Learn From Safe Experience: Safe Reinforcement Learning for Task Automation of Surgical Robot

Ke Fan, Ziyang Chen, Giancarlo Ferrigno, *Senior Member, IEEE*, and Elena De Momi, *Senior Member, IEEE*

***Abstract*—Surgical task automation in robotics can improve the outcomes, reduce quality-of-care variance among surgeons and relieve surgeons' fatigue. Reinforcement learning (RL) methods have shown considerable performance in robot autonomous control in complex environments. However, the existing RL algorithms for surgical robots do not consider any safety requirements, which is unacceptable in automating surgical tasks. In this work, we propose an approach called safe experience reshaping (SER) that can be integrated into any offline RL algorithm. First, the method identifies and learns the geometry of constraints. Second, a safe experience is obtained by projecting an unsafe action to the tangent space of the learned geometry, which means that the action is in the safe space. Then, the collected safe experiences are used for safe policy training. We designed three tasks that closely resemble real surgical tasks including 2-D cutting tasks and a contact-rich debridement task in 3-D space to evaluate the safe RL framework. We compare our framework to five state-of-the-art (SOTA) RL methods including reward penalty and primal-dual methods. Results show that our framework gets a lower rate of constraint violations and better performance in task success, especially with a higher convergence speed.**

***Impact Statement*—While RL methods have shown promise in the autonomous control of robots in complex environments, their use in surgical robotics has been limited by safety concerns. Integrating safety requirements into surgical robotics ensures that automation operates within well-defined boundaries, preventing unintended actions or errors that could jeopardize patient well being. The article on safe RL in surgical task automation addresses a critical gap in the existing RL algorithms for surgical robots: the lack of consideration for safety requirements. By introducing the SER approach, the article presents a novel solution to integrate safety into RL algorithms. The evaluation of the SER framework against five SOTA RL methods demonstrates its superiority in terms of lower safety violations and improved task success rates, with faster convergence. Future research aims to realize image-based learning in surgical tasks incorporating the SER approach in the meantime.**

***Index Terms*—Safe reinforcement learning (RL), surgical robot, surgical task automation.**

I. Introduction

ROBOT-ASSISTED minimally invasive surgery (RAMIS) has seen remarkable progress in the past two decades. RAMIS combines the precision of robotic systems with the advantages of minimally invasive procedures, reducing the physical trauma associated with open surgery and augmenting a surgeon's capabilities [1], [2]. This advancement has piqued interest in surgical task automation.

Surgical procedures are inherently complex, demanding a high level of adaptability, and real-time decision-making. Tasks such as suturing, tissue manipulation, and organ resection require dexterity and the ability to navigate intricate anatomical structures [3], [4], [5], [6]. The automation of these tasks is a compelling solution to mitigate the potential for human error, improve surgical outcomes, and reduce physical strain on surgeons who often perform these tasks over extended periods. Traditional control strategies, often based on preprogrammed rules and models, may struggle to adapt to the dynamic and unpredictable nature of surgery. Besides, they require substantial expertise and a complex development process. In contrast, reinforcement learning (RL), with its ability to learn from experience, provides a more flexible and adaptive framework to handle complex surgical tasks [7], [8]. Surgical environments can be highly variable, such as variations in patient anatomy and surgical complications. Traditional controllers, while capable of precise execution, often rely on static rules and lack the learning capabilities that RL provides. RL can continuously improve surgical procedures based on the knowledge gained from each surgery, ensuring the robotic system remains effective and safe under changing conditions.

In this context, RL is a promising approach for imparting surgical skills to robots. However, standard RL algorithms, such as Q-learning or policy gradient methods, often fall short in ensuring the safety of learned policies. For instance, in critical tasks like tumor resection, the robot must remove diseased tissue without harming healthy tissue or blood vessels. Therefore, it becomes imperative to develop safe RL algorithms that guarantee policy safety. However, this endeavor poses several challenges. Initially, both the environmental dynamics and constraint information are unknown, rendering the agent oblivious to which states are safe and which are perilous. Consequently, during the exploration phase, the agent cannot collect safe trajectories to improve the policy, thereby impeding the acquisition of safe behaviors. Furthermore, it is hard to tradeoff between exploration and safety, as overly stringent constraints may weaken the agent's ability to explore sufficiently for optimal policy learning. Conversely, overly lax

Manuscript received 15 August 2023; revised 26 October 2023; accepted 3 January 2024. Date of publication 9 January 2024; date of current version 9 July 2024. This article was recommended for publication by Associate Editor Ali Etemad upon evaluation of the reviewers' comments. *(Corresponding author: Ziyang Chen.)*

The authors are with the Department of Electronics, Information and Bioengineering, Politecnico di Milano, 20133 Milan, Italy (e-mail: ziyang.chen@polimi.it).

Digital Object Identifier 10.1109/TAI.2024.3351797

constraints may lead to a failure to fully respect these constraints. Incorporating constraints during learning can also impact the learning process itself. For instance, the addition of constraints to optimization problems amplifies their complexity, rendering them more arduous to resolve. Moreover, constraints may result in a sparse reward signal, making the learning process more challenging [9].

To overcome these challenges, we present the safe experience reshaping (SER) framework, and the contributions can be summarized as follows.

1) Constraint perception: We begin by pretraining a constraint perception network. In contrast to prior work, our training process allows the agent to autonomously explore the environment and collect constraint information. The constraint perception network effectively functions as a safety critic for constraints, determining the safety of specific states and also representing the manifold of constraints. Autonomous exploration is achieved through the optimization of a task-focused policy (guidance policy). This approach enables the network to rapidly learn the constraints affecting the task, as opposed to random exploration.
2) SER: During the pretraining phase, we obtain a guidance policy that does not consider constraints and a network that has learned the constraint manifold. When the agent executes risky actions under the guidance policy, these actions are projected onto the tangent space of the constraint manifold, ensuring compliance with the constraints. We collect trajectories containing corrective actions as safety experiences to train the safety policy. Compared to previous methods, this innovative approach accelerates the convergence of safety policies.
3) We introduce a comprehensive RL environment for the da Vinci research kit (dVRK) surgical robot (Intuitive Surgical, USA), compare five state-of-the-art (SOTA) methods, and validate our approach on three simulated surgical tasks. Finally, we implement our method with visual assistance on a real dVRK robot, thereby exploring the potential for sim-to-real transfer.

Regarding the article organization, the related work is documented in Section II; in Section III, we state the problem of RL with safety constraints; Section IV delves into the issues of constraint perception and SER; the experimental setup and results are presented in Section V; and Section VI draws conclusions.

II. Related Work

Researchers have developed various approaches that integrate safety constraints into the RL formulation. This section provides an overview of the key contributions and developments in the field. We also compare the nonlearning-based methods with RL methods in terms of their key characteristics, advantages, and limitations.

A. Joint Optimization for Safety and Task Performance

An approach commonly employed in safe RL involves the integration of safety constraints into the policy optimization process. An elementary approach of joint optimization is to add penalty terms into the reward function to deter unsafe behaviors [10], [11]. More advanced methodologies are rooted in the domain of constrained Markov decision processes (CMDPs) [12]. Primal-dual optimization (PDO) is a solution for CMDPs, which introduces Lagrange multipliers (LMs) or dual variables associated with the constraints, enabling the transformation of the original constrained optimization problem into an unconstrained (UN) problem [13], [14]. Constructing Lyapunov functions [15], [16] is another approach to solve the CMDPs problem. Lyapunov functions convert the trajectory-level constraints into discrete, state-dependent constraints at each time step. This method, however, necessitates the system's initialization with a policy that already satisfies the constraint. There are other safe RL methods that draw inspiration from control theory, which leverage control-theoretic principles such as stability analysis, robust control, and adaptive mechanisms to ensure safety [17], [18], [19], [20]. These methods all remain within the framework of joint optimization, which jointly optimizes task performance and constraint satisfaction. They usually do not require a prior model of the robot and constraints, but instead merely incentivize safe behavior and do not guarantee the satisfaction of hard constraints.

