Constrained Differentiable Cross-Entropy Method for Safe Model-based Reinforcement Learning

Anonymous Author(s)*

ABSTRACT

This paper proposes MPC-CDCEM, a model-based reinforcement algorithm (RL) that allows the agent to safely interact with the environment and explore without additional assumptions on system dynamics. The algorithm uses a Model Predictive Control (MPC) framework with a differentiable cross-entropy optimizer, which induces a differentiable policy that considers the constraints while addressing the objective mismatch problem in model-based RL algorithms. We evaluate our algorithm in Safety Gym environments and on a practical building energy optimization problem. In addition, we showed that in both experiments, our algorithms have the lowest number of constraint violations and achieve comparable rewards compared to baseline constrained RL algorithms.

CCS CONCEPTS

• **Computing methodologies → Reinforcement learning**; **Control methods**; **Planning and scheduling**; **Machine learning algorithms**.

KEYWORDS

Reinforcement Learning, Constrained Markov Decision Process, Cross-Entropy Methods, Differentiable Convex Optimization, Limited Multi-label Classification

ACM Reference Format:

Anonymous Author(s). 2022. Constrained Differentiable Cross-Entropy Method for Safe Model-based Reinforcement Learning. In *Proceedings of 9th ACM International Conference on Systems for Energy-Efficient Built Environments (BuildSys '22).* ACM, New York, NY, USA, 9 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

In recent years, reinforcement learning (RL) has shown exceptional success in various automated control and decision-making tasks. The RL algorithm can automatically learn a policy that satisfies the specified objective. However, current RL algorithms often require millions of interactions with the environment, which results in an expensive training process and primarily limits their application to simulated domains [38, 39]. In addition, transferring policies learned in a simulation environment has proven to be challenging due to model uncertainties [12, 43] and mismatch between real and simulated observations.

BuildSys '22, November 09–10, 2022, Boston, MA

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

In many real-world applications, safety considerations also prevent the agent from freely exploring the environment. For example, a self-driving agent cannot take any actions that could cause harm to pedestrians while learning to optimize its driving policies. The agent needs to be constrained to specific actions that do not violate the safety requirements. In general, it is usually non-trivial to transform constrained optimal control problems into unconstrained problems [45].

One common approach to addressing this issue is to enforce some operational constraints on the outputs of a machine learning algorithm. However, usually, these constraints are not enforced during the training process and can potentially negatively impact the overall performance of the system [9].

Other approaches have sought to develop constrained and policy gradient-based safe RL algorithms, however, current methods cannot guarantee strict feasibility of policies even when initialized with feasible initial policies [27]. This limitation precludes their use in safety-critical environments.

A third approach has generated RL algorithms that transform the reward optimization criteria into a combination of reward and constraint violation cost, however, such methods suffer when the task objective and safety objectives contradict each other [45]. In addition, learning the dynamics of the environment and black box cost function typically is very difficult, especially in high dimensional space [34].

The work in this paper is inspired by recent results applying the differentiable cross-entropy method (DCEM) [6], and we propose a new safe reinforcement learning algorithm we name the Constrained Model Predictive Differentiable Cross-Entropy Method (MPC-CDCEM) that builds upon the success of DCEM. In each iteration, the algorithm samples from the distribution of policies and selects a set of trajectories with the best objective values that satisfy constraint values. If there are not enough feasible trajectories, the algorithm uses trajectories with the best constraint satisfaction performance. Compared to similar proposed solutions that use the traditional cross-entropy method [45], using the differentiable cross-entropy method enables an end-to-end learning process for both optimizing the objective and learning system dynamics. The differentiable policy class parametrized by the model-based components is a solution to the objective mismatch problem in model-based control [25], which arises when the objective being optimized is different from a target, often uncorrelated metric that we wish to optimize. In the context of model-based reinforcement learning, the model that achieves better performance in one-step ahead prediction of system dynamics is not necessary better for control. Another benefit of a differentiable policy is that it allows us to learn a low dimensional latent action space. Learning lower dimensional latent space of reasonable candidates enables the policy to leverage spatial and temporal structure in the solution space of optimal action sequence and ignore irrelevant action sequences.

The contributions of this paper are as follows. First, we present a model-based constrained RL algorithm in continuous state and action spaces. We formulate the problem under the Constrained Markov Decision Process [4] framework with minimal additional assumptions on system dynamics and constraint function. The differentiable cross-entropy method induces a differential control policy that addresses the objective mismatch problem in model-based control problems. We show that our approach can achieve state-of-the-art performance in terms of constraint violation number and accumulated expected return on Safety Control Gym [46]. We also explore a microgrid energy management system to reduce energy consumption while ensuring thermal comfort satisfaction for occupants and equipment safety by preventing excessive cycling in chillers.

2 RELATED WORK

Our approach relies on recent developments in differentiable cross-entropy methods and is thematically similar to several recent works [27, 45]. Here we discuss these topics and refer the interested reader to [20] for a more comprehensive review of safe RL topics.

[20] considers two main approaches to safe RL. The first is based on modifying the optimality criterion to introduce the concept of risk, and the second modifies the exploration process to avoid actions that can lead to undesirable or catastrophic situations. Regarding the first approach, optimization-based methods can be further categorized as worst-case criterion [32, 41], risk-sensitive criterion [7, 8, 23], constrained criterion [4, 24, 30], and other optimization criteria such as r-squared value-at-risk (Var) [28], or density of return [31]

Regarding the second approach, in general, there are two main ways of modifying the exploration process. Prior knowledge can be incorporated into the exploration process [1, 19, 36, 42], and risk measures can be added to determine the probability of selecting an action during the exploration process [21, 26].

In this work, we focus on the constrained Markov decision process formulation for safe RL [4]. [44] proposed a projection-based constrained policy gradient method that relies on projected gradients to ensure feasibility. [2] proposed a model-free constrained optimization method based on trust-region methods. However, these methods suffer from errors in gradient and Hessian matrix estimation, which may lead to underperformance [45]. [3, 40] proposed Lagrangian methods that use adaptive penalty coefficients to ensure constraint feasibility, which requires target constraint violations to be set in advance.

