This paper appears in Journal of Medical Robotics Research, 2024.

Autonomous Soft-Tissue Needle Steering Using Reinforcement Learning Guided by Human Input

Yafei Ou, Mahdi Tavakoli

Department of Electrical and Computer Engineering, University of Alberta, 9211-116 Street NW, Edmonton, AB, T6G 1H9, Canada
E-mail: yafei.ou@ualberta.ca (corresponding author); mahdi.tavakoli@ualberta.ca

Soft-tissue needle steering, where a deformable needle is inserted into the tissue to guide its tip to a desired position, is a common minimally invasive surgery (MIS) procedure. The diverse types of needles and complex tissue dynamics limit the use of existing approaches that utilize models of the needle and the tissue for automating the task. In this work, we employ a data-driven approach using deep reinforcement learning (DRL) to achieve autonomous needle steering by viewing it as a multi-goal reinforcement learning problem. Human interventions are incorporated during training to accelerate learning and reduce catastrophic failures. Generative adversarial imitation learning (GAIL) is combined with regular DRL by utilizing a hindsight relabeling scheme for human interventions to encourage the agent to imitate human behavior.
To emulate the sim-to-real process, an agent is first trained in a simplistic simulation environment for needle steering and then transferred to a sophisticated one considered as the real world with fine-tuning (sim-to-sim). Experimental results show that with human interventions, the proposed method outperforms the other compared DRL approaches and can achieve good performance with only 2,000 training steps in the complex simulation environment, achieving an average return comparable to that of a 55,000-step agent trained from scratch.

Keywords: Surgical automation, flexible needle steering, reinforcement learning, learning from demonstration.

1. Introduction

Soft-tissue needle steering is a common minimally invasive surgery (MIS) procedure for purposes such as biopsy and brachytherapy. During the process, typically a beveled-tip needle is inserted in soft tissue and steered such that the needle tip is guided to a target location [1]. This procedure usually requires a high level of accuracy to achieve satisfactory surgical outcomes. However, during this process, needle deformation is caused by the interaction force between the needle and the tissue, making the guidance of the needle tip to the desired location challenging. In addition, the diverse properties of needles and tissues in practice make the task even more difficult for surgeons.

1.1. *Recent Advances in Autonomous Needle Steering*

A number of studies have attempted to automate needle steering or assist the surgeon during the procedure. One typical solution is to develop control strategies based on the modeling of needle-tissue interaction to predict the needle deflection in soft tissue. Various needle-tissue interaction models have been developed for designing needle steering control strategies. One simple yet commonly used approach is to model the needle tip path inside the tissue as a curvature produced by a unicycle or bicycle [2]. By employing the unicycle needle path model, Rucker et al. implemented a sliding mode controller for controlling the needle tip position [3], and Khadem et al. designed a two-step controller that first stabilizes the system on an equilibrium manifold and then on an equilibrium point [4]. Carriere et al. utilized the bicycle model and designed an event-triggered controller for steering the needle [5].

However, this kinematic modeling approach ignores tissue deformation caused by needle insertion that can result in additional force being applied to the needle. As a result, the needle tip path may deflect from what is predicted by the model. Some studies aim to address this problem by using mechanical models or finite-element approaches to take into account the dynamics of the needle-tissue interaction [6–8]. For instance, the local contact force between the needle and the tissue is modeled as linearly dependent on the magnitude of local tissue deformation in [6] and [8].

One limitation of the aforementioned methods is that

the models are usually specific to a fixed set of needle and tissue properties. Furthermore, tissue non-homogeneity is not considered in these modeling methods. While some adaptive modeling approaches such as [9] can adapt to some variation of the needle and tissue, model-based methods generally lack the ability to account for diverse needle and tissue behavior such as tissue anisotropy and needle buckling. Therefore, an alternative is to use data-driven approaches in order to achieve better generalizability. For example, in [10], a dataset with measurements of the needle behavior is first collected from several insertions and a Just-in-Time learning method is employed to predict the needle deflection and contact force during the insertion.

1.2. *Deep Reinforcement Learning for Needle Steering*

Deep reinforcement learning (deep RL, DRL) is a data-driven approach that has recently been extensively studied and applied in the field of automation and control, largely due to its high generalizability and low demand for human knowledge compared to traditional motion planning and control approaches. Recent works have shown promising results in automating some surgical tasks using DRL approaches [11–15]. While these results demonstrate the potential of applying DRL to soft-tissue needle steering, the extensive amount of interaction data from the environment required for training the RL agent is one of the major disadvantages of this method, due to the fact that interacting with the soft tissue in the real world to collect data is extremely expensive and impractical.

One common approach to mitigating the aforementioned sample efficiency problem is simulation-to-reality (sim-to-real) transfer, where a policy is first trained in a simulated environment by collecting artificial experiences in the simulator, and is then transferred to the real environment. In [16], a needle steering policy is first trained in the simulated environment reconstructed from segmented CT scan images, then transferred to the real world. Similarly, authors in [17] utilized MR angiography images to reconstruct a simulated environment and trained a DRL policy for flexible needle path generation. While these studies show the potential of training a policy in the simulator for direct use in the real world, it is important that the simulation environment is close to reality to ensure that the trained policy performs well in the real world. However, this is not always easy to be guaranteed in practice, due to the errors introduced by reconstruction, tissue uncertainty and inhomogeneity, and diverse needle properties. Therefore, the task of needle steering is particularly prone to the problem of sim-to-real gap, where the policy does not perform as well in reality as in simulation.

In general, it is common to further fine-tune the trained RL agent based on real-world experiences to increase task performance in the real world if the real-world environment differs from the simulation. For instance, in [18], an RL agent for robot grasping is first trained using a visual sim-to-real framework and is then further fine-tuned by mixing real online data from the real robot with simulated data. In [19], better performance is achieved after fine-tuning a robot grasping policy in the real world by utilizing progressive neural networks, compared with using other neural networks. However, online fine-tuning with real-world data, or more generally, exploring in the real world is traditionally considered impractical in surgical tasks including needle steering, since it is safety critical when making explorations in realistic surgical environments, such as using cadavers, as explorations for training the agent can cause significant damage to the environment. Without real-world explorations, bridging the sim-to-real gap has been more challenging for surgical tasks.

To make exploration possible in the real-world environment, one simple yet promising approach is utilizing real-time interventions from human experts during training. Prior works have shown that human interventions can prevent or reduce catastrophic failures during training, and accelerate the learning speed by providing appropriate guidance. In [20], a straightforward training mechanism is proposed where the human monitors the training process and overwrites agent actions in dangerous circumstances. In [21], proximal policy optimization (PPO) is modified to incorporate real-time human interventions and accelerate training by adding a behavior cloning (BC) loss. With improvements, this approach is further extended to off-policy algorithms in [22] and [23]. Recently, this training scheme has been applied to surgical robot learning as well [24–26]. In [26], an algorithm is proposed to leverage real-time human interventions by combining regular RL with generative adversarial imitation learning (GAIL). All of these works consider the regular RL situation with a single objective. However, in the needle steering task, there can be multiple different desired locations for the needle tip, making it a multi-goal reinforcement learning task that is more challenging to solve.

