A Case Study of Trajectory Transfer Through Non-Rigid Registration for a Simplified Suturing Scenario

John Schulman Ankush Gupta Sibi Venkatesan Mallory Tayson-Frederick Pieter Abbeel

***Abstract*— Suturing is an important, yet time-consuming part of surgery; a fast and robust autonomous procedure could reduce surgeon fatigue, shorten operation times, and enable remote tele-surgery. We present an approach that enables robots to learn to suture from human demonstrations. Our approach uses demonstrated trajectories as input to a closed-loop procedure which repeatedly adapts them to a previously unseen suturing environment.**

We use the *trajectory transfer* procedure proposed by the authors in a recent paper [13]; the key idea is to use non-rigid registration to find a 3D warping function which maps the demonstration scene onto the test scene, then use this function to transform the end-effector trajectory. Then a joint trajectory is generated by solving a trajectory optimization problem.

A first set of experiments in simulation and reality tested to what extent a single demonstration can be generalized to a variety of different initial conditions of the suturing scene. Our simulation experiments had a Raven II [6] suturing two flaps of tissue together. Our real-world experiments had a PR2 robot performing sutures in a scaled-up experimental setup. The simulation experiments were fully autonomous, but the real-world experiments required some human input for the perception. The success rate for learning from a single demonstration is nearly 100% for moderate perturbations from the demonstration's initial conditions, and it gradually decreases for larger perturbations. A second set of experiments, performed in simulation only, show that adding demonstrations from different initial conditions can increase success rate by better covering the state space. We can obtain these extra demonstrations using a "bootstrapping" procedure that uses successful autonomous executions as demonstrations.

I. Introduction

Robotic systems for minimally invasive surgery, such as the da Vinci ® system, are becoming increasingly widespread. Currently, these systems are operated in master-slave mode—the surgeon completely controls the movement of the robot. There are several reasons why it would be useful to perform some surgical subtasks—such as suturing—autonomously. First of all, some automation of repetitive tasks would give the surgeon much-needed rest and reduce fatigue, which is a serious consideration in operations that take many hours. Secondly, computer control could enable these tasks to be performed faster and more precisely by overcoming the inherent limitations of the human nervous system for speed and precision of motion. Third, autonomous low-level control would enable tele-surgery over long distances, so a surgeon could operate on a patient in a remote location hundreds of miles away. In these settings, transmission delays substantially degrade the surgeon's ability to perform continuous control. On the other hand, it is completely feasible for a surgeon to intermittently issue high-level commands, like "cut here", "suture here", which would then be controlled at a finer timescale by software.

Suturing, along with other surgical tasks, is very challenging to perform autonomously. It requires a long series of complex motions, and the margin of error is small. The operating environment, including the tissue to be sutured, may widely vary in position, shape, and material properties. Since the environment is so deformable, it is hard to perform a task repeatably without continually updating the planned movement based on perception.

Building on recent advances in learning from demonstrations and trajectory optimization, this paper presents promising results that bring us closer to real-world autonomous suturing capabilities. At the core of our procedure is the *trajectory transfer* algorithm recently proposed by the authors in [13], which takes trajectories from human demonstrations and adapts them to a new environment geometry.

Our experiments show that a single demonstration (provided by a human) can be generalized to a variety of initial conditions, for a simplified suturing scenario in reality and simulation.

Videos of our results are available at: `http://rll.berkeley.edu/suturing`

II. Related work

Some of the earliest work on automating laparoscopic procedures was performed by Kang et al. [7], [8], though that work focuses on the mechanical design and low-level control of a surgical robot. Recent work has addressed tying knots in the surgical setting based on human demonstrations. Mayer et al. [9] used ideas from fluid dynamics to modify trajectories to avoid obstacles. van den Berg et al. [14] used an iterative learning procedure to learn to tie an overhand knot rapidly with an imprecise robot. However, none of the aforementioned work addresses the problem of adapting the motion based on vision to match the current environment.

The core topics relevant to the current work are robotic programming by demonstration and trajectory optimization, where this work is closely based on the methods of [13] and [12], respectively. We refer the reader to the Related Work sections of those two papers for a comprehensive review of these topics.

Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, CA, USA. {joschu, mtaysonfrederick, ankush.gupta, sibi1992, pabbeel}@berkeley.edu

III. BACKGROUND

A. Trajectory transfer

Our method for generating end-effector trajectories in this work is based on the previous work of the authors in [13]. This section reviews that method; but the reader is referred to the original paper for a more complete exposition. The overall idea is to take the geometry data from perception and find a smooth transformation $\mathbf{f} : \mathbb{R}^3 \to \mathbb{R}^3$ that maps from the demonstration scene onto the test scene. Then this mapping is used to transform the end-effector trajectory from the demonstration, obtaining a new trajectory that matches the test scene's geometry. The transformation is chosen based on an an objective that encourages low fit error and rigidness (i.e., encourage the transformation to be nearly a rigid transformation).

The procedure for generating a trajectory, by generalizing a demonstration to the new "test" scene, is as follows:

Step 1: Find a transformation $\mathbf{f}$ *from the demonstration scene to the test scene*. Specifically, we assume that there is a list of 3D keypoints $\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_K$ in the demonstration scene, and there is list of corresponding keypoints $\mathbf{p}'_1, \mathbf{p}'_2, \ldots, \mathbf{p}'_K$ in the test scene. We use the method of thin plate splines [5], [15] to find a function $\mathbf{f}$ mapping $\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_K$ to their partners $\mathbf{p}'_1, \mathbf{p}'_2, \ldots, \mathbf{p}'_K$.

The method of thin plate splines solves the following optimization problem:

$$\underset{\mathbf{f}}{\text{minimize}} \sum_{i=1}^{K} \|\mathbf{p}'_i - \mathbf{p}_i\|^2 + \text{REGULARIZER}(\mathbf{f}) \qquad (1)$$

$\mathbf{f}$ is required to be an expansion in terms of radial basis functions

$$\mathbf{f}(\mathbf{x}) = \sum_{i=1}^{K} \mathbf{a}_i K(\mathbf{x}, \mathbf{x}_i) + \mathbf{B}\mathbf{x} + \mathbf{c} \qquad (2)$$

where K is the 3D thin plate spline kernel $K(\mathbf{x}, \mathbf{x}') = -\|\mathbf{x} - \mathbf{x}'\|$. The regularization term takes the form

$$\text{REGULARIZER}(\mathbf{f}) = \lambda \operatorname{tr}(\mathbf{A}^T \mathbf{K} \mathbf{A}) + \beta \|\log \mathbf{s}\|^2 \qquad (3)$$

where $\mathbf{s}$ is the vector of singular values of $\mathbf{f}$ and λ, β are parameters, and $\mathbf{K}$ is the kernel matrix. This regularization term encourages $\mathbf{B}$ to be a rotation with unit singular values. (This regularization on the singular values is not typically used with thin plate splines, so we had to modify the fitting procedure to account for it. See our technical report [11] for details.)

Step 2: Apply transformation $\mathbf{f}$ *to the demonstrated gripper trajectory*. The gripper poses along the demonstration trajectory are specified by positions $\mathbf{p}_1, \ldots, \mathbf{p}_T$ and orientations $\mathbf{R}_1, \ldots, \mathbf{R}_T$. We transform the positions and orientations as follows, to adapt the trajectory to the test situation:

$$\mathbf{p}_t \to \mathbf{f}(\mathbf{p}_t) \qquad (4)$$

$$\mathbf{R}_t \to \operatorname{orth}\left(\mathbf{J}_{\mathbf{f}}(\mathbf{p}_t)\mathbf{R}_t\right). \qquad (5)$$

Here, $\mathbf{J}_{\mathbf{f}}$ is the 3×3 Jacobian matrix

$$\mathbf{J}_{\mathbf{f}} = \begin{pmatrix} \partial f_x/\partial x & \partial f_x/\partial y & \partial f_x/\partial z \\ \partial f_y/\partial x & \partial f_y/\partial y & \partial f_y/\partial z \\ \partial f_z/\partial x & \partial f_z/\partial y & \partial f_z/\partial z \end{pmatrix}, \qquad (6)$$

and $\operatorname{orth}(\cdot)$ is a function that orthonormalizes a 3×3 matrix (e.g. using the SVD).