Several papers proposed a safe RL algorithm that uses a model-based learning framework. Model-based approaches often produce more sample-efficient control solutions while ensuring constraint feasibility [18, 35]. [16] proposed a method that combines model-free control with a model-based safety check to ensure action feasibility. [10, 11] proposed a Lyapunov-based approach that provides an effective way to guarantee global safety during training via a set of local, linear constraints. [15] extended PILCO [17] model based algorithm to enable active exploration using a metric for out-of-sample Gaussian Process that supports conditional-value-at-risk constraints.

The cross-entropy method (CEM) is a zeroth-order optimizer, which works by generating a sequence of samples from the objective function [37]. In recent works, CEM has shown state-of-the-art performance for solving a control optimization problem with neural network transition dynamics [13, 22]. Recently, [6] proposed a method to approximate the derivative through an unconstrained, non-convex, and continuous optimization process. The differentiable cross-entropy method allows us to embed action sequences in a lower-dimensional space. It induces a differentiable control policy that solves the objective mismatch problem in model-based control. [27, 45] proposed constrained cross-entropy methods that only select elite trajectories that satisfy constraint satisfaction criteria.

In this paper, we propose a constrained differentiable cross-entropy method (CDCEM) that effectively solves large safety-critical optimization problems in lower-dimensional latent space. In contrast to previous safe model-based algorithms, our algorithm induces a differentiable policy that can address objective mismatch problem by using the gradient information from a policy function and fine-tune controller components such as transition model or the cost model. In addition, backpropagating across all sampled trajectories is memory intensive and intractable in most practical problems; hence, this is only possible in lower-dimensional embedded action space [6].

3 PRELIMINARIES

3.1 Constrained Markov Decision Process

A Markov decision process (MDP) is defined as $(\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P}, \mu)$ where $\mathcal{S}$ is set of states, $\mathcal{A}$ is a set of actions, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ is the reward function, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \to \mathcal{D}(\mathcal{S})$ is the transition function and $\mu \in \mathcal{D}(\mathcal{S})$ is an initial state distribution. Let $\Pi : \mathcal{S} \to \mathcal{D}(\mathcal{S})$ be set of all stationary policies.

The objective of reinforcement learning is to select a policy that maximizes the discounted expected return

$$J(\pi) = \mathbb{E}_{\tau\sim\pi}[\sum_{t=1}^{H} \gamma^t R(s_t, a_t, s_{t+1})]$$

where $\gamma \in [0, 1)$ is the discount factor. Given a finite horizon H, a H-step trajectory is sequence of H state-action pairs. τ represents a trajectory ($\tau = s_1, a_1, \ldots, s_H, a_H$) and $\tau \sim \pi$ is distribution over trajectories.

A constrained Markov decision process (CMDP) [4] is an MDP with constraints that restrict the set of allowable policies over that MDP. The set of cost functions $C_i : S \times A \times S \to \mathbb{R}$ mapping transition tuple to real valued cost and limits $d_1, \ldots, d_n$. The expected discounted return $J_{C_i}(\pi) = \mathbb{E}_{\tau\sim\pi}[\sum_{t=1}^{H} \gamma^t C_i(s_t, a_t, s_{t+1})]$ with respect to cost function C_i. The set of feasible stationary policies for CMDP is then:

$$\Pi_C = \pi \in \Pi : \forall i, J_{C_i}(\pi) \leq d_i$$

and the solution to CMDP is:

$$\pi^* = \operatorname*{argmax}_{\pi\in\Pi_C} J(\pi)$$

3.2 Differentiable Constrained Cross-Entropy Method

The cross-entropy method (CEM) [37] is a zeroth-order optimization approach in the form of $\hat{x} := \text{argmin}_x f_\theta(x)$. CEM is an iterative solver which uses a sequence sampling distributions $g_\phi \in \mathbb{R}^n$. In each iteration N candidate points are sampled from the domain $[X_{t,i}]_{i=1}^N \sim g_{\phi_t}(\cdot)$, evaluated using $v_{t,i} := f_\theta(X_{t,i})$ and k elite candidates are selected to fit the new sampling distribution by solving the maximum-likelihood problem:

$$\phi_{t+1} := \underset{\phi}{\text{argmax}} \sum_i \mathbb{1}\{v_{t,i} \leq \pi(v_t)_k\} \log g_\phi(X_{t,i}) \tag{1}$$

The top-k operation in Equation (1) makes the $\hat{x}$ non-differentiable with respect to θ. The top-k operation can be made differentiable using a Multi-Label Projection (LML) layer [5].

4 APPROACH

4.1 Problem Formulation

Here we use Model Predictive Control (MPC) for our model-based RL approach for controlling discrete-time dynamical systems with continuous action-space, which allows our agent to adapt its plan based on new observations.

Let $\mathcal{A}^H$ be the space of control sequences over controller horizon length H. The goal is to learn a latent action space $\mathcal{Z}$ with parameterized decoder $f_\theta^{dec} : \mathcal{Z} \rightarrow \mathcal{A}^H$. For a special case of Constrained Markov Decision Process we aim to repeatedly solve the following optimization problem:

$$\begin{aligned} \hat{z} := J_\theta(z; s_{init}) := \underset{z \in \mathcal{Z}}{\text{argmax}} \sum_{t=1}^{H} J_\theta(a_t; s_t) \\ \text{subject to } s_1 = s_{init} \\ s_{t+1} = f_\theta^{trans}(s_t, a_t) \\ a_{1:H} = f_\theta^{dec}(z) \\ c(s_{t+1}) = 0 \end{aligned} \tag{2}$$

where s_{init} is the initial system state governed by deterministic system transition dynamics f^{trans} and $c(s_t)$ is a constraint violation cost function. The goal is to find a valid trajectory $s_{1:H}, a_{1:H}$ that optimizes the cost J_θ while adhering to the $c(s_t)$ constraint. In a receding horizon control setting [29] we only use the first action a_1 in the real system.

Here we adopt the model-based RL PETS [13] that uses an ensemble of models with trajectory sampling (TS) to estimate the epistemic uncertainty of the input data. Using an ensemble of B neural networks parametrized with θ_b, we train the models by minimizing the mean squared error (MSE) of loss function $\mathcal{L}(\theta) = \mathbb{E}_{(s_t, a_t, s_{t+1} \in \mathcal{D}_b)} \|s_{t+1} - f_{\theta_b}(s_t, a_t)\|$. Algorithm 1 describe the training pipeline for our MPC controller. The constraint violation cost function $c(s_t)$, and the reward function $r(s_t)$ can be learned from data using any classification model, or using a known cost function.