B. Safe Exploration

The key idea of safe exploration entails guiding the agent to explore within a safety-constrained subset of Markov decision processes (MDPs) [21]. Safety exploration strategies based on the filtering mechanism have been proposed in [22], [23], and [24]. Such methods all involve solving a constrained least squares problem. For example, Pham et al. [25] incorporates the differentiable quadratic programming (QP) solver into an extra safe action modification layer. The safe layer acts as a safety critic, which is used to convert an optimal but potentially unsafe action to the closest safe action that satisfies safety constraints. However, performing multiple steps of the QP solver in every interaction step is necessary to ensure that the QP problem converges and produces reliable results, which introduces additional computational complexity. In addition to safety filter mechanisms, recovery exploration mechanisms are proposed in [26] and [27]. Recovery policy updating also leverages the insights from the safety critic and ensures that the system can transition from any state to a safe state [27]. In the previous work, the safe critic is usually learned based on an offline dataset that includes expert-defined constraints. While our method can make the agents to explore and collect constraint information autonomously. This approach offers advantages in terms of learning efficiency and safety compared to methods that jointly optimize task performance and safety.

C. Comparison of Learning-Based and Nonlearning-Based Method

Automation under uncertainty while ensuring safety has been a longstanding focus in control theory. Model predictive control (MPC) is one of the most popular control strategies used in automation in recent years [28]. MPC allows the explicit

inclusion of safety constraints by optimizing a cost function that includes both control objectives and safety constraints [29], [30]. MPC is one of the pure planning techniques that never explicitly represent the policy compared with model-based RL (MBRL) [31], [32]. In MBRL, agents use such planning algorithms to generate candidate actions as the expert relative to the policy [33]. In fact, traditional nonlearning-based control strategies can often outperform RL in terms of tracking accuracy and other traditional control metrics, especially in situations where the dynamics of the system are well understood and well modeled [34]. However, typical assumptions of nonlearning-based control strategies involve the presence of the system and constraint model, often characterized by definite parameters or possessing bounded uncertainties in its dynamics and noise. Learning-based methods can excel in complex, nonlinear, and high-dimensional environments where manually defining constraints may be challenging. Furthermore, considering the specific application scenarios of RAMIS, in surgical areas such as the abdominal cavity, additional sensors could reduce the flexibility of surgical tools, increase system complexity, and potentially lead to tissue infection [35]. Surgical robots are often equipped with only endoscopic cameras and lack sensors on surgical tools. As a result, visual information becomes the sole means of environmental perception. The absence of sensors undoubtedly presents a challenge for nonlearning-based controllers. In contrast, learning-based methods can effectively extract and utilize features from high-dimensional image data.

III. Problem Statement

Under the concept of MDPs, RL can be formulated mathematically with a tuple: $M = \langle S, A, R, P, \rho_0, \gamma \rangle$ where S is the state space, a set of possible states that the agent can be in; A is the action space, a set of possible actions that the agent can take in each state; R is the reward function that specifies the reward that the agent receives for being in a particular state and taking a particular action; P is the transition probability function that specifies the probability of transitioning from one state to another when an action is taken; ρ_0 is the start state distribution; and γ is the discount factor that determines the importance of future rewards compared to immediate rewards.

In this article, we take safety constraints into account with CMDPs: $CM = \langle S, A, R, P, \rho_0, \gamma, C, \gamma_c \rangle$ where C represents the costs that the agent receives for violating safety constraints with the corresponding discount factor γ_c. $C = 1$ corresponds to constraint violation, otherwise $C = 0$. The goal of RL is to learn an optimal policy π that maximizes the expected cumulative reward (CR) over time. The policy guides the decision-making process of the agent by mapping the observed state to the appropriate action to be executed. The expected CR G_r^π is defined as the sum of discounted rewards over a time horizon T: $G_r^\pi = \mathbb{E}\left[\sum_{t=0}^{T} \gamma^t R(s_t, a_t, s_{t+1})\right]$. Correspondingly, the expected cumulative cost G_c^π is defined as $G_c^\pi = \mathbb{E}\left[\sum_{t=0}^{T} \gamma_c^t C(s_t, a_t, s_{t+1})\right]$. The optimal policy π^* is the policy that maximizes the expected CR over all possible policies

$$\pi^* = \underset{\pi \in \Pi}{\operatorname{argmax}} G_r^\pi \tag{1}$$

where Π is a set of policies. In the safe RL, we define Π_c as a feasible set of safety constraint-satisfying policies. The optimal policy π_c^* in safe RL is then given by

$$\begin{aligned} \pi_c^* &= \arg\max_{\pi \in \Pi_c} G_r^\pi \\ \Pi_c &= \{\pi : G_c^\pi \leq \xi\} \end{aligned} \tag{2}$$

where ξ is the safety threshold $\xi \in [0, 1]$ [36].

In a robotic system, the observations may include sensor readings such as joint angles, the position of the end effector, velocity, and acceleration. Thus, we can model the action a as a function of partial observations such as the velocity s_{vel} of the end effector: $a = f(s_{\text{vel}})$. In this article, we assume that the safety constraints purely depend on the position s_{pos_t} at time step t of the agent. Then, the expected discounted cumulative constraint violation is $G_c^\pi = \mathbb{E}\left[\sum_{t=0}^{T} \gamma_c^t C(s_{\text{pos}_t})\right]$, which is expected to be below the safety threshold $\xi \in [0, 1]$. Thus, we can extract the position of the agent from the state space as the input to train a neural network (NN) to fit a function G_c^π, which can be used to predict the manifold of the constraints. For example, we assume that the positions where $G_c^\pi \leq \xi$ are safe, the positions where $G_c^\pi > \xi$ are risky. We can project an unsafe action that may violate the constraints to the tangent space of the constraint manifold to get a safe action. Then, we execute the corrected safe action instead of the original unsafe action, which means the agent only explores in the tangent space of the constraint manifold. Based on this idea, we can collect trajectories with the corrected safe actions for the agent to learn safe behavior.

IV. Methodology

The main ideas behind SER are shown as follows. In Section IV-A, we show details on how to learn the constraint perception NN to predict the manifold of constraints. In Section IV-B, we show how the constraint perception NN is used to reshape safe experiences. In Section IV-C, we show how the safe policy is improved based on the safe experience.

