Proceedings of the 29th IEEE International Symposium
on Robot and Human Interactive Communication,
August 31 - September 4, 2020 - Virtual Conference

Collaborative Suturing: A Reinforcement Learning Approach to Automate Hand-off Task in Suturing for Surgical Robots

Vignesh Manoj Varier[1], Dhruv Kool Rajamani[1], Nathaniel Goldfarb[1], Farid Tavakkolmoghaddam[1], Adnan Munawar[2], Gregory S Fischer[1]

***Abstract*— Over the past decade, Robot-Assisted Surgeries (RAS), have become more prevalent in facilitating successful operations. Of the various types of RAS, the domain of collaborative surgery has gained traction in medical research. Prominent examples include providing haptic feedback to sense tissue consistency, and automating sub-tasks during surgery such as cutting or needle hand-off - pulling and reorienting the needle after insertion during suturing. By fragmenting suturing into automated and manual tasks the surgeon could essentially control the process with one hand and also circumvent workspace restrictions imposed by the control interface present at the surgeon's side during the operation. This paper presents an exploration of a discrete reinforcement learning-based approach to automate the needle hand-off task. Users were asked to perform a simple running suture using the da Vinci Research Kit. The user trajectory was learnt by generating a sparse reward function and deriving an optimal policy using Q-learning. Trajectories obtained from three learnt policies were compared to the user defined trajectory. The results showed a root-mean-square error of [0.0044mm, 0.0027mm, 0.0020mm] in $\mathbb{R}^3$. Additional trajectories from varying initial positions were produced from a single policy to simulate repeated passes of the hand-off task.**

I. Introduction

With about 1.2 million successful surgeries performed world-wide in 2019 using the da Vinci Surgical Systems [1], Robot Assisted Surgery (RAS) [2], [3], [4] has become increasingly prevalent in clinical environments. A particular field that has helped RAS gain recognition is Minimally Invasive Surgery (MIS). In MIS, surgeons attempt to minimize damage to tissue by operating through small incisions (laparoscopy) using advanced imaging tools. RAS have been successful in performing a wide range of complicated surgical procedures such as coronary artery bypass, skull-based surgeries, cardiac surgeries, cholecystectomies, and organ transplants [5], [6], [7]. Busch *et al.* [8] quantitatively compared robotic surgery using the da Vinci with conventional laparoscopy and concluded superiority of surgeries performed using the da Vinci for MIS. One caveat was a higher learning curve as compared to traditional open cavity surgery [9].

Teleoperated surgeries with the da Vinci can be particularly laborious for novice surgeons due to the repetitive use of *clutching* - an action used to reorient the Master Tool Manipulator (MTM) without moving the Patient Side Manipulator (*PSM*). A previous user study [10] conducted at Worcester Polytechnic Institute on using the MTM to manipulate grippers in a small workspace created in a simulation environment showed an average variation in a user's clutching frequency between 1 and 11 times, and an average path length coverage of 15.7cm - 356cm. Similar studies motivated other research groups to investigate techniques to automate surgical tasks. Automated procedures employ some iteration of the following concepts, Learning from Demonstration (LfD) [11], Deep Learning (DL) [12], Sequential Convex Optimization [13], Vision Guided Systems [14], etc. Although these works do justice to automating entire procedures, they fall short when addressing scenarios that require intraoperative assistance from a robot with the surgeon controlling the flow, also known as collaborative surgery. Padoy *et. al.* [11] demonstrated through a LfD technique that collaborative tasks between users and RAS were possible. In their research, recorded demonstrations by users were provided as inputs to a Dynamic Time Warping (DTW) algorithm to learn the trajectory using the da Vinci Research Kit (dVRK).

Fig. 1: User using the dVRK system to perform hand-off suturing task

This paper explores the use of discrete Reinforcement Learning (RL) to assist in collaborative suturing. The problem of suturing was first broken down into sub-tasks of which the task of needle hand-off was explored. User trajectories were collected and an algorithm was designed to generate sparse rewards and create a custom reward function. Q-learning was then implemented with this generated reward function to derive an optimal policy. Based on the state of the tool-tip, a trajectory was computed to complete the

[1]Authors with the Robotics Engineering Department, Worcester Polytechnic Institute, WPI, 100 Institute Road, MA 01609, USA
[2] Author with Laboratory for Computational Sensing and Robotics, Johns Hopkins University, Baltimore, Maryland 21218, USA
All correspondence should be to vvarier@wpi.edu

