Dual Variable Actor-Critic for Adaptive Safe Reinforcement Learning

Junseo Lee[1], Jaeseok Heo[2], Dohyeong Kim[2], Gunmin Lee[2], and Songhwai Oh[1,2]

***Abstract*— Satisfying safety constraints in reinforcement learning (RL) is an important issue, especially in real-world applications. Many studies have approached safe RL with the Lagrangian method, which introduces dual variables. However, applying a trained policy with the optimal dual variable to a new environment can be hazardous since the optimal value of the dual variable, which represents a level of safety, depends on the environmental setting. To this end, we propose a new framework, dual variable actor-critic (DVAC), that solves the safe RL problem by simultaneously training a single policy over different safety levels. We introduce a universal policy and universal Q-function, which have a dual variable as an argument. Then, we extend the soft actor-critic so that the universal policy is guaranteed to converge to the Pareto optimal policy sets. We evaluate the proposed method in simulation and real-world environments. The universal policy learned with the proposed method ranges from extremely safe to high performance according to the dual variables, and is nearly Pareto optimal compared to policies learned with the baseline methods. In addition, the agent is able to adapt to environments with unseen state distributions without additional training by identifying a suitable dual variable using the proposed method.**

I. INTRODUCTION

Reinforcement learning (RL) has been successful in various applications, including Atari games [1], [2], [3] and robot manipulation [4], [5], [6]. However, transferring an RL policy trained in simulations to real-world environments is difficult due to the lack of safety satisfaction, such as safe human-robot interaction, power consumption, and collision avoidance. To address this challenge, we can extend the RL problem to a constrained problem with explicit safety constraints, called *safe reinforcement learning*. A constrained Markov decision process (CMDP) [7] formulates a safe RL problem, enforcing to maximize the sum of rewards while the expected sum of discounted costs is below a given threshold.

To solve the safe RL problem, the Lagrangian approach [8], a classical method for solving constrained optimization problems, has been widely used [9], [10]. This approach transforms a CMDP problem into an unconstrained one by introducing dual variables [9], [10] , followed by a alternative update of policies and these variables via primal-dual optimization. However, the Lagrangian methods have two critical issues. First, the optimal policies and dual variables depend on environmental settings, such as the initial state distribution and obstacle density, making them not adaptable to other environments; it can be considered not robust to environmental changes. For example, a navigation policy optimized for rural areas may not be safe in crowded urban environments. Second, policies can only be trained for a single safety level at once. Generally, a value of the constraint threshold corresponding to a desired safety level is not known in advance. If a trained policy is dangerous or overly cautious, we need to retrain a policy with a different threshold to match the desired safety level, which is burdensome.

To address these issues, we propose a dual variable actor-critic (DVAC) framework for efficiently solving the safe RL problem over a wide range of constraint levels. In the proposed framework, we introduce a *universal policy* and *universal Q-function*, a novel policy and Q-function that take a dual variable as input. Since each dual variable corresponds to a different level of safety, we can generate policies with various safety levels after training by feeding proper dual variable inputs to the trained universal policy. To train the policy and Q-function, we propose universal policy evaluation and improvement. Through this iteration, we theoretically confirm that the universal policy, evaluated with a uniform convergence guarantee, is updated and converges to a Pareto optimal one. Practically, we parameterize universal policy and universal Q-function with a single network. In addition, we provide a search method to find a suitable dual variable in a new unseen environment using binary search, estimating the cost return with the universal cost-Q-function. The search method can efficiently find a policy that matches a user-specified safety level.

We evaluate DVAC for safe navigation tasks in simulation and real-world environments. The experimental results show that the universal policy trained by the proposed method can adapt to a wide range of safety levels by adjusting the dual-variable input. A policy set generated by applying diverse dual-variable values to the universal policy forms a nearly Pareto optimal frontier, which is as safe as, or safer than, safe RL baselines. The proposed dual variable-search method produces a policy with a 9% lower cost sum in safety-critical crowded environments and a 6% higher reward sum in sparse environments, where safety is not as critical, compared to a naive approach. In the Jackal real robot experiment, DVAC achieves a reward sum, which is compatible to other safe RL baselines, while keeping its cost sum below 21% of other methods. The main contributions are threefold:

- We propose a new framework to solve safe RL that learns a universal policy over various degrees of safety constraints. To the best of our knowledge, this is the first work that can adapt to different constraint thresholds or environments after training using the universal policy.
- We prove that the universal policy converges to the Pareto optimal policies through the proposed universal policy iteration.
- We evaluates DVAC in a number of experiments both in simulation and real-world environments. Empirical experiments demonstrate the effectiveness of the proposed method for various safety levels.

[1]Graduate School of Artificial Intelligence (GSAI) and ASRI, Seoul National University, Seoul, Korea (e-mail: junseo.lee@rllab.snu.ac.kr, songhwai@snu.ac.kr), [2]Department of Electrical and Computer Enginnering and ASRI, Seoul National University, Seoul, Korea (e-mail: jaeseok.heo, dohyeong.kim, gunmin.lee@rllab.snu.ac.kr).

This work was partly supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2019-0-01190, [SW Star Lab] Robot Learning: Efficient, Safe, and Socially-Acceptable Machine Learning, 50%) and Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT (NRF-2022R1A2C2008239, General-Purpose Deep Reinforcement Learning Using Metaverse for Real World Applications, 50%).