Equation (4) says that we apply the warping function $\mathbf{f}$ to all of the positions. As for rotations, the natural way to transform a vector $\mathbf{v}$ at a point $\mathbf{p}$ through a function $\mathbf{f}$ is to multiply it by $\mathbf{J}_{\mathbf{f}}(\mathbf{p})$, the Jacobian. Equation (5) applies this transformation to the x, y, and z axes of the gripper (which are the columns of matrix $\mathbf{R}_t$), and then orthogonalizes the resulting basis so it corresponds to a gripper pose.

Step 3: Convert the end-effector trajectory into a joint trajectory. To enable the robot to follow the trajectory as closely as possible while satisfying constraints, we formulate the following optimization problem on the joint trajectory $\boldsymbol{\theta}_{1:T}$:

$$\underset{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_T}{\text{minimize}} \left[\sum_{t=1}^{T-1} \|\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}_t\|^2 + \mu \sum_{t=1}^{T} \| \operatorname{err}\left(\tilde{\mathbf{T}}_t^{-1} \cdot \operatorname{fk}(\boldsymbol{\theta}_t)\right) \|_{\ell_1} \right]$$

subject to

No collisions, with safety margin d_{safe}

$\boldsymbol{\theta}_{\min} \le \boldsymbol{\theta}_{1:T} \le \boldsymbol{\theta}_{\max}$ (Joint limits)

Here, $\tilde{\mathbf{T}}_t$ is the desired end-effector pose at time t, $\operatorname{fk}(\cdot)$ indicates the robot's forward kinematics function applied to $\boldsymbol{\theta}_t$, and μ is a scalar parameter. $\operatorname{err}(\cdot)$ is an error function that maps a pose in $SE(3)$ to an error vector in $\mathbb{R}^6$. In particular, after decomposing a pose $\mathbf{T}$ into translation $\mathbf{p}$ and quaternion rotation $\mathbf{q}$, the error vector is simply given by $(p_x, p_y, p_z, q_x, q_y, q_z)$, i.e., the translation and the rotation part of the quaternion.

We will illustrate steps 1 and 2 of the above procedure with a two-dimensional toy example, where the task is to draw a two-dimensional curve through four guide-points. Note that this example merely illustrates the transformation of end-effector positions, not orientations. The left image of Figure 1 shows the training situation, environment shown in solid lines, gripper tip trajectory shown as a dotted line, coordinate grid lines shown as thin solid lines. The right image shows the test situation for which we want to predict a good gripper trajectory. The registered points are the four corners. First, we use the method of thin plate splines [15] to find a function that maps the four corners of the square in the training situation to the four vertices of the new quadrilateral. Then we applied the found warping function to the demonstrated path to obtain a new path (dotted line), which has the same topological characteristics. The warped coordinate-grid lines are shown.

IV. TECHNICAL DETAILS OF ALGORITHM

The training time and testing time procedures performed in our real-world experiments are described below. A schematic of the procedure applied at testing time is shown in Figure 2. The overall procedure used in the simulation experiments

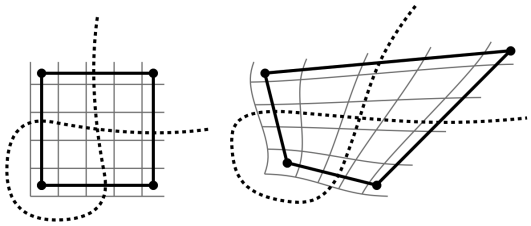

Fig. 1. Illustration of trajectory warping procedure on a cartoon 2-D example. Left: training situation. Right: testing situation.

was similar, except that we used the dense non-rigid registration method described in our previous work [13], using a set of points on the rope and tissue that were directly obtained from the simulation state.

A. Training and annotation procedure

At training time, we record a single high-quality demonstration of the complete task. This recording includes the joint states of the robot and a RGBD video of the task execution, recorded from a fixed camera.[1]

After data collection, we manually annotate the trajectory. We first break it into segments. The segment boundaries are chosen qualitatively, where the main consideration is that the subsequent movement can be executed in open loop based on the 3D data obtained at the beginning of the trajectory. Some examples of segments are as follows:

- Pick up the suture needle
- Grab the suture thread and pull it to tighten
- Wrap the suture thread around the wrist.