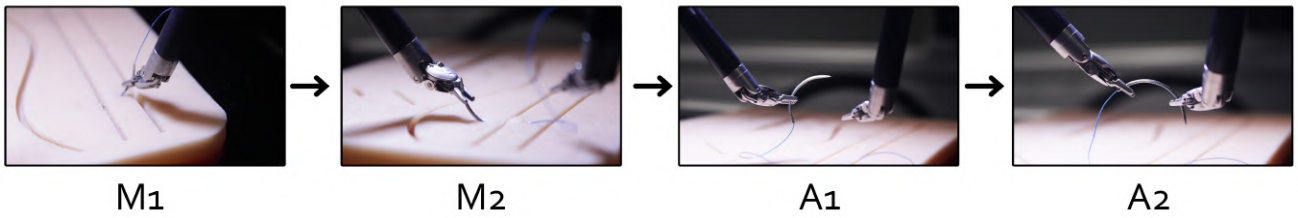

Fig. 2: Visual representation of the tasks being performed in this paper: M_1 - needle insertion performed by PSM_M; M_2 - grasping the needle manually performed by PSM_A, A_1 - pulling the needle and translating it, performed by PSM_A, and A_2 - reorient the needle and hand-off task performed by PSM_A. M_i indicates all manual tasks and A_i indicates all automated tasks. ($i \in 1,2$)

needle hand-off task. To verify the implementation of the algorithm, three optimal policies were derived from a single user trajectory (reference trajectory). Each policy was used to compute a trajectory which were quantitatively compared to the reference trajectory. To ascertain the robustness of the algorithm, the initial states were distributed away from the reference trajectory's initial state and trajectories were calculated from a single policy. An illustration showing consecutive passes of the hand-off task was also presented. Finally, a conclusion from the results was drawn and presented, along with the advantages and drawbacks of using this technique.

II. Background and Related Work

A. da Vinci Research Kit

The dVRK [15] system is an open-source mechatronics system based on the first generation da Vinci surgical robot developed by Intuitive Surgical Inc. The CISST libraries and Surgical Assistant Workstation (SAW) software allow for the control of the two *PSM*s, an Endoscopic Camera Manipulator (ECM) through the two MTMs and a foot pedal tray [2]. The MTM and *PSM* consists of 7 active joints with a gripper attached at the end of the *PSM* arms (shown in Figure (1)).

B. Reinforcement Learning for Motion Planning

RL is a subset of learning algorithms in which the surgical robot (an agent), learns how to perform in an environment based on numerical rewards. The aim is to make the robot learn from its own interactions with the environment while trying to maximize the rewards received. The problem is formed as a Markov Decision Process (MDP). Classical RL is based on discrete-event Dynamic Programming (DP). It works on two principles: Bellman Optimality Equation (BOE) and the Bellman Policy Equation (BPE). Motion planning from the context of RL is the process of finding the optimal policy to carry out a motion to match the desired trajectory. Some notable research in automating motion planning for surgical tasks in RAS include research done by Osa *et. al.* in which the authors used LfD, DTW and force control techniques to automate surgical tasks [16]. The authors mentioned one of the shortcomings as the performance depended on the quality of learned demonstrations and as a result the system could never outperform human users. Nonetheless,

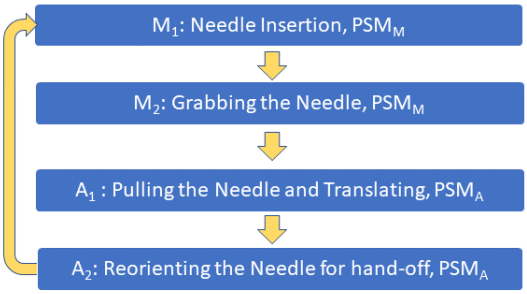

Fig. 3: Breakdown of suturing tasks and convention assumed

RL has the potential to find optimum trajectories to complete a task when applied to motion planning.

Selecting a suitable reward function is one of the most crucial and challenging tasks of practically applying RL. A good reward function would define how efficiently the *PSM* arm would learn to interact with the operating environment to find an optimal policy - how well it has learned to perform hand-off task after grabbing the needle.

C. Robotic Suturing

Suturing is the procedure of closing wounds using a thread to promote healing. It primarily consists of two main tasks, stitching and knot tying, with compounded sub-tasks such as needle aligning and tissue handling. In order to improve the performance of robotic suturing while still keeping the surgeon in charge, the process of collaborative suturing can be distributed into sets of automated and manual tasks. By automating certain sub-tasks of suturing, promising results can be achieved through clutch-less suturing, a step which can allow faster movements and reduce fatigue on the surgeon during teleoperated surgeries [11].

This paper explores a new technique to effectively assist surgeons during suturing tasks and goes on to accommodate the variant suturing styles of surgeons by generating sparse rewards for the desired trajectory. Figure 3 describes the repetitive flow of robotic suturing and a breakdown of tasks

considered in this paper. The tasks mentioned generalize gestures obtained from the JIGSAWS public data set [17] - a database of action recognition and skill assessment for automated surgery.