II. Related Work

A number of methods have been proposed for training a RL agent considering cost constraints. Achiam *et al.* [11] have proposed a method that optimizes a policy within a given trust-region. Kim and Oh [12] have proposed a trust region based method which utilizes the upper bound of conditional value at risk (CVaR) as a constraint function for the safe RL problem. Although trust-region based methods guarantee monotonic performance improvement for each policy update, they can be computationally expensive as the evaluation the Fisher information matrix and the second order Taylor expansion is mandatory in the optimization process. Ha *et al.* [8] have introduced a method that treats constrained RL problems as an unconstrained policy optimization problem by introducing a Lagrangian multiplier. Tessler *et al.* [10] have proposed a PID-Lagrangian method, which stabilizes the update process of the Lagrangian multiplier to prevent oscillations and cost-overshoots during training. However, the above methods do not consider multiple constraint level concurrently.

From another point of view, the safe RL problem can be considered as a multi-objective RL problem. Under the multi-objective RL setting, several works have studied to balance the trade-off from the conflicting objectives and learn policies over multiple preferences using a single network [13], [14]. Huang *et al.* [15] have proposed a framework that integrates multi-objective RL and constrained RL, where the dual variable in Lagrangian based safe RL can be interpreted as preference between the cost and reward. Despite the effectiveness of the above methods, they are unable to autonomously determine a suitable preference, or dual variable, adapting to an alternation of environmental distribution shift or constraint threshold.

III. Preliminaries

A. Constrained Markov Decision Process

A constrained Markov decision process (CMDP) [7] is a mathematical framework, which can be used to describe a constrained RL problem. A CMDP is a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, R, C)$, where $\mathcal{S}$ is a state space, $\mathcal{A}$ is an action space, $\mathcal{P}\colon \mathcal{S}\times\mathcal{A}\times\mathcal{S} \to \mathbb{R}$ is a transition model, $R\colon \mathcal{S}\times\mathcal{A} \to \mathbb{R}$ is a reward function, and $C\colon \mathcal{S}\times\mathcal{A} \to \mathbb{R}$ is a cost function. The expected return of the reward is defined as follows:

$$J_r^\pi := \mathbb{E}_\pi\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right], \tag{1}$$

where an initial state s_0 is sampled from ρ_0. Similarly, the expected return J_c^π of the cost is also defined as (1), but using costs instead of rewards. The goal of the constrained reinforcement learning is to maximize the expected reward return while limiting the expected cost return:

$$\underset{\pi}{\text{maximize}}\, J_r^\pi \text{ s.t. } J_c^\pi \le d, \tag{2}$$

where $d \in \mathbb{R}_{\ge 0}$ is a threshold for the constraint.

B. Primal-Dual Optimization

The dual problem of the constrained RL problem (2) is defined as the following min-max problem:

$$(\pi^*, \lambda^*) = \arg\min_{\lambda\ge 0}\max_{\pi} L(\pi, \lambda), \tag{3}$$

$$L(\pi, \lambda) = J_r^\pi - \lambda(J_c^\pi - d), \tag{4}$$

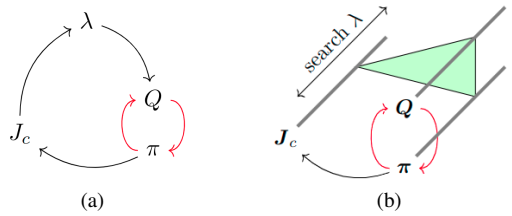


Fig. 1: Feedback loop in safe RL. (a) In traditional Lagrangian approach, the dual variable λ is updated according to the expected cost return J_c. When λ is updated, the objective of policy π also changes. (b) In the proposed framework, we illustrates policy and Q-functions with respect to the dual variable. Every training episode, λ is randomly sampled from a predefined distribution. Each λ has a corresponding safety level, so we can get a universal policy that can be applied to multiple safety levels after a single training.

where $\lambda \in \mathbb{R}_{\ge 0}$ is a non-negative dual variable, which is also called a Lagrange multiplier. Constrained reinforcement learning is known to have zero duality gap under the Slater's condition [16], which implies that the optimal policy π^* for the dual problem (3) is also optimal for the primal one (2).

For a given dual variable λ, maximizing (4) is equivalent to maximizing the expected return J_λ^π of the Lagrangian reward $\hat{R}_\lambda\colon \mathcal{S}\times\mathcal{A}\to\mathbb{R}$, which are defined as:

$$\hat{R}_\lambda(s,a) := (R(s,a) - \lambda C(s,a))/(1+\lambda), \tag{5}$$

where the factor $1/(1+\lambda)$ is multiplied to scale the Lagrangian reward.

C. Soft Actor-Critic

Soft actor-critic (SAC) is an off-policy RL algorithm first introduced by Haarnoja *et al.* [17]. SAC is an actor-critic algorithm that learns a stochastic policy π and a soft Q-function $Q\colon \mathcal{S}\times\mathcal{A}\to\mathbb{R}$. The soft policy iteration in SAC consists of soft policy evaluation and soft policy improvement.

In the policy evaluation step, the soft Q-function for the policy π is computed through the following Bellman operator:

$$\mathcal{T}^\pi Q(s,a) \triangleq R(s,a) + \gamma \mathbb{E}_{s'\sim\mathcal{P}(\cdot|s,a)}[V(s')], \tag{6}$$

$$V(s') = \mathbb{E}_{a'\sim\pi(\cdot|s')}[Q(s',a') - \alpha\log\pi(a'|s')]. \tag{7}$$

Here, α is a temperature parameter that determines the relative importance of the entropy term , which encourages exploration. In the policy improvement step, the policy is trained to approximate the distribution proportional to the exponential of the soft Q-function.

IV. Proposed Method

Existing Lagrangian methods find an optimal policy and dual variable by solving the min-max problem (4). Since the policy and variable influences each other in order to solve the min-max problem in the training process. As the dual variable and policy are updated in order to satisfy a given constraint, it is challenging to consider another constraint threshold after the training process. We propose a dual variable actor-critic (DVAC) for solving constrained RL by illustrating policy and Q-function with respect to the dual variable, as shown in Figure 1. We aim to find the λ-optimal policies and Q-function for any value of the dual variable

λ. We extend the actor-critic algorithm with the universal policy $\boldsymbol{\pi}(a|s,\lambda)$ and universal Q-function $\boldsymbol{Q}(s,a,\lambda)$.