1.3. *Objective and Contributions*

In this work, we consider the needle steering task as a multi-goal reinforcement learning problem, where the learned RL policy is able to guide the needle tip to multiple different target locations in the soft tissue. An RL agent is first trained in a simplistic simulator and then transferred to a more sophisticated simulation environment considered as the real world by fine-tuning. A DRL framework that incorporates human interventions is utilized by combining regular RL with GAIL for the multi-goal RL setting. By employing human interventions, fine-tuning in the real-world environment is safer and more efficient. The main contributions of this work are: (a) we present a novel DRL training approach that utilizes human interventions for learning multi-goal needle steering; (b) we automate soft-tissue needle steering by training an RL agent first in a simplistic simulator and then safely transfer it to a sophisticated simulation environment viewed as the real world by utiliz-

ing the proposed human-guided training approach; (c) we validate the proposed method can achieve safe and efficient transfer of a needle steering policy with human interventions.

The remainder of the paper is organized as follows. Section 2 outlines the mathematical preliminaries. Section 3 discusses the two simulation environments for needle steering. Section 4 introduces the proposed training methodology, followed by Section 5 which presents the experimental settings and results. Discussions on the proposed method and the results are provided in Section 6. Section 7 concludes the paper with remarks on potential future work.

2. Background

2.1. *Notations in RL*

RL typically considers the problem of a Markov decision process (MDP) described by a five-tuple $\langle \mathcal{S}, \mathcal{A}, P, r, \gamma \rangle$. $\mathcal{S}$ is the state space that consists of the state variables $\mathbf{s}$. $\mathcal{A}$ is the action space, with $\mathbf{a}$ being the corresponding action variables. $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the state transition function that maps a state-action pair $(\mathbf{s}_t, \mathbf{a}_t)$ at time step t to the next state $\mathbf{s}_{t+1}$. $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function that maps a state-action pair $(\mathbf{s}_t, \mathbf{a}_t)$ to a reward value. $\gamma \in [0, 1]$ is the discount factor.

2.2. *Soft Actor-Critic (SAC)*

Soft actor-critic (SAC) [27] is an off-policy actor-critic algorithm and is the primary RL algorithm used in this work. Actor-critic algorithms exploit the actor-critic structure where the actor $\pi_\phi(\mathbf{a}_t|\mathbf{s}_t)$ is the policy parameterized by ϕ that generates actions $\mathbf{a}_t$ from given states $\mathbf{s}_t$, and the critic is a function approximator $Q_\theta(\mathbf{s}_t, \mathbf{a}_t)$ parameterized by θ for estimating the state-action value function. In off-policy actor-critic algorithms, the target policy that is optimized through learning is different from the behavior policy used to collect the experiences during the exploration. In this case, a dataset $\mathcal{R}$ is usually employed for storing and retrieving previously collected experiences. The process is called experience replay and the dataset $\mathcal{R}$ is the experience replay buffer.

In SAC, the critic is optimized by minimizing the Bellman residual

$$J_Q(\theta) = \mathbb{E}_{(\mathbf{s}_t, \mathbf{a}_t) \sim \mathcal{R}} \left[\frac{1}{2} \left(Q_\theta(\mathbf{s}_t, \mathbf{a}_t) - \hat{y}_t \right)^2 \right] \tag{1}$$

to approximate the soft Q-value, where

$$\hat{y}_t = r(\mathbf{s}_t, \mathbf{a}_t) + \gamma \mathbb{E}_{\mathbf{s}_{t+1} \sim p}[V_\theta(\mathbf{s}_{t+1})] \tag{2}$$

is the temporal difference (TD) target, with

$$V_\theta(\mathbf{s}_t) = \mathbb{E}_{\mathbf{a}_t \sim \pi}[Q_\theta(\mathbf{s}_t, \mathbf{a}_t) - \alpha \log(\pi(\mathbf{a}_t|\mathbf{s}_t))] \tag{3}$$

being the soft state value function. α is a weighting factor. The actor is then optimized by maximizing the estimated soft Q-value and the policy entropy:

$$J_\pi(\phi) = \mathbb{E}_{\mathbf{s}_t \sim \mathcal{R}} \left[\mathbb{E}_{\mathbf{a}_t \sim \pi_\phi} \left[-Q_\theta(\mathbf{s}_t, \mathbf{a}_t) + \alpha \log(\pi(\mathbf{a}_t|\mathbf{s}_t)) \right] \right] \tag{4}$$

2.3. *Multi-Goal RL using Hindsight Experience*

The concept of multi-goal reinforcement learning, also known as goal-conditioned reinforcement learning (GCRL), refers to the problem of learning policies that are capable of achieving multiple objectives. GCRL considers a goal-augmented MDP (GA-MDP) [28] with an additional tuple $\langle \mathcal{G}, p_g, \phi_g \rangle$, where $\mathcal{G}$ is the goal space that consists of the corresponding goal variables $\mathbf{g}$, p_g is the distribution of the desired goal. $\phi_g : \mathcal{S} \to \mathcal{G}$ maps a state to a goal, and can be an identity mapping function if $\mathcal{S}$ and $\mathcal{G}$ are identical. The reward function $r : \mathcal{S} \times \mathcal{A} \times \mathcal{G} \to \mathbb{R}$ in GCRL also takes into account the desired goal. Compared with the regular situation where the reward function is fixed for one single objective, GCRL is particularly useful in multi-objective learning settings such as navigating a robot to any desired location in a 2D space.

Regular RL algorithms can be applied directly to GCRL problems by augmenting the state observation with the desired goal, and a general policy $\pi : \mathcal{S} \times \mathcal{G} \times \mathcal{A} \to [0, 1]$ that also considers the desired goal can be learned. However, learning a goal-conditioned policy is usually more challenging due to the fact that it is much more difficult for the agent to encounter successes during the exploration since the goal is resampled from $\mathcal{G}$ in each episode, and the reward function is usually sparse. One common approach to addressing the problem is using hindsight experience replay (HER) [29] to relabel the desired goals of the unsuccessful trajectories with the goals achieved by the agent in the same trajectory. Specifically, a transition $(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1}, \mathbf{g}, r_t = 0)$ can be transformed into $(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1}, \mathbf{g}', r_t = 1)$ where

$$\mathbf{g}' \in \{\phi_g(\mathbf{s}_0), \phi_g(\mathbf{s}_1), \ldots, \phi_g(\mathbf{s}_T)\} \tag{5}$$

with T being the horizon, and $\phi_g(\mathbf{s}_0), \phi_g(\mathbf{s}_1), \ldots, \phi_g(\mathbf{s}_T)$ are called the achieved goals. As a result, unsuccessful trajectories under the original desired goal are transformed into successful ones under another goal, making the training of a goal-conditioned policy tractable.

3. Simulation Environments for Soft-Tissue Needle Steering

Soft-tissue needle steering can be considered as a multi-goal RL task, where the goals are the different target needle tip locations [16]. The task involves diverse environment dynamics, depending on the physical properties of the needle and the tissue. We build two different simulation environments for needle steering. The first one is based on the kinematic bicycle model [2] which is less accurate but fast

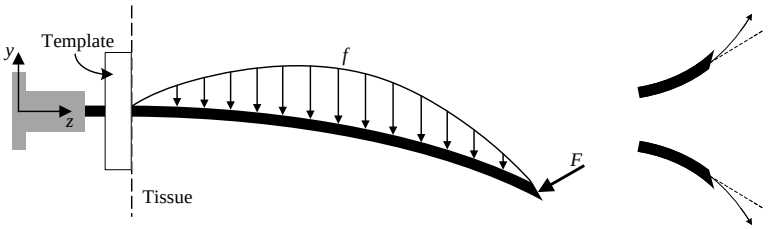


Fig. 1. Illustration of the needle insertion process and the two possible orientations of the bevel.

to solve, and the other one is based on a quasi-static cantilever beam model [30], which incorporates the mechanical properties of the tissue and the needle such as the tissue stiffness and needle tip cutting forces, making the model more accurate and close to reality, but more computationally demanding. To show the effectiveness of the proposed method, we view the first environment as a simplistic simulator and the second one as a sophisticated one that simulates the real world, and consider the problem of transferring a trained model from the simplistic simulation to the sophisticated one. This process simulates the problem of sim-to-real transfer, where here a more sophisticated simulation environment is considered as the real-world environment. For simplicity, we refer to it as the "real-world simulation".

