Multilateral Surgical Pattern Cutting in 2D Orthotropic Gauze with Deep Reinforcement Learning Policies for Tensioning

Brijen Thananjeyan,[2] Animesh Garg,[2,3] Sanjay Krishnan,[2] Carolyn Chen,[2] Lauren Miller,[2] Ken Goldberg[1,2]

***Abstract*— In the Fundamentals of Laparoscopic Surgery (FLS) standard medical training regimen, the Pattern Cutting task requires students to demonstrate proficiency by controlling two actuators, one with surgical scissors and the other with a grasping tool, to accurately cut a circular pattern on a sheet of surgical gauze suspended at the corners. Accuracy can be improved by "tensioning," where the grasping tool grasps and pulls the tissue along a sequence of directions to induce tension in the material as cutting proceeds. As deformation is difficult to sense, we explore how Deep Reinforcement Learning with Trust Region Policy Optimization (TRPO) can find tensioning policies using a finite element fabric simulator. We compare the Deep RL tensioning policies with fixed and analytic (orthogonal to cut direction) policies on a set of 17 open and closed curved patterns in simulation and 4 patterns in physical experiments with the da Vinci Research Kit. Our simulation results also suggest that learning to tension with Deep RL can significantly improve performance and robustness to noise and external forces.**

I. Introduction

Robotic surgical assistants (RSAs), such as Intuitive Surgical's da Vinci, facilitate precise minimally invasive surgery [35]. These robots currently operate under pure tele-operation control, but introducing supervised autonomy of subtasks has potential to assist surgeons, reduce fatigue, and facilitate tele-surgery. The Fundamentals of Laparoscopic Surgery (FLS) [27] and Fundamental Skills of Robotic Surgery (FSRS) [32] define a representative set of subtasks for surgical training and evaluation. We focus on the Pattern Cutting training task that is included in both FLS and FSRS which requires cutting a 50 mm diameter circular pattern marked on a 4×4 inch square of surgical gauze fixed at the corners. We consider how this task can be generalized to other desired open and closed cutting contours.

In principle, tracking the marked pattern and applying feedback control would be a reasonable solution [23]. However, the material properties of the gauze lead to a challenging multilateral control problem, as accurate cutting can only be achieved with the tissue phantom is held in tension. The direction and magnitude of the tensioning potentially deforms the gauze, introducing coupling between the two arms. We consider how Reinforcement Learning (RL) can address this problem by learning a control policy through iterative exploration. However, the number of trials necessary for exploration can be very high, which is impractical for physical tasks like Pattern Cutting because of time and resource costs such as resetting failed trials many times during exploration.

All authors are affiliated to AUTOLAB at UC Berkeley. [1] IEOR, [2] EECS, UC, Berkeley, CA USA; [3] CS, Stanford University, Stanford CA US; {bthananjeyan, animesh.garg, sanjaykrishnan, carolyn.chen, laurenm, goldberg}@berkeley.edu

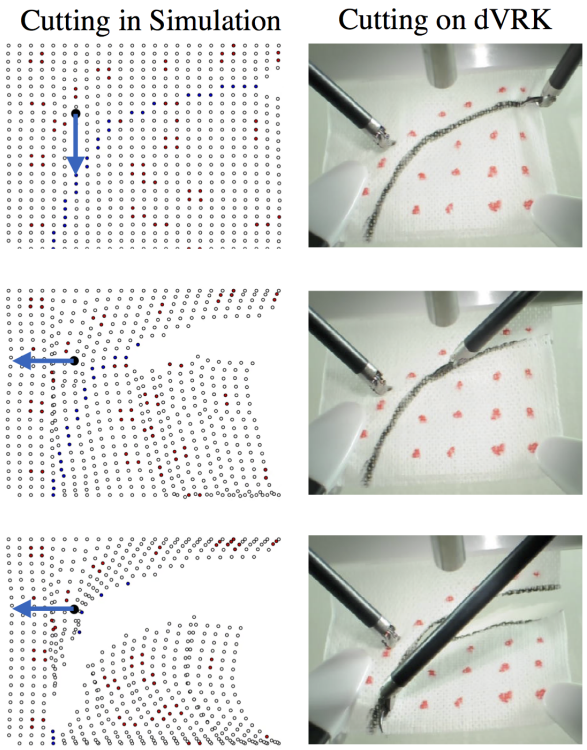

Fig. 1: Surgical Pattern Cutting involves tensioning and cutting deformable tissue phantoms. We propose the use of a simulator (left) and Deep Reinforcement Learning to learn tensioning policies that adaptively tension the phantom tissue during the Pattern Cutting task. The trained policy is then implemented on the physical system (right).

To address the prohibitive cost of real-world exploration, we use a spring-mass Finite Element simulation model to train a deep reinforcement learning policy for adaptive tensioning on a physical robot. The input is a desired cutting contour. The planner chooses a grasp point and sequence of tensioning directions as the surgical scissors follows a pre-planned trajectory. The reward is the negative symmetric difference between the desired contour and the actual cut contour. Given a desired contour, Trust Region Policy Optimization (TRPO), a deep variant of the Policy Gradient algorithm, is applied to a 4 layer fully-connected neural network to learn a tensioning policy.

Simulation results suggest that the Deep RL policy performs on average $43.3 \pm 8.6\%$ better than a non-tensioned baseline over a variety of desired open and closed contours.