Algorithm 1 Model-based MPC with CDCEM

Require: Initial collected trajectories $\mathcal{D}$, dynamics models, reward model, action sequence decoder, CDCEM parameters; Initialize dataset $\mathcal{D}$ with S random seed episodes;
while Not converged **do**
  for $t = 1, \ldots, T$ **do**
    $a_t \leftarrow$ CDCEM-solve$(h_{s_{t-1}})$
    $\{r_t, c_t, s_{t+1}\} \leftarrow env.step(a_t)$
    Add $\{r_t, s_t, a_t\}$ to $\mathcal{D}$
    if t mod update-interval = 0 **then**
      sample trajectories $\tau = [r_\tau, s_\tau, a_\tau]_{\tau=1}^H \sim \mathcal{D}$ from the dataset.
      Compute the loss: $\mathcal{L}(\tau, \hat{s}_\tau)$
      $\theta_{trans} \leftarrow$ grad-update$(\nabla_\theta \mathcal{L}(\tau, \hat{s}_\tau))$
      $\hat{z}_\tau \leftarrow \text{argmax}_{z \in \mathcal{Z}} J_\theta(z; \hat{s}_\tau)$
      $\theta_{dec} \leftarrow$ grad-update$(\nabla_\theta \sum_\tau J_\theta(\hat{z}_\tau))$
    end if
  end for
end while

4.2 Constrained Differentiable Cross-Entropy Algorithm

In order to solve the constrained optimization problem in Equation (2) we use constrained differentiable cross-entropy method (CDCEM) described in Algorithm 2. Here we use Multi-Label Projection (LML) layer [5] described in Equation (4) which allows us to implement differentiable top-k operation to select top trajectories based on the task cost function and feasibility cost. We can combine the two top-k operations with the weighted sum of reward and constraint cost for each each using a linear opinion pool [14]. This provides a belief aggregation method that combines the decision based on cost and reward objective which in the simplest case involves taking the weighted linear average of opinions:

$$\mathcal{I}_{combined,j} = \alpha \mathcal{I}_{1,j} + (1 - \alpha)\mathcal{I}_{2,j} \tag{3}$$

The α denotes the weight associated with the reward objective and $1 - \alpha$ is the weight associated with the cost objective respectively. The weighting parameters can be set as a hyperparameter or estimated during training,

$$\begin{aligned} \Pi_{\mathcal{L}_k(\frac{x}{\kappa})} := \text{argmin} - x^T y - \kappa U_b(y) \\ \text{subject to: } 1^T y = k, \\ 0 < y < 1, \end{aligned} \tag{4}$$

where U is binary cross-entropy function and κ is a hyperparameter that will induce vanilla CEM when $\kappa \rightarrow 0$. The derivative of Equation (4) can be computed by implicitly differentiating the KKT optimality conditions [5].

5 EXPERIMENTS

5.1 Experiment 1: Point Goal Environment

5.1.1 Problem Description. First, we evaluate our proposed safe reinforcement learning algorithm in the OpenAI Safety Gym [34]. We use Safety Gym because (1) it uses an auxiliary cost function to enforce safety requirements, and (2) state-of-the-art reinforcement

Algorithm 2 Constrained DCEM (CDCEM) $(r, c, g_\phi; \kappa, N, k, M)$

for $j = 1$ to M **do**
  $[X_{j,i}]_{i=1}^{N} \sim g_{\phi_j}(\cdot)$ ▹ Sample N points from the domain. Differentiate with reparameterization.
  $v_{j,i}^{reward} = r(X_{j,i})$ ▹ Evaluate the reward objective function at those points.
  $v_{j,i}^{safety} = c(X_{j,i})$ ▹ Evaluate the constraint objective function at those point.
  $\mathcal{I}_{1,j} = \Pi_{\mathcal{L}_k}(\frac{v_{j,i}^{reward}}{\kappa})$ ▹ Compute the soft top-k projection for reward objective.
  $\mathcal{I}_{2,j} = \Pi_{\mathcal{L}_k}(\frac{v_{j,i}^{safety}}{\kappa})$ ▹ Compute the soft top-k projection for constraint feasibility.
  $\mathcal{I}_{combined,j} = \alpha\mathcal{I}_{1,j} + (1-\alpha)\mathcal{I}_{2,j}$ ▹ Compute the soft top-k combining prediction $\mathcal{I}_{1,j}$ and $\mathcal{I}_{2,j}$.
  Update ϕ_{j+1} by solving the problem in Equation (1).
end for
Return: $\mathbb{E}[g_{\phi_{M+1}}(\cdot)]$

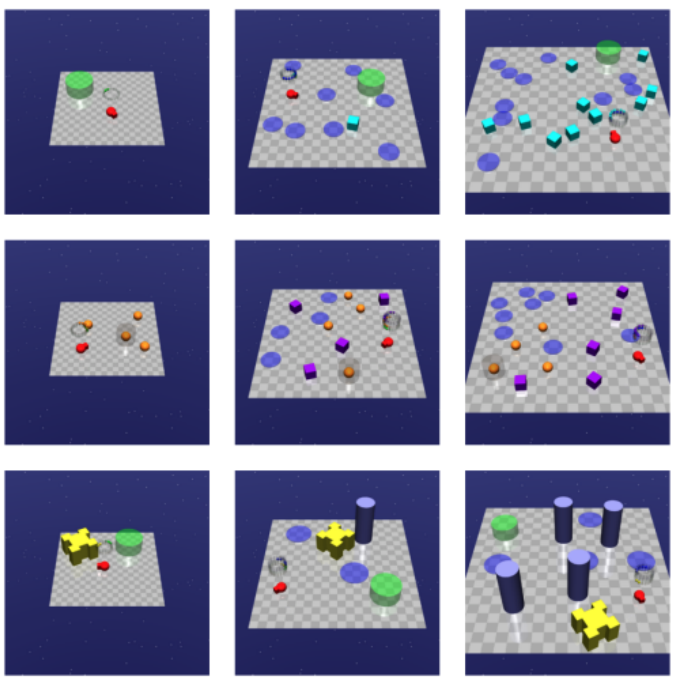


Figure 1: Point-Goal: Safety Gym environments for experiment 1.

learning algorithms with benchmarked performance are available in all environments. The Point-Goal Task (in Figure 1) requires the robot to navigate to the designated green point with two actuators for thrust and angle while avoiding hazards and vase.