The bicycle model is considered simplistic since it does not consider the coupled effects caused by needle deflection and tissue deformation, which is taken into account in the cantilever beam model to make needle deflection modeling more accurate. Additionally, the choice of these two models is related to their computational complexities, as the bicycle model is very fast to solve, making it suitable for fast pre-training. On the other hand, solving the cantilever beam model is time-consuming and is suitable for resembling a real-world environment where the cost of making explorations in it is high.

The simplistic simulation environment assumes that the needle tip bends and follows a path of constant curvature similar to the trajectory of a bicycle depending on the orientation of the needle tip, and models the 2D motion of the needle tip as

$$\dot{z}_e = v\cos\alpha_e, \quad \dot{y}_e = v\sin\alpha_e, \quad \dot{\alpha}_e = v\,b\,\kappa \tag{6}$$

where v is the velocity of insertion, z_e and y_e are the position of the needle tip along the Z and Y axis in the 2D plane, and α_e is the rotational angle of the needle tip. κ is the needle path curvature determined by the physical properties of the needle and the tissue. $b = \pm 1$ is dependent on the two possible orientations of the bevel, i.e. up or down, as shown in Fig. 1. Rotating the needle for 180° around the Z axis changes the sign of b and the future path of the needle tip.

The real-world simulation utilizes a mechanical model by viewing the needle as a cantilever beam. The deflection of the needle at the current insertion depth d is defined as the weighted summation of n deflection functions q_i representing the first n modes of vibration $v(d,z) = \sum_{i=0}^{n} g_i(d)q_i(z)$, where $v(d,z)$ is the deflection of the needle point whose Z-coordinate is z, $g_i(d)$ and $q_i(z)$ are the weighting coefficient and eigenfunctions for each of the vibration modes. For each insertion depth, $g_i(d)$ can be solved by n linear equations, each of which is defined as

$$\sum_{j=1}^{n}(EI\Psi_{ji} + K\Omega_{ji} + K_p\Gamma_{ji})g_j(d) - K\Phi_{ji} = F \tag{7}$$

for $i = 1$ to n, where

$$\begin{aligned} \Psi_{ij} &= \int_0^L \ddot{q}_i(z)\ddot{q}_j(z)\,\mathrm{d}z, \quad \Omega_{ij} = \int_{L-d}^{L} q_i(z)q_j(z)\,\mathrm{d}z \\ \Gamma_{ij} &= q_i(z_t)q_j(z_t), \quad \Phi_{ij} = \int_0^d g_i(d-\tau)q_j(\tau)\,\mathrm{d}\tau \end{aligned} \tag{8}$$

Here, L is the total length of the needle, and $z_t = L - d - c_t$ with c_t being the distance between the needle template and the tissue surface, F is the force applied by the tissue at the needle tip during the insertion, K is stiffness of the tissue, E and I are the Young's modulus and the second moment of inertia of the needle, respectively. The position and orientation of the needle tip can be obtained by

$$z_e \approx d, \quad y_e = v(d,L), \quad \alpha_e = \frac{\partial v}{\partial z}(d,L) \tag{9}$$

Both simulation environments are formulated in the reinforcement learning setting, with the state being the needle tip's location and orientation in the 2D plane, and the current orientation of the bevel (i.e. upward or downward):

$$\mathbf{s}_t = [z_e(t), y_e(t), \alpha_e(t), b(t)] \tag{10}$$

where t is the time step. The two actions are the change of insertion depth at each step $\Delta d(t)$, and the orientation of the bevel during the next step:

$$\mathbf{a}_t = [\Delta d(t), b(t+1)] \tag{11}$$

Similar to [30], we utilize a gradual needle tip force change when the needle is rotated for 180° while using the cantilever beam model to avoid the sudden change of the needle tip position during rotation. During the action step when rotation happens, the needle tip force F gradually decreases from F to reach $-F$. However, this can still yield sharp turns during the rotation due to the change in the beam's shape, as shown in Fig. 2b.

The goal of the task is to place the needle tip at a goal location $\mathbf{g} = [z_g, y_g] \in \mathcal{G}$. During each training episode, the goal position of the needle tip is randomly sampled from

a rectangular area, resulting in a multi-goal reinforcement learning problem.

Since both simulation environments use different approaches to modeling needle deflection, their underlying environment dynamics are different, making it challenging to transfer control policies learned from the simplistic environment to the real-world simulation directly. To make the task even more challenging and close to the real-world application, we add an obstacle in the real-world simulator that is not present in the simplistic one, as shown in Fig. 2. While adding the same obstacle to the simplistic simulator and training with obstacles in the simplistic simulation could result in learning a better policy as the difference between both simulators is smaller, we deliberately present the obstacle only in the real-world simulator to simulate a larger gap between simulation and the real world. In reality, this corresponds to scenarios where such obstacles in the real world are unknown in advance and cannot be reconstructed in the simulation environment. The reward in the bicycle model environment is

$$r_t^b(\mathbf{s}_t, \mathbf{a}_t, \mathbf{g}) = r^{succ} + \omega_1 r^{rot} \tag{12}$$

where ω_1 is a weighting factor and

$$r^{succ} = \begin{cases} 0, & \text{if } \|\mathbf{p}_e - \mathbf{g}\| \leq \epsilon \\ -1, & \text{otherwise} \end{cases} \tag{13}$$

with $\mathbf{p}_e = [z_e(t), y_e(t)]$ being the positions of the needle tip, and

$$r^{rot} = \begin{cases} -1, & \text{if the orientation of the bevel changes} \\ 0, & \text{otherwise} \end{cases} \tag{14}$$

This term punishes the action of rotating the needle to change the orientation of the bevel. Without this punishment term, the learned policy could result in a constant rotation of the needle, which is not desired and infeasible in real practice. While it is possible to design a dense reward function based on the distance between the needle tip and the goal position, this design will require much manipulation of the reward function to work because the insertion depth along the Z axis is much larger than the needle deformation along the Y axis, and a temporary large deviation of the needle tip from the goal position along the Y axis does not necessarily indicate low performance. To avoid a complex reward function design process, a sparse reward is used in this work.

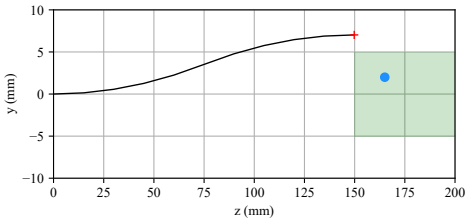
The reward function for the real-world simulation is

$$r_t^c(\mathbf{s}_t, \mathbf{a}_t, \mathbf{g}) = r^{succ} + \omega_1 r^{rot} + \omega_2 r^{obstacle} \tag{15}$$

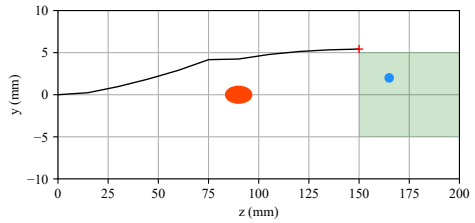
where

$$r^{obstacle} = \begin{cases} -20, & \text{if the needle tip hits obstacle} \\ 0, & \text{otherwise} \end{cases} \tag{16}$$

In practice, the needle tip is considered as hitting the obstacle if its position is inside the obstacle. The two simulation environments are shown in Fig. 2.