For each segment, we manually select a set of 3D keypoints in the demonstration scene. These keypoints indicate the parts of the scene that are most relevant to the next segment of the manipulation task. In our tasks, we use between zero and five keypoints. For example, $K = 0$ during knot tying for wrapping the thread around the robot arm, since no targets in the environment are relevant for the motion (except for collision avoidance, which is accounted for separately). $K = 1$ for grabbing the needle–the only keypoint is the tip of the needle (though we added in extra keypoints around the tip as described in the next paragraph.)

In some cases, we added in extra keypoints to constrain the first derivative of $\mathbf{f}$ in addition to its value. For some keypoints, we can measure the surface normal at the source and target location. Let us denote the source and target points and their normals by $(\mathbf{p}, \hat{\mathbf{n}})$ and $(\mathbf{p}', \hat{\mathbf{n}}')$. Then we add in the following extra keypoint pair:

$$\mathbf{p} + \epsilon\hat{\mathbf{n}} \leftrightarrow \mathbf{p}' + \epsilon\hat{\mathbf{n}}' \tag{7}$$

where we used $\epsilon = 1\,\mathrm{cm}$. For other keypoints we can measure a local coordinate frame. For example, when grabbing the needle, we used its entire frame. Let us denote positions and orientation 3x3 matrices by $(\mathbf{p}, \begin{pmatrix}\hat{\mathbf{x}} & \hat{\mathbf{y}} & \hat{\mathbf{z}}\end{pmatrix})$, $(\mathbf{p}', \begin{pmatrix}\hat{\mathbf{x}}' & \hat{\mathbf{y}}' & \hat{\mathbf{z}}'\end{pmatrix})$. Then we add three extra keypoint pairs

$$\mathbf{p} + \epsilon\hat{\mathbf{x}} \leftrightarrow \mathbf{p}' + \epsilon\hat{\mathbf{x}}' \tag{8}$$

$$\mathbf{p} + \epsilon\hat{\mathbf{y}} \leftrightarrow \mathbf{p}' + \epsilon\hat{\mathbf{y}}' \tag{9}$$

$$\mathbf{p} + \epsilon\hat{\mathbf{z}} \leftrightarrow \mathbf{p}' + \epsilon\hat{\mathbf{z}}' \tag{10}$$

These extra keypoints for the source geometry are illustrated in Figure 3.

B. Keypoint detection at testing time

At testing time, the robot executes the same sequence of trajectory segments that were demonstrated at training time. Consider, for example, the retraction maneuver, where the robot grabs a flap of tissue and lifts it up. At training time, we labeled five keypoints $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5$—three on the cut and one at each piercing location. To fit a transformation between the demonstration and current scene as described in Section III-A, we need to find five corresponding points $\mathbf{x}'_1, \mathbf{x}'_2, \mathbf{x}'_3, \mathbf{x}'_4, \mathbf{x}'_5$, in the test scene.

In our PR2 experiments, we had a human in the loop, clicking on all of the relevant keypoints for a given segment. We anticipate that in an actual surgical assistance system, some human guidance would be desirable: the surgeon would label where to suture, and a vision algorithm would track these keypoints over time. We plan to incorporate tracking into our suturing procedure, so it requires human input only on the first step of the suture; not after each segment.

C. End-effector trajectory generation

After obtaining the 3D keypoint correspondences, $\mathbf{x}_1 \leftrightarrow \mathbf{x}'_1, \mathbf{x}_2 \leftrightarrow \mathbf{x}'_2, \ldots, \mathbf{x}_K \leftrightarrow \mathbf{x}'_K$ we directly apply the method described in III-A to fit a 3D non-rigid transformation and use it to warp the trajectory of the end-effectors. Note that some trajectory segments move only one arm, while others move both arms simultaneously (and thus have two end-effectors).

The end-effector is either the robot's gripper or the needle tip. In the annotation stage, we indicate what the end-effector is for each segment. One important point is that whenever the end-effector is the needle tip, we need to know the precise pose of the needle relative to the robot's gripper. So every time the robot grabs the needle, we follow that segment with a "look-at-needle" motion where the robot holds up the needle and acquires the location of the tip.

V. Suturing experiments on PR2

Our first set of experiments simulates surgical suturing with a PR2 robot. The suture needle is a bent, rectangular shaft with 5mm edge length, and it sutures through a foam pad with pre-cut holes. For vision, we use an Asus Xtion Pro mounted to the robot's head.

