

Parallelism in Autonomous Robotic Surgery

Alaa Eldin Abdelaal , *Student Member, IEEE*, Jordan Liu, Nancy Hong, Gregory D. Hager , *Fellow, IEEE*, and Septimiu E. Salcudean , *Fellow, IEEE*

Abstract—Robots can perform multiple tasks in parallel. This work is about leveraging this capability in automating multilateral surgical subtasks. In particular, we explore, in a simulation study, the benefits of considering this parallelism capability in developing execution models for autonomous robotic surgery. We apply our work to two surgical subtask categories: (i) coupled-motion subtasks, where multiple robot arms share the same resources to perform the subtask, and (ii) decoupled-motion subtasks, where each robot arm executes its part of the task independently from the others. We propose and develop parallel execution models for the surgical debridement subtask, a representative of the first category, and the multi-throw suturing subtask, a representative of the second one. Comparing these parallel execution models to the state-of-the-art ones shows significant reductions in the subtasks completion time by at least 40%. In 20 trials, our results show that our proposed model for the surgical debridement subtask, that uses hierarchical concurrent state machines, provides a parallel execution framework that is efficient while greatly reducing collisions between the arms compared to a naive parallel execution model without coordination. We also show how applying parallelism can lead to execution models that go beyond the normal practice of human surgeons. We finally propose the notion of “automation for surgical manual execution” where we argue that autonomous robotic surgery research can be used as a tool for surgeons to discover novel manual execution models that can significantly improve their surgical practice.

Index Terms—Autonomous robotic surgery, laparoscopy, medical robots and systems, planning, surgical robotics, surgical robotics.

I. INTRODUCTION

ROBOT-ASSISTED surgery (RAS) has gained momentum over the last few decades with nearly 1200000 RAS procedures performed in 2019 alone [1] using the da Vinci Surgical

Manuscript received October 15, 2020; accepted January 27, 2021. Date of publication February 18, 2021; date of current version March 8, 2021. This letter was recommended for publication by Associate Editor L. Fichera and Editor P. Valdastri upon evaluation of the reviewers' comments. This work was supported in part by the Natural Sciences and Engineering Research Council of Canada (Discovery Grant), in part by the Canada Foundation for Innovation (infrastructure and operating funds), in part by Intuitive Surgical (equipment donation), in part by the C. A. Laszlo Chair in Biomedical Engineering held by Prof. Salcudean, and in part by the Vanier Canada Graduate Scholarship held by Alaa Eldin Abdelaal. (*Corresponding author: Alaa Eldin Abdelaal*)

Alaa Eldin Abdelaal, Nancy Hong, and Septimiu E. Salcudean are with the Electrical and Computer Engineering Department, University of British Columbia, Vancouver, BC V6T 1Z4, Canada (e-mail: aabdelaal@ece.ubc.ca; nancyhong321@gmail.com; tims@ece.ubc.ca).

Jordan Liu is with the Department of Computer Science, University of British Columbia, Vancouver, BC V6T 1Z4, Canada (e-mail: jliu0@student.ubc.ca).

Gregory D. Hager is with the Department of Computer Science, Johns Hopkins University, Baltimore, MD 21218 USA (e-mail: hager@cs.jhu.edu).

This letter has supplementary downloadable material available at <https://doi.org/10.1109/LRA.2021.3060402>, provided by the authors.

Digital Object Identifier 10.1109/LRA.2021.3060402

system, the most widely used surgical robotics platform [2]. RAS has been shown to be successful in many surgical specialties, most notably urology [3] and gynecology [4].

In the majority of RAS platforms currently used, the surgeon teleoperates the master to control robotic instruments inside the patient. An endoscopic camera provides visual feedback. Sophisticated computer processing of the information transfer between the surgeon and the instruments can provide additional functionality to the system to improve the teleoperation experience, e.g., by filtering tremor and scaling down the surgeon's motions for precise task execution [5]. This computer processing also provides an opportunity to control the motions of the patient-side manipulators (PSMs) to automate surgical subtasks.

RAS automation of some tedious and repetitive surgical subtasks, such as suturing, has been a topic of interest for several research groups [6]–[8]. RAS automation can facilitate tele-surgery, especially when there are significant delays between the control console and the PSMs. Moreover, automation of surgical subtasks using more than one robotic arm has the potential to reduce surgery and operating room (OR) time, lowering the cost of procedures and shortening the time the patient is under anesthesia.

This work focuses on the execution modeling aspect of automating surgical subtasks. Previous work in this area use execution models that are *sequential* in nature, meaning that the model itself allows one robot arm to move at any given time while the remaining arms are waiting for it to finish [9]–[11]. In this work, we explore the feasibility of devising *parallel execution models* for multilateral automation of surgical subtasks, where two or more robot arms can move at the same time to execute the surgical subtask, sometimes in a completely different way than one would expect from a human surgeon. In particular, the contributions of this letter are as follows:

- We apply the concept of parallelism for execution modeling to two surgical subtask categories: (i) Coupled-motion subtasks where multiple arms share the same resources (which requires some form of coordinated execution) (ii) Decoupled-motion subtasks where multiple arms can execute the same subtask independently from each other.
- We compare our parallel execution models with the sequential ones and show the significant improvements that result from applying parallelism to automate surgical subtasks.
- We show how the application of the parallelism concept can lead to execution models for surgical subtasks that go beyond the normal practice of human surgeons.
- We introduce the notion of “automation for surgical manual execution,” arguing that surgical subtask automation research can fundamentally change the way surgeons are

manually executing these same surgical subtasks by providing them with “out-of-the-box” manual execution models/plans.