(a)



(b)

Fig. 2. Simulation environments. (a) Simplistic environment based on the bicycle model; (b) Real-world simulator based on the quasi-static cantilever model. The red region is the obstacle. The goal positions (blue points) are sampled from the region marked in green.

4. Fine-Tuning an RL Agent with Human Interventions

4.1. *Human Interventions in Multi-Goal RL*

While prior works have shown that intermittent human interventions during RL training can be used as guidance for accelerating training and preventing catastrophic failures, these studies focus on the usage of human guidance in regular single-goal settings. However, applying RL algorithms designed for single-goal situations to the multi-goal case usually results in unsatisfactory training speed and results, primarily because of the large goal space [29, 31]. In this work, we implement a novel training scheme with human interventions for GCRL, where human interventions are incorporated into the training of an RL agent by combining the regular off-policy actor-critic methods with GAIL and utilizing a relabeling scheme for human actions.

In human-guided RL, a human expert monitors the training process and provides intermittent guidance by directly overwriting the agent's actions, and the actual action taken during training can be expressed by

$$\mathbf{a}_t = \mathcal{I}(\mathbf{s}_t, \mathbf{g})\mathbf{a}_t^H + (1 - \mathcal{I}(\mathbf{s}_t, \mathbf{g}))\mathbf{a}_t^A \tag{17}$$

where $\mathbf{a}_t^H$ is the action proposed by the human, and $\mathbf{a}_t^A$ is the action chosen by the agent. $\mathcal{I}(\mathbf{s}_t, \mathbf{g}) \in \{0, 1\}$ is a switching function determined by the human.

Similar to [26], we incorporate the human guidance by training a discriminator $D_\varphi(\mathbf{s}_t, \mathbf{a}_t)$ jointly with the RL training process for discriminating between human and

agent actions, whose parameters φ is trained by minimizing the loss

$$J_D(\varphi) = \mathbb{E}_{(\mathbf{s}_t,\mathbf{a}_t,\mathbf{g}_t)\sim\mathcal{R}_H}\left[\log D_\varphi(\mathbf{s}_t,\mathbf{a}_t,\mathbf{g}_t)\right] + \mathbb{E}_{(\mathbf{s}_t,\mathbf{a}_t,\mathbf{g}_t)\sim\mathcal{R}_A}\left[\log(1-D_\varphi(\mathbf{s}_t,\mathbf{a}_t,\mathbf{g}_t))\right] \tag{18}$$

where $\mathcal{R}_H$ is the experience replay buffer that stores the trajectories where human intervenes, and $\mathcal{R}_A$ is the one that stores trajectories relating to agent actions. The discriminator is used for predicting whether an action is close to the human's behavior and can be used to guide the training by incorporating the predicted value as an additional reward when updating the critic:

$$r_t^{train} = (1-\beta)r_t^{env} + \beta\, r_t^{GAIL} \tag{19}$$

where $r_t^{GAIL} = \log D_\varphi(\mathbf{s}_t,\mathbf{a}_t,\mathbf{g}_t) - 1$ is the GAIL reward and β is a weighting factor that can be gradually decreased during the training process. The critic can be updated using (1) with the augmented reward.

Furthermore, an additional imitation loss term should be added to the policy loss (4):

$$J_\pi(\phi) = \mathbb{E}_{(\mathbf{s}_t,\mathbf{g}_t)\sim\mathcal{R}}\left[\mathbb{E}_{\mathbf{a}_t\sim\pi_\phi}\left[-Q_\theta(\mathbf{s}_t,\mathbf{a}_t,\mathbf{g}_t) + \alpha\log(\pi(\mathbf{a}_t|\mathbf{s}_t,\mathbf{g}_t)) - \omega\log D_\varphi(\mathbf{s}_t,\mathbf{a}_t,\mathbf{g}_t)\right]\right. \tag{20}$$

where ω is a weighting term that can be gradually decreased, and $\mathcal{R} = \mathcal{R}_H \cup \mathcal{R}_A$ is the replay buffer that stores all trajectories. The general training framework for multi-goal RL with human guidance is shown in Fig. 3.

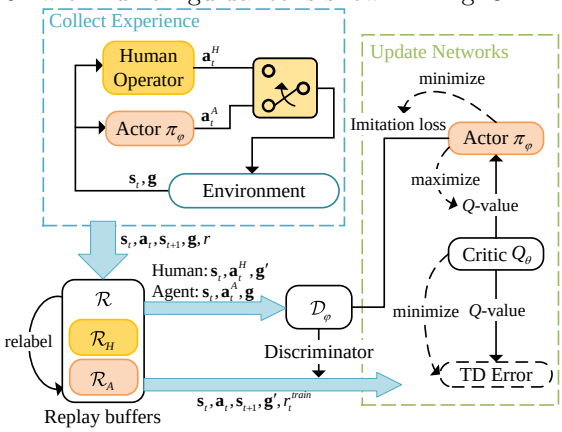


Fig. 3. Framework of the human-guided multi-goal RL scheme.

To further enrich the collected experiences, hindsight relabeling is applied when sampling from $\mathcal{R}$ and $\mathcal{R}_H$. Relabeling the goals when sampling from $\mathcal{R}$ can be easily realized without modifying the existing HER algorithm by viewing experiences caused by both the agent and the human jointly as a whole dataset. Relabeling the goals when sampling from $\mathcal{R}_H$ is performed when training the discriminator, and is achieved by assuming that the human's interventions are optimal not only for the actual desired goal, but also for all the achieved ones along the intervention. Consider a rollout trajectory from time step $t = 0$ to T, and assume that human intervention happens from $t = m$ to $m + k$ steps and the desired goal is $\mathbf{g}$. Each transition

$$(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1}, \mathbf{g}, r_t) \tag{21}$$

with $t = \{m, m+1, \ldots, m+k\}$ can be transformed into

$$(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1}, \mathbf{g}', r_t') \tag{22}$$

where

$$\begin{aligned} \mathbf{g}' &\in \{\phi_g(\mathbf{s}_m), \phi_g(\mathbf{s}_{m+1}), \ldots, \phi_g(\mathbf{s}_{m+k})\} \\ r_t' &= r(\mathbf{s}_t, \mathbf{a}_t, \mathbf{g}') \end{aligned} \tag{23}$$

In practice, r_t' is not required when training the discriminator and can be ignored, and the "future" strategy is employed where the relabeled goal is only sampled using states after the current transition. The hindsight relabeling scheme is shown in Fig. 4.

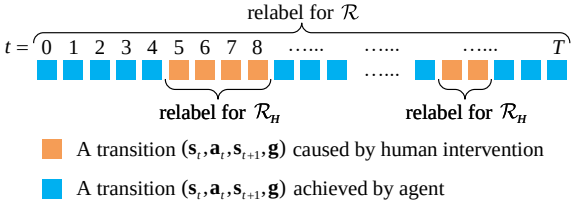


Fig. 4. Hindsight relabeling during the training process.