The steps for tying one suture stitch are as follows:

- Pick up the needle
- Pick up one piece of tissue
- Pierce both pieces of tissue with the needle
- Re-grasp the needle with the opposite hand
- Pull the needle and thread through both pieces of tissue

[1] In our PR2 experiments, the camera was a Xtion-Pro mounted to the robot's head. We anticipate that the methods we describe will straightforwardly generalize to a setting where 3D information is obtained through stereo.

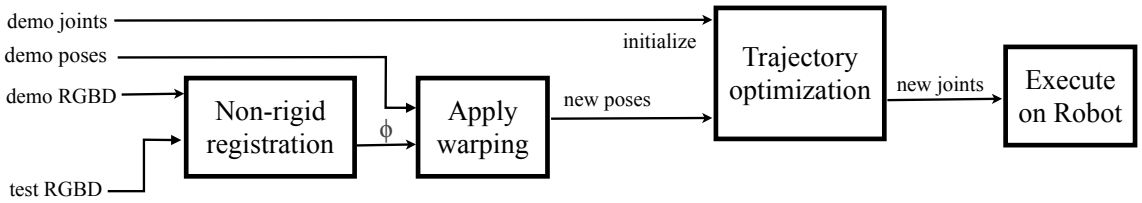

Fig. 2. Schematic diagram of the procedure applied at testing time for a given trajectory segment. "Demo" is short for "demonstration". "Poses" and "joints" refer to pose trajectories and joint trajectories, respectively.

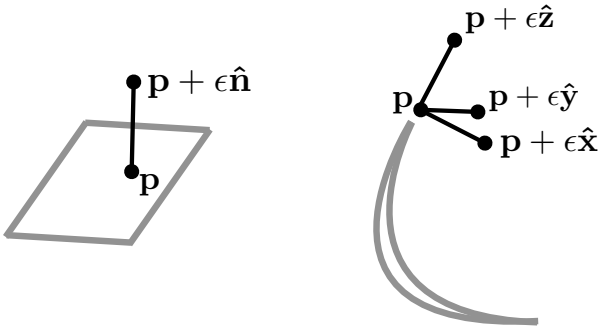

Fig. 3. Extra keypoints added when normal and pose information at the keypoints is relevant. Left: extra keypoint added for a surface normal. Right: three extra keypoints added for full rotation (at needle tip, needle shown in gray).

- Tie a surgeon's instrument-tied knot

Figure 4 shows several snapshots of this procedure. We did not include the knot tie in the quantitative experiment.

To investigate the extent to which the learned trajectory generalizes to different task space geometries, we applied translations, rotations, and bends to the suturing environment and measured the success rate. For these experiments, we focused on the first step of the manipulation: grasping the suture needle and passing it through the holes in the simulated tissue. This part requires precise movements based on perception, and always failed when executed in open loop unless the initial conditions exactly matched the demonstration.

Our procedure handled small translations easily with approximately 100% success, so we did not collect data on translations. For rotations, we found that the robot succeeded 100% of the time for small rotations (less than 10 degrees). However, as the rotation of the environment was increased, several sources of error in the overall pipeline became apparent. TODO: description of failures, results for non-rigid The results are shown in Table I.

TODO: update this. In summary, our experiments showed a good success rate even when the experimental setup was rotated or bent, and failures could be attributed to limitations of our current implementation that are readily fixable.

VI. Suturing experiments with Raven robot in physics simulation

We performed a separate set of experiments in a physics simulation environment, using a Raven II surgical robot [6]. These experiments in simulation provided a useful supplement to the PR2 experiments in several ways:

- The experiments are repeatable and it's possible to run a much larger number of them.
- The Raven has different kinematics than the PR2: non-redundant arm kinematics, remote center of motion.

We performed suture stitch (with puncture and knot), and we assumed that the robot has a mechanism to pass the needle between its gripper tips such as in the Endo Stitch™ system.

We used OpenRAVE [4] to load the model of the robot and do kinematics calculations, and we used a custom simulation environment based on Bullet physics engine. We teleoperated the simulated Ravens using a pair of Phantom OMNIs™.