Since the targeted application is that of collaborative suturing, a consistent task definition convention can be assumed for the remainder of the paper. Tasks with M_i denote teleoperated tasks performed by the user. Tasks with A_i denote automated tasks performed by the methods described in this paper.

- M_1 Inserting the needle through the tissue with *PSM* arm (PSM_M).
- M_2 Grabbing the needle with the *PSM* arm (PSM_M).
- A_1 Tightening the suture and translating to the next suture point.
- A_2 Aligning the needle with PSM_A's jaw for completing hand-off.

Figure 2 demonstrates various stages of the suturing.

III. Methodology

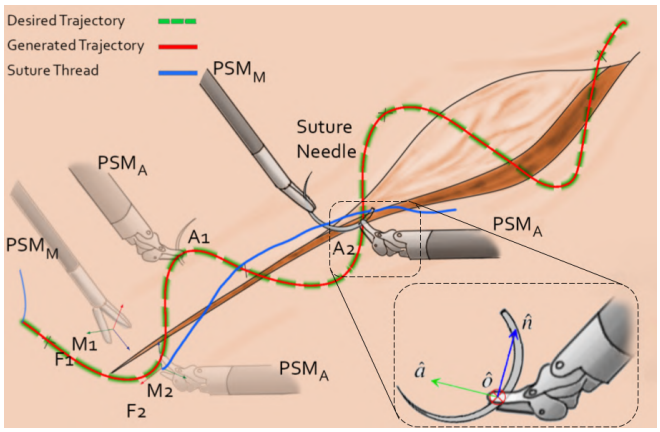

Fig. 4: Demonstration of the entire suturing procedure. At bottom right, pose of the needle grasped by PSM_A is shown, where $\hat{o}$ is the orientation vector, $\hat{a}$ is the approach vector, and $\hat{n}$ is the normal vector.

This section introduces a summary of problems - identifying an optimal trajectory, and addressing the varying suture style of surgeons. Users were instructed to perform a running suture (M_1, M_2, A_1, A_2) on a suture pad by teleoperating the *PSM* with the MTM. Data from a single hand-off task was collected. The collected data was sampled into a discrete gridspace defined in $\mathbb{R}^3$ to create sparse rewards to help follow a custom trajectory. Finally, Q-learning was implemented to identify the optimal policy for completing task A_1. The policy can then be tested on the dVRK with the user now having to perform tasks M_1 and M_2 only, with the tasks A_1 and A_2 automated.

A. Problem Definition

Problem 1: Identifying optimal trajectory for completing Task A_1.

Given a distinct value function of the environment, an optimal policy is to be computed to provide actions to the *PSM* to perform the hand-off task autonomously.

Problem 2: Addressing the varying suture styles of surgeons.

Surgeons are known to have varying suture styles. To accurately assist surgeons during procedures it is required to reciprocate their respective suture styles.

Assumption: Tasks M_1 - M_2 have been completed before hand-off task. The needle has been identified by the dVRK endoscope and the PSM_A has grabbed the needle in an orientation with the normal vector parallel to the needle profile, and the approach vector normal to the needle [Figure (4)].

Algorithm 1: Sparse Reward Generation

Result: Sparse rewards generated along the trajectory drawn by the operator

/* $Reward_{goal}$ much greater than $Reward_{visited}$ */

$Reward_{goal} \gg Reward_{visited} > Reward_{other}$
Input: $StateToIndex(array(x,y,z))$
Output: $Rewards[gridsize^3]$
while r_{id} *in Rewards* **do**
  if r_{id} == *goal* **then**
    $Rewards[r_{id}] = Reward_{goal}$
  else if r_{id} == *visited* **then**
    $Rewards[r_{id}] = Reward_{visited}$
  else
    $Rewards[r_{id}] = Reward_{other}$
  end
end

B. Data Collection

The users involved in the data collection process were members of the team, no external users were involved in this experiment. Since the purpose of this work is to learn a particular user's style rather than generalizing the whole population, only a small number of users were required for the study. Each of the user had less than 10 hours of experience in teleoperating the dVRK system prior to this experiment. The MTM was used to control the *PSM* arm during the suture needle hand-off task. The recorded data included the *PSM* pose and joint position data, the clutching frequency of the operator, and the MTM pose and joint positions. A python script implementing the *StateToIndex* function was created to process the operator specified trajectories and to generate sets of state-action pairs with appropriate discretization of action value ranges and concurrent states. The function *StateToIndex* converts each point P_i in the processed array to its associated index between 0 to $gridsize^3$. The *gridsize* decides the number of points present in the grid world. In this work, the *gridsize* is set to 11 and therefore a total of 1331 points are present in the grid world.