A. Constraint Geometry Perception

As shown in Fig. 1, on the left, a constraint perception function $G_c^{\pi_g}$ is learned to predict the discounted future probability of constraint violation of a policy π_g

$$G_c^{\pi_g}(s_{\text{pos}_t}) = \mathop{\mathbb{E}}_{\pi_g}\left[c_t \mid s_{\text{pos}_t} + \gamma_c G_c^{\pi_g}(s_{\text{pos}_{t+1}})\right] \tag{3}$$

where s_{pos_t} is the current position state, $s_{\text{pos}_{t+1}}$ is the next position state, and π_g is the guiding policy. To fit the function $G_c^{\pi_g}$, we train a NN $G_{c,\phi}^{\pi_g}$ with parameters ϕ. We collect a set $\mathcal{D}$ of experiences $\left\langle s_{\text{pos}_t}, a_t, s_{\text{pos}_{t+1}}, c_t \right\rangle$. To generate $\mathcal{D}$, we initialize the task target in a uniformly random location during the learning process of the guiding policy. The guiding policy learns to complete the task regardless of the constraints so that the agent can collect necessary data on constraint violating behavior, which means data including each position with a safety

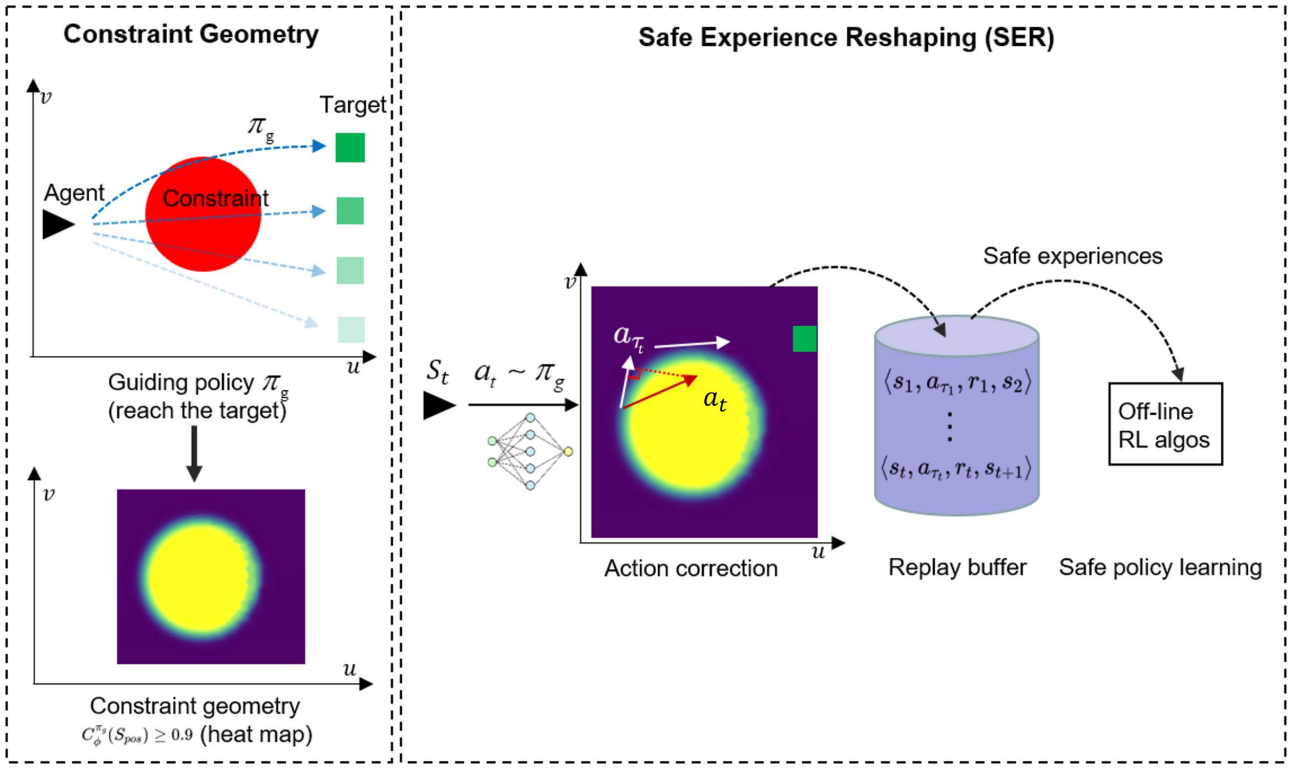

Fig. 1. We illustrate the safe RL by a 2-D obstacle avoidance task where the constraint is the red area. In the constraint perception phase, we pretrain a guidance policy π_g to reach the randomized target ignoring the constraint. The process of training π_g provides data of constraint violating behavior from which constraint manifold is learned. In the SER phase, the agent explores the environment by requesting actions a_t from π_g. When the agent is too close to the constraint boundary, a_t is projected onto the tangent space of the constraint manifold to get corrected action $a_{\tau t}$. Then, the experiences generated using these corrected actions are then stored in the replay buffer for subsequent safe policy training.

threshold ξ label. We train the $G_{c,\phi}^{\pi_g}$ by minimizing the mean-squared Bellman error (MSBE) loss function $J(\phi, \mathcal{D})$ using the sampled experiences $\mathcal{D}$

$$J(\phi,\mathcal{D}) = \mathop{\mathbb{E}}_{\langle s_{\text{pos}_t}, a_t, s_{\text{pos}_{t+1}}, c_t\rangle \sim \mathcal{D}} \Bigg[G_{c,\phi}^{\pi_g}(s_{\text{pos}_t}) - \left(c_t + \gamma_c G_{c,\phi}^{\pi_g}(s_{\text{pos}_{t+1}})\right)^2 \Bigg]. \tag{4}$$

The output of the NN $G_{c,\phi}^{\pi_g}(s_{\text{pos}})$ in fact constructs a safety critic. It quantifies the risk of the agent's position state s_{pos}, which is the collision probability in this context. Thus, if the agent acts in the area where the discounted probability of constraint violation is below the safety threshold $G_{c,\phi}^{\pi_g}(s_{\text{pos}}) \leq \xi$, the agent will not violate the constraints.

B. SER

The geometry of the constraints can be viewed as the manifold of the constraints parameterized by a set of coordinates that describe the position of a point on the manifold, which are the input (agent's position state s_{pos_t}) of the NN $G_{c,\phi}^{\pi_g}(s_{\text{pos}_t})$. For example, a manifold of a constraint represented by a circle area can be parameterized by two coordinates (u, v) that describe the position of a point on the constraint. We can project an arbitrary action onto the tangent space which is the set of all tangent vectors to the manifold as shown in Fig. 1, right.

If the agent acting under the guiding policy π_g is at position s_{pos_t} such that $G_{c,\phi}^{\pi_g}(s_{\text{pos}_t}) > \xi$, we can project the action a into the tangent space to get a safe action. Taking the 2-D case shown in Fig. 1, for example, we can prove that the projected action can guarantee the constraint.

We define the constraint $G_c^{\pi_g}(u, v)$ which is composed of all positions (u, v) such that

$$G_c^{\pi_g}(u,v) - \xi \leq 0. \tag{5}$$

The derivative of (5) is as follows:

$$\dot{G}_c^{\pi_g}(u,v) = J_c(u,v) \begin{bmatrix} \dot{u} \\ \dot{v} \end{bmatrix} \tag{6}$$

where $J_c(u, v)$ is the Jacobian matrix of the constraint. The row vector $[(\partial G_c^{\pi_g})/(\partial u), (\partial G_c^{\pi_g})/(\partial v)]$ of the Jacobian matrix is the normal vector of the constraint manifold. The tangent space of the constraint manifold is composed of a set of basis vectors that can be represented as the orthogonal vectors of the normal vector. These basis vectors actually form the null space of the Jacobian matrix: $N(u, v) = \text{null}[J_c(u, v)]$ such that $N(u, v)J_c(u, v) = 0$. We can select a set of coordinates ω in the tangent space of the constraint manifold to construct a velocity $[\dot{u}_\tau, \dot{v}_\tau]^T$ such that we can distinguish $[\dot{u}_\tau, \dot{v}_\tau]^T$ as a vector in the tangent space

$$N(u,v)\omega = \begin{bmatrix} \dot{u_\tau} \\ \dot{v_\tau} \end{bmatrix}. \tag{7}$$

Algorithm 1 SER

Require: pretrained guiding policy π_g, constraint perception NN $G^{\pi_g}_{c,\phi}$, empty replay buffer $\mathcal{D}$, task horizon T, number of episodes N, safety threshold ξ.