4.2. *Fine-Tuning with Human Input*

As discussed in Section 3, we consider the problem of transferring a policy trained in a simplistic simulation environment to the real-world simulation environment considered as the real world, which simulates the process of sim-to-real transfer. Since the underlying environment dynamics and the reward function are different, fine-tuning the policy online with new explorations in the real-world simulation is an efficient way for it to adapt to the new environment. However, while there is no need to ensure safety when training in a simulator, it is undesirable that dangerous actions are taken during fine-tuning in the real world as they may cause damage to the environment. It is also important that the fine-tuning process is efficient and does not require a large number of explorations since it is impractical in the real world, especially in the surgical task setting. For this purpose, we incorporate human interventions during fine-tuning, as discussed in Section 4.1. As a result, a regular agent with π_ϕ and critic Q_θ is first trained in the simplistic simulation environment, and further fine-tuned by exploring online in the real-world simulation environment, during which human guidance is incorporated to enable safer and faster learning. The overall fine-tuning procedure is summarized in Algorithm 4.1.

Algorithm 4.1. Fine-tuning for multi-goal RL with human input

1: Load pre-trained actor π_ϕ, critic Q_θ
2: Initialize discriminator D_φ

Initialize empty replay buffers $\mathcal{R}_H$, $\mathcal{R}_A$, $\mathcal{R} \triangleq \mathcal{R}_H \cup \mathcal{R}_A$
for each iteration **do**
  $\mathbf{g} \sim \texttt{Uniform}(\mathcal{G})$
  for each environment step **do**
    $\mathbf{a}_t^A \sim \pi_\phi(\mathbf{s}_t, \mathbf{g})$
    if human intervenes **then**
      $\mathcal{I}(\mathbf{s}_t, \mathbf{g}) = 1$
      sample $\mathbf{a}_t^H$
    else
      $\mathcal{I}(\mathbf{s}_t, \mathbf{g}) = 0$
    end if
    $\mathbf{a}_t \leftarrow \mathcal{I}(\mathbf{s}_t, \mathbf{g})\mathbf{a}_t^H + (1 - \mathcal{I}(\mathbf{s}_t, \mathbf{g}))\mathbf{a}_t^A$
    $\mathbf{s}_{t+1} \sim p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$
    $r_t \leftarrow r(\mathbf{s}_t, \mathbf{a}_t, \mathbf{g})$
    if $\mathcal{I}(\mathbf{s}_t, \mathbf{g}) = 1$ **then**
      $\mathcal{R}_H \leftarrow \mathcal{R}_H \cup \{(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1}, \mathbf{g}, r_t)\}$
    else
      $\mathcal{R}_A \leftarrow \mathcal{R}_A \cup \{(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1}, \mathbf{g}, r_t)\}$
    end if
  end for
  if train discriminator now **then**
    for each discriminator gradient step **do**
      Sample $\{(\mathbf{s}_t^i, \mathbf{a}_t^i, \mathbf{s}_{t+1}^i, \mathbf{g}^i, r_t^i)\} \sim \mathcal{R}_H$
      for each i with probability $p_{relabel}$ **do**
        $j^i \leftarrow \texttt{Uniform}(\{m^i + 1, \ldots, m^i + k\})$
        $\mathbf{g}^i \leftarrow \phi_g(\mathbf{s}_{j^i})$
      end for
      Update $\mathcal{D}_\varphi$ using Equation (18)
    end for
  end if
  for each policy gradient step **do**
    Sample $\{(\mathbf{s}_t^i, \mathbf{a}_t^i, \mathbf{s}_{t+1}^i, \mathbf{g}^i, r_t^i)\} \sim \mathcal{R}$
    for each i with probability $p_{relabel}$ **do**
      $j^i \leftarrow \texttt{Uniform}(\{t^i, t^i + 1, \ldots, T^i\})$
      $\mathbf{g}^i \leftarrow \phi_g(\mathbf{s}_{j^i})$
      Augment r_t^i using Equation (19)
    end for
    Update Q_θ using Equation (1)
    Update π_ϕ using Equation (20)
  end for
end for

5. Experiments and Results

5.1. *Training settings*

The fixed parameters for simulating needle insertion using the bicycle model and the quasi-static cantilever model are listed in table 1. These values are either directly taken or derived from [5] and [30] in order to represent the actual needle and tissue contact models found in previous research.

A regular SAC agent is first trained in the simplistic simulation environment that uses the bicycle model for 30,000 steps, and saved for further fine-tuning in the cantilever model-based simulator. During fine-tuning, the human interventions are achieved using a joystick controller. A total of 2,000 steps are trained during the fine-tuning phase, where a maximum of 200 steps of human interventions are allowed.

Table 1. Parameters for Simulation

Parameter	Value	Parameter	Value
l [mm]	200	I [$\text{m}^4 \times 10^{-13}$]	77.5
k [m]	1/800	K [kNm^{-2}]	59.8
E [GPa]	200	K_p [Nm^{-1}]	10^9
F [N]	0.5		

The implementation of SAC is based on [27], where two critic networks and two target networks are utilized to stabilize training. The actor and the critic networks are two-layer multilayer perceptron (MLP) networks with 256 hidden units at each layer. The learning rate is 3×10^{-4} and the batch size is 256. After pre-training the model in the simplistic environment, the optimizers for the actor and critic networks are saved and loaded before fine-tuning. The discriminator is a two-layer MLP with 100 hidden units at each layer, and the learning rate is 10^{-3}. The discriminator is trained for 5 epochs every 100 rollout steps with a batch size of 32. The relabeling ratio $p_{relabel}$ is 0.8. The GAIL reward weighting factor β is 0.1 and decays exponentially. Training is performed on a CPU device equipped with an Intel(R) Core(TM) i5-12400 processor. GPUs are not particularly helpful in our case, since the majority of the training time is spent solving equations for simulation, which relies heavily on the CPU.

To examine the effectiveness of human guidance in fine-tuning, we compare the training results with those obtained when fine-tuning the pre-trained agent without human guidance. For the agent without human guidance, 3 training instances are carried out using 3 different random seeds during fine-tuning. For the agent with human guidance, the same 3 random seeds are used, and for each random seed, 3 instances are trained to take into account the variance in human behavior. Therefore, a total of 9 training instances are performed. As a comparison, we also train an agent from scratch in the real-world simulator for 55,000 steps.

5.2. *Results*

Fig. 5 shows the learning curve during the pre-training phase in the original simplistic environment. After training for 30,000 steps, the model can achieve an average return of -15.53 in the original environment. The goal-reaching rate is 98% and the average number of needle rotations is 3.08 (not shown in the figures).

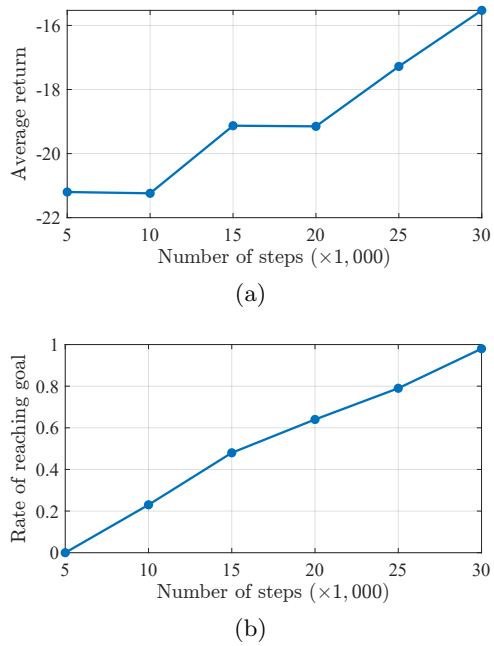


Fig. 5. Learning curves during pre-training in the simplistic simulator, evaluated for 100 episodes every 5,000 training steps: (a) Average return; (b) Rate of reaching the goal position.

