

Do No Harm: A Counterfactual Approach to Safe Reinforcement Learning

Sean Vaskov

ISEE AI, Cambridge, MA, 02139

SEAN.VASKOV@ISEE.AI

Wilko Schwarting

ISEE AI, Cambridge, MA, 02139

WILKO@ISEE.AI

Chris L. Baker

ISEE AI, Cambridge, MA, 02139

CHRISBAKER@ISEE.AI

Editors: A. Abate, K. Margellos, A. Papachristodoulou

Abstract

Reinforcement Learning (RL) for control has become increasingly popular due to its ability to learn rich feedback policies that take into account uncertainty and complex representations of the environment. When considering safety constraints, constrained optimization approaches, where agents are penalized for constraint violations, are commonly used. In such methods, if agents are initialized in, or must visit, states where constraint violation might be inevitable, it is unclear how much they should be penalized. We address this challenge by formulating a constraint on the counterfactual harm of the learned policy compared to a default, safe policy. In a philosophical sense this formulation only penalizes the learner for constraint violations that it caused; in a practical sense it maintains feasibility of the optimal control problem. We present simulation studies on a rover with uncertain road friction and a tractor-trailer parking environment that demonstrate our constraint formulation enables agents to learn safer policies than contemporary constrained RL methods.

Keywords: reinforcement learning, viability, causal models, counterfactual inference, harm

1. Introduction

Learning-based control, particularly reinforcement learning (RL) (Li, 2017; Sutton and Barto, 2018) has become increasingly popular due to its ability to learn powerful feedback policies that take uncertainty and complex environment representations into account. However, learning policies for applications in which safe behavior is required remains a challenging problem. Methods for safe RL typically fall into one of two categories: constrained optimization where the objective is to learn a policy whose output actions satisfy safety constraints; or shielding where a backup or safety-preserving policy corrects the agent’s actions to be safe. Constrained optimization techniques commonly use the method of Lagrange multipliers where the penalty for constraint violation is increased until the constraint is satisfied (Altman, 1998; Chow et al., 2017; Paternain et al., 2019). Epigraph forms avoid numerical issues associated with Lagrange multipliers at the cost of introducing a state and estimator for a cost budget (So and Fan, 2023). In shielding approaches, a safety-preserving policy, often designed with control barrier functions (Cheng et al., 2019) or reachability analysis (Shao et al., 2021; Selim et al., 2022; Hsu et al., 2023) restricts the agent’s actions to be safe during training. In the context of safety, hierarchical methods (Barto and Mahadevan, 2003; Lee

et al., 2023) are related to shielding methods in that a higher-level policy learns when to execute a safety-preserving policy; however both policies can be trained with constrained optimization.

In simulated RL applications where constraint violation is not associated with tangible damage, optimization-based methods are preferred because they allow for more exploration as the agents are not limited to choosing the actions of the shielding policy. A challenge associated with Lagrange methods is balancing the feasibility of the constraints with adjustment of the multipliers. Enforcing that constraint violation never happens limits the domain policies can be used in and may be impossible in some environments; on the other hand, penalizing agents for constraint violations that are unavoidable will cause the multipliers to increase indefinitely and destabilize learning.

We address this challenge using counterfactuals (Roese, 1997; Balke and Pearl, 1994). In the context of policy optimization, counterfactual metrics compare the outcome of one policy or action to another. The conditional average treatment effect (CATE) (Shpitser and Pearl, 2012) compares the expected utility. In RL, CATE-style objectives have been used for both credit (Foerster et al., 2018) and blame (Triantafyllou et al., 2021) assignment. Algorithms that minimize expected counterfactual regret have also been applied to extensive-form games with discrete action spaces (Zinkevich et al., 2007; Brown et al., 2019). Counterfactual harm (Richens et al., 2022) compares the outcome of a selected action to a default action across all realizations of uncertainty, as opposed to in expectation, making it a more appropriate metric for safety than CATE. Beckers et al. (2022) offer a competing definition of harm where a default utility instead of action is used, and all actions are simultaneously compared, as opposed to the pairwise comparison of each action to the default.

In this paper we develop both CATE and harm-based constraints for safe RL. We base our harm constraint off of Richens et al. (2022). For dynamic systems, assuming a default policy or action is more appropriate than a default utility; in practice, the later would be computed based on analysis of the agent’s state-action space and dynamics. Additionally, we focus on single agent environments where we compare the learned policy to a default policy, so the pairwise comparison is complete. Extensions to multi-agent environments and/or multiple policies may require incorporating elements of Beckers et al.’s definition. To the best of our knowledge this is the first application of counterfactual reasoning, especially harm, to constrained RL. Our contributions are three-fold:

- Relating safe RL and viability theory to counterfactual reasoning techniques that analyze harm; this provides a philosophically and practically sound definition of safe behavior;
- A constraint based on the conditional average treatment effect (CATE), which we modify to reduce conservatism;
- A constraint restricting counterfactual harm, which addresses CATE’s deficiency in safely handling uncertainty. In our formulation, we also connect notions of “stability” in the counterfactual literature (Lucas and Kemp, 2015) to TD(λ) methods (Sutton and Barto, 2018).