As one concern with Deep RL is overfitting, we report sensitivity experiments that suggest that the policies are not sensitive to small changes in values of gravity, elasticity, and resolution. We also evaluate the learned tensioning policies

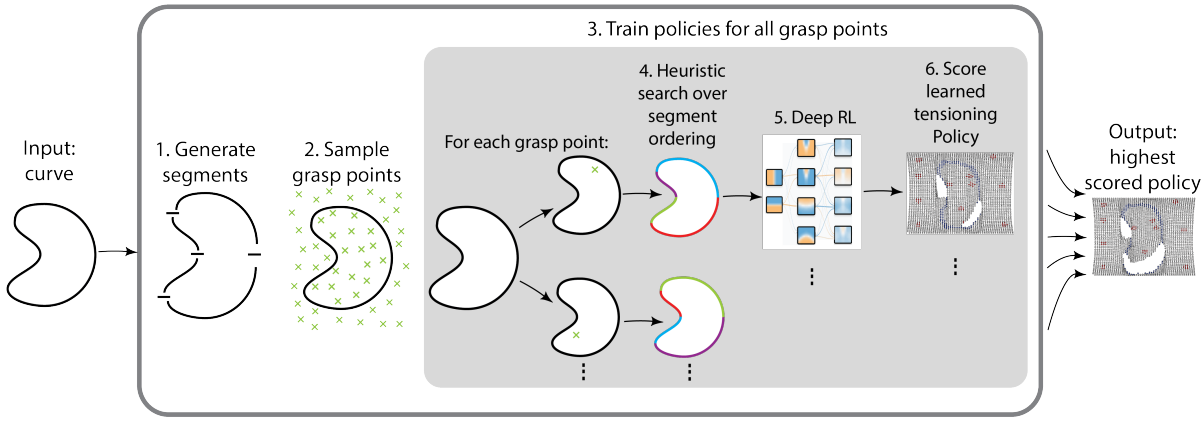

Fig. 2: The algorithm for generating tensioning policies for cutting using Deep RL involves several subcomponents: 1) the input curve is segmented, 2) a set of candidate grasp locations for tensioning are chosen, 3) search is performed to select the order to cut segments for all candidate grasp points, and 5) tensioning policies are found for each grasp point/segment ordering using Deep RL. The grasp point and policy which have the highest score are then selected. For details on each component, see Section VI.

with physical experiments on the da Vinci Research Kit (DVRK). The gauze is registered to the simulator using colored fiducial points tracked with a computer vision system. We add these visually tracked fiducial points to the system state for Deep RL training to represent the environment.

Summary of Contributions:

This paper:

1) Introduces a new class of "tensioning" policies for Multi-lateral Surgical Pattern Cutting in 2D Orthotropic Gauze.
2) Proposes an algorithm for learning such tensioning policies using Deep Reinforcement Learning.
3) Describes an implementation of the algorithm using 1) an FEM fabric simulator and 2) a working dVRK surgical robot.
4) Presents accuracy results from experiments with these policies on set of desired contours in simulated and real tissue phantoms.
5) Presents sensitivity results from experiments with these policies varying process noise, uncertainty in physical parameters and changes in simulation resolution.

II. Related Work

Deformable Manipulation in Robotic Surgery: Robotic surgical systems have been widely studied [2, 5, 22, 33]. Perception and control in surgical settings are challenging as most tasks inherently involve multilateral manipulation of deformable objects. While model-based deformation planning is relatively well-explored [16], highly accurate perception is often required for execution, e.g., surface tracking [15] – an assumption which is often challenging in surgery. Manipulation of deformable materials, particularly cutting, is an area of research interest in surgery [23, 24] as well as in computer graphics and computational geometry [9, 36]. In this paper, we explore the feasibility of Deep RL to improve control in surgical pattern cutting.

Prior work has explored the use of expert demonstrations to handle deformations in the environment without explicit models and simulations. Reiley et al. [26] proposed a demonstration-based framework that used Gaussian Mixture Models (GMMs) for motion generation. Van den Berg et al. [34] proposed an iterative technique to learn a reference trajectory and execute it at higher than demonstration speeds for suture knot tying. This work was recently extended by Osa et al. [25] to deal with dynamic changes in the environment, but with an industrial manipulator. Sen et al. [30] performed model based multi-throw suturing with feedback. Mayer et al. [21] used principles of fluid dynamics and Schulman et al. [28] used non-rigid registration techniques to generalize human demonstrations to similar, yet previously unseen, initial conditions for suturing. These approaches are broadly classified under the category of Learning From Demonstrations (LfD) [3, 8], where demonstration trajectories are directly modified for generalizing to test situations. In this paper, we explore the feasibility of a self-learning approach using Reinforcement learning (RL) and a simulator.

Reinforcement Learning for Deformable Manipulation: RL has been a popular control method in robotics when dynamics are unknown or uncertain [18]. There are a few examples of RL applied to deformable object manipulation, e.g. folding [4] and making pancakes [6]. The recent marriage between RL and Neural Networks (Deep RL) opened a number of new opportunities for control in non-linear dynamical systems [19, 20, 37]. An obstacle to applying Deep RL to physical robots is that large amounts of data are required for training, which makes sufficient collection difficult if not impossible using physical robotic systems. This is why we explore using a simulator in this work.

Simulation-based Training: There has been some recent work using simulators to train policies [1, 10, 11] before applying the policies to the physical system. For example, Frisken et al. [13] discuss a linked volume representation that enables physically realistic modeling of object interactions including collision detection and response, 3D object deformation, and interactive object modification. In this work, we explore a classical finite-difference model for simulating the deformation and cutting of the gauze [31]. The finite-difference model is computationally efficient, which is

Algorithm 1: Tensioning Policy Search

Data: Nonparametric curve $c(t)$

1 **gen_segments**(c): Decompose the curve (c) into cutting segments

2 $G \leftarrow$ **sample_grasp_points**(): Sample a set of candidate grasp points $G = \{g_1(t), \ldots, g_N(t)\}$.