The experimental results for using trajectory transfer on a single human demonstration are shown in Table II. They show that the task succeeds reliably for small perturbations but fails as the size of the perturbation is increased. TODO: comment on failure cases

It's worth noting that this task was very challenging for humans to perform under teleoperation with the Phantom OMNIs, and it took several hours of practice before we were able to record a successful demonstration.

VII. Leveraging multiple demonstrations with bootstrapping procedure

The above experiments showed that trajectory transfer enables task success for a variety of initial conditions around the demonstration's initial conditions. The success rate decreased for more distant initial conditions. These results suggest that by providing a large number of demonstrations that cover the space of initial conditions, we will achieve a much higher success rate, by simply looking up the nearest demonstration at each step of the execution. See [13] for more details on this nearest-neighbor lookup procedure.

Instead of covering the space of initial conditions with human demonstrations, we used the set of successful autonomous executions as demonstrations–we call this procedure "bootstrapping". After the first set of experiments, with results shown in Table II, we used the set of successful executions as an additional set of demonstrations.

As in [13], we measured closeness by performing non-rigid registration between the current state and the initial state of the corresponding segment in the demonstration, and considering the registration cost from Equation (1). To save on computation time, we only considered five demonstrations with nearby perturbation parameters.

Perturbation	Complete Success	Failed Needle Grasp	Failed Tissue Grasp	Failed Pierce
10° x rotation	3/3	0	0	0
15° x rotation	0/3	1	0	2
20° x rotation	2/3	1	0	0
10° y rotation	3/3	0	0	0
15° y rotation	1/3	2	0	0
20° y rotation	0/3	1	2	0
10° z rotation	3/3	0	0	0
15° z rotation	1/3	2	0	0
20° z rotation	0/3	2	0	1

TABLE I

Experimental results for PR2 performing pierce and regrasp. Three trials were performed in each experimental setting, so each row adds up to three. The last three columns are the different types of failures that occurred.

todo

TABLE II

Experimental results for the Raven suturing in simulation. The procedure included piercing, thread reorientation and knot tying. Piercing always succeeded.

We re-ran all of the experiments using this expanded set of demonstrations. The results are shown in Table III, and they show that the bootstrapping procedure was empirically very successful.

VIII. Future work

In future work, we plan to implement the procedure we described on a real surgical robot. We anticipate that this setting will present significant new challenges: perception on a smaller scale; working with lower precision of the robot relative to the scale of the experiment; and coping with the material properties of suture thread and tissue, which are more challenging to deal with than foam and rope.

The method we developed has demonstrated the significant promise of our approach, but we see many avenues for improvement. Currently the algorithm tries to stay near a reference trajectory but does not know in what ways the trajectory can be safely varied. Usually some parts need to be very precise (e.g. a grasp) while other parts (a motion through free space) can tolerate a large error. This information would be inconvenient to manually specify. One promising approach to automatically learn what is important is inverse reinforcement learning: collect a set of demonstrations and find the objective that reproduces the observed behavior [1], [10]. A closely related approach is to fit a Gaussian (or mixture of Gaussians) distribution to the collection of demonstrations and identify the directions of low variance– these are the important directions [3], [14], [2]. Alternatively, one could use a more domain-specific scheme that looks at the physical interactions between the robot and its environment. For example, the points where the robot contacts the environment need to be reached precisely, but the robot has more leeway when it is not in contact with anything.

IX. Conclusion

We have applied the trajectory transfer method to the problem of autonomous suturing and our experimental results with a real PR2 and simulated Raven robot validate this approach. These results suggest that trajectory transfer could be a powerful building block to bring us closer to having robots learn to perform challenging manipulation tasks from demonstrations. TODO: say something slightly more detailed based on experiments

Fig. 4. Snapshots of PR2 tying a suture stitch in our experimental setup, which used a foam block with pre-cut holes.