II. RELATED WORK

Many groups are developing autonomous robot systems for a variety of surgical tasks/subtasks. Some consider the use of single arm robot to perform the task at hand. For example, Shademan *et al.* [12] build an autonomous system for soft tissue surgery. Their system was tested in *ex vivo* and *in vivo* tasks that include linear suturing or anastomosis. Hu *et al.* [13] propose a semi-autonomous system for brain tumor ablation using the RAVEN surgical robot [14] and stereo visual feedback. They decompose the brain tumor ablation task into stages and represent it using a behavior tree [15]. To test their system, they simulate the entire task using a planar surface and the task then is to remove iron fillings from this surface using image guidance by one of the RAVEN arms that is equipped with a suction tool.

Autonomous multilateral task execution has the potential of decreasing task completion time and it may even be necessary for the execution of some surgical subtasks such as knot tying. Kehoe *et al.* [16] use explicit programming to automate a surgical subtask using two arms of the RAVEN surgical robot. They apply their work on the debridement subtask which involves removing damaged tissue samples from healthy ones. Their results show that using two arms of the RAVEN is faster in execution than using only one.

Learning from demonstration has been successfully used in automating multilateral execution of surgical subtasks. Van den Berg *et al.* [17] collect several demonstrations of surgical subtasks. They use these demonstrations to find a reference trajectory for the robot to follow to execute these tasks. After that, they use iterative learning control methods [18] to gradually increase the robot’s execution speed without having an accurate model of the robot dynamics at higher speeds. They apply their system to two subtasks; executing a figure eight trajectory and the two-handed knot tying subtask. Murali *et al.* [9] use human demonstrations to automate two surgical subtasks, namely a 3D debridement subtask and a 2D pattern cutting subtask. They analyze the collected demonstrations and they manually segment them into states or motion sequences and transition conditions between those states. Based on this segmentation step, they build a finite state machine for each of the two subtasks. Their system updates the parameters and states of the finite state machines by repeatedly executing the subtasks. Their system performs the two subtasks successfully 96% and 70% of the total evaluation trials for the debridement and pattern cutting subtasks, respectively. A similar approach was used to automate different variations of the suturing subtask as in [10] and [19].

In all the above works, the surgical subtask is executed using a sequential execution model. For example, the work in [9], [10] and [11] model each surgical subtask as a finite state machine which by definition means that the automation system is always in one single state at any given time. This leads to a sequential subtask execution where only one arm robot is moving at any given time while the other arms are waiting. In

this work, we argue that using parallelism in execution models of these subtasks can improve their automation, leading to faster completion times than using sequential models.

In this respect, the closest work to ours is reported in [16], where two arm robots move in parallel to execute a surgical debridement subtask. The paper is concerned with improving the state estimation pipeline to overcome the uncertainties resulting from dealing with cable-driven systems by using model predictive control methods [20]. In contrast, we focus here on the execution modeling aspect to improve the automation of surgical subtasks.

Another observation is that in the above autonomous robotic surgery papers, the execution model simply imitates how a human surgeon would perform the task. While this approach provides a good reference for the task execution, it does not necessarily take into account the differences in capabilities between humans and robots. An obvious example to this is that robots can move much faster than a human. Van den Berg *et al.* [17] exploit this capability by proposing a method to speed up the robot’s task execution of a knot tying subtask to be 10 times faster than the human. This work, however, uses the same execution model that a human is using to execute this subtask. In other words, the steps of the knot tying subtask are the same as what a human follows, but they are executed much faster.

In our letter, we propose rethinking the steps/plans of surgical subtasks to leverage the robot’s capability of performing more than one action in parallel. In particular, we argue that considering this capability in execution models can lead to task executions that go beyond the normal practice of human surgeons.

III. PROPOSED EXECUTION MODELS

Our fundamental observation is that multiple robot arms can be moving in parallel when planning the automation of surgical subtasks. This parallel motion of the robot arms decomposes the subtasks that they can carry out into two categories (i) Coupled-motion subtasks and (ii) Decoupled-motion subtasks. This decomposition is based on whether the arm robots share the same resources. In the next sub sections, we consider each of these two categories and we present execution models/plans for representative subtasks where the parallelism concept is considered.

It is important to note that the proposed plans are independent of their representation, e.g., using binary trees or state machines. They are also independent of the implementation details used to deploy these plans into surgical robots. That is why we follow a simple methodology to demonstrate the parallelism concept. Due to the COVID-19 situation and limited access to real robots in our laboratory, we demonstrate the work in simulation. Our plans described below are applied to a simulated da Vinci system and the two subtasks that we consider are performed using two da Vinci patient side manipulators (PSMs).

A. Coupled-Motion Subtasks

Coupled-motion subtasks in this context refer to subtasks where there is a need to coordinate the sharing of resources between the multiple arm robots that are moving in parallel.

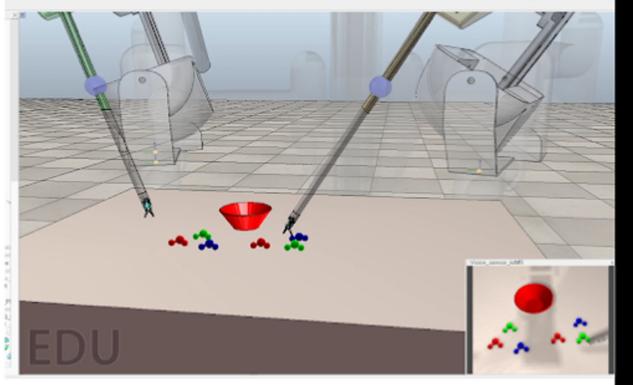


Fig. 1. Surgical debridement subtask simulation. Colored objects represent damaged tissue to be collected by the two PSMs and dropped into the red bowl. The robot camera view is shown in the bottom right corner.