The robot will receive a reward ($r_t = 1$) when it reaches the goal and a cost ($c_t(s_t) = 1$) when it violates the safety requirement. Here we use the available official baseline methods provided in the Safety Gym Environment, which are Constrained Policy Optimization (CPO) [2] as a constrained reinforcement learning baseline and cross-entropy based Model-Predictive Control (MPC-CEM) as a model-based unconstrained baseline. We follow the metrics proposed in the Safety Gym paper [34] which are episodic reward and episodic cost, and the number of samples required to reach convergence as a proxy for sample efficiency.

5.1.2 Implementation Details. We use the same hyperparameters provided in the Safety Gym official benchmark for the CPO and the same hyperparameters for both model-based (MPC-CDCEM and MPC-CEM). We evaluate each algorithm with three different seeds. For the dynamics models, we use a neural network with three hidden layers with 64 neurons, ReLU activation, 512 batch size, and the Adam optimizer with $1e^{-1}$ learning rate. We train the model for 50 epochs. We use a smaller neural network with two hidden layers and 128 neurons, ReLU activation, and Adam optimizer with a learning rate of $1e^{-3}$ to predict the constraint violation given. For MPC-CDCEM, we use a neural network as a decoder to map embedded action from the latent planning horizon to a larger planning horizon. For the decoder, we use a neural network with two hidden layers and 256 neurons, Swish [33] activation, and the Adam optimizer with $1e^{-4}$ learning rate.

5.1.3 Results. Figure 2 and Figure 3 show the accumulated reward and cost violation for the safety-gym point-goal task. The CPO algorithm's learning curve and violation cost are shown with a horizontal line since the model-free algorithm requires an order of magnitude more interaction (50 times) with the environment. It can be seen that our proposed algorithm converges to a slightly lower reward, but receives a significantly lower violation cost.

Table 1 compares the number of constraint violations during the first 5×10^3 iterations. It can be seen that the number of violations is significantly higher in the model-free case. Compared to the two model-based approaches, Table 1 demonstrates that the MPC-CDCEM incurs a lower cost while exploring the environment safely.

We further evaluate the effect of hyperparameter α in the MPC-CDCEM fused cost function on constraint violation. Figure 4 shows that constraint violation decreases sharply when α increases from 0 to 0.4. While a further increase in α reduces the number of violations, it negatively influences the obtained reward.

Table 1: Constraint violations in 5000 iteration

Algorithm	Constraint Violations	SD
MPC-CEM	56.21	3.52
MPC-CDCEM	29.75	1.21
CPO	812.4	12.2

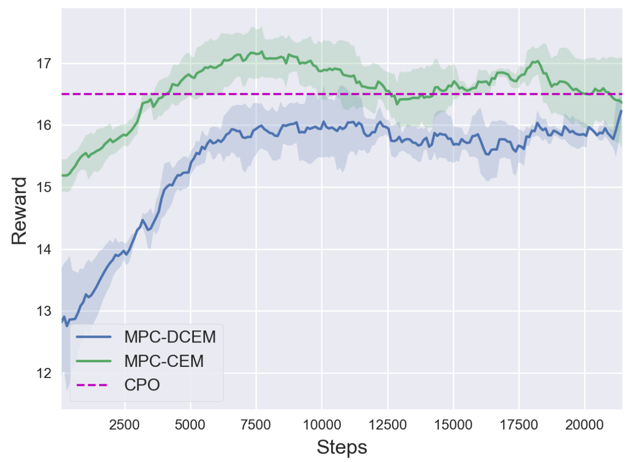


Figure 2: SafePoint-Goal task learning curves

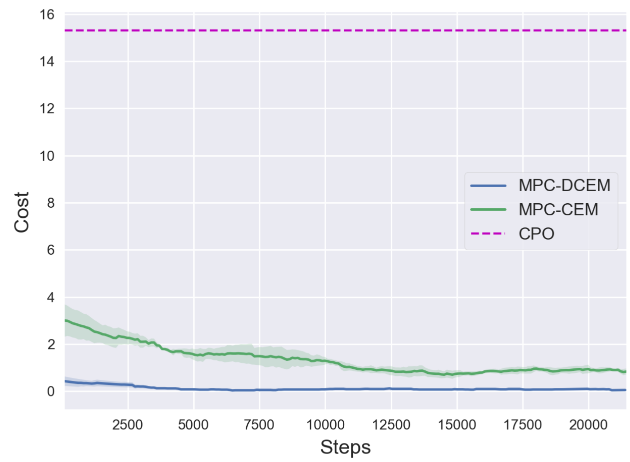


Figure 3: Constraint violation cost

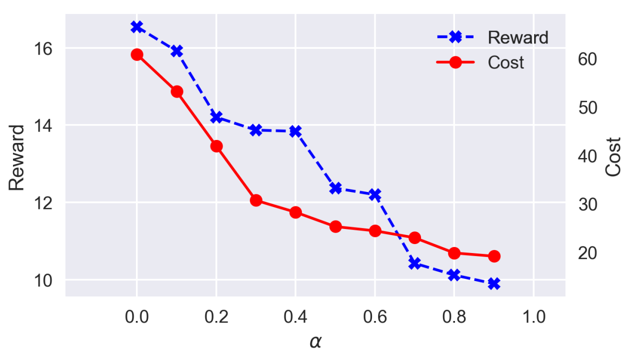


Figure 4: α in fused cost function

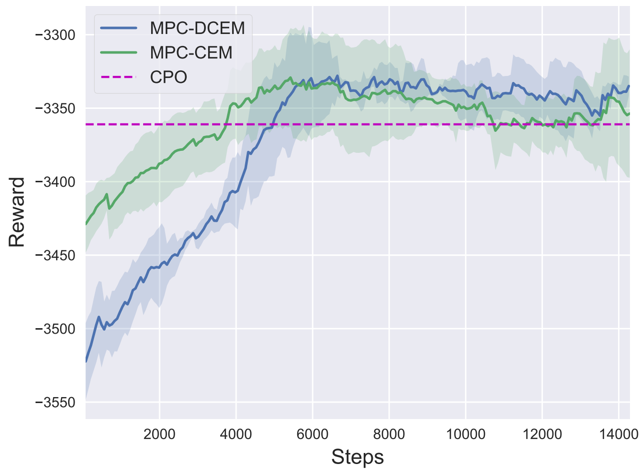
5.2 Experiment 2: Microgrid Energy Management System

5.2.1 Problem Description. One of the most critical constraints for utilizing building thermal mass and energy flexibility in a microgrid is maintaining a satisfactory occupant comfort level while minimizing energy consumption. A common approach to ensure policy feasibility is to penalize the violations of thermal comfort, but this does not guarantee the occupant's comfort requirements. In addition, it is also necessary to ensure the control strategies do not violate physical operating constraints of the equipment or cause premature equipment degradation. For example it is often desirable to slowly ramp large fan motors to avoid large pressure fluctiations in the duct systems, and it is also often desired to limit the cycling of large equipment like chillers.