The goal of this paper is to compare our proposed constraint formulations to those traditionally found in RL. We focus on the Lagrange multiplier approach; investigating compatibility with and comparing to epigraph forms is left to future work. We do not implement or compare to shielding and hierarchical methods; however, we discuss how to integrate our constraint formulations into them in Appendix A.¹ The paper is organized as follows. First, we give an overview of our notation, the constrained Markov decision process (CMDP) framework, and relevant concepts from causal reasoning and viability theory. Next, we propose three alternative constraint formulations for safe

1. An extended version, including the appendix, is available at <https://arxiv.org/abs/2405.11669>.

RL, the last two of which are contributions of this paper. Finally, we present two simulation studies showcasing the effectiveness of our constraints.

2. Preliminaries

In this section, we first define our notation and an instance of the general class of problems this paper solves. We then introduce concepts from causal modeling and viability theory that are relevant to our formulation and results. A Markov decision process (MDP) is a tuple $(\mathcal{S}, \mathcal{A}, p_s, r, \gamma, p_0)$ where \mathcal{S} and \mathcal{A} are compact state and action spaces, $p_s(\cdot|s_t, a_t)$ is the transition probability distribution conditioned on a state and action at time t , p_0 is the initial state distribution, $r : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ is an instantaneous reward function, and $\gamma \in (0, 1)$ is a discount factor. In the safe RL setting, we introduce a set of constraint functions $g_i : \mathcal{S} \rightarrow \mathbb{R}$, $i \in \{1, \dots, n_c\}$ whose 0-sublevel sets represent safe states. A policy is a mapping from states to actions and can either be deterministic or stochastic. Trajectories generated by the state transition distribution under a policy, π , are written as $s_{0:T}^\pi$. We can learn a safe, reward-maximizing policy by solving the following optimal control problem:

$$\pi^* = \operatorname{argmax}_{\pi} \mathbb{E} \left[\sum_{t \geq 0} \gamma^t r(s_t, a_t) \right] \quad (1a)$$

$$s_0 \sim p_0, s_{t+1} \sim p_s(\cdot|s_t, a_t), a_t \sim \pi(\cdot|s_t) \quad (1b)$$

$$\text{s.t. } \mathbb{E} \left[\max_{t \geq 0} \gamma^t g_i(s_t) \right] \leq 0, i \in \{1, \dots, n_c\}. \quad (1c)$$

Remark 1 Program (1) differs from the classical Constrained Markov Decision Process (Altman, 1998), which uses cumulative constraints (i.e. $\mathbb{E}[\sum_{t \geq 0} \gamma^t g_i(s_t)] \leq 0$). Here, (1c) ensures each constraint is non-positive at all times—the standard formulation for constrained optimal control problems (Johansen, 2011). All proposed constraint formulations in this paper can use either the max or sum operator.

Since (1) will be solved with RL we refer to π as the *learner policy*. Throughout this paper, we default to using the expectation operator without subscripts to indicate sampling π , with the state transition distribution p_s ; when needed we use subscripts to provide details, e.g. we write $\mathbb{E}_{p_0, \mu}$ to indicate that we are sampling actions from policy μ and initial states from p_0 . To solve (1) we leverage the method of Lagrange multipliers, and iteratively solve the unconstrained problem (Paternain et al., 2019):

$$(w^*, \pi^*) = \arg \min_{w \in \mathbb{R}_+^{n_c}} \max_{\pi} \mathbb{E} \left[\sum_{t \geq 0} \gamma^t r(s_t, a_t) \right] - \sum_{i=1}^{n_c} w_i \mathbb{E} \left[\max_{t \geq 0} \gamma^t g_i(s_t) \right] \quad (2a)$$

$$\text{s.t. } s_0 \sim p_0, s_{t+1} \sim p_s(\cdot|s_t, a_t), a_t \sim \pi(\cdot|s_t). \quad (2b)$$

Since the penalty for constraint violation, w , is automatically adjusted, the Lagrangian method is preferred to hand tuning in applications where constraint satisfaction conflicts with the performance objective (Roy et al., 2021). For notational brevity, we hereafter refer to a single constraint, g , as one can simply repeat our formulations for multiple constraints.

We next introduce assumptions and terminology related to counterfactual reasoning, structural causal models, and viability theory.

Assumption 2 Let (Ξ, \mathcal{F}, P) be a probability space. The state transition distribution can be modeled by a deterministic function, $f : \mathcal{S} \times \mathcal{A} \times \Xi \rightarrow \mathcal{S}$, with exogenous noise variables,

$$s_{t+1} = f(s_t, a_t, \xi_t), \quad \xi_t \in \Xi. \quad (3)$$

This assumption is always possible to satisfy using auto-regressive uniformization (Buesing et al., 2018, Lemma 2). Partially observable environments and the evolution of stochastic policies can be modeled by making the same assumption about the observation and action distributions.

Assumption 2 allows us represent the CMDP as a *structural causal model* (Pearl et al., 2000) where the state transitions, constraints, and rewards are endogenous variables and the noise values are exogenous variables. We refer to a trajectory of exogenous variables, $\xi_{0:T}$, as a *realization of uncertainty*. A *counterfactual query* is a triple of an observation, intervention, and outcome (X, I, Y) , that asks the question: given observation X , what would Y have been under intervention I ? The counterfactual query can be answered with *counterfactual inference* (Balke and Pearl, 1994). Counterfactual inference has three steps: infer the distribution of the (potentially unobserved) exogenous variables, $p_\xi(\cdot|X)$; simulate the system using (3) with the inferred distribution and intervention; and compute the outcome, Y . In our context X will be trajectories $s_{0:N}^\pi$, I will be a safe policy, and Y will be the constraint values.