The algorithm designed to generate an appropriate reward function is shown in Algorithm (1). The data collected from

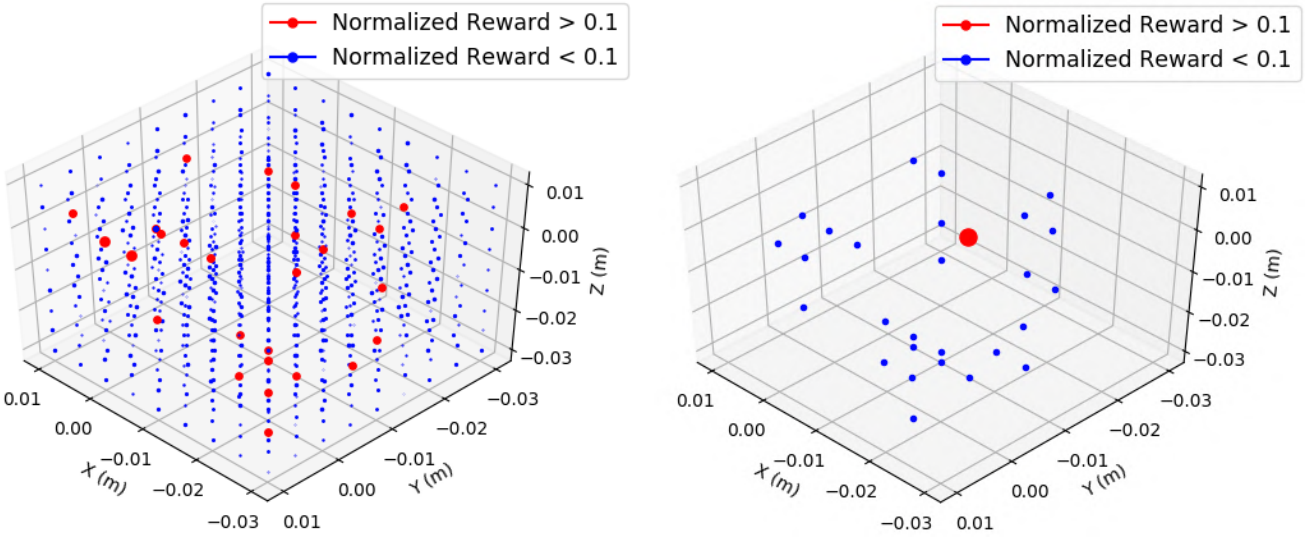

Fig. 5: Comparison of reward functions obtained for an user using IRL with Value Iteration (left) and sparse reward with Q-learning (right). The size and color of the dots indicate the rewards received by the *PSM* for visiting the corresponding states in the grid world.

the dVRK was discretized into an array of points $P_i(x, y, z)$ in the 3D grid world.

C. Motion Planning for RL

The *PSM* and it's workspace was considered to be the agent and environment respectively. The dVRK system was controlled using the CISST-SAW libraries. Since RL algorithms using DP are known to be computationally exhaustive, a minimal representation of the input states was considered. As a result, the problem was formulated as a discretized world problem in $\mathbb{R}^3$ with a grid of size $11 \times 11 \times 11$. An unit grid world corresponds to a $5 \times 5 \times 5mm^3$ cube in the real world. The range of the grid world was selected such that the robot tip followed a smooth trajectory in $\mathbb{R}^3$, without hampering the performance of the algorithm. From the collected user data, the change in orientation of the needle during the hand-off task was observed to be negligible until the point of handing the needle over to the PSM_M. This observation allowed sequential automation of the task, with the RL algorithm learning the desired trajectory as a simple translation, after which a set of rotation transformations were performed. In addition, the state space was significantly reduced, making the implementation of a discrete RL algorithm more plausible. The action space for the robot was designed to ensure all possible configurations out of a total of $[-ve, 0, +ve]^3 = 27$ actions from the current state, where 0 indicates no change in position value, $-ve$ and $+ve$ indicate a unit decrease and unit increase in the end-effector position value respectively.

Initially, Value Iteration was explored and implemented using the MDP environment created for the *PSM* [18]. However, the agent was unable to learn a trajectory despite iterating over varying parameters and integrating maximum entropy Inverse RL (IRL) to generate a task specific reward function. Maximum entropy IRL is a probabilistic approach in which equal probability is allocated to plans when they encounter equal rewards, and the probability of being chosen exponentially increases for plans with higher rewards. The implementation of this algorithm lacked sufficient exploration causing it to converge at a local minima instead of it following the desired trajectory.

Figure (5) shows the computed reward function using Maximum Entropy IRL (left) and Sparse Rewards (right). In the plot generated by Sparse rewards, equal rewards to the visited states ($threshold < 0.1$) and higher reward at the goal state can be seen ($threshold > 0.1$). This helped the Q-learning algorithm to track the visited states before reaching the goal state. In comparison, no such relation could be deduced in the Maximum Entropy IRL plot, which made it harder for the Value Iteration to track the desired trajectory.