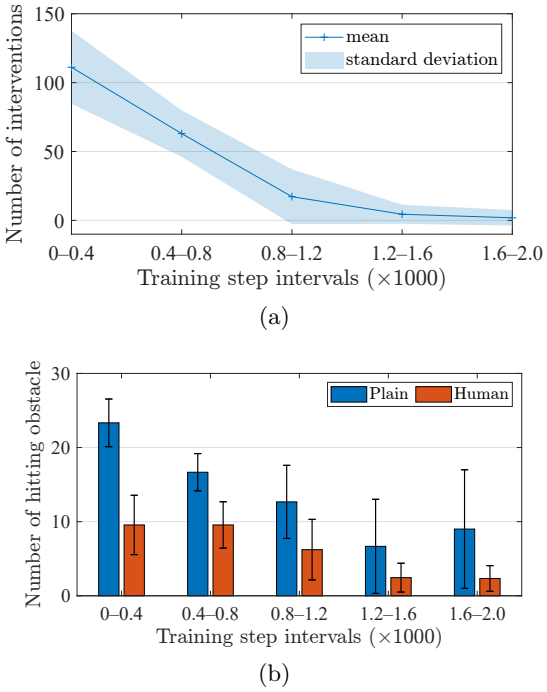
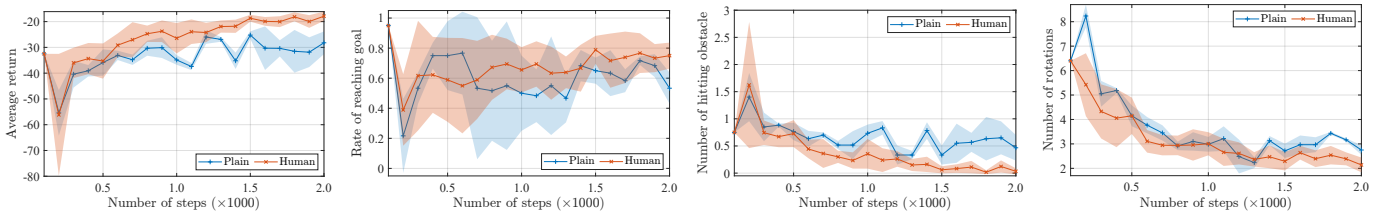


Fig. 6. (a) Number of interventions and (b) Number of hitting the obstacle during the fine-tuning. Data are grouped based on a 400-step interval.

Fig. 7 shows the performances of the two RL agents with and without human guidance throughout the fine-tuning process in the real-world simulation. The agents are evaluated every 100 training steps for 20 episodes (i.e. 20 insertions). As shown in the figures, the pre-trained model can achieve a high rate of reaching the goal in the initial phase without fine-tuning (100 steps), but the average return is low due to frequently hitting the obstacle and a large number of needle rotations, indicating that the pre-trained model cannot perform well in the new environment directly. By collecting new experiences in the new environment, both agents are able to improve their performances within 2,000 training steps. However, the agent with human guidance achieves a much better performance throughout the process, and the learning is generally faster compared with the one without human guidance. After training for 2,000 steps with human guidance, the agent is able to achieve an average return of around -18, while the one without human guidance only achieves around -28. The agent with human guidance has better performances regarding both the rate of reaching the goal and the number of needle rotations, and it reaches around 75% rate of reaching the goal (compared to 53% without human guidance), and an average of 2.13 needle rotations, compared to 2.75 without human guidance. One-way analysis of variance (ANOVA) using all insertion trials (180 for with human guidance and 60 for without human guidance) shows a statistically significant difference between the number of needle rotations performed by the agents trained with and without human intervention, with the p-value being 2.54×10^{-5}.

Furthermore, the performance regarding the average number of hitting the obstacle is much better when human guidance is incorporated. As shown in Fig. 7c, the average number of hitting the obstacle is still high after training for 2,000 steps for the agent without using human guidance (0.46), while the one with human guidance achieves around only 0.03. These results show that the training process for fine-tuning the pre-trained policy can be accelerated and avoiding significant failures (i.e. hitting the obstacle) can be learned much faster with human guidance. This aligns with our previous findings in [26] where a similar training scheme was used for single-goal regular RL. Fig. 8 shows an example of needle insertion with one specific goal using the two RL agents with and without human interventions trained for different steps. While both agents' performance improves throughout training and eventually reaches the goal with a decreasing number of needle rotations, the performance of the agent with human interventions is relatively better when trained for 1,000 steps, reaching the goal with only 3 needle rotations. In contrast, the agent trained for 1,000 steps without human intervention cannot reach the goal in this example.

The number of human interventions decreases throughout the training process for fine-tuning, as shown in Fig. 6a, which is due to the eventual improvement of the policy and is consistent with the findings of [22]. In addition, to show the effectiveness of human guidance in reducing significant failures during the training (i.e. hitting the obstacle) and increasing safety during exploration, we also



(a) (b) (c) (d)

Fig. 7. Performence during the fine-tuning phase, evaluated for 20 episodes every 100 training steps: (a) Average return; (b) Rate of reaching the goal position; (c) Average number of hitting the obstacle; (d) Average number of needle rotations. The solid lines are the mean values and the shaded areas represent the standard deviations.

record the numbers of hitting the obstacle throughout the human-guided training process and the one without human guidance, as shown in Fig. 6b. Although it is not possible to avoid hitting the obstacle completely due to the slow reaction of the human compared to the simulation step time, this problem can be mitigated by slowing the simulation down and making the needle move at a lower speed each step. Therefore, it remains reasonable to conclude that with human intervention the number of significant failures has been reduced and the exploration is safer.

Additionally, the agent trained from scratch can achieve an average return of -19.3. The rate of reaching the goal is 70%, the average number of hitting the obstacle is 0, and the average number of needle rotations is 2.5. This indicates that by utilizing the fine-tuning strategy where an agent is first trained in a simulator slightly different from the target environment, and then transferred to the target environment with only a few online explorations guided by a human, the agent can reach a comparable (even slightly better) performance than an agent trained from scratch for a large number of steps.

6. Discussions

6.1. *Performance of Pure Human Operation*

It is worth noting that the human who guides the training process is not an expert in completing this task. The human's own performance on this task is evaluated using 20 episodes by directly utilizing the joystick to control the needle's movement in the simulator, and the results show that the human can only achieve an average return of -23.2. The rate of reaching the desired goal is 10%, the average number of hitting the obstacle is 0, and the average number of needle rotations is 2.9. In fact, while the human is good at avoiding the obstacle and limiting the number of needle rotations, they are barely capable of guiding the needle to the desired position. This indicates that even though the human is not competent for completing the task, an RL agent can still benefit from their guidance, especially from their behavior regarding avoiding the obstacle and limiting the number of needle rotations. Furthermore, thanks to the relabeling scheme for human interventions, imperfect human interventions can still be transformed into successful ones under the new goals to help training.