We choose a subtask representative of surgical debridement. In surgical debridement, the goal is to remove damaged tissue so that the remaining healthy tissue can heal faster [21]. The task involves detecting damaged tissue, grasping it piece by piece, then moving each piece into a common location (e.g. a basket that can be removed through one of the surgical ports). It has been a subtask of interest in autonomous robotic surgery research as in [16] and [9]. We consider a simplified version of the surgical debridement environment where we have damaged tissue scattered on a common surface. The goal of the robot arms is to detect, grasp and place the damaged tissue pieces into a common location as shown in Fig. 1.

The proposed parallel execution model of this subtask is represented using a hierarchical concurrent state machine (HCSM) [22]. Each finite state machine in an HCSM can have sub-states machines (hence the term “hierarchical”) that can work in parallel (and hence the term “concurrent”). HCSMs allow the system they represent to be in multiple states at any given time which allows them to model and represent more complex behaviors compared with finite state machines (FSMs).

The main idea of our proposed execution model is to coordinate the motion of the two arm robots conducting the surgical debridement subtask so that one robot picks up a new piece of damaged tissue while the other one drops its own piece, at the same time. The overview representation of the entire execution model is shown in the HCSM in Fig. 2. The detailed representations of the components/sub-state machines are shown in Figs. 3, 4 and 5. In all these figures, we use the term “object” to refer to the tissue sample and the term “destination” to refer to the dropping location of the tissue samples.

The execution model/plan starts with the sub state machine on the left in Fig. 2 when the two arms prepare for the task by opening their jaws at the same time, as demonstrated by the two concurrent state machines named “open jaws”. The plan then continues to the next sub-state machine once both jaws are open. This sub-state machine, named “picking up objects,” consists in turn of two concurrent and identical state machines, one for each arm (or PSM). In each of these state machines, each arm approaches the closest damaged tissue to its location and

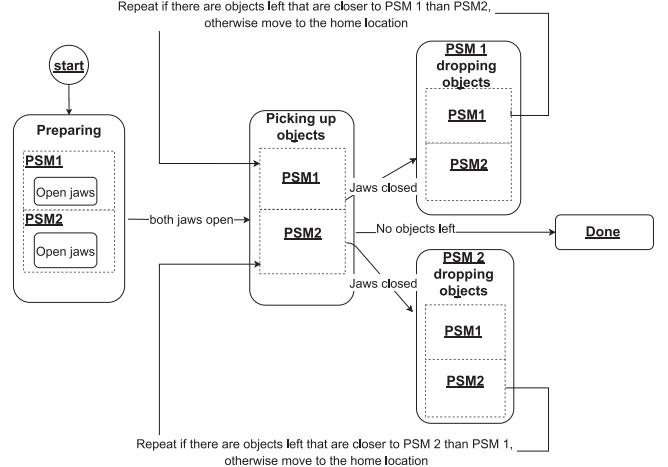


Fig. 2. An overview of the HCSM representation of the proposed parallel execution model for the surgical debridement subtask.

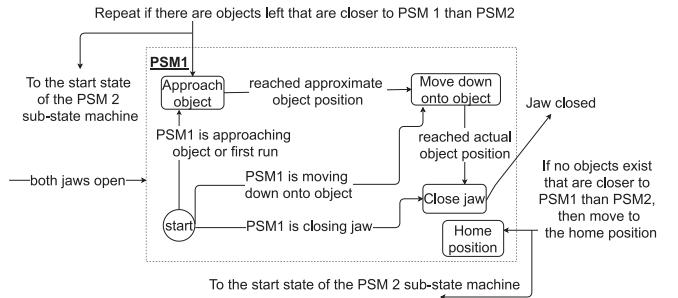


Fig. 3. The PSM1 sub state machine, which is a part of the “picking up objects” sub state machine shown in Fig. 2.

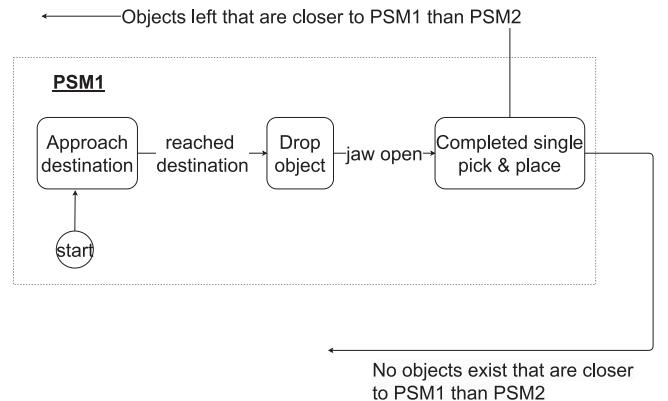


Fig. 4. The PSM1 sub state machine, which is a part of the “PSM1 dropping objects” sub state machine shown in Fig. 2.