A. Dual Q-Function

Given a dual variable λ, the dual Q-function $Q^{\pi}_{\lambda}\colon \mathcal{S}\times\mathcal{A}\to\mathbb{R}$ can be defined as:

$$Q^{\pi}_{\lambda}(s,a)=\mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty}\gamma^{t}\hat{R}^{\pi}_{\lambda}(s_t,a_t)|s_0=s,a_0=a\right], \quad (8)$$

where

$$\hat{R}^{\pi}_{\lambda}(s,a)=\hat{R}_{\lambda}(s,a)+\gamma\alpha\mathbb{E}_{s'\sim\mathcal{P}(\cdot|s,a)}[\mathcal{H}(\pi(\cdot|s'))], \quad (9)$$

where the Lagrangian reward $\hat{R}_{\lambda}$ is defined in (5). Consider a λ-optimal policy π^{*}_{λ} that maximizes the dual Q-function. Since the λ-optimal policy π^{*}_{λ} depends on the dual variable λ, the Q-function $Q^{\pi^{*}_{\lambda}}_{\lambda}$, which evaluates the λ-optimal policy, nonlinearly depends on λ. We define the optimal dual Q function $\boldsymbol{Q}^{*}\colon \mathcal{S}\times\mathcal{A}\times\mathbb{R}_{\geq 0}\to\mathbb{R}$ as:

$$\boldsymbol{Q}^{*}(s,a,\lambda)=Q^{\pi^{*}_{\lambda}}_{\lambda}(s,a). \quad (10)$$

We also define the optimal universal policy $\boldsymbol{\pi}^{*}(a|s,\lambda)$ as $\pi^{*}_{\lambda}(a|s)$. Notice that the dual Q-function $\boldsymbol{Q}^{*}(s,a,\lambda)$ and optimal universal policy $\boldsymbol{\pi}^{*}(a|s,\lambda)$ has the dual variable λ as an argument.

B. Soft Universal Policy Iteration

We propose universal policy evaluation and universal policy improvement by extending the SAC algorithm [17], which can compute an optimal universal policy by iteration. In the universal policy evaluation step, we aim to estimate the dual Q-function for a given universal policy. The dual Q-function can be computed from any function by repeatedly applying a modified Bellman operator $\mathcal{T}^{\boldsymbol{\pi}}$, defined as:

$$\mathcal{T}^{\boldsymbol{\pi}}\boldsymbol{Q}(s,a,\lambda)\triangleq\hat{R}_{\lambda}(s,a)+\gamma\mathbb{E}_{s'\sim\mathcal{P}(\cdot|s,a)}\big[\boldsymbol{V}(s',\lambda)\big], \quad (11)$$

$$\boldsymbol{V}(s',\lambda)=\mathbb{E}_{a'\sim\boldsymbol{\pi}(\cdot|s',\lambda)}\big[\boldsymbol{Q}(s',a',\lambda)-\alpha\log\boldsymbol{\pi}(a'|s',\lambda)\big]. \quad (12)$$

In the universal policy improvement step, we update the universal policy from $\boldsymbol{\pi}^{\text{old}}$ towards the target policy distribution:

$$\boldsymbol{\pi}^{\text{target}}(\cdot|s,\lambda)\propto\exp\left(\frac{1}{\alpha}\boldsymbol{Q}^{\boldsymbol{\pi}^{\text{old}}}(s,\cdot,\lambda)\right). \quad (13)$$

The universal policy is updated to $\boldsymbol{\pi}^{\text{new}}$ by minimizing the KL-divergence from the target policy distribution:

$$\boldsymbol{\pi}^{\text{new}}(\cdot|s,\lambda)=\arg\min_{\boldsymbol{\pi}'\in\Pi}D_{\text{KL}}\big(\boldsymbol{\pi}'(\cdot|s,\lambda)\big\|\boldsymbol{\pi}^{\text{target}}(\cdot|s,\lambda)\big). \quad (14)$$

Here, the target policy distribution is projected to a set of distributions Π, which can be a parameterized family of distributions.

C. Theoretical Analyses

In this section, we provide theoretical analyses of universal policy iteration, dual Q-function, and optimal universal policy. We assume that the number of elements in the action space $\mathcal{A}$ is finite. This assumption is required to guarantee that the Shannon entropy of the policy function is bounded.

Following a universal policy $\boldsymbol{\pi}$, we define the universal Q-function $\boldsymbol{Q}\colon \mathcal{S}\times\mathcal{A}\times\mathbb{R}_{\geq 0}\to\mathbb{R}$ as:

$$\boldsymbol{Q}(s,a,\lambda)=\mathbb{E}_{\boldsymbol{\pi}(\cdot|\cdot,\lambda)}\left[\sum_{t=0}^{\infty}\gamma^{t}\hat{R}^{\pi}_{\lambda}(s_t,a_t)|s_0=s,a_0=a\right], \quad (15)$$

where

$$\begin{aligned}\hat{R}^{\pi}_{\lambda}(s,a)=&\frac{1}{1+\lambda}R(s,a)+\frac{\lambda}{1+\lambda}C(s,a)\\&+\gamma\alpha\mathbb{E}_{s'\sim\mathcal{P}(\cdot|s,a)}[\mathcal{H}(\boldsymbol{\pi}(\cdot|s',\lambda))].\end{aligned} \quad (16)$$

Similar to Yang *et al.* [13], we define a distance between two universal Q-functions by the uniform metric.