3 **foreach** $g_i \in G$ **do**

4 $\quad k^* \leftarrow$ **heuristic_search**(g,K): Use a heuristic to select cut ordering from all possible permutations of cutting segment ordering $K := \{k_1(t), \ldots, k_N(t)\}$

5 $\quad \pi \leftarrow$ **learn**(g,k^*): Use Deep RL to learn a tensioning policy (π)

6 $\quad$ **score**(g,k^*,π)

Result: Return the policy (g,k,π) that receives the highest score.

important since we are running $10k$ trials with RL.

III. Problem Statement and Overview

In this section, we overview the system architecture, assumptions, and inputs and outputs of each component.

A. Problem Statement

We consider a generalization of the pattern cutting task from Fundamentals of Laparoscopic Surgery (FLS), which is a standard training regimen for medical students in laparoscopic surgery [27]. A typical pattern cutting task features a 50 mm diameter, 2 mm thick circular pattern marked on a 4×4 inch square of standard surgical gauze suspended by clips as shown in Figure 1. We consider a generalization where the gauze is marked with an arbitrary curve that does not self-intersect.

We assume that one arm of the robot is designated as the cutting arm and the other as the tensioning arm. The tensioning arm applies tension to the gauze from a fixed grasping point. The cutting arm operates along the initial position of the curve at a fixed speed regardless of the deformation. The task is complete once the cutting arm reaches the end point.

B. Evaluation Metric

Error is measured using the symmetric difference between the desired curve and the achieved curve cut. We used this metric in both the simulation training environment and in the physical evaluation experiments. More formally, let X be the set of points inside a closed intended trajectory and let Y be the set of points inside a closed simulated trajectory. The symmetric difference is then the exclusive-or $A \oplus B$ of the two sets. For open curves, curves are closed by forming boundaries with the edges of the cloth.

C. Tensioning Policy Search (TPS): Overview

An overview of the algorithm for Tensioning Policy Search is shown in Algorithm 1. A schematic of the TPS algorithm is shown in Figure 2. We describe the simulator model in Section IV and the learning approach (**learn**(g,k^*)) in Section V. The details for the subroutines **gen_segments**(c), **sample_grasp_points**(), and **heuristic_search**(g,K) are described in Section VI.

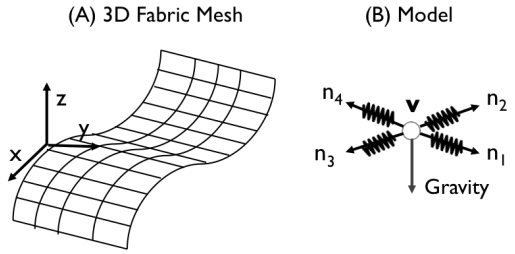

Fig. 3: We simulate the gauze with a finite difference method with a rectangular mesh of vertices coupled with ideal springs. The parameters of the simulated gauze are elasticity τ, a $-z$ external force $f(t)$, and time-constants α, δ that determine the rate at which the fabric reacts to $f(t)$

IV. Simulator Model

We use a fabric simulator to plan the cutting trajectories and learn the Deep RL tensioning policy in Tensioning Policy Search.

A. Surgical Gauze Simulation

Surgical gauze, a planar fabric, can be modeled as a rectangular mesh of point masses connected by ideal springs of equal length [7, 14] (Figure 3). Each point mass, a vertex of the mesh, can translate in three dimensions. A constant gravitational force is applied to each vertex, and tensioning is modeled as displacement of a single vertex. Initialized vertices are equally spaced in the x,y plane.

Suppose each vertex (i,j) of the fabric has coordinates $p_{ij} \in \mathbb{R}^3$. As a discrete-time approximation to the differential equation governing the system, each vertex is governed by the following equation of motion:

$$p(t+1) = \alpha p(t) + \delta(p(t) - p(t-1)) - \sum_{p' \in \text{neighbors}} \tau(p'(t) - p(t)) + f(t) \quad (1)$$

where τ is a spring constant, $\tau(p'(t) - p(t))$ is an idealized spring model of the interaction between vertices, and $f(t)$ is an external force applied to the system (e.g. gravity). α and δ are parameters that govern the time-constant of the system and how quickly the fabric reacts to applied forces. Cutting is simulated as removing vertices from the mesh (instantaneous and quasi-static) along the cutting trajectory, i.e., the vertex no longer has an effect on its neighbors:

$$p(t+1) = \alpha p(t) + \delta(p(t) - p(t-1)) - \sum_{p' \in \text{neighbors} \wedge \neg\text{cut}} \tau(p'(t) - p(t)) + f(t) \quad (2)$$

To model standard surgical gauze, we use the following parameters $\delta = 0.008$, $\alpha = 0.99$, $\tau = 1.0$ and $f(t)$ is gravity. We fit these parameters through trial and error, observing how cutting trajectories deformed in real gauze vs. the simulator. We use 625 vertices (25 x 25) in our experiments

to approximate the physical gauze. The simulator was implemented in Python using the Python package Cython to write C extensions. This sped up each run of the simulation by a factor of six. A single simulation for cutting a prototypical closed curve takes approximately 4 seconds on an Intel Core i7 desktop computer.

V. Learning the Tensioning Policy

Next, we describe learning a tensioning policy given a fixed cutting trajectory over the ordered segments $k(t)$ and grasping point g. k and g are returned by the planning algorithm in the following section. We learn a cutting policy for a specific curve and grasp point.

