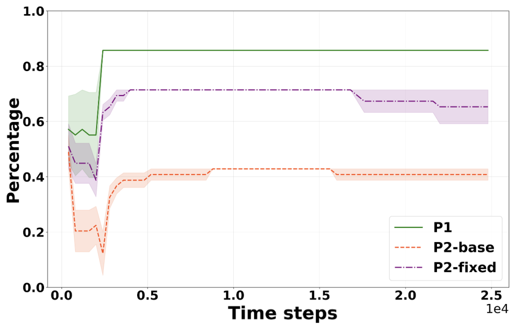


(b) Multi-Environment

Figure 2: Learning curves of Positive Counterfactual Percentage ($\rho_+$) for the Diabetes Control domain.



(a) Single-Environment



(b) Multi-Environment

Figure 3: Learning curves of Advantage Counterfactual Percentage ($\rho_{\text{adv}}$) for the Diabetes Control domain.

tive and efficient, while P2-fixed excels in generalization to multi-environment setting. P2-base, though weaker overall, demonstrates modest generalization potential.

## 5.2 Lunar Lander

**MDP Environment.** The Lunar Lander environment from OpenAI Gym [Brockman, 2016] simulates a 2D rocket attempting to land safely on a designated pad. The MDP state space includes eight continuous variables: the lander's x and y coordinates, x and y velocities, orientation angle, angular velocity, and two binary indicators for the left and right legs' contact with the ground. The action space includes two continuous variables controlling the throttle of the main engine and the lateral boosters. The reward function encourages safe landings, efficient fuel use, and minimal engine overuse.

**Data Generation.** The proposed approach was evaluated under single- and multi-environment settings. In the single-environment setting, a baseline policy was trained for 3,000 steps with a learning rate of 0.0001 and a gradient step size of 20. In the multi-environment setting, the baseline policy was trained across three gravity values, with 500 steps per environment per round, until stable performance was achieved. As in the Diabetes domain, trajectories were generated using a sliding window method, with each 20-step segment forming a trajectory. Both settings included 12 randomly sampled trajectories in each training and test set.

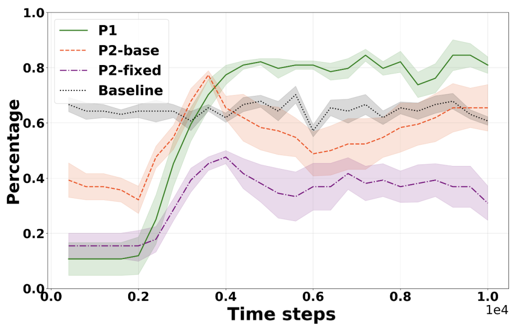
**Experimental Setup.** The proposed approach was used to generate counterfactual trajectories with a learning rate of 0.00001 and a gradient step size of 20, following a process similar to the diabetes experiments. Three variants were evaluated: P1, P2-base, and P2-fixed. In P2-base and P2-fixed, constrained states $S^c$ were defined as those with x-velocity in $[-0.18, 0.18]$. The policy $\pi^c$ was set as the baseline policy in P2-base, while P2-fixed used a fixed policy with action vectors set to $(0, 0)$.

**Results Analysis.** We conducted seven independent trials for each method and analyzed the results below.

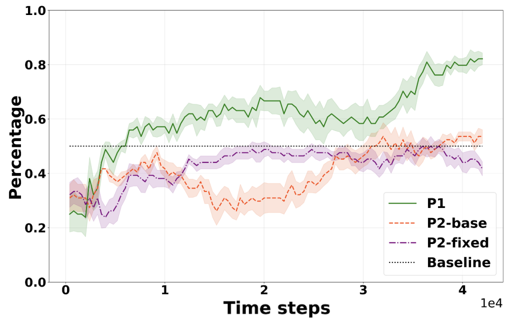
*Effectiveness:* As shown in Tables 1 and 2, P1 achieves the highest Positive Counterfactual Percentage ($\rho_+$) and Advantage Counterfactual Percentage ($\rho_{\text{adv}}$) in both single- and multi-environment settings, surpassing all other methods and the baseline. While P2-base falls short of P1 in both metrics, it outperforms the baseline. P2-fixed shows the weakest performance, slightly below the baseline in $\rho_+$.