closes its jaws once it reaches the tissue to perform the grasping as shown in Fig. 3. Based on which arm closes its jaws first, the plan continues so that the first arm to close its jaws moves towards the common location to drop the damaged tissue as shown in Fig. 4 (assuming that PSM1 is the first arm to close its jaws, but the same sequence applies to PSM2 using a similar sub-state machine). In the unlikely case when the two arms close

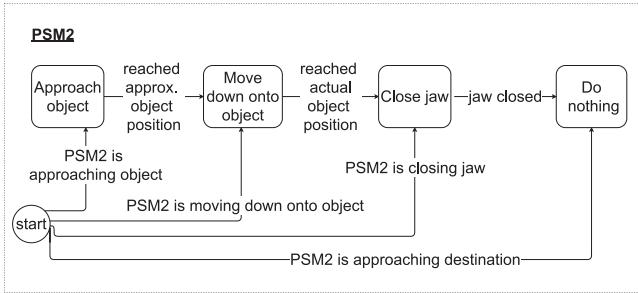


Fig. 5. The PSM2 sub state machine, which is a part of the “PSM1 dropping objects” sub state machine shown in Fig. 2.

their jaws at exactly the same time, the arm closer to the dropping location will be the first to move. If the two arms are at the same distance from the dropping location in this case, one arm will be randomly chosen to move first. At the same time, the plan makes sure that the other arm continues its motion to approach its own (assigned) tissue and grasp it by closing its jaws once it reaches the tissue location. This same arm then waits until the first arm drops its tissue as shown in Fig. 5 (assuming that PSM2 is the second arm to close its jaws, but the same sequence applies to PSM1 using a similar sub state machine). After that, the plan continues in a similar way but this time to let the second arm drop its tissue while the first arm goes back to collect a new damaged one. The same sequence is then repeated until there is no damaged tissue left for any of the two arms.

B. Decoupled-Motion Subtasks

Decoupled-motion subtasks refer to subtasks that can be carried out by multiple robot arms that do not share the same resources after applying the parallelism concept to their execution model. We choose the multi-throw suturing (MTS) subtask as a representative of this category. MTS is a suturing technique that is used to decrease the wound detension and it does not involve knot tying [23]. This subtask falls under the knotless suturing techniques similar to barbed suturing, which is used in many surgical applications [24]. One throw of the MTS subtask involves the insertion of a needle with a suture thread from one side of the wound and pulling it out from the other side until the suture is tight. Multiple throws are carried out to cover the entire wound. Suturing in general has been studied extensively in the context of autonomous robotic surgery [12], [25] and MTS was first studied in this context in [10] where a sequential execution model was used to automate it.

We consider a simplified version of the MTS that does not include a thread, as shown in Fig. 6, where we assume that the needle trajectory can be fitted by half a circle with known entry and exit points. We also assume that these entry and exit points are given *a priori* similar to the case in [26]. The autonomous execution starts with the needle being grasped by one of the robot arms. The goal of the autonomous system is to first detect the locations of the entry and exit points for each throw using vision-based methods. The needle is then moved to the entry point where it is inserted until it appears from the corresponding

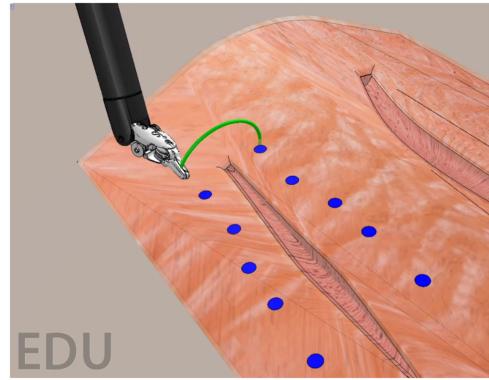


Fig. 6. The MTS subtask setup. The needle is shown in green and the blue points along the wound represent the entry and exit points for all the throws.

exit point. It is then pulled back from the tissue and the same sequence is repeated until all the throws are finished.

Our proposed execution model for MTS involves using two of the robot arms, PSM 1 and PSM 2. The first arm starts the suturing execution from one end of the wound and the second arm starts a similar suturing sequence but at the middle of the wound. From example, if the wound needs six throws in total, PSM 1 starts from the first throw and PSM 2 starts from the fourth throw at the same time. The two arms then move along the same direction of the wound, which eliminates the possibility of collisions between the two arms. The last throw for the first arm is the one right before the first throw for the second arm. The second arm’s last throw is the one at the distal end of the wound. In the above example, this means that PSM 1 carries out the first three throws and PSM 2 carries out the last three.

The proposed execution model in this case consists of two parallel FSMs, one for each arm. Each FSM starts with locating the entry and exit points of the throw at hand. Then, the arm moves towards the entry point to insert the needle. The needle is inserted after that until its end appears from the corresponding exit point. The arm then drops the needle (which stays stationary as the tissue is holding it) and moves to grasp it from the exit point side. The needle is finally pulled out from the tissue and the arm reorients itself to start the next throw. The execution of each FSM continues until the corresponding arm finishes all its assigned throws. Using one arm to insert the needle without the need to support the tissue is consistent with autonomous suturing frameworks in the literature as in [27] and [26].

The proposed MTS parallel execution model transforms a “coupled-motion subtask,” where the two arms are usually part of each throw’s execution as in [10], into a “decoupled-motion subtask” where two single-arm throws are carried out in two locations at the same time. Another important aspect is that this proposed parallel execution model is completely different from the traditional manual execution of the MTS subtask (which is similar to the automated execution in [10]).