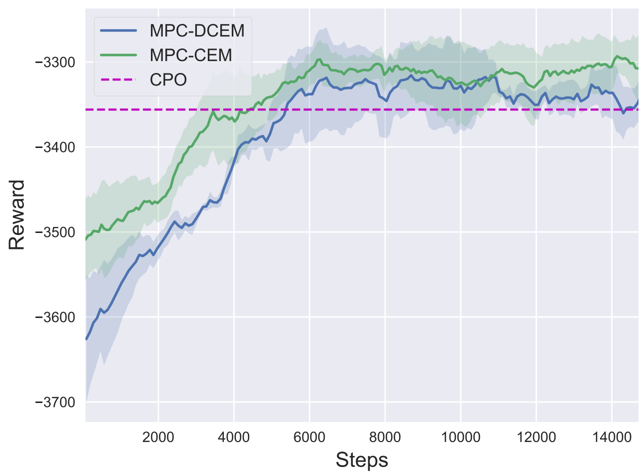
Here we evaluate our safe reinforcement learning algorithm on a building-level microgrid energy management test-bed. The simulation environment is implemented using the EnergyPlus model for a large office building, PV system, wind turbine, inverters, and a battery storage facility connected to the main grid. The additional electricity can be bought from the grid if the renewable energy and battery storage cannot meet the demand. The building model used here is a large commercial office building with a Chicago weather file. Cooling is provided to the building zones by chilled-water variable-air-volume (VAV) air-handlers and cooling-only terminal boxes. Zone heating is performed by electric resistance baseboard heaters. The central plant features two centrifugal chillers, two cooling towers, and water pumps. In this experiment, we control the zone cooling set-points to maintain the zone temperature to ensure occupants' comfort while minimizing the electricity consumption in the microgrid. The constraints here are to maintain a satisfactory comfort level as measured by the zone Predictive Mean Vote (PMV) index during the occupied hours and prevent excessive switching of large chilled water plant equipment. PMV index values range from −3 to +3, which describes the feeling from cold to hot, respectively. Based on ASHRAE 55 the agent will violate comfort constraints if PMV is outside the recommended limits (−0.5 and 0.5). The objective is described in Eq. (5).

$$\begin{aligned} \max J &= -\lambda_1 E_{hvac,t} - \lambda_2 \|u_t\|_1 \\ \text{s.t. } & |PMV_t| \leq 0.5, \quad \forall t \in \{\text{occupied hours}\} \\ & \text{chiller-cycles} \leq 2\text{per day} \end{aligned} \tag{5}$$

5.2.2 Implementation Details. We use the same hyperparameter for both baselines and MPC-CDCEM over the weather file from May 2018 to September 2018. The EnergyPlus model uses a 10-min control time step with a planning horizon of $H = 24$. The same hyperparameters are used for the dynamics model, constraint cost prediction model, and the decoder neural network, as mentioned in experiment 1. The decoder maps the latent horizon length of $H_l = 4$ to the task horizon $H = 24$. We train the induced MPC policy by iterating over-collected samples for 20 epochs with a batch size of 256. We follow the metrics proposed in the Safety Gym paper [34] which are episodic reward and episodic cost, and the number of samples required to reach convergence as a proxy for sample efficiency. We use the same hyperparameters provided in the Safety

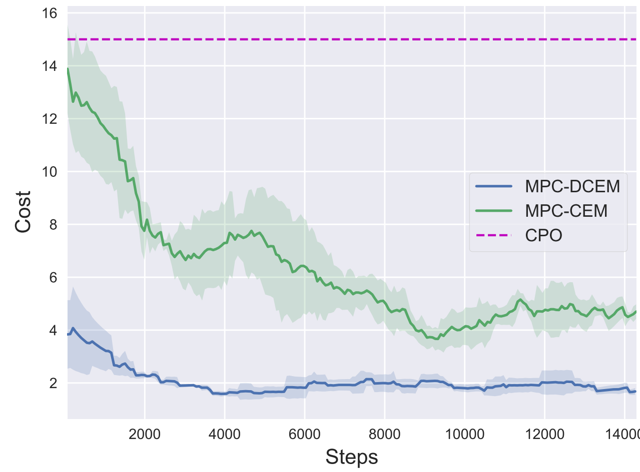


(a) Thermal comfort experiment

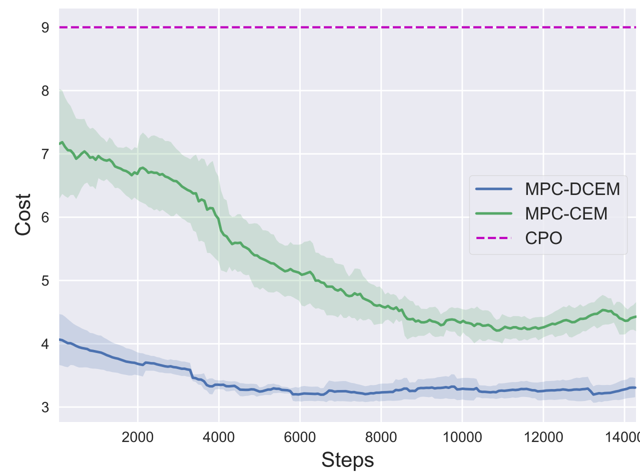


(b) Chiller cycle experiment

Figure 5: Learning curves for thermal comfort and chiller cycling constraint experiments.



(a) Thermal comfort experiment



(b) Chiller cycle experiment

Figure 6: Cost trend for thermal comfort and chiller cycling constraint experiments.

Gym official benchmark for the CPO and the same hyperparameters for both model-based MPC (MPC-CDCEM and MPC-CEM).