Definition 1: Consider two bounded functions $\boldsymbol{Q},\boldsymbol{Q}'\colon \mathcal{S}\times\mathcal{A}\times\mathbb{R}_{\geq 0}\to\mathbb{R}$. The distance between two functions are defined as :

$$D(\boldsymbol{Q},\boldsymbol{Q}')=\sup_{s\in\mathcal{S},a\in\mathcal{A},\lambda\in\mathbb{R}_{\geq 0}}|\boldsymbol{Q}(s,a,\lambda)-\boldsymbol{Q}'(s,a,\lambda)|. \quad (17)$$

Note that $D(\boldsymbol{Q},\boldsymbol{Q}')$ is always finite since $\boldsymbol{Q}$ and $\boldsymbol{Q}'$ are bounded.

Lemma 1: Let $\boldsymbol{Q}$, $\boldsymbol{Q}'$ be bounded functions from $\mathcal{S}\times\mathcal{A}\times\mathbb{R}_{\geq 0}$ to $\mathbb{R}$. Then, $D(\mathcal{T}^{\boldsymbol{\pi}}\boldsymbol{Q},\mathcal{T}^{\boldsymbol{\pi}}\boldsymbol{Q}')\leq\gamma D(\boldsymbol{Q},\boldsymbol{Q}')$.

Proof: From (11), $\mathcal{T}^{\boldsymbol{\pi}}\boldsymbol{Q}(s,a,\lambda)$ is equal to

$$\begin{aligned}&\hat{R}_{\lambda}(s,a)+\gamma\mathbb{E}_{s'\sim\mathcal{P}(\cdot|s,a),a'\sim\boldsymbol{\pi}(\cdot|s,\lambda)}[\boldsymbol{Q}(s',a',\lambda)]\\&-\gamma\alpha\mathbb{E}_{s'\sim\mathcal{P}(\cdot|s,a),a'\sim\boldsymbol{\pi}(\cdot|s,\lambda)}[\log\boldsymbol{\pi}(a'|s',\lambda)] \quad (18)\\=&\hat{R}^{\pi}_{\lambda}(s,a)+\gamma\mathbb{E}_{s'\sim\mathcal{P}(\cdot|s,a),a'\sim\boldsymbol{\pi}(\cdot|s,\lambda)}[\boldsymbol{Q}(s',a',\lambda)]. \quad (19)\end{aligned}$$

For all $s\in\mathcal{S},a\in\mathcal{A},\lambda\in\mathbb{R}_{\geq 0}$,

$$\begin{aligned}&|\mathcal{T}^{\boldsymbol{\pi}}\boldsymbol{Q}(s,a,\lambda)-\mathcal{T}^{\boldsymbol{\pi}}\boldsymbol{Q}'(s,a,\lambda)|\\=&\big|\gamma\mathbb{E}_{s'\sim\mathcal{P},a'\sim\boldsymbol{\pi}}\big[\boldsymbol{Q}(s',a',\lambda)-\boldsymbol{Q}'(s',a',\lambda)\big]\big| \quad (20)\\\leq&\gamma\mathbb{E}_{s'\sim\mathcal{P},a'\sim\boldsymbol{\pi}}\big[|\boldsymbol{Q}(s',a',\lambda)-\boldsymbol{Q}'(s',a',\lambda)|\big] \quad (21)\\\leq&\gamma D(\boldsymbol{Q},\boldsymbol{Q}'). \quad (22)\end{aligned}$$

Thus, $D(\mathcal{T}^{\boldsymbol{\pi}}\boldsymbol{Q},\mathcal{T}^{\boldsymbol{\pi}}\boldsymbol{Q}')\leq\gamma D(\boldsymbol{Q},\boldsymbol{Q}')$. ■

Lemma 1 states that the Bellman operator $\mathcal{T}^{\boldsymbol{\pi}}$ is a contraction on a metric space with a uniform metric.

Theorem 2: (Universal Policy Evaluation.) Consider a bounded function $\boldsymbol{Q}^{0}\colon \mathcal{S}\times\mathcal{A}\times\mathbb{R}_{\geq 0}\to\mathbb{R}$ and define $\boldsymbol{Q}^{k+1}=\mathcal{T}^{\boldsymbol{\pi}}\boldsymbol{Q}^{k}$. Then the sequence $\boldsymbol{Q}^{k}$ will uniformly converge to the universal Q-function $\boldsymbol{Q}^{\boldsymbol{\pi}}$ as $k\to\infty$.

Proof: By the Banach fixed point theorem, the sequence will uniformly converge to a unique fixed point. Applying the Bellman operator to the universal Q-function $\boldsymbol{Q}^{\boldsymbol{\pi}}$ confirms that it is a fixed point. ■

Note that Theorem 2 differs from the analyses for SAC [17] in that it addresses various dual variables and states that the convergence is uniform with respect to the dual variable.

Theorem 3: (Universal Policy Iteration.) Iterations of universal policy evaluation and improvement allows any universal policy $\boldsymbol{\pi}$ to converge into an optimal universal policy $\boldsymbol{\pi}^{*}$ with respect to a dual variable λ.

Proof: Similar to soft policy improvement in SAC, the universal Q-function increases with universal policy improvement. By repeating universal policy iteration, the universal Q-function converges to some universal Q-function $\boldsymbol{Q}^{\boldsymbol{\pi}^{\infty}}$ that satisfies $\boldsymbol{Q}^{\boldsymbol{\pi}^{\infty}}(s,a,\lambda)\geq\boldsymbol{Q}^{\boldsymbol{\pi}'}(s,a,\lambda)$ for all $\boldsymbol{\pi}'\in\Pi,s\in S,a\in A,\lambda\in\mathbb{R}_{\geq 0}$. Therefore, policy $\boldsymbol{\pi}^{\infty}(\cdot|\cdot,\lambda)$ is λ-optimal for any dual variable $\lambda\in\mathbb{R}_{\geq 0}$. ■

Ignoring the entropy term by assuming that the temperature parameter α is zero, the Q-function decomposes into a reward-Q-function and a cost-Q-function as follows:

$$Q^{\pi}(s,a)=\frac{1}{1+\lambda}Q^{\pi}_{r}(s,a)-\frac{\lambda}{1+\lambda}Q^{\pi}_{c}(s,a). \quad (23)$$

Since the λ-optimal policy π^{*}_{λ} maximizes Q^{π}, $J^{\pi'}_{r}\leq J^{\pi^{*}_{\lambda}}_{r}$ or $J^{\pi'}_{c}\geq J^{\pi^{*}_{\lambda}}_{c}$ always holds for any dual variable λ and policy

π'. This implies that the optimal universal policy represents the Pareto frontier.