A. Reinforcement Learning Model

The goal is to learn a policy for the tensioning arm that such that the error from the cutting trajectory to the marked curve is minimized. We model the tensioning problem as a Markov Decision Process (MDP) $\mathcal{M}$:

$$\mathcal{M} = \langle S, A, T(\cdot,\cdot), R(\cdot,\cdot), N \rangle.$$

where the actions A are $1mm$ movements of the tensioning arm in each of the cardinal directions, and the states S are described below. The dynamics model $T()$ is unknown and the time horizon N is fixed. Reward is measured using the symmetric difference between the desired curve and the achieved curve cut. The robot receives 0 reward at all time-steps prior to the last step, and at the last time-step $N-1$ receives the symmetric difference. It is worth noting that we do not shape the reward, symmetric difference is exactly the error metric used for evaluation as well.

In RL, the objective is to find a function $\pi : S \mapsto A$ that optimizes the expected reward. The policy is often denoted as π_θ, parameterized by a vector θ. When applied to $\mathcal{M}$, π_θ generates a distribution over N-step trajectories. Each π_θ has an expected reward:

$$R(\theta) = \mathbf{E}_{(s_n,a_n)\sim\pi_\theta}[\sum_{n=0}^{N} R(s,a)] \tag{3}$$

B. Choice of State Space and Action Space

Our experiments require a state space that is observable for RL in both the simulator and in the real robot. The state-space is a tuple consisting of the time index of the trajectory t, the displacement vector from the original grasp point $\delta \in \mathbb{R}^3$, and the $x_i \in \mathbb{R}^3$ location of fiducial points chosen randomly on the surface of the gauze. In all simulation-only experiments we use 12 fiducial points and for robot experiments we use between 8-12 fiducial points. Fiducials are randomly placed in all cases. The 12 randomly chosen points are registered to the workspace of the real robot using standard machine vision contour detectors. When a point is occluded by the robot arm or cloth deformation, its last visible position is used until it is visible again. The action space A consists of a unit translation vector and its additive inverse for each of the simulator frame's axes as well as the zero vector.

It is worth noting that at each time step the policy can only apply a small amount of tension but it can learn to build up the tension over time in a particular direction. The policy gradient algorithm learns a state-feedback tensioning policy as a function of these states. Also note that the distance from pattern is *not* a feature because it is not available in real robot experiments.

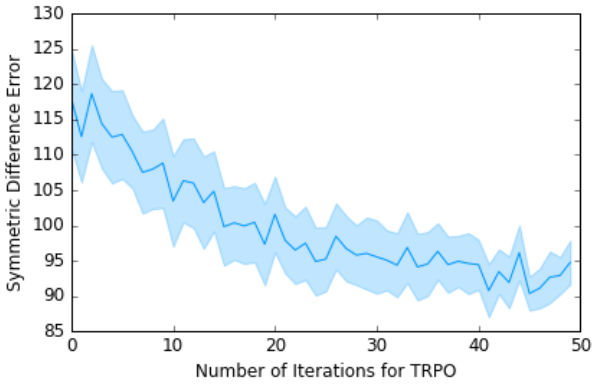

Fig. 4: **Convergence Analysis**: The figure shows the mean error with a 95% C.I. vs number of iterations of TRPO over 14 runs of simulated training over shape #4 in Table I.

C. Trust Region Policy Optimization

To optimize the parameter θ, we apply a class of a reinforcement learning algorithms– policy gradient. Specifically we leverage the TRPO implementation [29] in Rllab [12]. Policy gradient algorithms optimize the objective in Equation 3 using gradient ascent:

$$\theta^{(i+1)} = \theta^{(i)} + \lambda \cdot \nabla_\theta R(\theta)$$

Since $R(\theta)$ is a stochastic quantity, policies repeatedly compute noisy estimates of the gradient of the expected reward of a policy. The policy is then updated with the estimated gradient direction.

We use a neural network to parametrize the policy π_θ, which maps an observation vector to a discrete action. We use a two 32x32 Hidden Layer Multi-Layer Perceptron to represent this mapping. Since neural networks are differentiable, we can optimize the quantity $R(\theta)$.

As the estimate of the gradient can be very noisy, one solution is to use a technique called trust region optimization. Trust region methods take gradient steps in a small neighborhood around the current parameter value, and if the update exceeds this neighborhood, the update is clipped. This prevents excessive oscillation due to noise.

We plot the decrease in error as training progresses in Figure 4. We observe that the convergence takes about 5 minutes on a Intel core i7 desktop and for computational reasons, the experiments presented here are trained upto 25 iterations.

VI. Planning and Ordering Cut Segments

In order to evaluate tensioning policies, we need to A) plan a cutting trajectory, prior to tensioning, and B) select grasp locations for tensioning. This section describes our methods for planning a cutting trajectory and a grasp point.

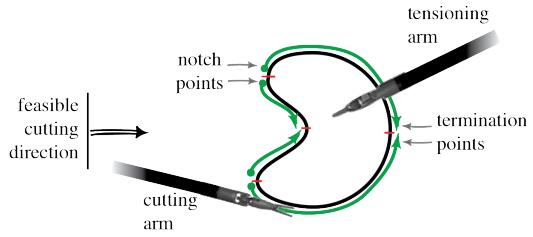

Fig. 5: The dVRK arms have limited approach angles, and cutting can only happen in the forward direction from the base of the arm, which requires segmenting curves into multiple sections for cutting. Notch and termination points are locations at the limits of the cutting arm's range of motions, or those orthogonal to the main approach direction of the arm, and the direction of cutting is always in the main approach direction.