In RL applications the practitioner often chooses the initial state distribution, p_0 . To understand how this relates to the feasibility of (1) we introduce concepts from viability theory (Aubin et al., 2011). Let p_ξ be a probability distribution that describes realizations of uncertainty. The *viability kernel* is the set of initial states from which the constraints can be satisfied:

$$\text{Viab}_{p_\xi} = \{s_0 \in \mathcal{S} \mid \forall \xi_{0:\infty} \sim p_\xi, \exists \{a_t\}_{t \geq 0} \text{ s.t. } \max_{t \geq 0} g(s_t) \leq 0, \text{ where } s_{t+1} = f(s_t, a_t, \xi_t)\}. \quad (4)$$

We define the viability kernel for a policy, μ , as

$$\text{Viab}_{p_\xi}^\mu = \{s_0 \in \mathcal{S} \mid \forall \xi_{0:\infty} \sim p_\xi, \max_{t \geq 0} g(s_t) \leq 0, \text{ where } s_{t+1} = f(s_t, \mu(s_t), \xi_t)\}. \quad (5)$$

Counterfactual inference allows us to understand how an agent should be penalized for constraint violation. If the agent was in the viability kernel, there is an intervention on the agent’s actions that can satisfy the constraint so the agent should be penalized harshly. As the agent leaves the viability kernel, the change in outcome for any intervention diminishes, so the agent’s penalty should as well. In the following section we present reformulations of (1c) that capture this relationship.

3. Formulations

In this section, we outline three alternative formulations of the constraint (1c), the last of which is the main contribution of this paper. We are particularly interested in the relationship between the initial state distribution, p_0 , the viability kernel (4), and the values of the constraints (1c). Leaving (1c) as is requires us to engineer the initial state distribution to be a subset of the viability kernel in order for (1) to be feasible. In this case, agents may never or rarely visit certain boundaries of the viability kernel, and produce unsafe behavior if they do. Furthermore, in some environments, engineering p_0 to be feasible may be impossible if uncontrollable forces or bad actors cause the agent to exit the viability kernel (regardless of how it is initialized). We reformulate (1c) to allow agents to be initialized in and out of the viability kernel. In order to do so, we make the following assumption:

Assumption 3 We are given access to a default action or policy, μ , that the agent can execute to avoid constraint violation. In the event that agents have exited the viability kernel, the default policy can minimize the severity of violation; for example, a vehicle could brake to reduce the impact energy of a collision. We assume there is one default μ for all constraints. μ can be hand-designed, computed using reachability analysis (Bansal et al., 2017; Fisac et al., 2019), learned with behavior cloning (Torabi et al., 2018), or a combination thereof.

Our first reformulation of (1c) makes the threshold a function of the default policy:

$$\mathbb{E}_{p_0, \pi} \left[\max_{t \geq 0} \gamma^t g(s_t) \right] \leq \mathbb{E}_{p_0, \mu} \left[\text{ReLU} \left(\max_{t \geq 0} \gamma^t g(s_t) \right) \right]. \quad (6)$$

This option ensures the feasibility of (1) but will not result in safe policies because it only enforces the learner policy to violate the constraints less than the default policy in expectation over p_0 and p_s . Using a risk metric such as CVaR $_\alpha$ (Chow et al., 2017) will not address this problem since we can apply the same argument about averaging to violations occurring in the α -quantile. In the following subsections we propose constraints that remedy the effects of averaging over p_0 and p_s .

3.1. Clipped Conditional Average Treatment Effect

The second option is a modified version of the conditional average treatment effect (CATE) (Shpitser and Pearl, 2012), which enforces the learner policy to be safer than the default policy in expectation:

$$\text{CCATE}_g^\mu(s_t) = \mathbb{E}_\pi \left[\max_{\tau > t} \gamma^{\tau-t} g(s_\tau) \right] - \text{ReLU} \left(\mathbb{E}_\mu \left[\max_{\tau > t} \gamma^{\tau-t} g(s_\tau) \right] \right). \quad (7)$$

We refer to this as the *clipped conditional average treatment effect* (CCATE). The clipping ensures that the learner policy is not required to satisfy the constraints more than the default policy, only to violate them less, which reduces conservatism while preserving safety. We replace (1c) with $\mathbb{E}[\max_{t \geq 0} \gamma^t \text{CCATE}_g^\mu(s_t)] \leq 0$. CCATE is stronger than (6) since it is enforced at each state the agent visits, instead of averaging over p_0 . CCATE can be implemented efficiently if we precompute a function approximation of μ 's expected constraint violation. However, CCATE only compares the violations in expectation over p_s —it is not robust to dynamics uncertainty.