6.2. *Effectiveness of Relabeling Scheme and Discriminator*

We remove the relabeling scheme for human interventions when training the discriminator to investigate its effectiveness and train 2,000 steps for fine-tuning following the same training settings. Furthermore, a modified version of [21] and [22] for SAC (IA-SAC), in which behavior cloning (BC) loss is added to the policy loss in the case of human interventions to encourage the agent to imitate human behavior when human intervention occurs, is also used as a comparison to the proposed method that uses a discriminator. In addition, a naive version of [20] where no modification to the training algorithm is made when human overwrites the agent actions is also implemented for SAC and compared, which is named HI-SAC. Each agent is evaluated for 50 episodes (i.e. 50 insertion trials) after training in the real-world simulation and the results are listed in Table 2. It is shown that without relabeling human interventions for training the discriminator, the performance significantly degrades compared to the originally proposed approach. Additionally, the proposed method achieves better performance compared to IA-SAC and HI-SAC due to the fact that human interventions are conditioned on different goals, and that the human is not competent enough to provide interventions as perfect demonstrations.

6.3. *Connections with Imitation Learning*

Apart from preventing significant failures and dangerous situations in the exploration phase of RL such as hitting an obstacle in this work, human interventions can be further considered as intermittent human demonstrations [23]. The proposed training scheme with human guidance can be viewed as a variation of goal-conditioned imitation learning [31] with an adaptation for intermittent human interventions. While in imitation learning, the human demonstrates the completion of the task before training begins so that the agent can imitate the human, in human-guided

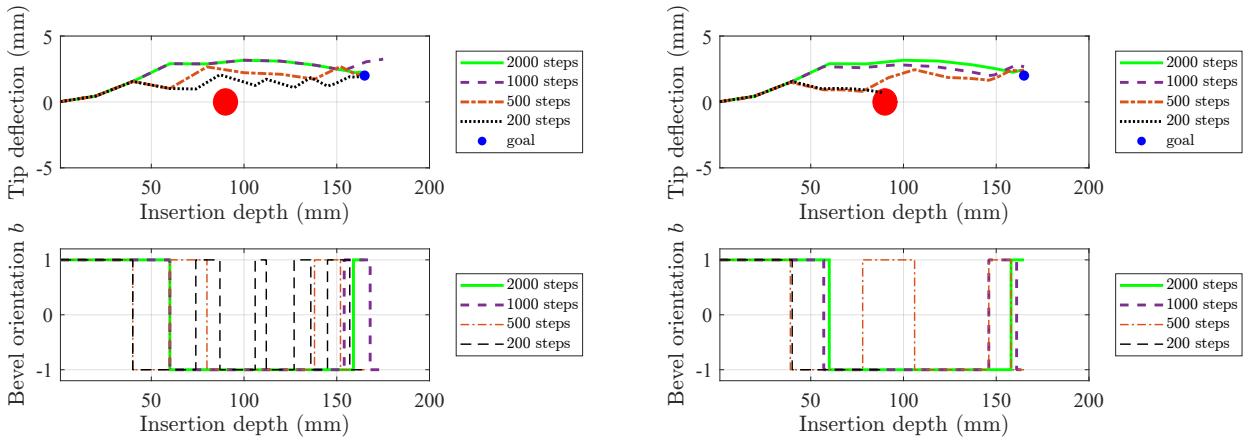


Fig. 8. Insertion examples using the RL agent after training for different steps. (a) without and (b) with human interventions.

Table 2. Agent Performances After Fine-Tuning Using Different Approaches (Each Agent is Evaluated for 50 Trials)

Method	Return	Rate of reaching goal	No. of hitting obstacle	No. of needle rotations
Proposed	$\mathbf{-18.25 \pm 1.06}$	$\mathbf{0.78 \pm 0.08}$	$\mathbf{0.04 \pm 0.03}$	$\mathbf{2.38 \pm 0.14}$
Proposed (w/o human relabeling)	-22.21 ± 1.84	0.58 ± 0.20	0.14 ± 0.08	2.66 ± 0.31
IA-SAC	-21.89 ± 2.30	0.57 ± 0.25	$\mathbf{0.12 \pm 0.07}$	$\mathbf{2.52 \pm 0.10}$
HI-SAC	$\mathbf{-21.46 \pm 2.90}$	$\mathbf{0.70 \pm 0.12}$	0.14 ± 0.13	2.63 ± 0.45

RL, the human supervises the training process and provides intermittent guidance, without necessarily completing the task, as discussed in the Section 6.1.

6.4. *Limitations*

One major limitation of this work is that both the bicycle model and the cantilever beam model are still far from the real-world environment. Both simulation environments are simplified, ignoring factors such as inhomogeneous tissue properties. The cantilever beam model yields unrealistic sharp turns during frequent needle rotations. Therefore, there can be a larger gap between the simulation and the real world in reality, requiring longer training time and more human interventions. Additionally, the observation space may be redesigned to better accommodate the real-world scenario, as observing only the needle tip position and orientation may not be sufficient or be very inaccurate in reality.

Furthermore, as only 2D needle insertion is considered in this work for the sake of simplicity, the environments including how human intervenes should be redesigned when extending the approach to 3D needle insertion. Visualization of the needle insertion procedure in the 3D space and a teleoperation device can be utilized for straightforward human intervention. The action space should incorporate the continuous rotation of the needle from 0 to 360°. The proposed algorithm may not require modification since it can already handle continuous action spaces. However, longer training time and more human interventions may again be the major challenge.

7. Conclusion

In this work, we propose a training framework for learning autonomous soft-tissue needle steering with multiple target positions, by considering a multi-goal RL problem. An RL agent is first trained in a simplistic simulator and transferred to a sophisticated simulation environment that resembles the real world by incorporating human interventions in the fine-tuning phase. It is shown that the proposed method can achieve safer and more efficient learning during the fine-tuning phase, and the learning process requires fewer explorations. By leveraging human interventions, online explorations for fine-tuning are made possible although it is typically considered impractical to conduct explorations in the real world. While this work considers a sophisticated simulation environment as the real world, these results show the potential of applying the approach to real-world experiments. Despite the fact that this work considers 2D needle steering as a proof of concept, the approach can easily be extended to 3D as well. It is therefore possible in the future to conduct real-world experiments of needle steering in the 3D space.

Acknowledgments

This research was supported by the Canada Foundation

for Innovation (CFI), the Natural Sciences and Engineering Research Council (NSERC) of Canada, the Canadian Institutes of Health Research (CIHR), the Alberta Jobs, Economy and Innovation Ministry's Major Initiatives Fund to the Center for Autonomous Systems in Strengthening Future Communities, and the China Scholarship Council (CSC).

References

[1] C. Rossa and M. Tavakoli, Issues in closed-loop needle steering, *Control engineering practice* **62** (2017) 55–69.

[2] R. J. Webster III, J. S. Kim, N. J. Cowan, G. S. Chirikjian and A. M. Okamura, Nonholonomic modeling of needle steering, *The International Journal of Robotics Research* **25**(5-6) (2006) 509–525.

[3] D. C. Rucker, J. Das, H. B. Gilbert, P. J. Swaney, M. I. Miga, N. Sarkar and R. J. Webster, Sliding mode control of steerable needles, *IEEE Transactions on Robotics* **29**(5) (2013) 1289–1299.

[4] M. Khadem, C. Rossa, N. Usmani, R. S. Sloboda and M. Tavakoli, Geometric control of 3d needle steering in soft-tissue, *Automatica* **101** (2019) 36–43.