A. Grasp Point Planning

We first select a point on the gauze for the tensioning arm to apply the tensioning policy. This is referred to as **`sample_grasp_points`**() in our algorithm. We uniformly sample $N = 30$ mesh points randomly to generate a candidate set of grasp points G. We also include the centroid of the curve $c(t)$ in the set G as a grasp point in addition to the sampled points. We train policies, running the entire pipeline, for each of the candidate grasp points, and select the point that results in the highest over-all cutting accuracy. Since we do not have a smooth, analytic, differentiable metric, random search is among the most efficient options. Random search can be easily parallelized allowing to efficiently try out all of the samples.

B. Segment Planning

Using the da Vinci's arms, complex curves cannot be completed in a single, smooth trajectory. The arms have limited approach angles, and cutting can only happen in the forward direction from the base of the arm. Additionally, collision between the two arms (tensioning arm and the cutting arm) must be avoided. We segment the curves and identify *notch points* (where the cutting arm enters the gauze) and *termination points* for cutting. We use the methods **`gen_segments`**(c) and **`heuristic_search`**(g,K) in Section III-C for segmentation of a parametric curve and creating an ordering over the segments for executing the cut.

`gen_segments`(c) – Given a registered pattern, represented parametrically by a finite sequence of points, $c(t)$, we find local minima and maxima of the curves to locate a set of candidate segments. We calculate an estimated directional derivative along the trajectory towards the cutting arm. Given two points p_i and $p_{i+1} \in \mathbb{R}^3$, where $i \in 0 \ldots N-1$ for an N-length trajectory, we compute $\Delta x = p_{i+1}^{(x)} - p_i^{(x)}$. Δx changing from positive to negative signifies the location of a notch. A negative to positive change of Δx marks the end point of a segment. A conceptual illustration of this procedure is shown in Figure 5.

`heuristic_search`(g,K) – Given a complete set of executable segments, the sequence of cuts needs to be specified. We posit that for any given set of segments, there is an optimal ordering such that the simulated cutting error is minimized. For experiments in this paper, we take a brute-force approach and iterate through all possible permutations of cutting ordering for each of the candidate grasp points G, choosing the ordering and subsequently trajectory plan $\chi(n)$ that maximizes the simulation score using a fixed tensioning policy. Although this means $K!$ simulations are required for K registered segments, we assume that K is small for the majority of relevant cases.

TABLE I: **Evaluation of Deep RL**: For the 17 curves shown, we evaluate the three tensioning policies described in Section VII-A. We measure and report performance in terms of relative percentage improvement in symmetric difference over a baseline of no tensioning for the tensioning trials. The 95% confidence interval for 10 simulated trials is shown for fixed, analytic, and Deep RL tensioning, while the mean absolute symmetric difference error is reported for the no-tensioning baseline experiments. The data suggest that Deep RL performs significantly better in comparison to the fixed and analytic baseline. The corresponding grasp points used for fixed and Deep RL are indicated in red. The analytic grasp point is the centroid of the shape.

Shape		Tensioning Method			
		No-Tension	Fixed	Analytic	Deep RL
1		17.4	-20.69±4.44	-149.43±10.24	**64.37±5.77**
2		22.5	32.44±1.74	-117.78±0.00	**55.11±7.40**
3		23.2	-21.12±5.41	18.10±1.78	**38.36±9.43**
4		102.9	7.48±0.62	30.42±1.45	**36.15±3.97**
5		41.1	0.49±7.99	9.98±0.00	**52.31±7.26**
6		42.0	**55.00±1.77**	11.43±1.52	45.95±6.60
7		40.2	22.64±1.14	3.73±1.46	**33.83±3.99**
8		40.0	-1.25±0.82	1.75±2.83	**35.50±4.67**
9		66.6	2.85±2.60	**34.68±2.25**	28.38±4.98
10		63.6	25.63±2.98	20.60±3.60	**41.35±9.90**
11		73.6	2.31±1.66	24.32±0.98	**56.11±8.99**
12		79.3	22.82±4.51	55.49±0.38	**63.56±3.61**
13		94.3	3.29±2.05	27.15±0.32	**34.04±5.87**
14		71.7	3.07±7.15	-2.51±0.61	**39.89±8.20**
15		178.7	74.71±1.22	80.75±0.88	**81.25±1.38**
16		114.6	-8.03±2.16	**31.06±0.62**	29.06±8.81
17		74.8	10.29±2.08	**28.34±2.07**	0.80±7.03
Mean (%)			13.54±9.84	9.70±21.96	**43.30±8.61**

VII. Simulation Experiments

In the first set of experiments, we compare Deep RL to alternative tensioning approaches. We first evaluate the techniques in terms of performance by calculating the

symmetric difference between the target pattern and actual cut. Then, we compare the techniques in terms of robustness by tuning the techniques for one simulated parameter setting and then applying them to perturbations.

A. Cutting Accuracy

We manually drew 17 different closed and open curves to test in simulation, which are illustrated in Table I. For each of the curves, we evaluated four different tensioning methods:

(a) **Non-tensioned:** Single-arm cutting is performed with no assisted tensioning other than the stationary corner clips that fix the gauze in place.

(b) **Fixed Tensioning:** The material is grasped at a single point with the grasper arm (with corner points still fixed in place), but no directional tensioning is applied. We simulate area contact by grasping a circular disc.

(c) **Analytic Tensioning:** Tensioning is proportional to the direction and magnitude of the error in the 3D position of the cutting tool and the closest point on the desired curve. The gain is fixed to 0.01.

(d) **Deep RL** This policy is described in detail in Section V. A separate policy is trained for each shape, this requires 20 iterations of TRPO in the simulator.

For the fixed and Deep RL, the grasp point used was chosen by the algorithm described in Section III-C. For the analytic tensioning model, the centroid of the curve is used as the grasp point. Performance is measured using the symmetric difference between the desired curve and the curve cut.