Proposition 4: A cost Q-function $Q_c^{\pi_\lambda^*}(s,a)$ for λ-optimal policy monotonically decreases with respect to dual variable λ.

Proof: From definition, λ-optimal policy π_λ^* maximizes Q-function, $Q^{\pi_\lambda^*} = (Q_r^{\pi_\lambda^*} - \lambda Q_c^{\pi_\lambda^*})/(1+\lambda)$. Consider $\lambda, \lambda' \in \mathbb{R}_{\geq 0}$ such that $\lambda \leq \lambda'$. Then,

$$Q_r^{\pi_\lambda^*} - \lambda Q_c^{\pi_\lambda^*} \geq Q_r^{\pi_{\lambda'}^*} - \lambda Q_c^{\pi_{\lambda'}^*}, \tag{24}$$

$$Q_r^{\pi_{\lambda'}^*} - \lambda' Q_c^{\pi_{\lambda'}^*} \geq Q_r^{\pi_\lambda^*} - \lambda' Q_c^{\pi_\lambda^*}. \tag{25}$$

Subtract two equations.

$$(\lambda' - \lambda)(Q_c^{\pi_\lambda^*} - Q_c^{\pi_{\lambda'}^*}) \geq 0, \tag{26}$$

which leads to $Q_c^{\pi_\lambda^*} \geq Q_c^{\pi_{\lambda'}^*}$. Here, inputs s, a are omitted for convenience. ∎

Corollary 4.1: Expected cost return $J_c^{\pi_\lambda^*}$ for λ-optimal policy monotonically decreases with respect to λ.

Similarly, expected reward return also decreases as the dual variable increases. These relationships show that a trade-off between safety and performance exists and is related to the dual variable. It means that the expected cost return can be reduced by applying a smaller dual variable. Moreover, increasing the dual variable within the limit allowed by the expected cost return leads to a higher expected reward return.

D. Searching Dual Variable after Training

To sample actions from the learned universal policy $\boldsymbol{\pi}(\cdot|s,\lambda)$, we need to determine which value of the dual variable λ to use. Since this value can be determined after training the universal policy, one naive way to decide the dual variable is by testing all possible values.

In this paper, we propose a method to automatically search optimal dual variable λ^* for a given constrained threshold d. Since the expected cost return $J_c^{\pi_\lambda^*}$ for the λ-optimal policy π_λ^* decreases with respect to the dual variable λ according to Corollary 4.1, the optimal dual variable λ^* satisfying $J_c^{\pi_\lambda^*} = d$ can be found easily with the binary search. Here, the expected cost return $J_c^{\pi_\lambda^*}$ can be computed as the expectation of the dual cost-Q-function $\boldsymbol{Q}_c(s,a,\lambda) = Q_c^{\pi_\lambda^*}(s,a)$:

$$J_c^{\pi_\lambda^*} = \mathbb{E}_{s\sim\rho_0, a\sim\pi_\lambda^*(\cdot|s)}[\boldsymbol{Q}_c^*(s,a,\lambda)], \tag{27}$$

where ρ_0 is initial state distribution. Here, dual cost-Q-function can be trained similar to the dual Q-function. See section IV-E for details.

Estimating the expectation can be challenging due to the lack of information on the initial state distribution ρ_0 in a new test environment. To address this issue, we propose two practical methods for estimating $J_c^{\pi_\lambda^*}$ using dual cost-Q-function. The first method is to sample states from the replay buffer saved during the training process. The second method uses the dual cost-Q-function for the latest states. The latter approach has the advantage that it is applicable even if the initial state distribution is different during training and testing. In this method, the dual variable may change at each environment step, so an exponential moving average is performed to ensure stability.

E. Dual Variable Actor-Critic

Practically, we parameterize the dual Q-function, the dual cost-Q-function, and the universal policy with neural networks. θ, ψ, ϕ are the parameters of the dual Q-network $\boldsymbol{Q}^\theta$, the dual cost-Q-network $\boldsymbol{Q}_c^\psi$, and the universal policy network $\boldsymbol{\pi}^\phi$, respectively.

We sample state-action pairs (s,a) from the replay buffer $\mathcal{D}$ and dual variable λ from a prior distribution Λ. Then we update the parameters θ, ψ of the dual Q-networks $\boldsymbol{Q}^\theta$, $\boldsymbol{Q}_c^\psi$ to minimize the following Bellman residuals:

$$\mathcal{L}_Q(\theta) = \mathbb{E}_{(s,a)\sim\mathcal{D},\lambda\sim\Lambda}\left[\frac{1}{2}(\boldsymbol{Q}^\theta(s,a,\lambda) - \hat{\boldsymbol{Q}}(s,a,\lambda))^2\right], \tag{28}$$

$$\mathcal{L}_c(\psi) = \mathbb{E}_{(s,a)\sim\mathcal{D},\lambda\sim\Lambda}\left[\frac{1}{2}(\boldsymbol{Q}_c^\psi(s,a,\lambda) - \hat{\boldsymbol{Q}}_c(s,a,\lambda))^2\right], \tag{29}$$

with the target value

$$\hat{\boldsymbol{Q}}(s,a,\lambda) = \mathcal{T}^{\boldsymbol{\pi}^\phi}\boldsymbol{Q}^{\bar{\theta}}(s,a,\lambda), \tag{30}$$

$$\hat{\boldsymbol{Q}}_c(s,a,\lambda) = \mathcal{T}_c^{\boldsymbol{\pi}^\phi}\boldsymbol{Q}_c^{\bar{\psi}}(s,a,\lambda), \tag{31}$$

where $\bar{\theta}$ and $\bar{\psi}$ are exponential moving averages of the parameters.