*Efficiency:* The learning curves in Figures 4 and 5 depict the efficiency of each method in terms of Positive Counterfactual Percentage ($\rho_+$) and Advantage Counterfactual Percentage ($\rho_{\text{adv}}$) for the Lunar Lander domain. In the single-environment setting (Figures 4a and 5a), P1 converges quickly and achieves the highest $\rho_+$ and $\rho_{\text{adv}}$. P2-base steadily improves, surpassing the baseline, while P2-fixed converges slowly and lags behind other methods. In the multi-environment setting (Figures 4b and 5b), P1 consistently outperforms others in both $\rho_+$ and $\rho_{\text{adv}}$, with rapid convergence and strong results. P2-base and P2-fixed demonstrate less consistent progress in $\rho_+$ but maintain competitive performance in $\rho_{\text{adv}}$.

*Generalization:* All three methods generalize effectively

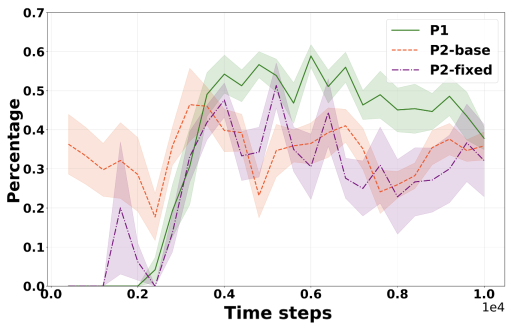


(a) Single-Environment

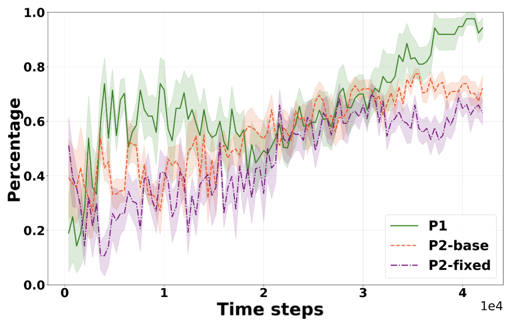


(b) Multi-Environment

Figure 4: Learning curves of Positive Counterfactual Percentage ($\rho_+$) for the Lunar Lander domain.



(a) Single-Environment



(b) Multi-Environment

Figure 5: Learning curves of Advantage Counterfactual Percentage ($\rho_{\text{adv}}$) for the Lunar Lander domain.

from single- to multi-environment settings. Notably, Advantage Counterfactual Percentage ($\rho_{\text{adv}}$) improves significantly in the multi-environment setting, while the learning curves remain comparable across both settings.

*Summary:* In the lunar landing experiments, P1 consistently outperforms other methods across both settings. P2-base demonstrates moderate effectiveness and generalization, with gradual improvements and variability in performance. P2-fixed, while initially competitive, lags behind and performs worse than the baseline in some cases.

## 5.3 Discussion

**Comparing Single- and Multi-Environment.** The $\rho_{\text{adv}}$ results in multi-environment settings show marked improvement over single-environment settings across all methods and domains (Table 2). This suggests that training in diverse environments enhances model performance by exposing it to a wider range of scenarios, enabling better generalization and the integration of more varied information. These findings highlight the value of tailoring counterfactual generation methods to the unique characteristics of each environment and leveraging diverse training conditions to achieve superior outcomes.

**Comparing Results of Two Domains.** The results from the Diabetes Control and Lunar Lander experiments highlight the influence of domain-specific characteristics on the performance of counterfactual generation methods. While P1 consistently performs best in both domains, P2-fixed outperforms P2-base in Diabetes Control, whereas P2-base surpasses P2-fixed in Lunar Lander. These differences arise from the nature of state transitions and action impacts in each environment. In Lunar Lander, actions produce immediate and short-lived effects—adjusting the engine throttle instantly changes the rocket's velocity, with the effects quickly dissipating. Conversely, Diabetes Control involves delayed and prolonged effects, as an insulin dose takes time to affect blood glucose levels and its influence persists for an extended period. This delay likely contributes to the more stable learning curves in the Diabetes Control domain compared to the more variable curves observed in Lunar Lander.

## 6 Conclusion

This work presents a novel approach for generating counterfactual explanations in continuous action RL. By extending the TD3 algorithm with mechanisms for constraint handling and reward shaping, our approach efficiently generates counterfactual trajectories that improve outcomes while minimizing deviations from observed actions. Experiments in the Diabetes Control and Lunar Lander domains demonstrate our approach's effectiveness, efficiency, and generalization across diverse environments. Notably, the P1 variant (without constrained states) achieves the best overall performance, while the P2 variants (with constrained states) offer flexibility for incorporating user-defined constraints and preferences. Future directions include enhancing the interpretability of counterfactuals through real-time feedback and visualization tools, further improving user trust and understanding.

## Acknowledgements

This work was supported in part by the U.S. National Science Foundation under Grants CCF-1942836 and CCF-2131511. The opinions, findings, conclusions, or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the sponsoring agencies.