TRAIN

$s_0 \leftarrow \mathsf{env.reset}()$ ▷ Initial state

for $t = 1, 2, \cdots, T$ **do**

  if $G^{\pi_g}_{c,\phi}(s_{pos_t}) \leq \xi$ **then**

    $a_t \leftarrow \pi_g$, $\mathsf{env.step}(a_t)$ ▷ Execute actions of π_g

    observe s_{t+1} and r

    $\mathcal{D} \leftarrow \langle s_t, a_t, s_{t+1}, r \rangle$ ▷ Store trajectory in replay buffer

  else if $G^{\pi_g}_{c,\phi}(s_{pos_t}) > \xi$ **then**

    $a^*_t \leftarrow a_\tau$, $\mathsf{env.step}(a^*_t)$ ▷ Execute safe action

    $\mathcal{D} \leftarrow \langle s_t, a^*_t, s_{t+1}, r \rangle$ ▷ Safe experience reshape

  end if

  Train safe policy π_s on $\mathcal{D}$

end for

Substituting $[\dot{u}_\tau, \dot{v}_\tau]^T$ into (6)

$$\dot{G}^{\pi_g}_c(u, v) = J_c(u, v)N(u, v)\omega = J_c(u, v)\begin{bmatrix}\dot{u}_\tau \\ \dot{v}_\tau\end{bmatrix} = 0. \quad (8)$$

Equation (8) means that no matter what coordinate ω is selected, the constraint remains unchanged. We model the action a as the function of velocity of partial state: $a = f(\dot{s}_{\text{pos}})$. Starting from a safe area $\{(u, v)|G^{\pi_g}_c(u, v) \leq \xi\}$, and then projecting unsafe actions into the tangent space, the action $a_\tau = f([\dot{u}_\tau, \dot{v}_\tau]^T)$ can ensure that the agent is moving in the safe area.

C. Safe Experience Improvement

In UN RL, we gather experiences of tuples $m = \langle s, a, r, p, \rho_0, \gamma \rangle$ by repeatedly performing policy network predictions in the environment. We have recognized the potential risks associated with executing UN action a. Thus, the agent will execute the corrected action a_τ during the interaction with the environment. The proposed safe RL mechanism offers protection to the agent by avoiding regions where constraint violations are likely, assuming that the estimated risk of the constraint perception network is accurate and that executing the corrected action decreases this risk. In order to learn such corrected safe behavior, we substitute the original action a with the safe action a_τ into the tuples: $m' = \langle s, a_\tau, r, p, \rho_0, \gamma \rangle$ and improve the safe policy with this safe experience. See Algorithm 1 and Fig. 1 for further illustration.

V. Experiments

The experiment aims at exploring the effectiveness of SER within the context of automating surgical tasks. The study focuses on learning three tasks utilizing a six-DoF manipulator of dVRK surgical system in the simulation environment. Then, we conduct a physical experiment on the first task based on the real dVRK robot system with visual assistance of an endoscopic camera. The primary objective is to investigate whether SER techniques can effectively balance task performance and constraint satisfaction compared to SOTA algorithms.

A. Simulation Experimental Setup

The experiments focus on policy learning under state space constraints. If the agent violates that constraint during an episode, the episode is terminated. This makes sense in real-world environments where constraint violations may necessitate the halting of robots to prevent damage to themselves or their surrounding environment. Learning under such constraints poses significant challenges as it restricts the agent's ability to explore the environment sufficiently. The scenarios try to closely mirror real surgical tasks. For example, the goal of cancer surgery is to remove all of the cancer from the body. To do this, the surgeon uses cutting tools to remove the tumor and some healthy tissue around it. During tumor removal procedures, efforts are made to avoid puncturing or damaging the tumor. Puncturing the tumor can potentially lead to the spread of cancer cells or cause bleeding. While during some debridement surgeries, which involve the removal of damaged, infected, or dead tissue, there is a risk of damaging internal organs and blood vessels.

1) Circumnavigation: This task is simulating tumor removal. it is crucial to ensure that the surgical tool does not puncture the round tumor. The constraint is the red area, as shown in Fig. 2(a). The goal is to bring the surgical tool agent to the target by controlling the movement on the u axis (horizontal) and v axis (vertical). The action space is continuous velocity change $a = [\dot{u}, \dot{v}]^{\text{T}}$. The state space is the position and velocity of the agent. The reward is sparse: 50 rewards are obtained once reaching the target and 0 elsewhere. Episodes terminate once one of the following events happens: the agent violates the constraint, the target is successfully reached, or the max time horizon ($t = 3000$) is reached.
2) Navigation: This task is simulating the cutting surgery. We aim to preserve healthy tissue and minimize unnecessary damage. The path should carefully consider the course of blood vessels, nerves, and other vital structures to avoid accidental injury during the cutting process. The constraint is the red wall, as shown in Fig. 2(b). The goal is to bring the surgical tool agent to the target by controlling the movement on the 2-D plane. The state space, action space, and reward function are the same as the circumnavigation task.
3) Debridement: This is a 3-D task set up with the dVRK surgical robot as shown in Fig. 2(c). In this task, a patient side manipulator (PSM) equipped with the surgical tool which is a gripper that aims to retract the diseased tissue under the blood vessel. During debridement, it is important for the surgical tool to be carefully maneuvered to bypass blood vessels and minimize the risk of injury. The action space is the velocity change of the tooltip in the 3-D space. The state space is the position and velocity of the tooltip and the position of the vessel and diseased tissue. Since exploration in 3-D space is inefficient, we use a dense reward function: getting 30 rewards when reaching

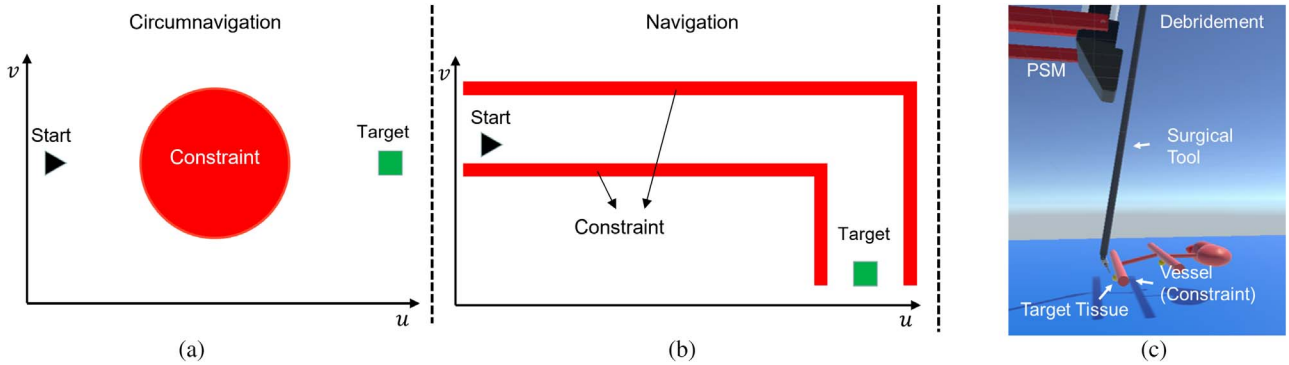

Fig. 2. Simulation experiments domains: (a) surgical toll agent should bypass the red circle area to reach the green target; (b) is similar to (a), the 2-D tasks are designed to mirror the cutting task in surgeries; and (c) is designed to mirror the debridement surgery. The surgical tool should get the tissue without collisions with the vessel.

the target and $1 - d/d_{max}$ rewards before reaching the target where d is the distance between the end effector and the target and $d_{\max}$ is the distance between the start position and the target position.