3.2. Counterfactual Harm

The safest option uses *counterfactual harm* (Richens et al., 2022), which we apply as a constraint as follows: we first define the counterfactual harm at a given state, s_t , as

$$H_g^\mu(s_t) = \mathbb{E} \left[\text{ReLU} \left(\max_{\tau > t} \gamma^{\tau-t} g(s_\tau) - \text{ReLU} \left(\mathbb{E}_{\substack{\tilde{s}_{\tau+1}=f(\tilde{s}_\tau, \mu(\tilde{s}_\tau), \tilde{\xi}_\tau) \\ \tilde{\xi}_\tau \sim p_\xi(\cdot | s_{t+1}, s_t) \\ \tilde{s}_t = s_t}} \left[\max_{\tau > t} \gamma^{\tau-t} g(\tilde{s}_\tau) \right] \right) \right) \right]; \quad (8)$$

then we replace (1c) with the equality constraint $\mathbb{E}[\max_{t \geq 0} \gamma^t H_g^\mu(s_t)] = 0$.

Remark 4 Composing any monotonic function satisfying $f(0) = 0$, with counterfactual harm does not change the underlying constraint. E.g., the indicator function constrains the probability of harm.

Our definition of harm differs from Richens et al.’s in two ways. First, as in (7), we clip the outcome of μ for the constrained setting. Second, we compare outcomes over an infinite time horizon; we require this when exiting the viability kernel takes multiple timesteps. Counterfactual harm differs from CCATE in two ways. First, the exogenous variables are conditioned on the observed trajectory ($p_\xi(\cdot|s_{t+1}, s_t)$ in 8); this is important because we want to compare outcomes had the same or a similar scenario unfolded. Second, there is an additional clipping inside the expectation, which enforces the constraint for all realizations of uncertainty. Defining p_ξ is an active area of research. If the exogenous variables are observable, we can use the twin-network model (Pearl et al., 2000), where their values are equivalent ($p_\xi(\cdot|s_{t+1}, s_t) \leftarrow \delta_{\xi_t}$). Lucas and Kemp (2015) propose a stability parameter that probabilistically interpolates between sampling the observed or random noise.

To estimate (8) we perform counterfactual inference for N steps, summarized in Algorithm 1. We estimate the infinite time horizon outcome by using a learned approximation of the default policy’s *constraint value function*, $V_g^\mu(s_t) \approx \mathbb{E}_\mu[\max_{\tau \geq t} \gamma^{\tau-t} g(s_\tau)]$, and the recursive estimator

$$\hat{V}_g^{\mu, \lambda}(\tilde{s}_\tau) = \max\{g(\tilde{s}_\tau), \gamma(\lambda \hat{V}_g^{\mu, \lambda}(\tilde{s}_{\tau+1}) + (1 - \lambda)V_g^\mu(\tilde{s}_{\tau+1}))\}, \quad (9)$$

with $\hat{V}_g^{\mu, \lambda}(\tilde{s}_{\tau+N}) = V_g^\mu(\tilde{s}_{\tau+N})$ and $\lambda \in [0, 1)$. In RL, (9) is referred as a TD(λ) method (Sutton and Barto, 2018), but its application to the max operator is a contribution of this paper. Previous works using the max operator in RL (Fisac et al., 2019; Li et al., 2022) only present single step ($\lambda = 0$) estimates. λ is analogous to the stability parameter in Lucas and Kemp (2015) since increasing (decreasing) it weights trajectories generated from the observed (marginal) exogenous variables higher. We also apply (9) to estimate the learner policy’s constraint violation and the maximum harm over the episode as follows:

$$\hat{V}_H^\lambda(s_t) = \max\{\text{ReLU}(\hat{V}_g^{\pi, \lambda}(s_t) - \text{ReLU}(\hat{V}_g^{\mu, \lambda}(s_t))), \gamma(\lambda \hat{V}_H^\lambda(s_{t+1}) - (1 - \lambda)V_H(s_{t+1}))\}. \quad (10)$$

Algorithm 1: N -step Counterfactual Inference

Data: State trajectory $s_{t:t+N}^\pi$, default policy μ , constraint g , value function V_g^μ , stability parameter λ , N

Result: Counterfactual outcome: $\max_{\tau \geq t} \gamma^{\tau-t} g(s_t)$, of intervention: replace π with μ

```

1  $\tilde{s} \leftarrow \{s_t\}$ 
2 for  $\tau \in \{0, \dots, N-1\}$  do
3    $\tilde{s}_{\tau+1} = f(\tilde{s}_\tau, \mu(\tilde{s}_\tau), \tilde{\xi}_\tau)$ , with  $\tilde{\xi}_\tau \sim p_\xi(\cdot|s_{\tau+t+1}, s_{\tau+t})$  //  $p_\xi \leftarrow \delta_{\xi_t}$  if  $\xi$  observable
4    $\tilde{s} \leftarrow \tilde{s} \cup \tilde{s}_{\tau+1}$ 
5 end
6 return  $\hat{V}_g^{\mu, \lambda}(s_t)$ ; estimated with  $\tilde{s}$ , (9), and  $\hat{V}_g^{\mu, \lambda}(\tilde{s}_{t+N}) = V_g^\mu(\tilde{s}_{t+N})$ 
```

4. Learning Implementation

We solve both CCATE and Harm constrained algorithms using reinforcement learning and the Lagrange multiplier technique (2). For the policy optimization step in Algorithm 2, we use PPO (Schulman et al., 2017) with the addition of counterfactual inference. We use separate critics for the reward, default and learner constraints, and counterfactual harm (or CCATE). After collecting data from the learner rollouts we estimate returns of the reward using generalized advantage estimation (GAE) (Schulman et al., 2015) and constraint violation using (9). We apply Algorithm 1 at each state the agent visits (Line 6), and compute the counterfactual harm with (10). Alternatively,