To avoid underestimation of the cost-Q-function, we use additional loss, called *conservative loss*. Inspired by Kumar *et al.* [18], the additional loss increases the cost-Q-function for actions that are not in the replay buffer. The conservative loss $\mathcal{L}_{cc}$ for updating the cost-Q-network $\boldsymbol{Q}_c^\psi$ is defined as:

$$\mathcal{L}_{cc} = \mathcal{L}_c + \mathbb{E}_{(s,a)\sim\mathcal{D},\lambda\sim\Lambda}\Big[\boldsymbol{Q}_c^\psi(s,a,\lambda)\Big] - \mathbb{E}_{s\sim\mathcal{D},\lambda\sim\Lambda,a\sim\boldsymbol{\pi}^\phi(\cdot|s,\lambda)}\Big[\boldsymbol{Q}_c^\psi(s,a,\lambda)\Big]. \tag{32}$$

The universal policy parameter ϕ is trained to minimize expected KL-divergence in (14). Since the temperature parameter α and the normalization factor are independent of the action, the loss becomes:

$$\mathcal{L}_\pi(\phi) = \mathbb{E}\Big[\alpha\log(\boldsymbol{\pi}^\phi(a|s,\lambda)) - \boldsymbol{Q}^\theta(s,a,\lambda)\Big], \tag{33}$$

where $s\sim\mathcal{D}$, $\lambda\sim\Lambda$, and $a\sim\boldsymbol{\pi}^\phi(\cdot|s,\lambda)$.

The complete algorithm for training DVAC is described in Algorithm 1. To summarize, the DVAC algorithm trains a dual Q-network, a dual cost-Q-network, and a universal policy network by sampling state-action pairs and dual variables as inputs. The dual Q-network and dual cost-Q-network are updated to minimize the Bellman residual, while the universal policy network is trained to minimize the KL divergence with the exponential of the dual Q-network. The universal policy network can sample actions based on the state with a dual variable. The dual variable is randomly sampled for each training episode to collect the replay buffer.

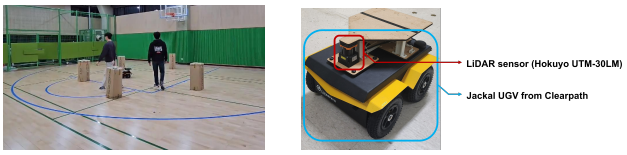
V. Experiments

In this section, we evaluate the proposed method in various environments with different tasks and robots, to answer the following questions:

- Can DVAC learn a universal policy that ranges from extremely safe to high performance according to the dual variables?
- Is the universal policy learned via DVAC Pareto optimal compared to the policies from the baseline methods?
- Can we identify a suitable dual variable to adapt to environments with unseen state distributions?
- Is it feasible to transfer the policy learned from simulated environments to the real world using DVAC?

Algorithm 1 Dual Variable Actor-Critic (DVAC)

Input: $\theta_1, \theta_2, \psi, \phi$
$\bar{\theta}_1 \leftarrow \theta_1$, $\bar{\theta}_2 \leftarrow \theta_2$, $\bar{\psi} \leftarrow \psi$
$\mathcal{D} \leftarrow \emptyset$
for each iteration **do**
  $\lambda' \sim \Lambda$
  for each environment step **do**
    $a_t \sim \boldsymbol{\pi}_\phi(\cdot|s_t, \lambda')$
    $s_{t+1} \sim \mathcal{P}(s_t, a_t)$
    $\mathcal{D} \leftarrow \mathcal{D} \cup \{(s_t, a_t, r(s_t, a_t), c(s_t, a_t), s_{t+1})\}$
  end for
  for each gradient step **do**
    $\theta_i \leftarrow \theta_i - \eta_Q \hat{\nabla}_{\theta_i} \mathcal{L}_Q(\theta_i)$ for $i \in \{1, 2\}$
    $\psi \leftarrow \psi - \eta_c \hat{\nabla}_\psi \mathcal{L}_{cc}(\psi)$
    $\phi \leftarrow \phi - \eta_\pi \hat{\nabla}_\phi \mathcal{L}_\pi(\phi)$
    $\bar{\theta}_i \leftarrow \tau\theta_i + (1-\tau)\bar{\theta}_i$ for $i \in \{1, 2\}$
    $\bar{\psi} \leftarrow \tau\psi + (1-\tau)\bar{\psi}$
  end for
end for



(a) (b)

Fig. 2: Jackal platform and real-world environment. (a) Jackal real-world environment, (b) Jackal robot. We installed a LiDAR with a 270-degree field of view on the robot.

A. Safety Gym Environments

The simulation experiments are conducted using four environments in Safety Gym [19] with some modifications based on the works of Kim and Oh [12] and Liu *et al.* [20]. Two types of robots, `Point` and `Car`, are used for two tasks: `Goal` and `Button`. In the `Goal` task, robots move to a series of goal positions without entering hazards. In the `Button` task, robots press a series of goal buttons without entering hazards, pressing wrong buttons, and colliding with moving Gremlins.