B. Comparison

We implement our SER RL and other five existing methods in the scenarios mentioned in Section V-A.

1) UN [37]: Ignoring the constraint, just try to maximize the task reward. UN can be a benchmark of the reward learning curve. Other methods considering the constraint cannot perform better than UN in CRs and convergence speed.
2) Recovery RL (RRL) [27]: The core idea behind RRL is to learn a safe critic to monitor the agent's actions and environment during execution to detect unsafe states. The safe critic is learned based on the offline data that contain expert-defined constraint information. Once a failure or unsafe state is detected, the recovery mechanism will pull the agent back to the former safe state. Since the agent will always fall back to the former state, the exploration of this mechanism is inefficient. The policy can be safe, but it needs more time to learn how to reach the goal.
3) Constrained policy optimization (CPO) [38]: CPO employs trust region methods that iteratively update the policy within a trust region. This trust region acts as a safety buffer, ensuring that the policy changes are not too drastic and remain within the defined constraints. By enforcing this trust region constraint, CPO guarantees that the agent's behavior stays close to the original policy, preventing drastic policy updates that may lead to violating safety requirements.
4) Proximal policy optimization with Lagrange multiplier (PPO-LAG) [36]: This method can be used with any other standard RL algorithm. We use PPO-LAG in the control experiment. It uses an adaptive LM to enforce constraints. With $f(\theta)$ as the objective and $c(\theta)$ as the constraint, the LM solves the dual-primal optimization problem

$$\max_{\theta} \min_{\lambda \geq 0} = f(\theta) - \lambda c(\theta) \tag{9}$$

by gradient ascent on θ and descent on λ.

5) Reward penalty (RP) [10]: The key idea behind RP is to modify the reward function to include penalties for actions or states that should be avoided or discouraged

$$r_t = r_t^{\text{ori}} - c_t \tag{10}$$

where r_t is the immediate reward, r_t^{ori} is the original immediate reward, and c_t is the immediate safe penalty. By assigning negative rewards or penalties to such actions or states, the agent learns to avoid them during the learning process. This approach provides a mechanism for shaping the agent's behavior toward safer or more desired outcomes.

C. Evaluation Metrics

We proposed metrics to evaluate the performance and safety aspects of the proposed safe RL algorithms. Metrics provide quantitative measures and visualizations that assess the effectiveness of the learning algorithms in achieving the desired balance between task performance and constraint adherence. The evaluation metrics used in the study are as follows.

1) CR: The reward performance metric assesses the CRs obtained by the agent over an episode of T time horizon

$$\text{CR} = \sum_{t=0}^{T} r_t. \tag{11}$$

It provides an understanding of the agent's ability to optimize the task-specific objectives while ensuring compliance with safety constraints. The learning curve plots CRs against the number of learning iterations. It visualizes the agent's learning progress, illustrating how its performance evolves over time.
2) Cumulative task successes (CTSs): Since episodes terminate when violating safety constraints during the learning process, cumulative successes can measure episodes in which the agent successfully completes the task while adhering to safety constraints

$$\text{CTS} = \sum_{n=1}^{N} o_n \tag{12}$$

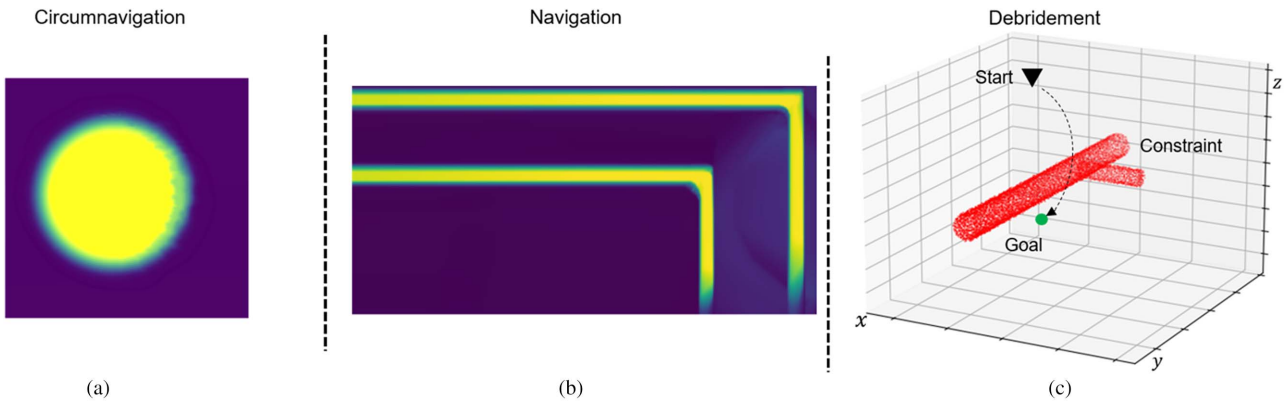

Fig. 3. Learned geometry of the constraint. (a) and (b) Use the heatmap to visualize the geometry of constraints of tasks 1 and 2 as shown in Fig. 2. (c) Use point cloud to show the geometry of the constraint in task 3.

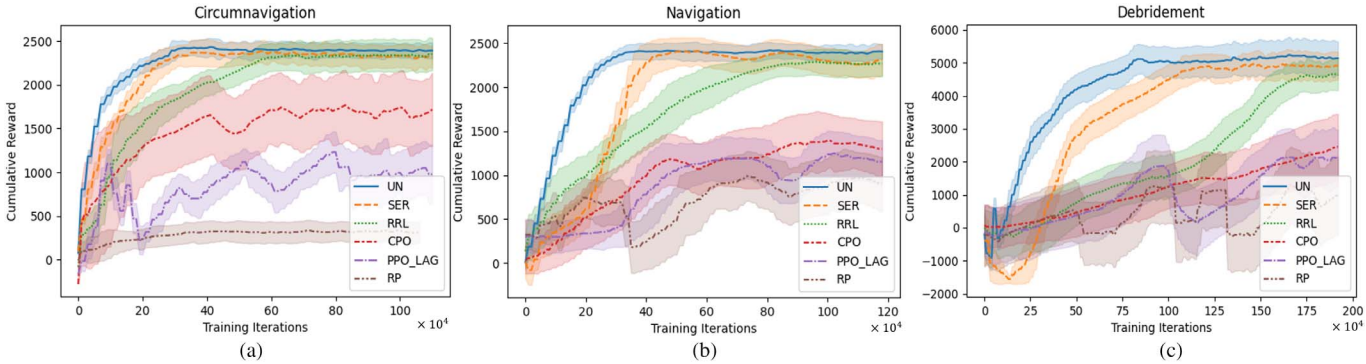

Fig. 4. Reward learning curve: the curves show the CR in each episode and averaged over every ten episodes. We run each algorithm ten times with different seeds. The average value is represented by the line, while the shaded area surrounding the line represents the standard deviation.

where N is the total episodes over the course of learning. If the episode is successful, $o_n = 1$, otherwise $o_n = 0$. It provides insight into how effectively the agent learns to achieve the desired goal while avoiding violations.