Due to the unsatisfactory outcome of the aforementioned techniques, the authors explored the use of Q-learning - a technique that uses function approximation techniques to find the measure of the overall expected reward received by the agent (Q values) of a state. The state-action value function $Q_\pi(S,A)$ was derived to follow a policy $\pi(S,A)$ and take an action A from state S using Equation (1) [18]. Upon the completion of each iteration, the agent's expected value of a state was evaluated and stored to be used to calculate the expected reward. The following conditions warranted termination:

- To ensure the *PSM* doesn't keep moving around in the environment indefinitely, a maximum number of steps was assigned to terminate an episode (set to 100)
- During learning phase it is possible that the *PSM* can get stuck at a position. Therefore, the maximum amount

of time allotted to the *PSM* to complete an episode was set to 30 minutes.

- Task completion (when the *PSM* reaches the goal state)

$$Q_*(S,A) \longleftarrow Q_*(S,A) + \alpha[R + \gamma max_a Q_*(S',a) - Q_*(S,A)] \quad (1)$$

where R is the reward received when moving from the state S to the next state S' and α is the learning rate, γ represents the discount factor ($0 < \alpha, \gamma \leq 1$), . Once the agent estimates the value functions for each state, it chooses the path leading to the goal state by selecting the maximum value of rewards and repeats itself until the goal state is reached.

D. Realigning PSM_A Jaw

After gathering the suture data, the user indicated a preference for the PSM_A's tip to approach the PSM_M's tip orthogonally, and have the normal vector of PSM_A's jaw be parallel to the approach vector of PSM_M's jaw to help make the hand-off task easier [Figure (4)].

IV. Results

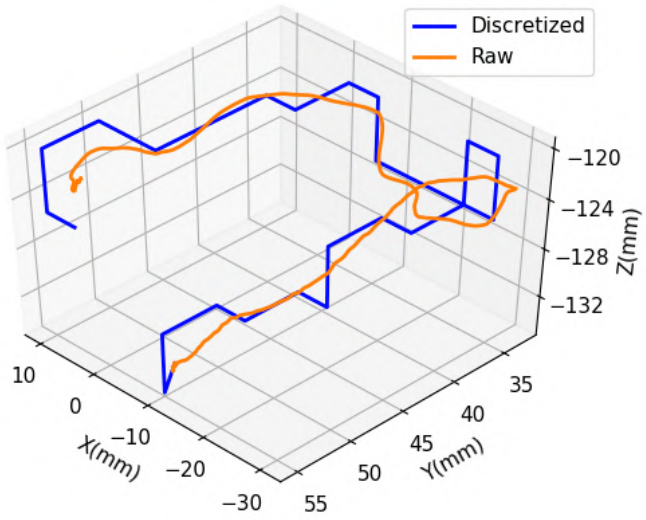

Fig. 6: Raw and discretized input trajectory for one of the user's data

For finding the ideal discretization factor, the factors considered was the collected user's data, required computational resources and the largest step the *PSM* could move without resulting in jerky movements. The grid size was set to $11\times11\times11$ and all the values were rounded off to nearest 5mm value as data loss observed was minimal without compromising the performance of the algorithm. Figure (6) shows that the discretized trajectory manages to retain the features of the desired raw input trajectory of the user.

Given the required suture points are computed beforehand and provided to the algorithm, the same model can be used to perform repeated suturing hand-off task (A_1, A_2) by aligning and stacking the grid world with respect to the goal position as shown in Figure (7). This is possible since the agent has

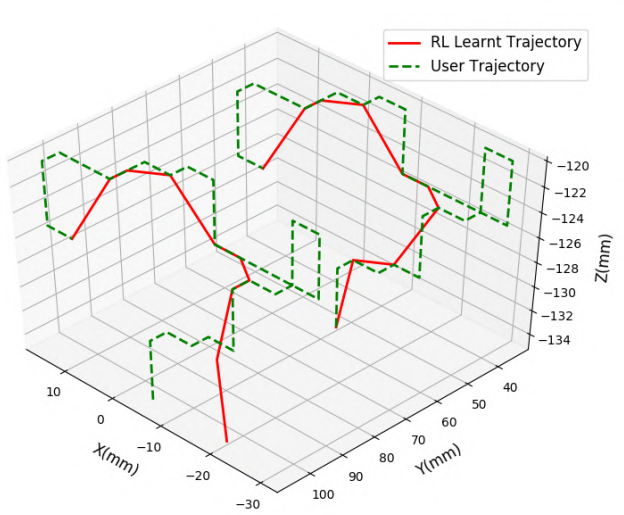

Fig. 7: Application of learnt RL policy for successive A_1 and A_2 tasks

learnt the policy to follow the user trajectory, even if there is a change in initial position. Since the exploration factor in Q-learning is dependent on the random numbers generated by the system, different iterations of the algorithm generated a slightly new policy. Each policy differs in state-action pairs, but as it can be seen from Figure (8), the agent was still able to successfully learn the user's trajectory.