Algorithm 2: Counterfactual Harm Constrained Policy Update (Actor Critic)**Data:** r, g, λ, M, N , Lagrange multipliers w , default policy μ , critics $V_r, V_g^\pi, V_g^\mu, V_H$ and actor π **Result:** Optimized policy π

```

1 for  $k \in \{0, \dots, M\}$  do
2   | Step forward environment with  $a_{t+k} \sim \pi_{\text{old}}(s_{t+k})$  and store  $s_{t+k}, a_{t+k}, r(s_{t+k}, a_{t+k}), g(s_{t+k})$ 
3 end
4 Estimate reward advantage  $\hat{A}_r^\lambda$  with GAE and constraint violations,  $\hat{V}_g^{\pi, \lambda}$  with (9)
5 for  $k \in \{M, \dots, 0\}$  do
6   | Estimate  $N$ -step counterfactual outcome,  $\hat{V}_g^{\mu, \lambda}(s_{t+k})$ , with Algorithm 1
7   | Estimate maximum harm over episode  $\hat{V}_H^\lambda(s_{t+k})$  with (10)
8 end
9 Optimize  $\pi$  with (11) plus critics with L2 (or other suitable) loss, e.g.  $V_g^\pi$  with  $\sum_i (\hat{V}_g^{\pi, \lambda}(s_i) - V_g^\pi(s_i))^2$ 

```

per Remark 4, the γ -discounted probability of harm can be optimized by applying an indicator function to the computed harm. For CCATE we replace $\hat{V}_g^{\mu, \lambda}$ with V_g^μ and remove the outer ReLU. To optimize the actor, π , we use the loss (omitting clipping terms in PPO for brevity):

$$\mathcal{L}_{\text{actor}} = \frac{-1}{N_{\text{batch}}} \sum_i \left(\frac{\pi(a|s_i)}{\pi_{\text{old}}(a_i|s_i)} (\hat{A}_r^{\pi, \lambda}(s_i) - w \cdot (\hat{V}_H^\lambda(s_i) - V_H(s_i))) + \mathcal{E}(\pi(a|s_i)) \right), \quad (11)$$

which maximizes the advantage of the unconstrained objective (2). $\mathcal{E}(\pi(a|s_i))$ in (11) is a small reward for entropy of the policy. The critics are trained by minimizing the mean squared error or cross-entropy loss (in chance-constrained settings) from returns computed with (9) (Line 9). Convergence proofs for using (9) to estimate the constraint value function are given in Appendix B.

5. Experiments

In this section we compare the performance of different constraint formulations. We compare to two formulations from the literature: Direct Behavior Specification (DBS) (Roy et al., 2021), which composes an indicator function with the constraint; and Instantaneous Constrained RL (IC) (Li et al., 2021), which applies a clipping function. These methods use cumulative constraints and initialize agents within a heuristically chosen subset of the viability kernel, referred to as the *feasible initialization*, which is common practice in RL. The additional formulations are as follows: MC_0 uses (1c) and the feasible initialization; MC uses (6); CCATE (7); and HARM (8). We also test chance-constrained formulations of (1c), CC_0, and (6), CC, which apply an indicator function to the constraint; and CCATE_C and HARM_C which follow from Remark 4.

For the experiments we assume we have access to the simulation model, meaning (3) is the environment model and the exogenous variables are observable. This paradigm is applicable to RL methods that use an accessible simulation model to generate synthetic data (Buesing et al., 2018; Levine et al., 2020). The goal of this study is to understand the benefits of the counterfactual harm constraint, not to produce the best possible agents for benchmarking, so we use hand-designed default policies and quantify their suboptimality. All methods use the Lagrange multiplier approach (2) with PPO. To improve efficiency we increase the multiplier update rate as training progresses; analogous to the penalty parameter in Li et al. (2021). We train all algorithms using the same hyperparameters; which we tuned to achieve efficient training as a group—we did not optimize hyperparameters for methods individually. If the performance worsened at the end of training, we select the checkpoint with the least constraint violations followed by the highest success rate.

We compare the methods on two environments, a rover navigating a U-shape track with uncertain road friction and a tractor-trailer parking task. We compare the ability of the learned policies to stay within the viability kernel, the severity of constraint violation, and the success rate. Analytically computing the true viability kernel (4) is intractable, so we use the *default viability kernel*, Viab_δ^μ (5), where the noise distribution follows the twin network model (Pearl et al., 2000). We initialize 20,000 agents randomly in and out of Viab_δ^μ and compare the recall and discovery rate (DR) of the learner policy. The recall assesses safety by counting instances where the learner policy violates the constraints and the default policy does not. DR counts instances where the learner policy satisfies the constraints and the default policy does not; which quantifies how much Viab_δ^μ underapproximates the true viability kernel. We also measure the probability that agents reach their goal, “Success”, safely given they start within Viab_δ^μ ; and the probability that they incur harm, P_{harm} . We round all values to the nearest percent. For the tractor trailer environment, we plot the cumulative distributions of harm and constraint violation. Extended descriptions of each environment, plots of performance and safety during training, and further discussion are in Appendix C.