3) Cumulative violations (CVs): This metric quantifies the episode in which the agent violates safety constraints during the learning process

$$\mathrm{CV} = \sum_{n=1}^{N} v_n \tag{13}$$

where N is the total episodes over the course of learning. If the agent violates the constraint in the episode, $v_n = 1$, otherwise $v_n = 0$. It should not increase once a policy can generate safe actions. However, if both cumulative successes and cumulative violations no longer increase, the agent may be trapped in local optima, which means agents exhibit excessive caution and risk aversion, thereby limiting their exploration of potentially better solutions. When agents are overly cautious, they may prioritize safety constraints to such an extent that they become hesitant to take actions that could lead to better performance. This cautious behavior can prevent them from exploring different possibilities and finding more optimal solutions. Instead, they may settle for suboptimal outcomes that satisfy safety requirements but do not reach the goal.

D. Simulation Results

In our method (SER), the risky actions are corrected based on the geometry of the constraint. Therefore, we first show the pretrained constraint manifold results in Fig. 3. We use heatmaps in tasks 1 and 2 and use point cloud in task 3 to visualize the position where the safety threshold $\xi \geq 0.9$. Then, we report the results corresponding to the metrics mentioned in Section V-C. We ran all the methods ten times with random seeds and reported the mean and standard errors. This allows us to capture the inherent variability in the learning process and evaluate the stability and consistency of the algorithms' performance. By reporting the mean and standard error across the ten runs, we can quantify the central tendency and measure the variability of the results. This statistical analysis further strengthens the validity and reliability of the experimen<title>-tal findings.

The results of our study indicate that our safe RL approach with SER mechanisms outperforms prior algorithms across all tasks. Notably, SER demonstrates a faster convergence speed compared to other algorithms.

To illustrate the performance, we present the reward learning curves in Fig. 4. The curves show the total rewards achieved

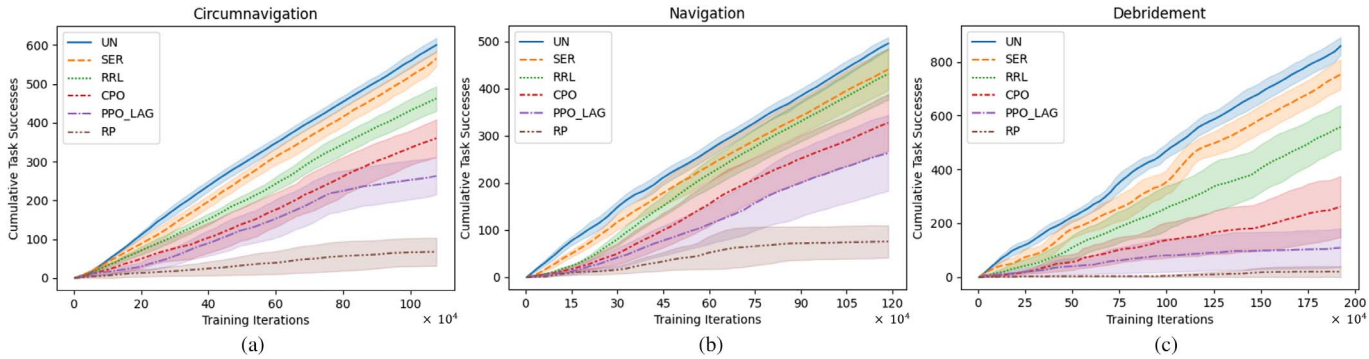

Fig. 5. Cumulative task success curve: the cumulative successes in each task for each algorithm, with results averaged over ten runs for all algorithms.

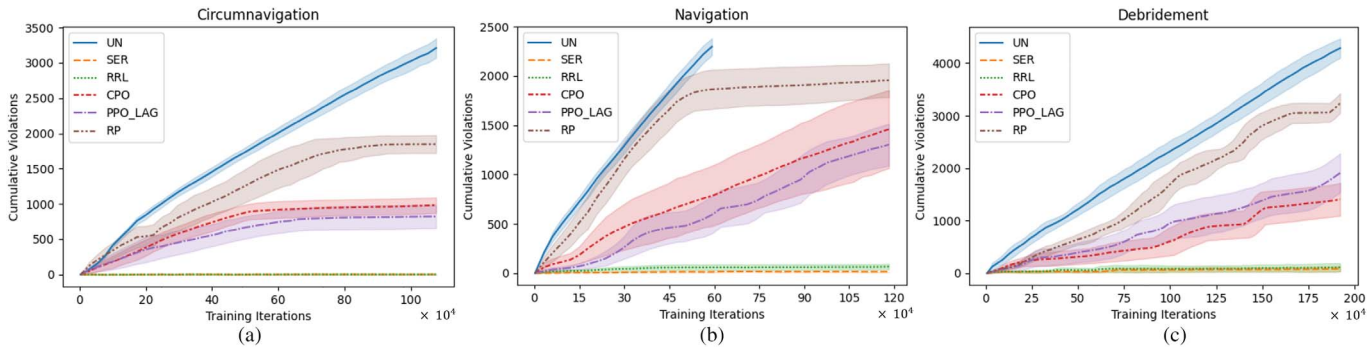

Fig. 6. Cumulative constraint violation curve: the cumulative violations in each task for each algorithm, with results averaged over ten runs for all algorithms.

TABLE I
WILCOXON RANK-SUM TEST BETWEEN STATE-OF-THE-ART METHODS AND OURS

	UN	RRL	CPO	PPO_LAG	RP
Circumnavigation	$*$	ns	$***$	$***$	$***$
Navigation	$***$	$**$	$***$	$***$	$***$
Debridement	$**$	$**$	$***$	$***$	$***$

Note: Wilcoxon rank-sum test to compute the significant differences between different SOTA methods and our method SER. The result is shown as follows: $ns: 0.05 < p \leq 1$, $*: 0.01 < p \leq 0.05$, $**: 0.001 < p \leq 0.01$, $***: 0.0001 < p \leq 0.001$, $****: p \leq 0.0001$, where p is the p_value of Wilcoxon rank-sum test.

in each episode, smoothed over a ten-episode window, for each simulation domain. The results are averaged over ten runs for all algorithms. Among the algorithms, the UN converges the fastest and achieves the highest rewards. This is because UN disregards constraints and solely focuses on reaching the goal. However, it should be noted that UN may not guarantee safety.

On all domains, SER exhibits quicker convergence to higher quality solutions in terms of the task reward function compared to prior safe RL algorithms. This suggests that SER is not only able to learn more safely but also more efficient than the comparison algorithms. RRL also achieves high rewards but with a slower convergence speed compared to SER. As shown in Table I, the statistical results based on the Wilcoxon rank-sum test show that our method is different from the SOTA methods in the CR.

Figs. 5 and 6 illustrate the performance of the algorithms in terms of task successes and constraint violations. UN shows the highest number of successes and violations simultaneously since it does not consider constraints. After pretraining the constraint perception, both SER and RRL can avoid violations from the start of the training phase. Lagrangian methods demonstrate intermediate performance, where algorithms such as CPO and PPO-LAG are able to accomplish the task as the policy improves, as seen in Fig. 5, with an increase in the number of successful episodes. However, Fig. 6 also shows that the number of failed episodes due to constraint violations is also increasing. This suggests that such methods cannot always guarantee constraint satisfaction.

RP performs the worst, as it does not show significant increases in reward. This suggests that simply penalizing violations may not effectively train the agent. In the 2-D tasks, we observe a plateau in both successes and violations, indicating that the agent becomes overly cautious. In the 3-D task, RP achieves limited success, highlighting the difficulty in striking a balance between task performance and constraint adherence.