5.2.3 Results. Figure 5 and 6 show the learning curve and reward and constraint violation for the occupancy comfort and chiller cycle constraint experiments. The figures show MPC-CDCEM quickly learns the underlying constraint function and converges to a reward that is inline or slightly lower than MPC-CEM and CPO. This observation is reasonable since the agent can ignore the constraint and maximize the task reward.

After training the agent, the trained agent directly operated in a simulation environment with the Washington D.C. weather sequence on the first week of June. Table 2 and Table 3 highlight the HVAC electric use, reward, and constraint violation during a weekday in the testing period. Compared to other methods, the reward and constraint violations show that MPC-CDCEM achieves rewards comparable to MPC-CEM and CPO while constraint violations are always less than other methods. The proposed MPC-CDCEM agent (average over three random seeds) saves 12.3% energy compared to the default nighttime setup (NSU) and consumes approximately 1% more compared to MPC-CEM.

Figures 7 and 8 show the simulation results for a weekday during the testing period in order to compare the NSU, MPC-CDCEM, MPC-CEM, and CPO indoor thermal comfort and zone temperatures. The zone temperature represents the weighted average zone temperature based on zone area, and the PMV is the occupancy weighted average PMV. It can be seen that MPC-CDCEM is trying to maintain the zone temperature at a temperature that does not violate the comfort constraint. MPC-CEM and CPO are more unstable than MPC-CDCEM, which resulted in violations of the

Table 2: Testing period result for thermal comfort experiment.

Case	Energy Use [kWh]	Reward	Constraint Violation
NSU	31,320	-4,186	0
MPC-CEM	27,151	-3,811	3
MPC-CDCEM	27,429	-3,804	0
CPO	27,109	-3,834	4

Table 3: Testing period result for chiller cycling experiment.

Case	Energy Use [kWh]	Reward	Constraint Violation	
			Chiller	Comfort
NSU	31,320	-4,186	6	0
MPC-CEM	27,285	-3,912	8	0
MPC-CDCEM	27,394	-3,888	4	0
CPO	27,147	-3,907	7	2

comfort constraints in the morning and afternoon. In the chiller cycle experiment, the MPC-CDCEM agent learns to maintain a lower temperature in the morning, possibly preventing excessive switching of chillers during the day. This confirms that the proposed algorithm observes the constraints during policy learning.

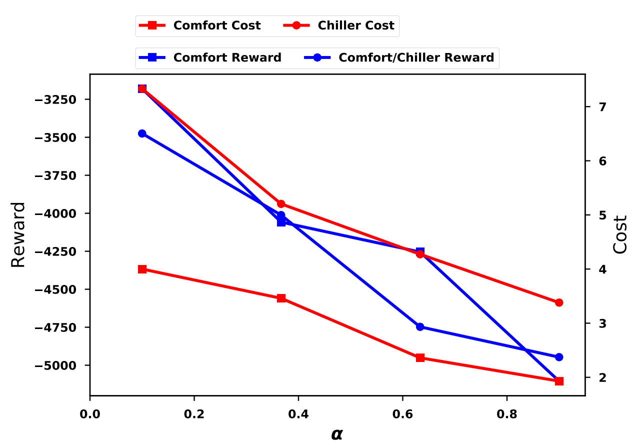


Figure 9: α in fused cost function.

We further evaluated the effect of fused cost function parameter α as show in Figure 9, which shows that a good balance between reward and cost constraint tends result in good compromise between cost and safety.

6 CONCLUSION

In this study, we present an effective constrained RL algorithm formulated under the Constrained Markov Decision Process framework with no additional assumption on the system dynamics. The proposed algorithm induces a differentiable control policy that addresses the objective mismatch problem and enables an end-to-end learning process while enforcing constraint feasibility. First, we evaluated our algorithm in the Safety Gym environment, which showed superior constraint satisfaction while maintaining task performance compared to other constrained RL algorithms. Next, we evaluated MPC-CDCEM in a microgrid environment to minimize energy consumption while ensuring occupants' thermal comfort and preventing excessive chiller cycles. In both cases, MPC-CDCEM achieved better constraint satisfaction while maintaining good reward performance compared to other baseline algorithms.

REFERENCES

[1] Pieter Abbeel, Adam Coates, and Andrew Y Ng. 2010. Autonomous helicopter aerobatics through apprenticeship learning. *The International Journal of Robotics Research* 29, 13 (2010), 1608–1639.
[2] Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. 2017. Constrained policy optimization. In *International Conference on Machine Learning*. PMLR, 22–31.
[3] Eitan Altman. 1998. Constrained Markov decision processes with total cost criteria: Lagrangian approach and dual linear program. *Mathematical methods of operations research* 48, 3 (1998), 387–417.
[4] Eitan Altman. 1999. *Constrained Markov decision processes*. Vol. 7. CRC Press.
[5] Brandon Amos, Vladlen Koltun, and J Zico Kolter. 2019. The limited multi-label projection layer. *arXiv preprint arXiv:1906.08707* (2019).
[6] Brandon Amos and Denis Yarats. 2020. The differentiable cross-entropy method. In *International Conference on Machine Learning*. PMLR, 291–302.
[7] Arnab Basu, Tirthankar Bhattacharyya, and Vivek S Borkar. 2008. A learning algorithm for risk-sensitive cost. *Mathematics of operations research* 33, 4 (2008), 880–898.
[8] Vivek S Borkar. 2001. A sensitivity formula for risk-sensitive cost and the actor–critic algorithm. *Systems & Control Letters* 44, 5 (2001), 339–346.
[9] Bingqing Chen, Priya Donti, Kyri Baker, J Zico Kolter, and Mario Berges. 2021. Enforcing Policy Feasibility Constraints through Differentiable Projection for Energy Optimization. *arXiv preprint arXiv:2105.08881* (2021).
[10] Yinlam Chow, Ofir Nachum, Edgar Duenez-Guzman, and Mohammad Ghavamzadeh. 2018. A lyapunov-based approach to safe reinforcement learning. *arXiv preprint arXiv:1805.07708* (2018).
[11] Yinlam Chow, Ofir Nachum, Aleksandra Faust, Edgar Duenez-Guzman, and Mohammad Ghavamzadeh. 2019. Lyapunov-based safe policy optimization for continuous control. *arXiv preprint arXiv:1901.10031* (2019).
[12] Paul Christiano, Zain Shah, Igor Mordatch, Jonas Schneider, Trevor Blackwell, Joshua Tobin, Pieter Abbeel, and Wojciech Zaremba. 2016. Transfer from simulation to real world through learning deep inverse dynamics model. *arXiv preprint arXiv:1610.03518* (2016).
[13] Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. 2018. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *arXiv preprint arXiv:1805.12114* (2018).
[14] Roger Cooke et al. 1991. *Experts in uncertainty: opinion and subjective probability in science*. Oxford University Press on Demand.
[15] Alexander I Cowen-Rivers, Daniel Palenicek, Vincent Moens, Mohammed Abdullah, Aivar Sootla, Jun Wang, and Haitham Ammar. 2020. Samba: Safe model-based & active reinforcement learning. *arXiv preprint arXiv:2006.09436* (2020).
[16] Gal Dalal, Krishnamurthy Dvijotham, Matej Vecerik, Todd Hester, Cosmin Paduraru, and Yuval Tassa. 2018. Safe exploration in continuous action spaces. *arXiv preprint arXiv:1801.08757* (2018).
[17] Marc Deisenroth and Carl E Rasmussen. 2011. PILCO: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on machine learning (ICML-11)*. Citeseer, 465–472.
[18] Marc Peter Deisenroth, Dieter Fox, and Carl Edward Rasmussen. 2013. Gaussian processes for data-efficient learning in robotics and control. *IEEE transactions on pattern analysis and machine intelligence* 37, 2 (2013), 408–423.
[19] Kurt Driessens and Sašo Džeroski. 2004. Integrating guidance into relational reinforcement learning. *Machine Learning* 57, 3 (2004), 271–304.
[20] Javier Garcıa and Fernando Fernández. 2015. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research* 16, 1 (2015), 1437–1480.
[21] Clement Gehring and Doina Precup. 2013. Smart exploration in reinforcement learning using absolute temporal difference errors. In *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*. 1037–1044.
[22] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. 2019. Learning latent dynamics for planning from