The averaged symmetric difference scores for each shape/curve and tensioning method are reported in Table I. The success of the different tensioning algorithms are presented as the percentage of improvement in symmetric difference score over the non-tensioned baseline. The average of the scores over all 17 curves are also included in the table. For the selected set of curves, Deep RL achieves an average relative improvement of 43.30%, the analytical method 9.70%, and the fixed approach 13.54%. This suggests that the learned adaptive policy with Deep RL leads to more accurate cutting in the simulator.

B. Robustness

Since it is learned, one concern with Deep RL is that it could overfit to the simulation environment and would not be as robust as the alternatives. We evaluate the robustness of Deep RL by training it with particular simulation parameters and then testing it in an alternative parameter setting.

1. Robustness to Resolution We first evaluate the learned policy's sensitivity to the resolution of the simulator. We trained four different policies, each on a mesh composed of different number of vertices (100, 400, 625, 2500). These experiments were performed on shape 1, as illustrated in Table I. We used each of the four policies to cut meshes with six different resolutions for ten rollouts each. Results are presented in Figure 6 as the absolute symmetric difference error (in number of vertices) per each trained policy when simulated on a mesh of x points. Our results indicate that

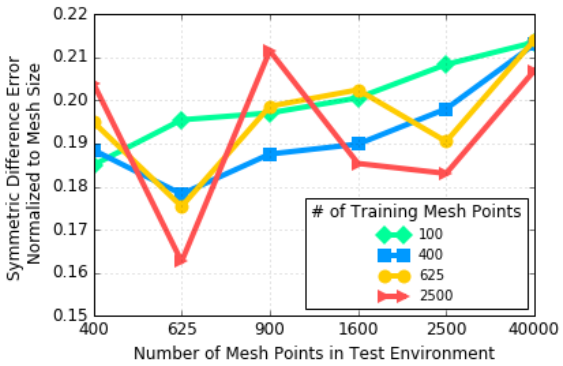

Fig. 6: **Robustness to resolution**: Deep RL was trained for four different mesh resolutions (plotted as individual lines), and evaluated on six different resolutions (x-axis). The symmetric difference error (calculated in unites of vertices) is normalized by the test environment resolution. Deep RL is relatively robust to changes in resolution but the performance suffers as the difference between the mesh resolution used for training and testing increases.

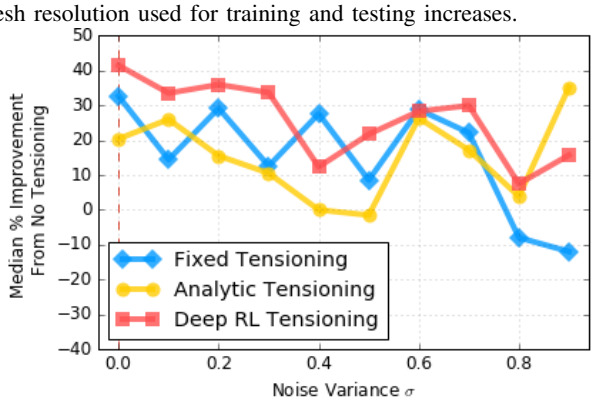

Fig. 7: **Robustness to Process Noise**: Deep RL was trained on a noise-free mesh and evaluated with different levels of Gaussian noise added to the process model (Equation (1)). We plot the improvement of the fixed, analytic, and Deep RL tensioning over no tensioning. The median of three trials is shown for each noise level. Deep RL is at least as robust as the alternatives to noise.

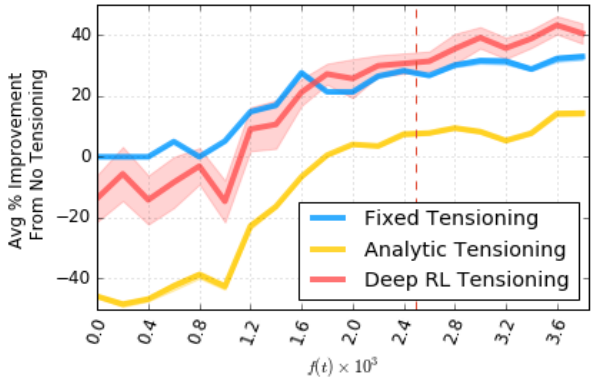

Fig. 8: **Robustness to Model Error**: We evaluate the sensitivity of the policies to varying levels of an unmodeled external force in Equation (1). For shape 9, Deep RL is trained using $f(t) = 2500$ indicated by the red dashed line. The mean and variance for twenty trials is represented as a percentage of improvement over no tensioning.

policies trained on lower resolution simulation models still result in comparable performance up to a very low resolution (400 vertices).

2. Robustness to Process Noise Next, we evaluate the effect

of process noise on the performance of the tensioning policies. Deep RL was trained in a noise-free simulated environment. Then all of the approaches were applied to shape 9 (see Table I) with increasing levels of Gaussian noise, i.e. $N(0,k)$ for $k \in \sigma[0,1]$, added to the mesh vertex position update (Eq. (1)). We ran three trials for each policy at 10 different values of σ. Figure 7 depicts the median relative percentage scores in terms of symmetric difference for fixed tensioning, analytic tensioning, and deep RL tensioning, over no-tensioning. A surprising result is that Deep RL is at least as robust as the other approaches, which are not learned.

3. Robustness to Model Error Next, we evaluated how parametric modeling error affects the performance of the tensioning policies. We vary the magnitude of the external force applied to each vertex in the $-z$-direction. Deep RL is trained on $f(t) = 2500$ in (1). All four policies are tested on varying magnitudes of $f(t)$ on shape 9, for twenty trials. When external forces are greater than this reference value, the Deep RL tensioning policy performs consistently better than the other policies. Both the Deep RL policy and the analytic policy perform worse than no tensioning and fixed tensioning for low force values. Intuitively, as the downward vertical force (gravity) tends toward zero, the gauze behaves more like a rigid sheet than a deformable material, so a simpler policy will work better, and the analytic and Deep RL policies are overcompensating to error.