## References

[Amitai *et al.*, 2024] Yotam Amitai, Yael Septon, and Ofra Amir. Explaining reinforcement learning agents through counterfactual action outcomes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 10003–10011, 2024.

[Bica *et al.*, 2021] Ioana Bica, Daniel Jarrett, Alihan Huyuk, and Mihaela van der Schaar. Learning "what-if" explanations for sequential decision-making. In *International Conference on Learning Representations*, 2021.

[Brockman, 2016] G Brockman. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[Chen *et al.*, 2024] Ziheng Chen, Fabrizio Silvestri, Gabriele Tolomei, Jia Wang, He Zhu, and Hongshik Ahn. Explain the explainer: Interpreting model-agnostic counterfactual explanations of a deep reinforcement learning agent. *IEEE Transactions on Artificial Intelligence*, 5(04):1443–1457, 2024.

[Frost *et al.*, 2021] Julius Frost, Olivia Watkins, Eric Weiner, Pieter Abbeel, Trevor Darrell, Bryan Plummer, and Kate Saenko. Explaining reinforcement learning policies through counterfactual trajectories. *ICML 2021 Workshop on Human in the Loop Learning*, 2021.

[Fujimoto *et al.*, 2018] Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*, pages 1587–1596. PMLR, 2018.

[Gajcin and Dusparic, 2024] Jasmina Gajcin and Ivana Dusparic. Redefining counterfactual explanations for reinforcement learning: Overview, challenges and opportunities. *ACM Computing Surveys*, 56(9):1–33, 2024.

[Guidotti, 2022] Riccardo Guidotti. Counterfactual explanations and how to find them: literature review and benchmarking. *Data Mining and Knowledge Discovery*, pages 1–55, 2022.

[Man *et al.*, 2014] Chiara Dalla Man, Francesco Micheletto, Dayu Lv, Marc Breton, Boris Kovatchev, and Claudio Cobelli. The uva/padova type 1 diabetes simulator: new features. *Journal of Diabetes Science and Technology*, 8(1):26–34, 2014.

[Milani *et al.*, 2023] Stephanie Milani, Nicholay Topin, Manuela Veloso, and Fei Fang. Explainable reinforcement learning: A survey and comparative review. *ACM Computing Surveys*, 2023.

[Olson *et al.*, 2021] Matthew L Olson, Roli Khanna, Lawrence Neal, Fuxin Li, and Weng-Keen Wong. Counterfactual state explanations for reinforcement learning agents via generative deep learning. *Artificial Intelligence*, 295:103455, 2021.

[Raffin *et al.*, 2021] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021.

[Schulman *et al.*, 2017] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[Tang *et al.*, 2024] Chen Tang, Ben Abbatematteo, Jiaheng Hu, Rohan Chandra, Roberto Martín-Martín, and Peter Stone. Deep reinforcement learning for robotics: A survey of real-world successes. *Annual Review of Control, Robotics, and Autonomous Systems*, 8, 2024.

[Tejedor *et al.*, 2020] Miguel Tejedor, Ashenafi Zebene Woldaregay, and Fred Godtliebsen. Reinforcement learning application in diabetes blood glucose control: A systematic review. *Artificial intelligence in medicine*, 104:101836, 2020.

[Tsirtsis *et al.*, 2021] Stratis Tsirtsis, Abir De, and Manuel Rodriguez. Counterfactual explanations in sequential decision making under uncertainty. *Advances in Neural Information Processing Systems*, 34:30127–30139, 2021.

[Verma *et al.*, 2024] Sahil Verma, Varich Boonsanong, Minh Hoang, Keegan Hines, John Dickerson, and Chirag Shah. Counterfactual explanations and algorithmic recourses for machine learning: A review. *ACM Computing Surveys*, 56(12):1–42, 2024.

[Wachter *et al.*, 2018] S Wachter, B Mittelstadt, and C Russell. Counterfactual explanations without opening the black box: automated decisions and the gdpr. *Harvard Journal of Law and Technology*, 31(2), 2018.

[Yu *et al.*, 2021] Chao Yu, Jiming Liu, Shamim Nemati, and Guosheng Yin. Reinforcement learning in healthcare: A survey. *ACM Computing Surveys (CSUR)*, 55(1):1–36, 2021.

[Zhu *et al.*, 2020] Taiyu Zhu, Kezhi Li, Pau Herrero, and Pantelis Georgiou. Basal glucose control in type 1 diabetes using deep reinforcement learning: An in silico validation. *IEEE Journal of Biomedical and Health Informatics*, 25(4):1223–1232, 2020.