5.1. Rover

In this environment a rover is tasked to navigate a U-shaped corridor and reach a goal depicted in Figure 1. The rover is modeled as a kinematic bicycle model with a circular footprint. The action commands are longitudinal acceleration and wheel angle; which, along with the speed, are limited in the dynamics model. The surface has a friction coefficient randomly sampled from the interval $[0.3, 1.0]$ for each environment. The longitudinal and lateral acceleration are clipped to lie within the friction circle. The rover receives a noisy observation of its state and the friction coefficient. Noise is also added to the rover’s action commands and the true friction coefficient at each timestep. The rover reaches the goal if it arrives within 0.5 m of the origin with a speed ≤ 0.1 m/s. The constraint is the signed distance of the center of mass to the outer boundary and obstacle in the middle.

The feasible initialization randomly places agents in the center of the corridor at 0 speed and a heading within $\pm\pi/2$ relative to the corridor direction. To initialize agents out of the viability kernel, we allow the center of mass to be anywhere in the corridor and for the agents to have nonzero speed; as a result 50% of agents are initialized out of Viab_δ^μ . The policy is a normal distribution with a state-independent variance parameter for each action. The network architecture used for the critics and the policy mean has 2 layers of dimension 256 and Tanh activation functions. The default policy is braking with maximum deceleration and steering toward the corridor center. We use a time discretization of 0.5 s and episode length of 100 steps. We train with 3,000 parallel environments for 15,000 policy updates and use $N = 5$ counterfactual steps and $M = 24$ steps for each rollout in Algorithm 2. The addition of counterfactual inference and critics increases training time by 5%. The results are shown in Table 1.

5.2. Tractor-Trailer Parking

The second environment is a tractor-trailer parking task where the ego agent must park a trailer into a variety of spots as shown in Figure 2. The actions are the acceleration and wheel angle rate and are limited, along with the speed, within the dynamics. Normally distributed noise proportional to the speed is added to the dynamics. Additionally the trailers take random, unobservable, wheelbases. The static environment is an occupancy grid and the agent performs raycasting to return the distance to the closest occupied points and their relative velocity. Noise proportional to their distance is

	Rec	DR	Success	P_{harm}
DBS	0.96	0.02	0.95	0.06
IC	0.93	0.01	0.93	0.09
MC_0	0.93	0.02	0.92	0.10
CC_0	0.96	0.02	0.96	0.05
MC	0.96	0.03	0.96	0.06
CC	0.97	0.03	0.97	0.04
CCATE	0.97	0.02	0.97	0.05
CCATE_C	0.98	0.03	0.89	0.03
HARM	0.96	0.02	0.96	0.05
HARM_C	0.99	0.03	0.99	0.02

Table 1: Viability Statistics for rover

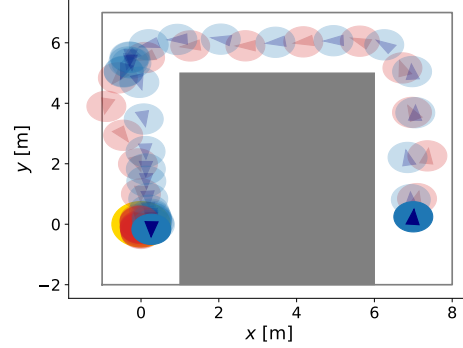


Figure 1: CC (red) and HARM_C (blue) policies

added to the observation. The constraint we consider is a modified version of the signed distance to obstacles where, when violated, we multiply it by $1+v^2/2$. The addition of $v^2/2$ penalizes collision severity by assuming the tractor absorbs 50% of its kinetic energy upon impact.

Agents are initialized in collision-free states near their parking spots with random positions, orientations, velocities and wheel angles. 22% of the agents are initialized outside of the default viability kernel. The feasible initialization sets their velocity to 0. The default policy, μ , is braking with maximum deceleration and maintaining the current wheel angle. The network architecture consists of a shared two-layer encoder network that processes the raycasting and state observations, then passes the concatenated output to the actor and critic networks, which are similar to those used for the rover. The training hyperparameters are the same as the rover except the episode length is extended to 300 steps and the number of counterfactual steps is $N = 4$. The agents are trained with 10,000 parallel environments for 15,000 policy updates. The addition of counterfactual inference and critics increases training time by 25%. The performance statistics are shown in Table 2 and the cumulative distributions of the harm and constraint violation in Figure 3.