VIII. Physical Experiments

dVRK: Hardware and Software

We use the Intuitive Surgical da Vinci Research Kit (dVRK) surgical robot assistant, as in [15, 23, 30]. We interface with the dVRK using open-source electronics and software developed by [17]. The software system is integrated with ROS and allows direct robot pose control. We use the standard laparoscopic stereo camera for tracking fiducials on the surgical gauze.

Physical Evaluation of Deep RL

We show the physical results on 4 curves using Fixed Tensioning and Deep RL using the symmetric difference as the evaluation metric in Table II. In this set of experiments, we used fixed tensioning as the baseline. The no tensioning policy frequently failed in the physical experiment so we excluded that from the results table for a fair baseline comparison. We were unable to evaluate the analytic method in the physical experiments because the state-estimation needed for feedback control was not possible outside of the simulated environment.

Figure 1 also illustrates the results in the case of a circular curve. We do not attempt analytic tensioning since it requires real-time tracking of the pattern to estimate local error. We observed that 3 out of 4 of the Deep RL experiments performed better than the fixed tensioning policy.

IX. Future work

This paper introduced a new class of "tensioning" policies for Multilateral Surgical Pattern Cutting in 2D Orthotropic Gauze and an algorithm for learning such tensioning policies using Deep Reinforcement Learning. We report accuracy and sensitivity results with an implementation of the algorithm using 1) an FEM fabric simulator and 2) a working dVRK surgical robot.

In future work, we will explore more complex tensioning policies where the grasp point can change and where more than 4 tensioning directions are possible. We will also explore methods to avoid collisions between the arms and other obstacles and how this approach can be generalized to cutting contours on more complex surfaces and manifolds for possible application to anastomosis.

TABLE II: **dVRK Physical Experiments**: This table compares the relative percentage improvement in terms of symmetric difference to a baseline of fixed tensioning to Deep RL in experiments performed on the dVRK robot. The black dot indicates the grasp point of the gripper arm

	Shape	Deep RL
1		**118.63 %**
2		**36.77 %**
3		**75.33 %**
4		-44.59 %

Acknowledgment

This research was performed at the AUTOLAB at UC Berkeley in affiliation with the AMP Lab, BAIR, and the CITRIS "People and Robots" (CPAR) Initiative: http://robotics.citris-uc.org in affiliation with UC Berkeley's Center for Automation and Learning for Medical Robotics (Cal-MR). The authors were supported in part by the U.S. National Science Foundation under NRI Award IIS-1227536: Multilateral Manipulation by Human-Robot Collaborative Systems, and by Google, UC Berkeley's Algorithms, Machines, and People Lab, and by a major equipment grant from Intuitive Surgical.

References

[1] P. Abbeel, M. Quigley, and A. Y. Ng, "Using inaccurate models in reinforcement learning", in *Proceedings of the 23rd international conference on Machine learning*, ACM, 2006, pp. 1–8 (cit. on p. 2).

[2] R. Alterovitz and K. Goldberg, *Motion Planning in Medicine: Optimization and Simulation Algorithms for Image-guided Procedures*. Springer, 2008 (cit. on p. 2).

[3] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration", *Robotics and Autonomous Systems*, 2009 (cit. on p. 2).

[4] B. Balaguer and S. Carpin, "Combining imitation and reinforcement learning to fold deformable planar objects", in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2011, pp. 1405–1412 (cit. on p. 2).

[5] R. A. Beasley, "Medical robots: current systems and research directions", *Journal of Robotics*, 2012 (cit. on p. 2).

[6] M. Beetz, U. Klank, I. Kresse, A. Maldonado, L. Mösenlechner, D. Pangercic, T. Rühr, and M. Tenorth, "Robotic roommates making pancakes", in *Humanoid Robots (Humanoids), 2011 11th IEEE-RAS International Conference on*, IEEE, 2011, pp. 529–536 (cit. on p. 2).

[7] A. Brooks, *A tearable cloth simulation using vertlet integration*, `https://github.com/Dissimulate/Tearable-Cloth`, 2016 (cit. on p. 3).

[8] S. Calinon, "Robot programming by demonstration", in *Springer handbook of robotics*, 2008 (cit. on p. 2).

[9] N. Chentanez, R. Alterovitz, D. Ritchie, L. Cho, K. Hauser, K. Goldberg, J. R. Shewchuk, and J. F. O'Brien, "Interactive simulation of surgical needle insertion and steering", *ACM Transactions on Graphics*, vol. 28, no. 3, 2009 (cit. on p. 2).

[10] M. Cutler and J. P. How, "Autonomous drifting using simulation-aided reinforcement learning", in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, May 2016 (cit. on p. 2).

[11] M. Cutler, T. J. Walsh, and J. P. How, "Real-world reinforcement learning via multifidelity simulators", *IEEE Transactions on Robotics*, 2015 (cit. on p. 2).

[12] Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel, "Benchmarking deep reinforcement learning for continous control", in *International Conference on Machine Learning (ICML-16)*, 2016 (cit. on p. 4).

[13] S. F. Frisken-Gibson, "Using linked volumes to model object collisions, deformation, cutting, carving, and joining", *IEEE transactions on visualization and computer graphics*, 1999 (cit. on p. 2).

[14] S. Gale and W. J. Lewis, "Patterning of tensile fabric structures with a discrete element model using dynamic relaxation", *Computers & Structures*, 2016 (cit. on p. 3).