The proposed MTS parallel execution model shows that by considering the parallelism capability of multi-arm robotic systems, we can generate a faster and completely different execution sequence that goes beyond the manual execution of

expert surgeons. We argue that it is generally hard for humans to pay simultaneous attention to two (or more) locations when performing a delicate subtask such as MTS. The reader can gauge the difficulty of the task by imagining holding two needles, one in each hand, then accurately targeting them simultaneously towards two different locations, inserting them, and pulling them back out along a circular trajectory! Robots do not have the same “cognitive” limitation in these tasks and our proposed execution model simply shows an instance of what can be done when we leverage this capability in automating surgical subtasks.

IV. SIMULATION ENVIRONMENT AND EXPERIMENTS

We tested the proposed execution models on a simulated first generation da Vinci surgical system. We used the open source da Vinci simulator proposed in [28] which simulates a full da Vinci patient-side cart with two PSMs and the da Vinci endoscope. The simulator is interfaced with the Robot Operating System (ROS) and the simulated robot can be controlled using the da Vinci Research Kit (dVRK) software [29] similar to controlling the real patient-side cart. The simulator also has some built-in scenes representing different surgical subtasks, two of which are used in this work. The simulator has been used before in RAS applications as in [30] and [31]. In the next two subsections, we present the specific setup used for evaluating each of the two proposed execution models.

A. Surgical Debridement Simulation Experiment

The experimental setup used for the surgical debridement subtask is shown in Fig. 1. We compared our parallel execution model to the following three task execution models, all described by FSMs:

- One arm execution (Case I): This case is the baseline. The execution starts with the arm moving towards the closest tissue sample to grasp it. The arm then approaches the bowl location and drops the sample. The execution continues until there are no tissue samples left.
- Two-arm sequential execution (Case II): This is where two arms perform the subtask in a sequential manner. That is, at any given time, there is only one arm moving while the other is waiting for it to finish. The subtask execution is represented as an FSM similar to the one used in the surgical debridement subtask in [9]. Equal number of tissue samples is assigned to each arm. The subtask execution starts with one arm moving its assigned tissue samples one by one, while the other arm is waiting. After the first arm finishes, the second arm carries out its part of the subtask in the same way. This case is considered to quantify the differences between sequential task execution and our proposed parallel execution model.
- Two-arm parallel execution without coordination (Case III): In this case, the two arms are moving independently and in parallel to execute the subtask. Each arm is exactly following the same FSM used in the first case above. There is no coordination between the two arms in this case, unlike the proposed parallel execution model. This case is considered to quantify the effect of coordinating the motion

of the two arms in a coupled-motion subtask and compare that with the proposed parallel execution model that takes coordination into account.

To compare between the proposed parallel execution model and the above three cases, we performed 20 trials of the surgical debridement subtask for each case (a total of 80 trials). We generated 20 sets of random locations for the tissue samples in the subtask environment and all the four cases were tested on each of these random sets. We used subtask completion time and the number of collisions between the arms as our performance metrics.

As for the implementation details, we used a simple vision-based method to locate both the damaged tissue and bowl. Our method uses upper and lower Hue, Saturation, Value (HSV) bounds to find the contours around objects of particular colors. We then find the centroids of each of these contours in image coordinates. The images from the left and right stereo cameras are then used to find the corresponding 3D location of each object of interest in the scene. In a real scenario, perception frameworks such as the one in [32] can be used to extract the visual information needed for our proposed execution models. For grasping each tissue sample, each arm is instructed to move right above the tissue sample location, then move down towards the sample until it reaches a pre-defined distance, and finally close the jaws to grasp the sample. Each arm is instructed to move above the midpoint of the bowl, where it opens its jaws to drop the tissue sample. We used an even number of tissue samples and they were evenly assigned to each arm based on how close they are to each of them.

B. MTS Simulation Experiment

The experimental setup used for the MTS subtask is shown in Fig. 6. We compared the proposed parallel plan with two cases as follows:

- Single-arm execution of the subtask (Case A): In this case, only one arm was used to perform the entire subtask. The execution starts with the arm holding the needle and moving towards the entry point. The arm then moves/rotates to insert the needle into the simulated tissue until the needle’s other end appears from the exit point. After that, the arm releases the needle and moves to the exit point side to grasp the other side of the needle. The arm rotates again to almost pull the entire needle out of the tissue. A small part of the needle is intentionally left inside the tissue after that so that the arm can release the needle and pick it up again from the exit point side. The arm then pulls the needle entirely from the tissue. This last step is necessary so that the same end of the needle is always inserted into the tissue. The same sequence is then repeated until the needle goes through all the throws. This case serves as the baseline case.
- Two-arm traditional subtask execution (Case B): In this case, we simulated the traditional two-arm execution of the MTS as an FSM. The execution starts with the same first step as the first case above (Case A). Then, the second arm is instructed to grasp the other side of the needle and rotate to pull it out, leaving a small part of it inside the tissue. The

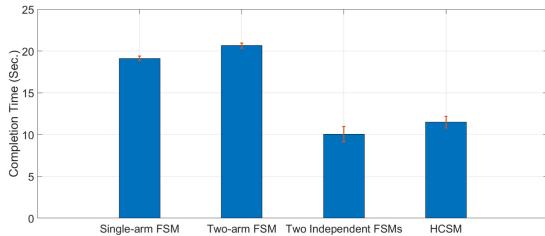


Fig. 7. The completion time results of the surgical debridement subtask experiments.

first arm approaches the needle again to completely pull it out from the tissue. The first arm then reorients itself to perform the next throw. The sequence is repeated until all throws are performed. The execution in this case is similar to the automated MTS subtask in [10] except that it does not include a handover of the needle between the two arms as this is not needed in our setup. This case is considered to quantify the differences between a human-like plan to automate the subtask execution and the proposed parallel plan.