	Rec	DR	Success	P_{harm}
DBS	0.93	0.07	0.90	0.18
IC	0.90	0.06	0.89	0.20
MC_0	0.91	0.07	0.90	0.18
CC_0	0.94	0.07	0.91	0.16
MC	0.80	0.09	0.79	0.23
CC	0.93	0.08	0.92	0.19
CCATE	0.92	0.08	0.90	0.14
CCATE_C	0.93	0.08	0.92	0.18
HARM	0.94	0.09	0.93	0.11
HARM_C	0.93	0.08	0.93	0.17

Table 2: Viability statistics for tractor-trailer

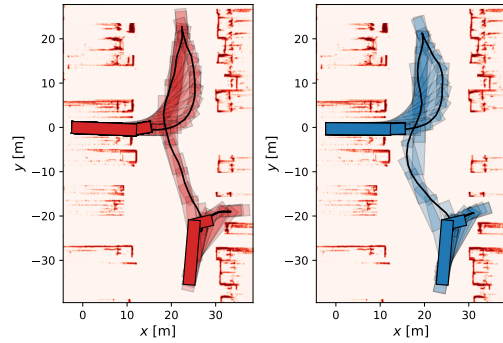


Figure 2: CC_0 (red) and HARM (blue) policies

5.3. Discussion

HARM_C and HARM achieve the highest recall, success rate, and lowest P_{harm} in the rover and tractor-trailer environments, respectively. The cumulative distribution of HARM in the tractor-

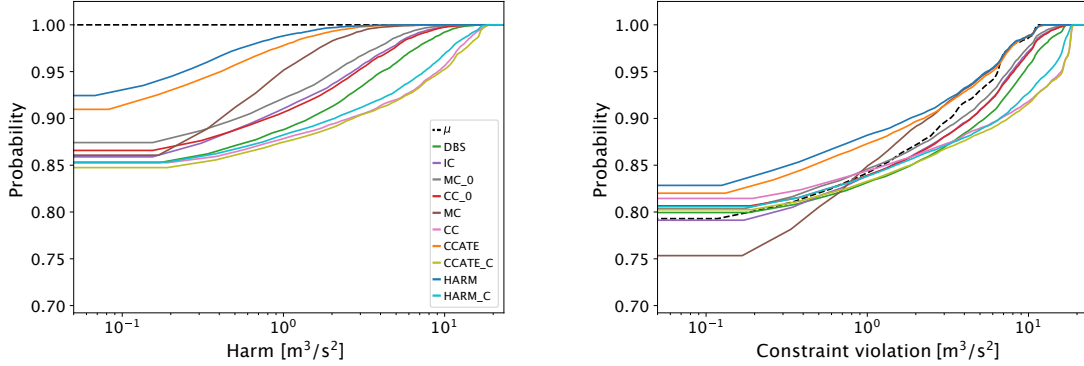


Figure 3: Cumulative distribution of harm (left) and constraint violations (right) for tractor-trailer. The black dashed lines are generated by executing the default policy, μ , from the initial states.

tractor environment upperbounds those of the other methods, indicating it incurs less constraint violations both in terms probability and severity. Optimizing the probability of harm performs worse in the tractor-trailer environment because the impact energy constraint requires the agent to exactly apply the maximum deceleration to achieve 0 harm, which PPO cannot easily sample from a normally distributed policy. It is easier to achieve 0 harm with the signed distance constraint of the rover, by sampling actions that steer the agent toward the centerline.

In the tractor-trailer environment, CC_0 matches and DBS is close to achieving the same recall, despite using the feasible initialization; however, they incur more severe constraint violations as shown in Figure 3. MC and CC are able to satisfy their constraints but do not generate safe policies due to the issues with (6) outlined in Section 3. Compared to those for the rover, every method’s DR is higher due to the higher speed and suboptimality of the default policy. Considering a default policy with evasive steering action would reduce the DR, although conditioning on the default policy did not impede the safety of HARM. In this experiment, the DRs are similar across methods and HARM incurred the fewest and least severe constraint violations. HARM and/or HARM.C also achieves the highest success rate in both environments. We hypothesize that the counterfactual constraint enables the policy to learn a better understanding of the viability kernel since it is only penalized for state-action sequences that cause or increase constraint violation.

6. Conclusion

We presented novel counterfactual constraints for safe reinforcement learning; most notably, a constraint on counterfactual harm. Our formulation performs counterfactual inference to only penalize agents when they cause a constraint to be violated or increase in magnitude, which ensures feasibility when violation is inevitable without sacrificing safety when violation is avoidable. In rover and tractor-trailer environments we showed that our counterfactual harm constraint enables agents to learn safer and more performant policies than alternative constraint formulations. The cost of our method is additional computations for counterfactual inference, which was modest in our experiments but could be significant if simulation of the environment model is expensive and counterfactual inference has to be performed over a long time horizon; this could possibly be addressed with off-policy methods. Finally, our experiments were conducted in single-agent environments with hand-designed default policies. In future work we will study the value of counterfactual harm constraints in multi-agent settings and explore applications with learned default policies.

Acknowledgments

We thank Feng Xiang for the network design and development of the tractor-tractor environment. We would also like to thank Michal Čáp, Spencer Sharp, Xiaobai Liu, Yibiao Zhao, and Tobias Gerstenberg for helpful discussions and comments.