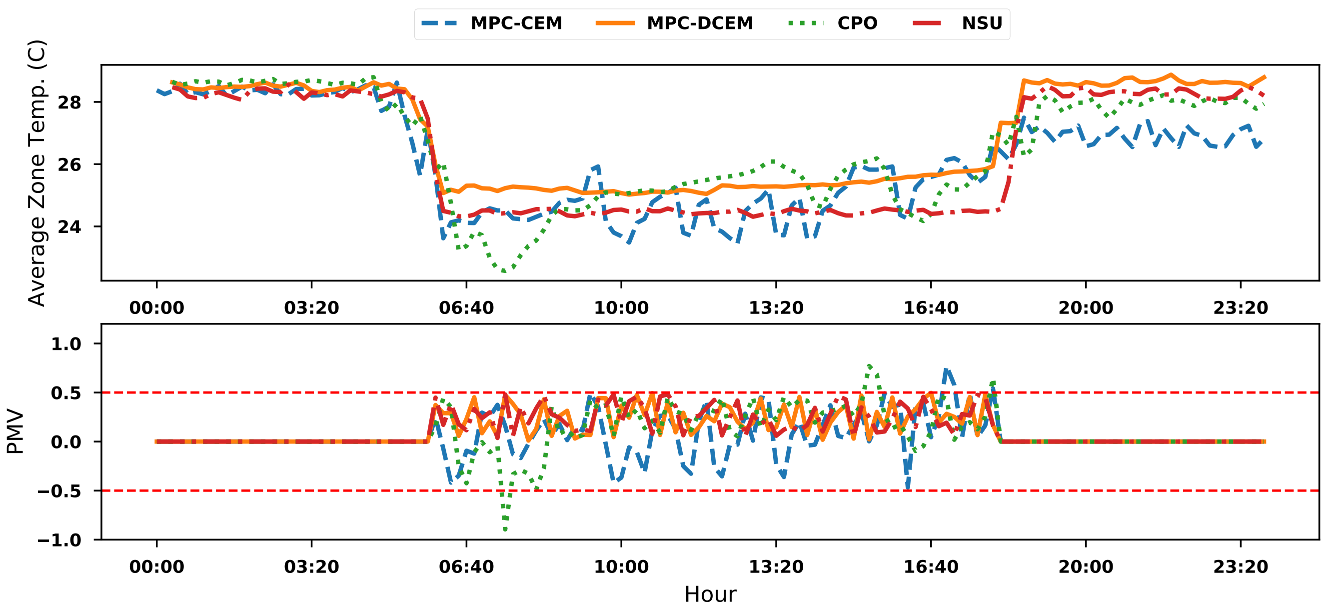


Figure 7: Performance evaluation for thermal comfort experiment.