To compare between the proposed parallel plan and the above two cases, we performed 20 trials of the MTS subtask using each of the three cases. In each trial, we randomize the location of the tissue by rotating it around its vertical axis by a random angle, chosen between -15° and 15° . After each random rotation, we test each one of the three cases. For performance metrics, we choose completion time only because in this case collisions between arms in all the three cases are eliminated by the design of the subtask execution model.

As for the implementation details, we used the same vision-based method described in IV-A to locate the entry and exit points as well as the needle's two ends. For the needle insertion, we used a method similar to the algorithm proposed in [26] where, for each throw, a circular trajectory with the same radius as the needle is computed such that it passes through both the entry and exit points marked onto the tissue.

It is important to note that while the evaluation environment is in simulation with some ideal parameters, the same ideal conditions were used towards the other cases with which we compare the proposed parallel plans in both the surgical debridement and MTS subtasks. In other words, this setup allows us to fix all variables except for the subtask plans and hence enables us to quantify the effect of changing this variable on the performance of the automated subtasks.

V. RESULTS

A. Surgical Debridement Results

Fig. 7 shows the completion time of each case in the surgical debridement subtask experiments. The two parallel execution models represented in the last two bars (the two independent FSMs representing case III in IV-A and HCSM representing the proposed execution model) achieved the best completion times. Our proposed execution model took an average of 11 seconds to complete, slightly longer than case III which took 10 seconds

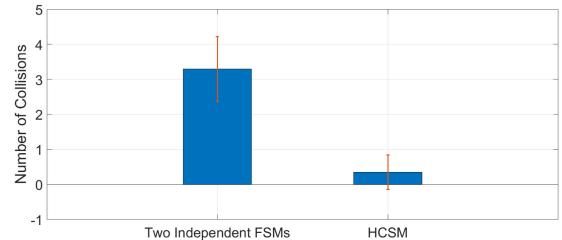


Fig. 8. The number of collisions in each case of the surgical debridement subtask experiments.

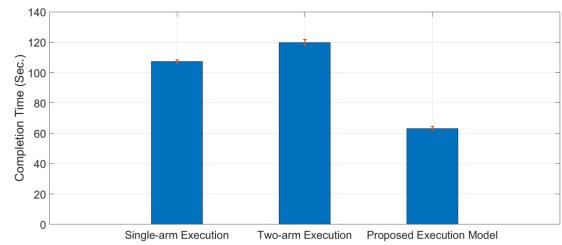


Fig. 9. The completion time results of the MTS subtask experiments.

on average. The two parallel execution models are at least 40% faster than any of the two sequential plans which shows the large improvements resulting from applying the parallelism concept for automating this subtask.

Fig. 8 shows the number of collisions in the two parallel execution models as the two sequential models have no collisions. An average of 0.35 collision per trial occurred under our execution model compared with an average of three collisions per trial in the case of using two independent FSMs in case III. The reported collisions in our case occurred when the following occurred at the same time: (i) one arm has picked up an object that is very close to the center line dividing half of the objects, and (ii) the other arm has dropped an object at the bowl and is approaching another object close to the center line.

The collision results show the advantage of considering the coordination between multiple arms when using the parallelism concept. These results also show that the edge of case III in terms of completion time is outweighed by the large improvement of our proposed execution model when it comes to the number of collisions. Furthermore, we were only able to detect the collisions as they occur in the simulator, but we could not add any reasonable physics-based consequences on the arms motion if they collide. This means that the completion time of case III, had these consequences been added, would have been much longer than the reported time, which most likely would have resulted in having our proposed execution model the fastest case.

B. MTS Results

Fig. 9 shows the completion time of the three cases considered in the MTS subtask simulation experiments. Our proposed execution model achieved the best completion time by taking 63 seconds on average to finish the subtask. This completion time is 47% shorter than the completion time in the traditional two-arm

execution model and 41% shorter than the completion time in the single arm execution. This shows the significant potential of using parallel execution models to automate decoupled surgical subtasks.

VI. DISCUSSION

The above results show the significant potential of considering the parallelism capability of multi-arm robotic systems in autonomous robotic surgery. A direct advantage is the large improvements in the subtask execution times compared with using sequential execution models. An important aspect to achieve the full potential of this capability is the coordination of the motion of multiple arm robots, especially in coupled-motion subtasks such as the surgical debridement. Another advantage of the parallelism capability is that innovative parallel execution models of surgical subtasks can actually turn a normal coupled-motion subtask into a decoupled-motion one as we presented in our proposed model for the MTS subtask. Our proposed model allows each arm to have its own workspace that does not intersect with the workspace of the other arm(s) and hence preventing collisions from happening. This advantage can greatly reduce the probability of collisions between the robot arms which in turn can facilitate the design and implementation of the parallel execution models.