[5] J. Carriere, M. Khadem, C. Rossa, N. Usmani, R. Sloboda and M. Tavakoli, Event-triggered 3d needle control using a reduced-order computationally efficient bicycle model in a constrained optimization framework, *Journal of Medical Robotics Research* **4**(01) (2019) p. 1842004.

[6] S. P. DiMaio and S. E. Salcudean, Needle insertion modeling and simulation, *IEEE Transactions on robotics and automation* **19**(5) (2003) 864–875.

[7] R. J. Roesthuis, Y. R. Van Veen, A. Jahya and S. Misra, Mechanics of needle-tissue interaction, *2011 IEEE/RSJ international conference on intelligent robots and systems*, IEEE (2011), pp. 2557–2563.

[8] M. Khadem, C. Rossa, R. S. Sloboda, N. Usmani and M. Tavakoli, Ultrasound-guided model predictive control of needle steering in biological tissue, *Journal of Medical Robotics Research* **1**(01) (2016) p. 1640007.

[9] P. Moreira and S. Misra, Biomechanics-based curvature estimation for ultrasound-guided flexible needle steering in biological tissues, *Annals of biomedical engineering* **43** (2015) 1716–1726.

[10] C. Rossa, T. Lehmann, R. Sloboda, N. Usmani and M. Tavakoli, A data-driven soft sensor for needle deflection in heterogeneous tissue using just-in-time modelling, *Medical & biological engineering & computing* **55** (2017) 1401–1414.

[11] Z. Chen, A. Malpani, P. Chalasani, A. Deguet, S. S. Vedula, P. Kazanzides and R. H. Taylor, Virtual fixture assistance for needle passing and knot tying, *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE (2016), pp. 2343–2350.

[12] Z.-Y. Chiu, F. Richter, E. K. Funk, R. K. Orosco and M. C. Yip, Bimanual regrasping for suture needles using reinforcement learning for rapid motion planning, *2021 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE (2021), pp. 7737–7743.

[13] F. Richter *et al.*, Autonomous robotic suction to clear the surgical field for hemostasis using image-based blood flow detection, *IEEE Robotics and Automation Letters* **6**(2) (2021) 1383–1390.

[14] F. Liu, Z. Li, Y. Han, J. Lu, F. Richter and M. C. Yip, Real-to-sim registration of deformable soft tissue with position-based dynamics for surgical robot autonomy, *2021 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE (2021), pp. 12328–12334.

[15] Y. Ou and M. Tavakoli, Sim-to-real surgical robot learning and autonomous planning for internal tissue points manipulation using reinforcement learning, *IEEE Robotics and Automation Letters* **8**(5) (2023) 2502–2509.

[16] X. Tan, Y. Lee, C.-B. Chng, K.-B. Lim and C.-K. Chui, Robot-assisted flexible needle insertion using universal distributional deep reinforcement learning, *International journal of computer assisted radiology and surgery* **15** (2020) 341–349.

[17] J. Kumar, C. S. Raut and N. Patel, Automated flexible needle trajectory planning for keyhole neurosurgery using reinforcement learning, *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE (2022), pp. 4018–4023.

[18] K. Rao *et al.*, Rl-cyclegan: Reinforcement learning aware simulation-to-real, *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, (June 2020).

[19] A. A. Rusu, M. Večerík, T. Rothörl, N. Heess, R. Pascanu and R. Hadsell, Sim-to-real robot learning from pixels with progressive nets, *Conference on robot learning*, PMLR (2017), pp. 262–270.

[20] W. Saunders, G. Sastry, A. Stuhlmueller and O. Evans, Trial without error: Towards safe reinforcement learning via human intervention, *arXiv preprint arXiv:1707.05173* (2017).

[21] F. Wang *et al.*, Intervention aided reinforcement learning for safe and practical policy optimization in navigation, *Conference on Robot Learning*, PMLR (2018), pp. 410–421.

[22] J. Wu, Z. Huang, Z. Hu and C. Lv, Toward human-in-the-loop ai: Enhancing deep reinforcement learning via real-time human guidance for autonomous driving, *Engineering* (2022).

[23] J. Wu, Z. Huang, W. Huang and C. Lv, Prioritized experience-based reinforcement learning with human guidance for autonomous driving, *IEEE Transactions on Neural Networks and Learning Systems* (2022).

[24] Y. Long, W. Wei, T. Huang, Y. Wang and Q. Dou, Human-in-the-loop embodied intelligence with interactive simulation environment for surgical robot learning, *arXiv preprint arXiv:2301.00452* (2023).

[25] Y. Ou and M. Tavakoli, Towards safe and efficient reinforcement learning for surgical robots using real-time human supervision and demonstration, *2023 In-*

ternational Symposium on Medical Robotics (ISMR), (2023).
[26] Y. Ou, S. Zargarzadeh and M. Tavakoli, Robot learning incorporating human interventions in the real world for autonomous surgical endoscopic camera control, *Journal of Medical Robotics Research* (2023).
[27] T. Haarnoja *et al.*, Soft actor-critic algorithms and applications, *arXiv preprint arXiv:1812.05905* (2018).
[28] M. Liu, M. Zhu and W. Zhang, Goal-conditioned reinforcement learning: Problems and solutions, *arXiv preprint arXiv:2201.08299* (2022).
[29] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, O. Pieter Abbeel and W. Zaremba, Hindsight experience replay, *Advances in neural information processing systems* **30** (2017).
[30] C. Rossa, M. Khadem, R. Sloboda, N. Usmani and M. Tavakoli, Adaptive quasi-static modelling of needle deflection during steering in soft tissue, *IEEE Robotics and Automation Letters* **1**(2) (2016) 916–923.
[31] Y. Ding, C. Florensa, P. Abbeel and M. Phielipp, Goal-conditioned imitation learning, *Advances in neural information processing systems* **32** (2019).

Yafei Ou received his B.Sc. degree in Mechanical Design, Manufacturing and Automation from the University of Electronic Science and Technology of China (UESTC) in 2021. He is currently pursuing a Ph.D. degree in Electrical and Computer Engineering at the University of Alberta. His research interests focus on surgical robotics and automation.

Mahdi Tavakoli is a Professor in the Department of Electrical and Computer Engineering, University of Alberta, Canada. He received his BSc and MSc degrees in Electrical Engineering from Ferdowsi University and K.N. Toosi University, Iran, in 1996 and 1999, respectively. He received his PhD degree in Electrical and Computer Engineering from the University of Western Ontario, Canada, in 2005. In 2006, he was a post-doctoral researcher at Canadian Surgical Technologies and Advanced Robotics (CSTAR), Canada. In 2007-2008, he was an NSERC Post-Doctoral Fellow at Harvard University, USA. Dr. Tavakoli's research interests broadly involve the areas of robotics and systems control. Specifically, his research focuses on haptics and teleoperation control, medical robotics, and image-guided surgery. Dr. Tavakoli is the lead author of Haptics for Teleoperated Surgical Robotic Systems (World Scientific, 2008). He is a Senior Member of IEEE, Specialty Chief Editor for Frontiers in Robotics and AI (Robot Design Section), and an Associate Editor for the International Journal of Robotics Research, IEEE Transactions on Medical Robotics and Bionics, IEEE Robotics and Automation Letters, IEEE TMECH/AIM Emerging Topics Focused Section, and Journal of Medical Robotics Research.