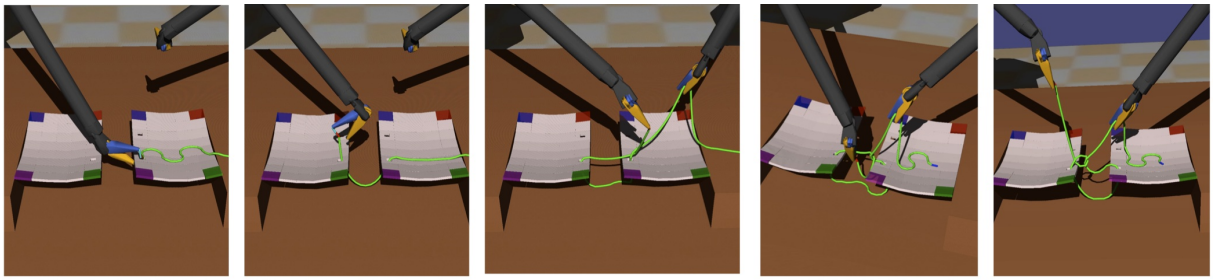

Fig. 5. Snapshots of Raven II tying a suture stitch in simulation.

todo

TABLE III

Experimental success after botostrapping.

X. Acknowledgements

Thanks to Jonathan Ho and Alex Lee for their contributions to the software we used. Sachin Patil, Zoe McCarthy, and Preetum Nakkiran provided useful comments and advice.

References

[1] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proceedings of the twenty-first international conference on machine learning (ICML)*. ACM, 2004, p. 1.

[2] S. Calinon, F. Guenter, and A. Billard, "On learning, representing, and generalizing a task in a humanoid robot," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 37, no. 2, pp. 286–298, 2007.

[3] A. Coates, P. Abbeel, and A. Y. Ng, "Learning for control from multiple demonstrations," in *Proceedings of the 25th international conference on Machine learning*. ACM, 2008, pp. 144–151.

[4] R. Diankov and J. Kuffner, "OpenRAVE: A planning architecture for autonomous robotics," *Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-08-34*, 2008.

[5] J. Duchon, "Splines minimizing rotation-invariant semi-norms in sobolev spaces," *Constructive theory of functions of several variables*, pp. 85–100, 1977.

[6] B. Hannaford, J. Rosen, D. Friedman, H. King, P. Roan, L. Cheng, D. Glozman, J. Ma, S. Nia Kosari, and L. White, "Raven-II: an open platform for surgical robotics research," *Transations on Biomedical Engineering*, vol. 10, no. 10, 2012.

[7] H. Kang and J. T. Wen, "Autonomous suturing using minimally invasive surgical robots," in *Control Applications, 2000. Proceedings of the 2000 IEEE International Conference on*. IEEE, 2000, pp. 742–747.

[8] ——, "Robotic assistants aid surgeons during minimally invasive procedures," *Engineering in Medicine and Biology Magazine, IEEE*, vol. 20, no. 1, pp. 94–104, 2001.

[9] H. Mayer, I. Nagy, A. Knoll, E. Braun, R. Lange, and R. Bauernschmitt, "Adaptive control for human-robot skill transfer: Trajectory planning based on fluid dynamics," in *Robotics and Automation, 2007 IEEE International Conference on*. IEEE, 2007, pp. 1800–1807.

[10] N. D. Ratliff, J. A. Bagnell, and M. A. Zinkevich, "Maximum margin planning," in *Proceedings of the 23rd international conference on Machine learning*. ACM, 2006, pp. 729–736.

[11] J. Schulman, "Notes on iterative registration with thin plate splines," July 2013. [Online]. Available: http://eecs.berkeley.edu/~joschu/docs/registration-techreport.pdf

[12] J. Schulman, J. Ho, A. Lee, I. Awwal, H. Bradlow, and P. Abbeel, "Finding locally optimal, collision-free trajectories with sequential convex optimization," in *Proc. Robotics: Science and Systems*, 2013.

[13] J. Schulman, J. Ho, C. Lee, and P. Abbeel, "Generalization in Robotic Manipulation Through The Use of Non-Rigid Registration," *Submitted. Manuscript at http://rll.berkeley.edu/isrr2013lfd/*, 2013.

[14] J. van den Berg, S. Miller, D. Duckworth, H. Hu, A. Wan, X.-Y. Fu, K. Goldberg, and P. Abbeel, "Superhuman performance of surgical tasks by robots using iterative learning from human-guided demonstrations," in *Proceedings of the International Conference on Robotics and Automation (ICRA)*, 2010.

[15] G. Wahba, *Spline models for observational data*. Siam, 1990, no. 59.