The environments are compatible with the `PointGoal1`, `CarGoal1`, `PointButton1`, and `CarButton1` from the original Safety Gym except the following three modifications. First, the goal position is provided in the state directly, not in LIDAR form, for a more realistic setting. Second, we use cost function $C(s, a) = \text{sigmoid}(10 \cdot \delta)$, where δ is the minimum distance (positive or negative) to the area to be avoided. Third, the positions of the button and Gremlin are deterministic in the `Button` task. Especially for the `PointGoal` environment, we also test a policy in environments where the number of obstacles are different from the number of obstacles used in during training.

B. Jackal Environments

The sim-to-real experiment is to move a Jackal [21] robot to a series of goal positions while avoiding walls, static obstacles, and dynamic pedestrians. Figure 2(b) shows the Jackal robot platform used in experiment, including a LiDAR installed with a range of 270 degrees. In this experiment, we train the agents in the MuJoCo simulator and evaluate in the real world without additional training. Figure 2(a) show images from simulation and the real-world experiment. This environment is similar to the one from Kim and Oh [12] except for the dynamic pedestrians. Each episode has up to six obstacles and up to three pedestrians. For safety, we immediately end the episode and add an additional cost if the robot approaches a wall or obstacle within 0.2 meters.

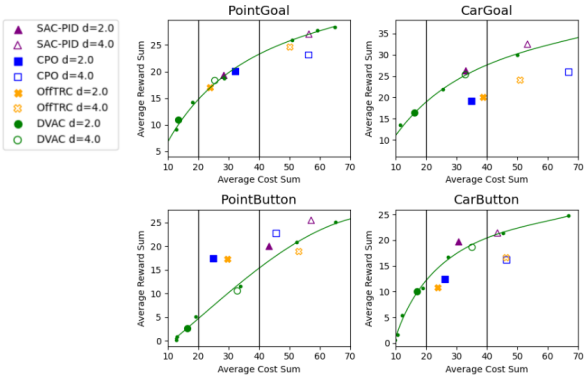


Fig. 3: Average reward and cost sums of the policy trained. For DVAC, green marker is evaluated by applying the searched dual variable on the trained universal policy.

C. Baselines

We compare the proposed method with three safe RL baselines: SAC-PID, CPO [11], and off-policy TRC (OffTRC) [12]. Like the proposed method, SAC-PID is a Lagrange-based approach based on SAC. Inspired by Stooke *et al.* [10], SAC-PID controls the dual variable with a PI controller. CPO and OffTRC are a trust region-based method that ensures monotonic policy improvement by constraining policy updates within a trust region. For fair comparisons, we use the cut-off point of CVaR for OffTRC to one, which makes the CVaR constraint become an expectation constraint, like other methods. We trained all of the baseline methods for 8e6 environment steps in the Safety Gym environment and for 4e6 environment steps in the Jackal environment.

D. Results

Figure 3 shows the average sums of rewards and costs of the trained policies in the Safety Gym environment. The green line shows the result of the proposed algorithm, tested across a wide range of dual variable. Note that in DVAC, repeated training is not required for testing each dual variable, as it learns the universal policy. The proposed method learns a diverse set of policies, ranges from an extremely safe policy with an average cost sum of 10.1 to a high-performing policy with an average reward sum of 28.0 in the `PointGoal` environment. The proposed method forms a frontier that is as safe as the safe RL baselines in `PointGoal`, `CarGoal` and `CarButton` environments. However, frontier formed with our method is less safe and less efficient than the baselines in the `PointButton` environment. As it is difficult to gain low cost sum while completing the task in `PointButton` environment, the proposed method is unable to learn the entire wide-ranged universal policy. The green markers represent the results of applying dual variables found using the method described in Section IV-D using replay buffer. In all four environments, the average cost sum was below the threshold and above 63% of the threshold.

TABLE I: **PointGoal adaptation results.** The table presents a comparison between the two dual variable-search methods, described in IV-D. Agents are first trained in `PointGoal` environments with eight hazards and then adapt to the sparse and crowded `PointGoal` environments, where four and twelve hazards exist, respectively.

	Reward Sum ↑		Cost Sum ↓	
Method	Sparse	Crowded	Sparse	Crowded
Replay Buffer	24.2	13.4	12.4	36.6
Latest State	25.8	11.1	17.2	33.2

TABLE II: **Jackal experiment results.** The table shows the result of the safe RL comparison on the Jackal environment. Sparse, Crowded, and Real stands for sparse simulation, crowded simulation, and real-world environment. The sparse simulation environment includes two obstacles and no pedestrians, while the crowded simulation environment includes two obstacles and four pedestrians.

	Reward Sum ↑			Cost Sum ↓		
Method	Sparse	Crowded	Real	Sparse	Crowded	Real
SAC-PID	24.5	22.3	26.1	19.8	83.8	27.6
CPO	22.6	20.8	22.8	18.4	81.4	27.5
OffTRC	22.9	19.8	27.4	12.0	69.8	17.6
DVAC	21.0	17.9	25.0	16.1	68.4	3.7

The SAC algorithm with a fixed dual variable of $\lambda = 0.6$ achieves a reward sum of 9.0, a cost sum of 17.2, and a Lagrangian reward sum of 2.6 in the `PointGoal` environment. Applying $\lambda = 0.6$ to the universal policy learned by DVAC results in a Lagrangian reward sum of 4.6, which is 1.8 times higher than that achieved by the SAC algorithm, despite both algorithms being trained with the same number of environment and training steps.

Table I shows the results of the proposed methods using different methodologies when finding the dual variable, trained from `PointGoal` environment and evaluated with different numbers of obstacles. Compared to the method using a replay buffer, the method using the latest state achieves a 9% lower cost sum in a crowded environment where safety is critical. Conversely, in a sparse environment, the method using the latest state achieves a 6% higher reward sum. This demonstrates that the method using the latest state adapts to changes in the environment.