Mean			Standard Deviation		
X	Y	Z	X	Y	Z
2.857	1.488	0.774	3.388	2.286	1.808

TABLE I: The mean and standard deviation of the dissimilarity between the average of the three learnt trajectories [Figure 8] and a single user trajectory (all values are in mm)

To show the effectiveness of the technique presented in the paper, the authors decided to measure the similarity of the generated trajectory and the user defined trajectory. DTW [19] with Manhattan distance as the distance measure, was used to establish this metric. DTW is a common method to compare the similarity between two sets of data which vary in length or time. Table (I) shows the mean [2.857mm, 1.488mm, 0.774mm] and standard deviation [3.388mm, 2.286mm, 1.808mm] of three learnt policies shown in Figure (8) from the user defined trajectory shown in Figure (6). For reference, perfectly similar trajectories would have a 0 mean and 0 deviation from the reference trajectory. Figure (8) also shows the deviations of the learnt policies from the user trajectory, where the black dashed lines symbolize the DTW produced mapping from the policy to the user trajectory. Root-Mean-Square Error (RMSE) was used to evaluate the *goodness* of the generated trajectories. The RMSE was found to be [0.0044mm, 0.0027mm, 0.0020mm] for the three trajectories shown in Figure (8) with reference to the user trajectory.

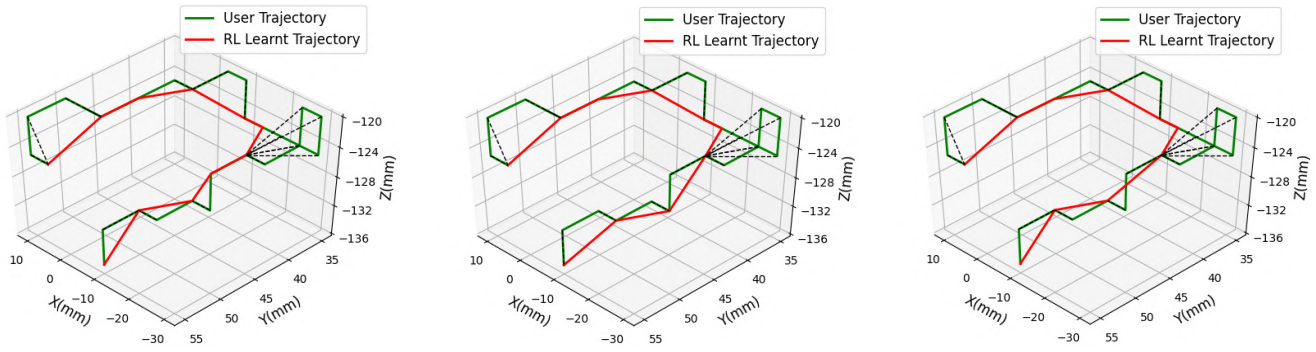

Fig. 8: Learnt RL and discretized user trajectory for 3 different computed policies with DTW to measure similarity

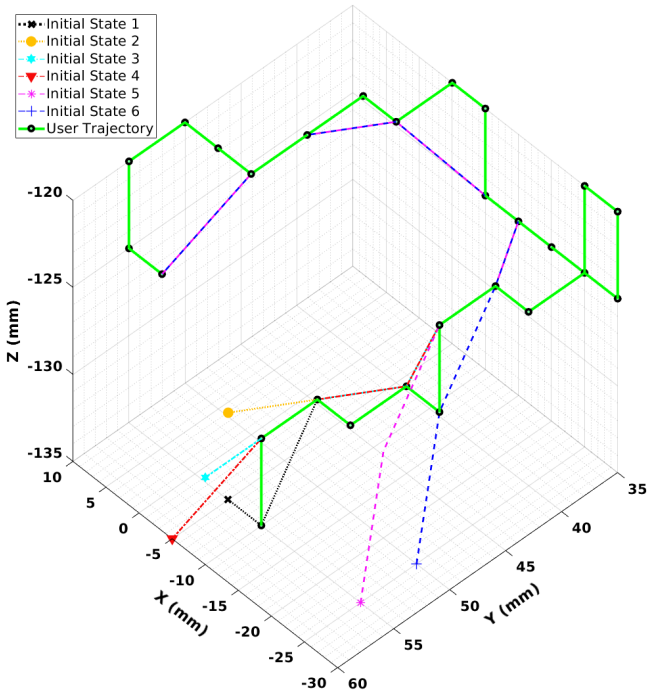

Fig. 9: Trajectory followed by *PSM* after change in the initial state based on the learnt RL policy

Figure (9) shows the agent following the user's trajectory with a deviated initial state. The user's initial position was [-10mm, 55mm, -135mm] and the value of different initial state tried for the learnt agent ranged between [-25mm, -5mm] along the X-axis, [50mm, 60mm] along the Y-axis and [-130mm, -135mm] along the Z-axis. As shown in the figure, the agent was successful at learning the user's trajectory even after significant deviations in initial state of the agent.