In summary, our results demonstrate the superiority of our safe RL approach with SER mechanisms over prior algorithms across various tasks. SER exhibits faster convergence, achieving higher quality solutions in terms of task rewards. While

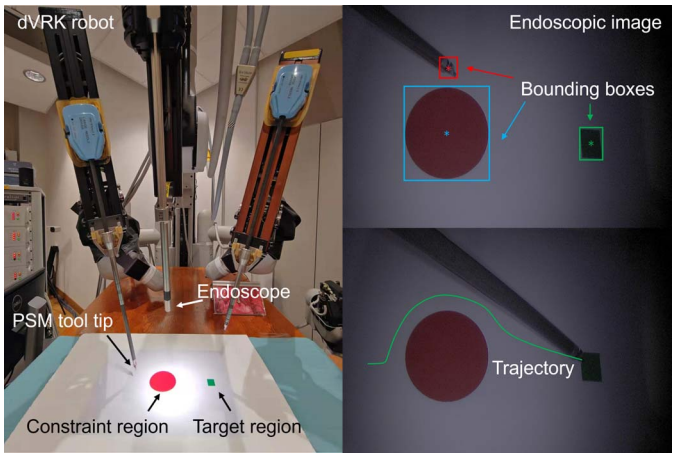

Fig. 7. Physical experiment setup. The left side depicts the robotic system equipped with an endoscopic camera to provide visual information. The PSM tool tip aims at reaching the green target region while avoiding entry into the red constraint region. The upper right portion displays the results of tool tip, constraint region, and target region detection based on endoscopic images, with asterisks denoting the central points of the bounding boxes. The lower right part exhibits potential trajectories of the tool tip.

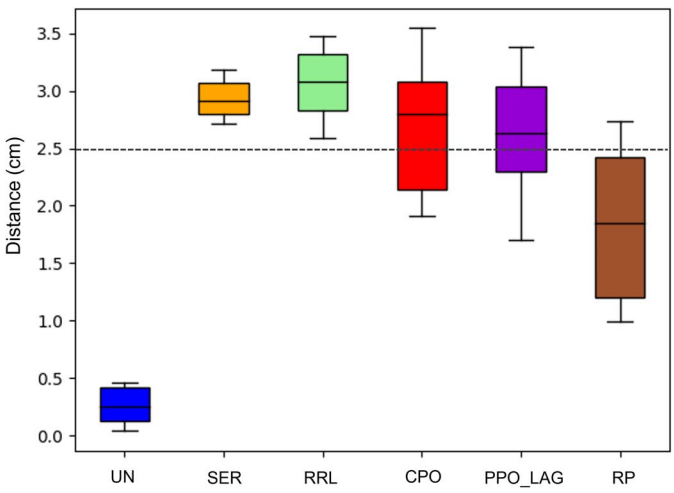

Fig. 8. Minimum distance between the center point of the PSM tool tip and the center point of the constraint region of a total of 20 runs of each method. Instances where the distance exceeded 2.5 cm were considered as not violating the constraint.

RRL shows high performance in terms of rewards and safety, it shows a lower convergence speed. Lagrangian methods have intermediate performance, and RP struggles to optimize both task performance and constraint satisfaction effectively.

E. Physical Experiment

Physical experiments are run on the dVRK robotic system. RL training is typically not conducted directly on real robots for several reasons, including safety concerns and the potential risk of damaging expensive equipment. The primary focus of this section lies in the transfer of strategies learned in a simulated environment to practical applications on real robots.

As shown in Fig. 7, due to the absence of conventional sensors on our robotic platform, we can only use an endoscopic camera to provide visual information. Thus, we use a visual method for pretraining the constraint perception network. A dataset comprising 3000 endoscopic images as shown on the right side of Fig. 7 was curated for the training of an detection network. This detection network is tasked with detecting the PSM tool tip, constraint region, and target region within the images and outputting their respective positions. Subsequently, the output of the detection network is employed to train the constraint perception network we introduced before and also serves as observations for our trained policies.

We ran each method 20 times, during which we recorded the minimum distance between the PSM tool tip and the center of the constraint region, as shown in Fig. 8. If this distance exceeded 2.5 cm, it was considered as nonviolation of the constraint. Since UN disregards constraints, allowing the tool tip to directly traverse through the constraint center region, the distances are close to zero. Our method SER demonstrated impeccable safety assurance. The shorter whisker exhibited minimal dispersion of data, which means better stability of the method. RRL also assured safety but displayed a comparatively larger dispersion. CPO and PPO yielded similar results, with their medians surpassing the 2.5 cm threshold, but without an absolute guarantee of safety. RP, on the other hand, exhibits the worst performance, with almost no guarantee of safety.

VI. Conclusion

In conclusion, this article introduced the SER method for safe RL. The SER approach incorporates constraint geometry prelearning and safe policy training to achieve a balance between task performance and safety. We designed three simulation tasks that are closely integrated with real surgical tasks. The experimental results demonstrate the effectiveness of the SER method compared to other safe RL algorithms. The SER approach shows superior convergence speed, task success rate, and safety performance. The integration of constraint geometry prelearning and safe policy training enables the SER method to efficiently explore the environment and achieve a balance between task completion and constraint satisfaction. In addition, we also validated our safety policy on the real dVRK surgical robot, utilizing a vision model to convert high-dimensional image features into low-dimensional position coordinates. The results indicate that the safety policy remains effective in the real robotic system, providing an initial exploration of the potential for sim-to-real transfer. However, we need to collect the geometric information of the constraints first, the process is not real time. Thus, it is challenging to handle dynamically changing constraints.

Further research can explore extensions and variations of the SER method to address specific challenges in different domains and tasks. Then, we wish to realize end-to-end learning in surgical tasks. The development of image-based deep safe RL can benefit surgical automation more, since robotic surgical systems, such as the da Vinci surgical system, rely on accurate

and detailed visual information to perform complex surgical operations with precision and efficiency.

REFERENCES

[1] T. Haidegger, "Autonomy for surgical robots: Concepts and paradigms," *IEEE Trans. Med. Robot. Bionics*, vol. 1, no. 2, pp. 65–76, May 2019.

[2] S. A. Bowyer, B. L. Davies, and F. R. y Baena, "Active constraints/virtual fixtures: A survey," *IEEE Trans. Robot.*, vol. 30, no. 1, pp. 138–157, Feb. 2014.

[3] F. Richter, R. K. Orosco, and M. C. Yip, "Open-sourced reinforcement learning environments for surgical robotics," 2019, *arXiv:1903.02090*.

[4] V. M. Varier, D. K. Rajamani, N. Goldfarb, F. Tavakkolmoghaddam, A. Munawar, and G. S. Fischer, "Collaborative suturing: A reinforcement learning approach to automate hand-off task in suturing for surgical robots," in *Proc. 29th IEEE Int. Conf. Robot Human Interactive Commun. (RO-MAN)*, Piscataway, NJ, USA: IEEE Press, pp. 1380–1386.

[5] N. D. Nguyen, T. Nguyen, S. Nahavandi, A. Bhatti, and G. Guest, "Manipulating soft tissues by deep reinforcement learning for autonomous robotic surgery," in *Proc. IEEE Int. Syst. Conf. (SysCon)*, Piscataway, NJ, USA: IEEE Press, pp. 1–7.