Table II shows the results in Jackal environment. Here, DVAC uses a latent state to determine the dual variable. In the real-world environment, DVAC achieves a reward sum, which is compatible to other safe RL baselines, while keeping its cost sum below 21% of other baselines.

VI. Conclusion

In this paper, we have proposed DVAC, a dual variable actor-critic, which finds optimal universal policy over a wide range of the dual variable. Under DVAC, the balance between safety and performance can be controlled by choosing the appropriate value for the dual variable. Our proposed method has learned a universal policy that ranges from extremely safe to high performance according to the dual variables. Furthermore, the agent has been able to adapt to environments with unseen state distributions by identifying a suitable dual variable with the proposed method.

The proposed method has a limitation in that it does not guarantee safety during the training process. In particular, since random dual variables are used to collect trajectories, unsafe policies are often executed even in the later stage of training. Future research may explore strategies for sampling dual variables to reduce this risk while collecting trajectories. Also, this paper does not address multiple constraints while we believe that a similar approach can be applied.

References

[1] D. Hafner, T. P. Lillicrap, M. Norouzi, and J. Ba, "Mastering Atari with discrete world models," in *Proc. of the International Conference on Learning Representations, May 3-7*, 2021.

[2] J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel, L. Timothy, and D. Silver, "Mastering Atari, Go, Chess and Shogi by planning with a learned model," *Nature*, vol. 588, no. 7839, pp. 604–609, 2020.

[3] A. P. Badia, B. Piot, S. Kapturowski, P. Sprechmann, A. Vitvitskyi, Z. D. Guo, and C. Blundell, "Agent57: Outperforming the Atari human benchmark," in *Proc. of the International Conference on Machine Learning, July 13-18*, 2020.

[4] R. Mendonca, O. Rybkin, K. Daniilidis, D. Hafner, and D. Pathak, "Discovering and achieving goals via world models," in *Proc. of the Advances in Neural Information Processing Systems, December 6-14*, 2021.

[5] S. Gu, E. Holly, T. P. Lillicrap, and S. Levine, "Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates," in *Proc. of the International Conference on Robotics and Automation, May 29 - June 3*, 2017.

[6] K. Lee, J. Choy, Y. Choi, H. Kee, and S. Oh, "No-regret Shannon entropy regularized neural contextual bandit online learning for robotic grasping," in *Proc. of the International Conference on Intelligent Robots and Systems*, October 24, 2020 - January 24, 2021.

[7] E. Altman, "Constrained Markov decision processes with total cost criteria: Lagrangian approach and dual linear program," *Mathematical methods of operations research*, vol. 48, no. 3, pp. 387–417, 1998.

[8] S. Ha, P. Xu, Z. Tan, S. Levine, and J. Tan, "Learning to walk in the real world with minimal human effort," in *Proc. of the Conference on Robot Learning, November 16-18*, 2020.

[9] Q. Yang, T. D. Simão, S. H. Tindemans, and M. T. J. Spaan, "WCSAC: worst-case soft actor critic for safety-constrained reinforcement learning," in *Proc. of the AAAI Conference on Artificial Intelligence, February 2-9*, 2021.

[10] A. Stooke, J. Achiam, and P. Abbeel, "Responsive safety in reinforcement learning by PID Lagrangian Methods," in *Proc. of the International Conference on Machine Learning, July 13-18*, 2020.

[11] J. Achiam, D. Held, A. Tamar, and P. Abbeel, "Constrained policy optimization," in *Proc. of the International Conference on Machine Learning, August 6-11*, 2017.

[12] D. Kim and S. Oh, "Efficient off-policy safe reinforcement learning using trust region conditional value at risk," *IEEE Robotics Automation Letters*, vol. 7, no. 3, pp. 7644–7651, 2022.

[13] A. Abels, D. M. Roijers, T. Lenaerts, A. Nowé, and D. Steckelmacher, "Dynamic weights in multi-objective deep reinforcement learning," in *Proc. of the International Conference on Machine Learning, June 9-15*, 2019.

[14] R. Yang, X. Sun, and K. Narasimhan, "A generalized algorithm for multi-objective reinforcement learning and policy adaptation," in *Proc. of the Advances in Neural Information Processing Systems, December 8-14*, 2019.

[15] S. H. Huang, A. Abdolmaleki, G. Vezzani, P. Brakel, D. J. Mankowitz, M. Neunert, S. Bohez, Y. Tassa, N. Heess, M. A. Riedmiller, and R. Hadsell, "A constrained multi-objective reinforcement learning framework," in *Proc. of the Conference on Robot Learning, November 8-11*, 2021.

[16] S. Paternain, L. F. O. Chamon, M. Calvo-Fullana, and A. Ribeiro, "Constrained reinforcement learning has zero duality gap," in *Proc. of the Advances in Neural Information Processing Systems, December 8-14*, 2019.

[17] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proc. of the International Conference on Machine Learning, July 10-15*, 2018.

[18] A. Kumar, A. Zhou, G. Tucker, and S. Levine, "Conservative Q-learning for offline reinforcement learning," in *Proc. of the Advances in Neural Information Processing Systems, December 6-12*, 2020.

[19] A. Ray, J. Achiam, and D. Amodei, "Benchmarking safe exploration in deep reinforcement learning," *CoRR*, vol. abs/1910.01708, 2019.

[20] Z. Liu, Z. Cen, V. Isenbaev, W. Liu, Z. S. Wu, B. Li, and D. Zhao, "Constrained variational policy optimization for safe reinforcement learning," in *Proc. of the International Conference on Machine Learning, July 17-23*, 2022.

[21] "Jackal UGV - small weatherproof robot - Clearpath," Dec 2020. [Online]. Available: https://clearpathrobotics.com/jackal-small-unmanned-ground-vehicle/. [Accessed: 06-Sep-2022]