[15] A. Garg, S. Sen, R. Kapadia, Y. Jen, S. McKinley, L. Miller, and K. Goldberg, "Tumor localization using automated palpation with gaussian process adaptive sampling", in *IEEE Conf. on Automation Science and Engineering (CASE)*, 2016 (cit. on pp. 2, 7).

[16] P. Jiménez, "Survey on model-based manipulation planning of deformable objects", *Robotics and computer-integrated manufacturing*, 2012 (cit. on p. 2).

[17] P. Kazanzides, Z. Chen, A. Deguet, G. Fischer, R. Taylor, and S. DiMaio, "An open-source research kit for the da vinci surgical system", in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2014 (cit. on p. 7).

[18] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey", *The International Journal of Robotics Research*, p. 0 278 364 913 495 721, 2013 (cit. on p. 2).

[19] I. Lenz, H. Lee, and A. Saxena, "Deep learning for detecting robotic grasps", *The International Journal of Robotics Research*, 2015 (cit. on p. 2).

[20] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies", *ArXiv preprint arXiv:1504.00702*, 2015 (cit. on p. 2).

[21] H. Mayer, I. Nagy, D. Burschka, A. Knoll, E. Braun, R. Lange, and R. Bauernschmitt, "Automation of manual tasks for minimally invasive surgery", in *Int. Conf. on Autonomic and Autonomous Systems*, 2008, pp. 260–265 (cit. on p. 2).

[22] G. Moustris, S. Hiridis, K. Deliparaschos, and K. Konstantinidis, "Evolution of autonomous and semi-autonomous robotic surgical systems: a review of the literature", *Int. Journal of Medical Robotics and Computer Assisted Surgery*, vol. 7, no. 4, pp. 375–392, 2011 (cit. on p. 2).

[23] A. Murali, S. Sen, B. Kehoe, A. Garg, S. McFarland, S. Patil, W. D. Boyd, S. Lim, P. Abbeel, and K. Goldberg, "Learning by observation for surgical subtasks: Multilateral cutting of 3d viscoelastic and 2d orthotropic tissue phantoms", in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2015, pp. 1202–1209 (cit. on pp. 1, 2, 7).

[24] H.-W. Nienhuys and A. F. van der Stappen, "A surgery simulation supporting cuts and finite element deformation", in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2001*, 2001 (cit. on p. 2).

[25] T. Osa, N. Sugita, and M. Mamoru, "Online trajectory planning in dynamic environments for surgical task automation", in *Robotics: Science and Systems (RSS)*, 2014 (cit. on p. 2).

[26] C. E. Reiley, E. Plaku, and G. D. Hager, "Motion generation of robotic surgical tasks: learning from expert demonstrations", in *Engineering in Medicine and Biology Society (EMBC), 2010 Annual International Conference of the IEEE*, IEEE, 2010, pp. 967–970 (cit. on p. 2).

[27] E. Ritter and D. Scott, "Design of a proficiency-based skills training curriculum for the fundamentals of laparoscopic surgery", *Surgical Innovation*, 2007 (cit. on pp. 1, 3).

[28] J. Schulman, A. Gupta, S. Venkatesan, M. Tayson-Frederick, and P. Abbeel, "A case study of trajectory transfer through non-rigid registration for a simplified suturing scenario", in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2013, pp. 4111–4117 (cit. on p. 2).

[29] J. Schulman, S. Levine, P. Moritz, M. I. Jordan, and P. Abbeel, "Trust region policy optimization", *CoRR, abs/1502.05477*, 2015 (cit. on p. 4).

[30] S. Sen, A. Garg, D. V. Gealy, S. McKinley, Y. Jen, and K. Goldberg, "Automating multiple-throw multilateral surgical suturing with a mechanical needle guide and sequential convex optimization", in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2016 (cit. on pp. 2, 7).

[31] J. E. Shigley, "Mechanical engineering design", 1972 (cit. on p. 2).

[32] A. P. Stegemann, K. Ahmed, J. R. Syed, S. Rehman, K. Ghani, R. Autorino, M. Sharif, A. Rao, Y. Shi, G. E. Wilding, *et al.*, "Fundamental skills of robotic surgery: A multi-institutional randomized controlled trial for validation of a simulation-based curriculum", *Urology*, 2013 (cit. on p. 1).

[33] R. Taylor, A. Menciassi, G. Fichtinger, and P. Dario, "Medical robotics and computer-integrated surgery", *Springer Handbook of Robotics*, pp. 1199–1222, 2008 (cit. on p. 2).

[34] J. Van Den Berg, S. Miller, D. Duckworth, H. Hu, A. Wan, X. Fu, K. Goldberg, and P. Abbeel, "Superhuman performance of surgical tasks by robots using iterative learning from human-guided demonstrations", in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2010, pp. 2074–2081 (cit. on p. 2).

[35] R. Veldkamp, E. Kuhry, W. Hop, J. Jeekel, G. Kazemier, H. J. Bonjer, E. Haglind, L. Pahlman, M. A. Cuesta, S. Msika, *et al.*, "Laparoscopic surgery versus open surgery for colon cancer: short-term outcomes of a randomised trial", *Lancet Oncol*, vol. 6, no. 7, pp. 477–484, 2005 (cit. on p. 1).

[36] H. Zhang, S. Payandeh, and J. Dill, "On cutting and dissection of virtual deformable objects", in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2004 (cit. on p. 2).

[37] M. Zhang, S. Levine, Z. McCarthy, C. Finn, and P. Abbeel, "Policy learning with continuous memory states for partially observed robotic control", *CoRR*, vol. abs/1507.01273, 2015 (cit. on p. 2).