[6] T. Nguyen, N. D. Nguyen, F. Bello, and S. Nahavandi, "A new tensioning method using deep reinforcement learning for surgical pattern cutting," in *Proc. IEEE Int. Conf. Ind. Technol. (ICIT)*, Piscataway, NJ, USA: IEEE Press, 2019, pp. 1339–1344.

[7] X. He and C. Lv, "Robotic control in adversarial and sparse reward environments: A robust goal-conditioned reinforcement learning approach," *IEEE Trans. Artif. Intell.*, vol. 5, no. 1, pp. 244–253, Jan. 2024.

[8] J. García and F. Fernández, "A comprehensive survey on safe reinforcement learning," *J. Mach. Learn. Res.*, vol. 16, no. 1, pp. 1437–1480, 2015.

[9] L. Brunke et al., "Safe learning in robotics: From learning-based control to safe reinforcement learning," *Annu. Rev. Control, Robot., Auton. Syst.*, vol. 5, pp. 411–444, May 2022.

[10] A. Pore et al., "Safe reinforcement learning using formal verification for tissue retraction in autonomous robotic-assisted surgery," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Piscataway, NJ, USA: IEEE Press, 2021, pp. 4025–4031.

[11] M. El-Shamouty, X. Wu, S. Yang, M. Albus, and M. F. Huber, "Towards safe human-robot collaboration using deep reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, 2020, pp. 4899–4905.

[12] S. Gu et al., "A review of safe reinforcement learning: Methods, theory and applications," 2022, *arXiv:2205.10330*.

[13] X. Shi, Y. Xu, G. Chen, and Y. Guo, "An augmented Lagrangian-based safe reinforcement learning algorithm for carbon-oriented optimal scheduling of EV aggregators," *IEEE Trans. Smart Grid*, vol. 15, no. 1, pp. 795–809, Jan. 2024.

[14] S. Paternain, M. Calvo-Fullana, L. F. O. Chamon, and A. Ribeiro, "Safe policies for reinforcement learning via primal-dual methods," *IEEE Trans. Autom. Control*, vol. 68, no. 3, pp. 1321–1336, Mar. 2023.

[15] A. B. Jeddi, N. L. Dehghani, and A. Shafieezadeh, "Memory-augmented Lyapunov-based safe reinforcement learning: End-to-end safety under uncertainty," *IEEE Trans. Artif. Intell.*, vol. 4, no. 6, pp. 1767–1776, Dec. 2023.

[16] Y. Zhang et al., "Barrier Lyapunov function-based safe reinforcement learning for autonomous vehicles with optimized backstepping," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Jul. 1, 2022.

[17] Y. Chow, O. Nachum, E. Duenez-Guzman, and M. Ghavamzadeh, "A Lyapunov-based approach to safe reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, 2018.

[18] M. Han, Y. Tian, L. Zhang, J. Wang, and W. Pan, "Reinforcement learning control of constrained dynamic systems with uniformly ultimate boundedness stability guarantee," *Automatica*, vol. 129, 2021, Art. no. 109689.

[19] P. Liu, D. Tateo, H. B. Ammar, and J. Peters, "Robot reinforcement learning on the constraint manifold," in *Proc. Conf. Robot Learn.*, PMLR, 2022, pp. 1357–1366.

[20] Y. Chow, O. Nachum, A. Faust, E. Duenez-Guzman, and M. Ghavamzadeh, "Lyapunov-based safe policy optimization for continuous control," 2019, *arXiv:1901.10031*.

[21] J. F. Fisac, A. K. Akametalu, M. N. Zeilinger, S. Kaynama, J. Gillula, and C. J. Tomlin, "A general safety framework for learning-based control in uncertain robotic systems," *IEEE Trans. Autom. Control*, vol. 64, no. 7, pp. 2737–2752, Jul. 2019.

[22] G. Dalal, K. Dvijotham, M. Vecerik, T. Hester, C. Paduraru, and Y. Tassa, "Safe exploration in continuous action spaces," 2018, *arXiv:1801.08757*.

[23] S. Li and O. Bastani, "Robust model predictive shielding for safe reinforcement learning with stochastic dynamics," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA),* Piscataway, NJ, USA: IEEE Press, 2020, pp. 7166–7172.

[24] L. Zhang, Q. Zhang, L. Shen, B. Yuan, X. Wang, and D. Tao, "Evaluating model-free reinforcement learning toward safety-critical tasks," in *Proc. AAAI Conf. Artif. Intell.*, vol. 37, no. 12, 2023, pp. 15313–15321.

[25] T.-H. Pham, G. De Magistris, and R. Tachibana, "Optlayer-practical constrained optimization for deep reinforcement learning in the real world," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, Piscataway, NJ, USA: IEEE Press, 2018, pp. 6236–6243.

[26] J. F. Fisac, N. F. Lugovoy, V. Rubies-Royo, S. Ghosh, and C. J. Tomlin, "Bridging Hamilton-Jacobi safety analysis and reinforcement learning," in *Proc. Int. Conf. Robot. Automat. (ICRA)*, Piscataway, NJ, USA: IEEE Press, 2019, pp. 8550–8556.

[27] B. Thananjeyan et al., "Recovery RL: Safe reinforcement learning with learned recovery zones," *IEEE Robot. Autom. Lett.*, vol. 6, no. 3, pp. 4915–4922, Jul. 2021.

[28] H. Fukushima, T. Yanagiya, Y. Ota, M. Katsumoto, and F. Matsuno, "Model predictive path-following control of snake robots using an averaged model," *IEEE Trans. Control Syst. Technol.*, vol. 29, no. 6, pp. 2444–2456, Nov. 2021.

[29] D. Nicolis, F. Allevi, and P. Rocco, "Operational space model predictive sliding mode control for redundant manipulators," *IEEE Trans. Robot.*, vol. 36, no. 4, pp. 1348–1355, Aug. 2020.

[30] T. Gold, A. Völz, and K. Graichen, "Model predictive interaction control for robotic manipulation tasks," *IEEE Trans. Robot.*, vol. 39, no. 1, pp. 76–89, Feb. 2023.

[31] T. M. Moerland et al., "Model-based reinforcement learning: A survey," *Found. Trends Mach. Learn.*, vol. 16, no. 1, pp. 1–118, 2023.

[32] T. Koller, F. Berkenkamp, M. Turchetta, and A. Krause, "Learning-based model predictive control for safe exploration," in *Proc. IEEE Conf. Decis. Control (CDC)*, Piscataway, NJ, USA: IEEE Press, 2018, pp. 6059–6066.

[33] A. Nagabandi, G. Kahn, R. S. Fearing, and S. Levine, "Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, 2018, pp. 7559–7566.

[34] J. Ibarz, J. Tan, C. Finn, M. Kalakrishnan, P. Pastor, and S. Levine, "How to train your robot with deep reinforcement learning: Lessons we have learned," *Int. J. Robot. Res.*, vol. 40, nos. 4–5, pp. 698–721, 2021.

[35] G. G. Muscolo and P. Fiorini, "Force–torque sensors for minimally invasive surgery robotic tools: An overview," *IEEE Trans. Med. Robot. Bionics*, vol. 5, no. 3, pp. 458–471, Aug. 2023.

[36] A. Ray, J. Achiam, and D. Amodei, "Benchmarking safe exploration in deep reinforcement learning," 2019, *arXiv:1910.01708*.

[37] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*.

[38] J. Achiam, D. Held, A. Tamar, and P. Abbeel, "Constrained policy optimization," in *Proc. Int. Conf. Mach. Learn.*, PMLR, 2017, pp. 22–31.