V. Discussion

A. Note on the Results

The results show the robustness of learnt RL algorithms and repeatability of the model for successive hand-off tasks. The results from Figure (9) show that the agent was able to follow a trajectory close to that of the user's despite varying the initial position. This is a relevant scenario since the needle may not present itself at the same position over each step. The user's trajectory was successfully learned and reproduced on the dVRK to perform multiple passes of the hand-off task with varied start points, with a similarity mean of 1.706mm of the learned trajectory from the user's trajectory, and a cumulative standard deviation of 2.494mm over the trial conducted by the user and the robot. By automating the hand-off task, clutch-less suturing can be performed and the user can ignore the workspace constraints of the PSM_A.

B. Applications

Due to the high dexterity required for suturing tissue with the da Vinci, obtaining certification to be a trained da Vinci user mandates certain requirements [20]. The methods mentioned in this paper could be employed to train novice users by replicating a reference trajectory of the hand-off task performed by an expert user. Further modifications to the presented methods could be performed to allow haptic feedback to jog the novice user through the trajectory and/or score the novice user's trajectory by evaluating it's standard deviation and mean from the expert user's trajectory [10]. Furthermore, a database of the varying suture techniques can be created using Algorithm (1) to store expert user's trajectories for multiple types of sutures - running suture, figure of 8 suture, simple buried suture etc. [21]. In comparison to current state of the art supervised learning techniques that require a significant amount of data to learn from demonstration, the algorithm presented only required a single trial.

C. Drawbacks

The major drawback of employing discrete RL techniques that follow Bellman's Optimality equation (Dynamic Programming etc.) is the curse of dimensionality [22]. As the number of states and actions increase, deriving the optimal policy becomes computationally exhaustive. Therefore, the algorithm presented is not scalable and requires limited observations at each state. For a system with s states and a actions, a total of a^s computations are performed to obtain the optimal policy at that state for one iteration [18]. Possible workarounds would be to move towards employing Deep RL techniques such as Deep Q-network (DQN) [23] or

explore continuous state space RL algorithms such as Deep Deterministic Policy Gradient (DDPG) [24].

D. Future Work

Although this paper assumes identification of the needle and that the needle has been grasped using a traditional control technique, attempts to further this proposed technique would involve extending it's domain to grasping and identifying the needle as well. The authors are already exploring the Deep RL and continuous state space techniques mentioned in the prior section to overcome the curse of dimensionality observed in this presented work. Hence, future work would include implementing continuous action space algorithms and using policy gradient methods such as DDPG [24], Trust Region Policy Optimization (TRPO) [25] to automate suturing task.

VI. Conclusion

This work is limited to demonstrating the viability of a discrete RL technique to automate the needle hand-off task for collaborative suturing. An algorithm to generate a sparse reward function was designed and a successful implementation of Q-learning was presented. Trajectories obtained from three learnt policies were compared to the user defined trajectory. A RMSE of [0.0044mm, 0.0027mm, 0.0020mm] was observed. The robustness of the algorithm was calculated using multiple trajectories with varying initial positions from a single policy. Finally, a qualitative analysis of the results was presented.

VII. Acknowledgement

This work is supported by the National Science Foundation (NSF) through National Robotics Initiative (NRI) grant: IIS-1637759 and NSF AccelNet grant-1927275. The research work involved using computational resources supported by the Academic and Research Computing group at Worcester Polytechnic Institute. The authors would also like to thank Dhruv Mishra for helping in designing some of the graphics shown in the paper.

References

[1] I. Intuitive Surgical, "2019 annual report," 2019. [Online]. Available: http://www.annualreports.com/Company/intuitive-surgical-inc

[2] Z. Chen, A. Deguet, R. H. Taylor, and P. Kazanzides, "Software architecture of the da vinci research kit," in *2017 First IEEE International Conference on Robotic Computing (IRC)*. IEEE, 2017, pp. 180–187.

[3] G. A. Fontanelli, F. Ficuciello, L. Villani, and B. Siciliano, "Modelling and identification of the da vinci research kit robotic arms," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 1464–1469.