References

- Eitan Altman. Constrained Markov decision processes with total cost criteria: Lagrangian approach and dual linear program. *Mathematical methods of operations research*, 48:387–417, 1998.
- Jean-Pierre Aubin, Alexandre M Bayen, and Patrick Saint-Pierre. *Viability theory: new directions*. Springer Science & Business Media, 2011.
- Alexander Balke and Judea Pearl. Counterfactual probabilities: Computational methods, bounds and applications. In *Uncertainty Proceedings 1994*, pages 46–54. Elsevier, 1994.
- Somil Bansal, Mo Chen, Sylvia Herbert, and Claire J Tomlin. Hamilton-Jacobi reachability: A brief overview and recent advances. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 2242–2253. IEEE, 2017.
- Andrew G Barto and Sridhar Mahadevan. Recent advances in hierarchical reinforcement learning. *Discrete event dynamic systems*, 13:341–379, 2003.
- Sander Beckers, Hana Chockler, and Joseph Y Halpern. Quantifying harm. *arXiv preprint arXiv:2209.15111*, 2022.
- Noam Brown, Adam Lerer, Sam Gross, and Tuomas Sandholm. Deep counterfactual regret minimization. In *International conference on machine learning*, pages 793–802. PMLR, 2019.
- Lars Buesing, Theophane Weber, Yori Zwols, Sebastien Racaniere, Arthur Guez, Jean-Baptiste Lespiau, and Nicolas Heess. Woulda, coulda, shoulda: Counterfactually-guided policy search. *arXiv preprint arXiv:1811.06272*, 2018.
- Richard Cheng, Gábor Orosz, Richard M Murray, and Joel W Burdick. End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 3387–3395, 2019.
- Yinlam Chow, Mohammad Ghavamzadeh, Lucas Janson, and Marco Pavone. Risk-constrained reinforcement learning with percentile risk criteria. *The Journal of Machine Learning Research*, 18(1):6070–6120, 2017.
- Jaime F Fisac, Neil F Lugovoy, Vicenç Rubies-Royo, Shromona Ghosh, and Claire J Tomlin. Bridging Hamilton-Jacobi safety analysis and reinforcement learning. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8550–8556. IEEE, 2019.
- Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

- Kai-Chieh Hsu, Allen Z Ren, Duy P Nguyen, Anirudha Majumdar, and Jaime F Fisac. Sim-to-lab-to-real: Safe reinforcement learning with shielding and generalization guarantees. *Artificial Intelligence*, 314:103811, 2023.
- Tor A Johansen. Introduction to nonlinear model predictive control and moving horizon estimation. *Selected topics on constrained and nonlinear control*, 1:1–53, 2011.
- Kyowoon Lee, Seongun Kim, and Jaesik Choi. Adaptive and explainable deployment of navigation skills via hierarchical deep reinforcement learning. *arXiv preprint arXiv:2305.19746*, 2023.
- Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.
- Jingqi Li, David Fridovich-Keil, Somayeh Sojoudi, and Claire J Tomlin. Augmented Lagrangian method for instantaneously constrained reinforcement learning problems. In *2021 60th IEEE Conference on Decision and Control (CDC)*, pages 2982–2989. IEEE, 2021.
- Jingqi Li, Donggun Lee, Somayeh Sojoudi, and Claire J Tomlin. Infinite-horizon reach-avoid zero-sum games via deep reinforcement learning. *arXiv preprint arXiv:2203.10142*, 2022.
- Yuxi Li. Deep reinforcement learning: An overview. *arXiv preprint arXiv:1701.07274*, 2017.
- Christopher G Lucas and Charles Kemp. An improved probabilistic account of counterfactual reasoning. *Psychological review*, 122(4):700, 2015.
- Santiago Paternain, Luiz Chamon, Miguel Calvo-Fullana, and Alejandro Ribeiro. Constrained reinforcement learning has zero duality gap. *Advances in Neural Information Processing Systems*, 32, 2019.
- Judea Pearl et al. Models, reasoning and inference. *Cambridge, UK: CambridgeUniversityPress*, 19(2):3, 2000.
- Jonathan Richens, Rory Beard, and Daniel H Thompson. Counterfactual harm. *Advances in Neural Information Processing Systems*, 35:36350–36365, 2022.
- Neal J Roese. Counterfactual thinking. *Psychological bulletin*, 121(1):133, 1997.
- Julien Roy, Roger Girgis, Joshua Romoff, Pierre-Luc Bacon, and Christopher Pal. Direct behavior specification via constrained reinforcement learning. *arXiv preprint arXiv:2112.12228*, 2021.
- John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.
- John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
- Mahmoud Selim, Amr Alanwar, Shreyas Kousik, Grace Gao, Marco Pavone, and Karl H Johansson. Safe reinforcement learning using black-box reachability analysis. *IEEE Robotics and Automation Letters*, 7(4):10665–10672, 2022.

- Yifei Simon Shao, Chao Chen, Shreyas Kousik, and Ram Vasudevan. Reachability-based trajectory safeguard (RTS): A safe and fast reinforcement learning safety layer for continuous control. *IEEE Robotics and Automation Letters*, 6(2):3663–3670, 2021.
- Ilya Shpitser and Judea Pearl. Identification of conditional interventional distributions. *arXiv preprint arXiv:1206.6876*, 2012.
- Oswin So and Chuchu Fan. Solving stabilize-avoid optimal control via epigraph form and deep reinforcement learning. *arXiv preprint arXiv:2305.14154*, 2023.
- Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.
- Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral cloning from observation. *arXiv preprint arXiv:1805.01954*, 2018.
- Stelios Triantafyllou, Adish Singla, and Goran Radanovic. On blame attribution for accountable multi-agent sequential decision making. *Advances in Neural Information Processing Systems*, 34:15774–15786, 2021.
- Martin Zinkevich, Michael Johanson, Michael Bowling, and Carmelo Piccione. Regret minimization in games with incomplete information. *Advances in neural information processing systems*, 20, 2007.