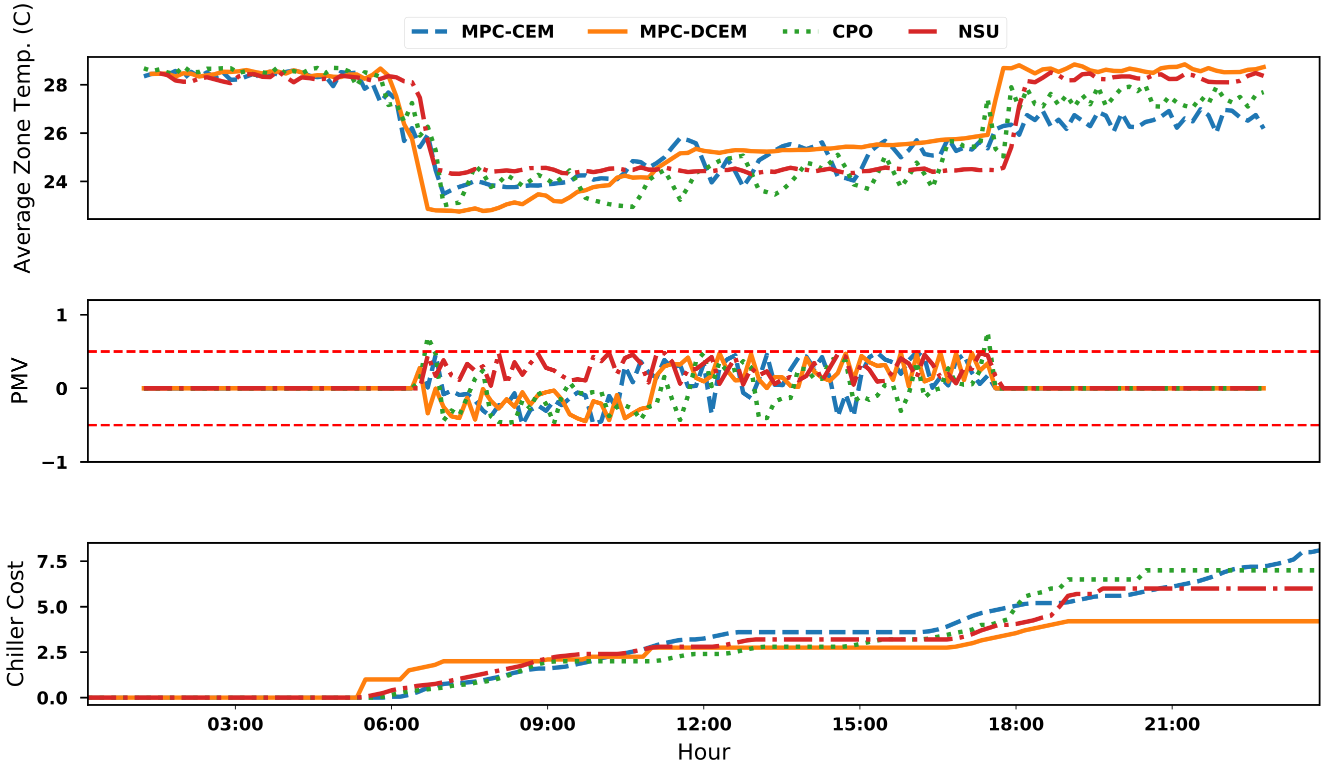


Figure 8: Performance evaluation for chiller cycle experiment.

pixels. In *International Conference on Machine Learning*. PMLR, 2555–2565.

[23] Ronald A Howard and James E Matheson. 1972. Risk-sensitive Markov decision processes. *Management science* 18, 7 (1972), 356–369.

[24] Yoshinobu Kadota, Masami Kurano, and Masami Yasuda. 2006. Discounted Markov decision processes with utility constraints. *Computers & Mathematics with Applications* 51, 2 (2006), 279–284.

[25] Nathan Lambert, Brandon Amos, Omry Yadan, and Roberto Calandra. 2020. Objective mismatch in model-based reinforcement learning. *arXiv preprint arXiv:2002.04523* (2020).

[26] Edith LM Law. 2005. Risk-directed exploration in reinforcement learning. (2005).

[27] Zuxin Liu, Hongyi Zhou, Baiming Chen, Sicheng Zhong, Martial Hebert, and Ding Zhao. 2020. Constrained Model-based Reinforcement Learning with Robust Cross-Entropy Method. *arXiv preprint arXiv:2010.07968* (2020).

[28] Helmut Mausser. 1998. Beyond VaR: From measuring risk to managing risk. *ALGO research quarterly* 1, 2 (1998), 5–20.

[29] David Q Mayne and Hannah Michalska. 1988. Receding horizon control of nonlinear systems. In *Proceedings of the 27th IEEE Conference on Decision and Control*. IEEE, 464–465.

[30] Teodor Mihai Moldovan and Pieter Abbeel. 2012. Risk Aversion in Markov Decision Processes via Near Optimal Chernoff Bounds.. In *NIPS*. 3140–3148.

[31] Tetsuro Morimura, Masashi Sugiyama, Hisashi Kashima, Hirotaka Hachiya, and Toshiyuki Tanaka. 2010. Nonparametric return distribution approximation for reinforcement learning. In *ICML*.

[32] Arnab Nilim and Laurent El Ghaoui. 2005. Robust control of Markov decision processes with uncertain transition matrices. *Operations Research* 53, 5 (2005), 780–798.

[33] Prajit Ramachandran, Barret Zoph, and Quoc V Le. 2017. Searching for activation functions. *arXiv preprint arXiv:1710.05941* (2017).

[34] Alex Ray, Joshua Achiam, and Dario Amodei. 2019. Benchmarking safe exploration in deep reinforcement learning. *arXiv preprint arXiv:1910.01708* 7 (2019).

[35] Benjamin Recht. 2019. A tour of reinforcement learning: The view from continuous control. *Annual Review of Control, Robotics, and Autonomous Systems* 2 (2019), 253–279.

[36] Michael T Rosenstein, Andrew G Barto, Jennie Si, Andy Barto, Warren Powell, and Donald Wunsch. 2004. Supervised actor-critic reinforcement learning. *Learning and Approximate Dynamic Programming: Scaling Up to the Real World* (2004), 359–380.

[37] Reuven Y Rubinstein and Dirk P Kroese. 2013. *The cross-entropy method: a unified approach to combinatorial optimization, Monte-Carlo simulation and machine learning*. Springer Science & Business Media.

[38] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. 2016. Mastering the game of Go with deep neural networks and tree search. *nature* 529, 7587 (2016), 484–489.

[39] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. 2017. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815* (2017).

[40] Adam Stooke, Joshua Achiam, and Pieter Abbeel. 2020. Responsive safety in reinforcement learning by pid lagrangian methods. In *International Conference on Machine Learning*. PMLR, 9133–9143.

[41] Aviv Tamar, Huan Xu, and Shie Mannor. 2013. Scaling up robust MDPs by reinforcement learning. *arXiv preprint arXiv:1306.6189* (2013).

[42] Jie Tang, Arjun Singh, Nimbus Goehausen, and Pieter Abbeel. 2010. Parameterized maneuver learning for autonomous helicopter flight. In *2010 IEEE International Conference on Robotics and Automation*. IEEE, 1142–1148.

[43] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. 2017. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 23–30.

[44] Eiji Uchibe and Kenji Doya. 2007. Constrained reinforcement learning from intrinsic and extrinsic rewards. In *2007 IEEE 6th International Conference on Development and Learning*. IEEE, 163–168.

[45] Min Wen and Ufuk Topcu. 2020. Constrained cross-entropy method for safe reinforcement learning. *IEEE Trans. Automat. Control* (2020).

[46] Zhaocong Yuan, Adam W Hall, Siqi Zhou, Lukas Brunke, Melissa Greeff, Jacopo Panerati, and Angela P Schoellig. 2021. safe-control-gym: a Unified Benchmark Suite for Safe Learning-based Control and Reinforcement Learning. *arXiv preprint arXiv:2109.06325* (2021).