Based on our results, we propose a novel application for autonomous robotic surgery research: *the use of novel automation strategies to guide the manual execution of surgical subtasks*. This is what we refer to by the term “automation for surgical manual execution”. Automation research allows us to explore new areas of the task/subtask planning space, that can go beyond normally used areas of this space for the manual execution of surgical subtasks. Our above results in the MTS subtask show a single instance of the manual exploration of this space and this leads to the large improvements described above. This was done by devising hand-crafted execution models which was a tedious task. We hope that automated task design, employing optimization and learning techniques such as those described in [33], will provide significant and quantifiable improvements across a broad range of tasks. This approach is conceptually similar to the use of reinforcement learning in the AlphaGo project on board games where the learning agent was able to apply novel strategies beyond what is used by the world champions of this game [34]. We envision a similar approach in autonomous robotic surgery, but instead of applying the newly discovered strategies directly to the surgical practice, we discuss them with surgeons with the goal of finding novel manual execution plans of the same subtask and/or developing human-in-loop systems where the surgeon and an autonomous agent can share the task execution steps [35]. We argue that our results in the MTS subtask above is a first promising step towards realizing the proposed vision of “automation for surgical manual execution” in robot-assisted surgery.

We plan to build upon the current work to overcome some of its limitations. Motion planning methods can be employed to eliminate collisions between arms in the surgical debridement subtask execution model. In addition, more complex versions

of the chosen subtasks will be used to test the feasibility of applying the parallelism concept on real robots. For example, we plan to test this on the MTS subtask when using a thread. This may require more robust design of the execution models to avoid problems such as collisions between the arms and threads, and suture thread entanglement. Furthermore, surgical planning and port placement methods can be used to avoid joint limits during the autonomous execution of the subtasks. In the current work, we manually adjusted the locations of the arms and subtask spaces to avoid this problem. Moreover, comparing the performance of our proposed execution models with humans is needed. We showed that parallel execution models are faster than sequential ones. It would be interesting to see how this plays out when compared with expert surgeons.

VII. CONCLUSIONS AND FUTURE WORK

This work explored the benefits of applying the concept of parallelism in generating novel execution models to automate multilateral surgical subtasks. We proposed two parallel execution models to automate the surgical debridement and multi-throw suturing subtasks. The former was a representative of coupled-motion subtasks where the coordination between multiple robot arms is necessary for the task execution. The latter was an example of how applying the parallelism concept can turn a traditional coupled-motion subtask into a decoupled-motion one, which can facilitate the design of execution models that significantly reduce the possibility of collisions between the robot arms. Our simulation experiments showed that our proposed parallel execution models can lead to at least 40% decrease in the subtasks completion time.

Our work opens up new directions on multiple fronts. On the task planning aspect, a promising direction is researching how to automate the generation of parallel execution models which can facilitate the application of these models in a wider range of surgical subtasks. On the multi-robot systems front, we believe that our work, on one hand, paves the way to applying established methods in this huge area to automate multilateral surgical subtasks. On the other hand, our work also introduces surgical robotics as an area of study for multi-robot systems research with all its unique challenges. On the surgical robotics design front, we hope that this line of work can inspire the design of robotic systems that can actually facilitate the execution of surgical subtasks in a more parallel fashion, which could increase the throughput of using these new systems in hospitals. Finally, on the simulation-based surgical robotics research, we argue that our proposed notion of “automation for surgical manual execution” opens up a new application area for this research regardless of how accurate the available simulations are. Techniques such as reinforcement learning can be used, based on our notion, to explore new areas of the execution modeling space which could lead to fundamental changes in the manual surgical practice in robot-assisted surgery.

ACKNOWLEDGMENT

We would like to thank Linh Tran and Edward Kim for their assistance in the early stages of this work.