[4] Y. Wang, R. Gondokaryono, A. Munawar, and G. S. Fischer, "A dynamic model identification package for the da vinci research kit," *arXiv preprint arXiv:1902.10875*, 2019.

[5] A. R. Lanfranco, A. E. Castellanos, J. P. Desai, and W. C. Meyers, "Robotic surgery: a current perspective." *Annals of surgery*, vol. 239 1, pp. 14–21, 2004.

[6] A. Sepehripour, G. Garas, T. Athanasiou, and R. Casula, "Robotics in cardiac surgery."

[7] K. Bumm, J. Wurm, J. Rachinger, T. Dannenmann, C. Bohr, R. Fahlbusch, H. Iro, and C. Nimsky, "An automated robotic approach with redundant navigation for minimal invasive extended transsphenoidal skull base surgery," *Minimally invasive neurosurgery : MIN*, vol. 48, pp. 159–64, 07 2005.

[8] C. Busch, R. Nakadate, M. Uemura, S. Obata, T. Jimbo, and M. Hashizume, "Objective assessment of robotic suturing skills with a new computerized system: A step forward in the training of robotic surgeons," *Asian Journal of Endoscopic Surgery*, vol. 12, no. 4, pp. 388–395, 2019. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/ases.12672

[9] J. H. Palep, "Robotic assisted minimally invasive surgery," *Journal of Minimal Access Surgery*, vol. 5, no. 1, p. 1, 2009.

[10] A. Munawar, "An asynchronous simulation framework for multi-user interactive collaboration: Application to robot-assisted surgery," *PhD dissertation*.

[11] N. Padoy and G. D. Hager, "Human-machine collaborative surgery using learned models," in *2011 IEEE International Conference on Robotics and Automation*, May 2011, pp. 5285–5292.

[12] J. 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," *2010 IEEE International Conference on Robotics and Automation, Anchorage, AK, 2010, pp. 2074-2081.*

[13] S. Sen, A. Garg, D. V. Gealy, S. McKinley, Y. Jen, and K. Goldberg, "Automating multi-throw multilateral surgical suturing with a mechanical needle guide and sequential convex optimization," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 4178–4185.

[14] A. Shademan, R. S. Decker, J. D. Opfermann, S. Leonard, A. Krieger, and P. C. W. Kim, "Supervised autonomous robotic soft tissue surgery," *Science Translational Medicine*, vol. 8, no. 337, pp. 337ra64–337ra64, 2016. [Online]. Available: https://stm.sciencemag.org/content/8/337/337ra64

[15] P. Kazanzides, Z. Chen, A. Deguet, G. S. Fischer, R. H. Taylor, and S. P. DiMaio, "An open-source research kit for the da vinci surgical system," in *IEEE Intl. Conf. on Robotics and Auto. (ICRA)*, Hong Kong, China, 2014, pp. 6434–6439.

[16] T. Osa, N. Sugita, and M. Mitsuishi, "Online Trajectory Planning and Force Control for Automation of Surgical Tasks," *IEEE Transactions on Automation Science and Engineering*, vol. 15, no. 2, pp. 675–691, Apr. 2018. [Online]. Available: https://ieeexplore.ieee.org/document/7888981/

[17] N. Ahmidi, L. Tao, S. Sefati, Y. Gao, C. Lea, B. B. Haro, L. Zappella, S. Khudanpur, R. Vidal, and G. D. Hager, "A dataset and benchmarks for segmentation and recognition of gestures in robotic surgery," *IEEE Transactions on Biomedical Engineering*, vol. 64, no. 9, pp. 2025–2041, Sep. 2017.

[18] R. S. Sutton and A. G. Barto, *Introduction to Reinforcement Learning*, 1st ed. Cambridge, MA, USA: MIT Press, 1998.

[19] S. Salvador and P. Chan, "Toward accurate dynamic time warping in linear time and space," *Intell. Data Anal.*, vol. 11, no. 5, p. 561–580, Oct. 2007.

[20] S. Estes, D. Goldenberg, J. Winder, R. Juza, and J. Lyn-Sue, "Best practices for robotic surgery programs." *JSLS*, vol. 21(2):e2016.00102, 01 2017.

[21] J. Meyle, "Suture materials and suture techniques," *Perio*, vol. 3, 01 2006.

[22] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey." [Online]. Available: https://arxiv.org/abs/cs/9605103

[23] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," 2013, cite arxiv:1312.5602Comment: NIPS Deep Learning Workshop 2013. [Online]. Available: http://arxiv.org/abs/1312.5602

[24] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning." [Online]. Available: https://arxiv.org/abs/1509.02971

[25] J. Schulman, S. Levine, M. I. Moritz, Philipp amd Jordan, and P. Abbeel, "Trust region policy optimization." [Online]. Available: https://arxiv.org/abs/1502.05477