REFERENCES

- [1] *Intuitive Surgical Investor Overview*. 2019. [Online]. Available: <https://isrg.gcs-web.com/>
- [2] C. P. Childers and M. Maggard-Gibbons, "Estimation of the acquisition and operating costs for robotic surgery," *JAMA*, vol. 320, no. 8, pp. 835–836, 2018.
- [3] R. Autorino and F. Porpiglia, "Robotic surgery in urology: The way forward," *World J. urol.*, vol. 38, no. 4, 2020, Art. no. 809.
- [4] S. E. Kristensen *et al.*, "Robot-assisted surgery in gynecological oncology: Current status and controversies on patient benefits, cost and surgeon conditions-a systematic review," *Acta Obstetricia Et Gynecologica Scandinavica*, vol. 96, no. 3, pp. 274–285, 2017.
- [5] S. DiMaio, M. Hanuschik, and U. Kreaden, *The Da Vinci Surgical System*. Boston, MA: Springer US, 2011, pp. 199–217. [Online]. Available: https://doi.org/10.1007/978-1-4419-1126-1_9
- [6] M. Yip and N. Das, "Robot autonomy for surgery," *Encyclopedia Med. Robot.*, (In 4 Volumes), 2018, Art. no. 281.
- [7] A. Attanasio, B. Scaglioni, E. De Momì, P. Fiorini, and P. Valdastri, "Autonomy in surgical robotics," *Annu. Rev. Control. Robot. Auton. Syst.*, vol. 4, no. 1, 2021, doi: [10.1146/annurev-control-062420-090543](https://doi.org/10.1146/annurev-control-062420-090543).
- [8] G. Fagogenis *et al.*, "Autonomous robotic intracardiac catheter navigation using haptic vision," *Sci. Robot.*, vol. 4, 2019, Art. no. eaaw1777.
- [9] A. Murali *et al.*, "Learning by observation for surgical subtasks: Multilateral cutting of 3d viscoelastic and 2d orthotropic tissue phantoms," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2015, pp. 1202–1209.
- [10] 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 *Proc. IEEE Int. Conf. Robot. Automat.*, 2016, pp. 4178–4185.
- [11] S. McKinley *et al.*, "Autonomous tumor localization and extraction: Palpation, incision, debridement and adhesive closure with the da vinci research kit," in *Proc. Hamlyn Surg. Robot. Conf.*, 2015, p. 27.
- [12] A. Shademan, R. S. Decker, J. D. Opfermann, S. Leonard, A. Krieger, and P. C. Kim, "Supervised autonomous robotic soft tissue surgery," *Sci. Transl. Med.*, vol. 8, no. 337, pp. 337ra64–337ra64, 2016.
- [13] D. Hu, Y. Gong, B. Hannaford, and E. J. Seibel, "Semi-autonomous simulated brain tumor ablation with ravenii surgical robot using behavior tree," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2015, pp. 3868–3875.
- [14] B. Hannaford *et al.*, "Raven-II: An open platform for surgical robotics research," *IEEE Trans. Biomed. Eng.*, vol. 60, no. 4, pp. 954–959, Apr. 2013.
- [15] A. Marzinotto, M. Colledanchise, C. Smith, and P. ögren, "Towards a unified behavior trees framework for robot control," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2014, pp. 5420–5427.
- [16] B. Kehoe *et al.*, "Autonomous multilateral debridement with the raven surgical robot," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2014, pp. 1432–1439.
- [17] J. Van Den *et al.*, "Superhuman performance of surgical tasks by robots using iterative learning from human-guided demonstrations," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2010, pp. 2074–2081.
- [18] D. A. Bristow, M. Tharayil, and A. G. Alleyne, "A survey of iterative learning control," *IEEE Control Syst. Mag.*, vol. 26, no. 3, pp. 96–114, Jun. 2006.
- [19] T. Osa, N. Sugita, and M. Mitsuishi, "Online trajectory planning and force control for automation of surgical tasks," *IEEE Trans. Automat. Sci. Eng.*, vol. 15, no. 2, pp. 675–691, Apr. 2018.
- [20] J. B. Rawlings, "Tutorial overview of model predictive control," *IEEE Control Syst. Mag.*, vol. 20, no. 3, pp. 38–52, Jun. 2000.
- [21] M. Granick, J. Boykin, R. Gamelli, G. Schultz, and M. Tenenhaus, "Toward a common language: Surgical wound bed preparation and debridement," *Wound Repair Regeneration*, vol. 14, pp. S1–S10, 2006.
- [22] O. Ahmad, J. Cremer, J. Kearney, P. Willemse, and S. Hansen, "Hierarchical, concurrent state machines for behavior modeling and scenario control," in *Proc. 5th Annu. Conf. AI, Plan. High Autonomy Syst.*, 1994, pp. 36–42.
- [23] J. Hochberg, K. M. Meyer, and M. D. Marion, "Suture choice and other methods of skin closure," *Surg. Clin. North Amer.*, vol. 89, no. 3, pp. 627–641, 2009.
- [24] C. Iavazzo, I. Mamais, and I. D. Gkegkes, "The role of knotless barbed suture in gynecologic surgery: Systematic review and meta-analysis," *Surg. Innovation*, vol. 22, no. 5, pp. 528–539, 2015.
- [25] S. A. Pedram, P. Ferguson, J. Ma, E. Dutson, and J. Rosen, "Autonomous suturing via surgical robot: An algorithm for optimal selection of needle diameter, shape, and path," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2017, pp. 2391–2398.
- [26] S. Iyer, T. Looi, and J. Drake, "A single arm, single camera system for automated suturing," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2013, pp. 239–244.
- [27] S. A. Pedram, C. Shin, P. W. Ferguson, J. Ma, E. P. Dutson, and J. Rosen, "Autonomous suturing framework and quantification using a cable-driven surgical robot," *IEEE Trans. Robot.*, to be published, doi: [10.1109/TRO.2020.3031236](https://doi.org/10.1109/TRO.2020.3031236).
- [28] G. A. Fontanelli, M. Selvaggio, M. Ferro, F. Ficuciello, M. Vendittelli, and B. Siciliano, "A v-rep simulator for the da vinci research kit robotic platform," in *Proc. IEEE Int. Conf. Biomed. Robot. Biomechatronics*, 2018, pp. 1056–1061.
- [29] 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 *Proc. IEEE Int. Conf. Robot. Automat.*, 2014, pp. 6434–6439.
- [30] M. Selvaggio, G. A. Fontanelli, F. Ficuciello, L. Villani, and B. Siciliano, "Passive virtual fixtures adaptation in minimally invasive robotic surgery," *IEEE Robot. Automat. Lett.*, vol. 3, no. 4, pp. 3129–3136, Oct. 2018.
- [31] M. Selvaggio, R. Moccia, F. Ficuciello, and B. Siciliano, "Haptic-guided shared control for needle grasping optimization in minimally invasive robotic surgery," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 3617–3623.
- [32] Y. Li *et al.*, "Super: A surgical perception framework for endoscopic tissue manipulation with surgical robotics," *IEEE Robot. Automat. Lett.*, vol. 5, no. 2, pp. 2294–2301, Apr. 2020.
- [33] A. Colomé and C. Torras, *Reinforcement Learning of Bimanual Robot Skills*. Berlin, Germany: Springer, 2020.
- [34] D. Silver *et al.*, "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, pp. 354–359, 2017.
- [35] N. Padoy and G. D. Hager, "Human-machine collaborative surgery using learned models," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2011, pp. 5285–5292.