Proceedings Paper:

Attanasio, A, Marahrens, N orcid.org/0000-0002-5982-4990, Scaglioni, B orcid.org/0000-0003-4891-8411 et al. (1 more author) (2021) An Open Source Motion Planning Framework for Autonomous Minimally Invasive Surgical Robots. In: 2021 IEEE International Conference on Autonomous Systems (ICAS). 2021 IEEE International Conference on Autonomous Systems (ICAS), 11-13 Aug 2021, Montreal, QC, Canada. IEEE . ISBN 978-1-7281-7290-3

https://doi.org/10.1109/icas49788.2021.9551134

AN OPEN SOURCE MOTION PLANNING FRAMEWORK FOR AUTONOMOUS MINIMALLY INVASIVE SURGICAL ROBOTS

Aleks Attanasio, Nils Marahrens*, Bruno Scaglioni and Pietro Valdastri*

STORM Lab, University of Leeds, United Kingdom

ABSTRACT

Planning and execution of autonomous tasks in minimally invasive surgical robotic are significantly more complex with respect to generic manipulators. Narrow abdominal cavities and limited entry points restrain the use of external vision systems and specialized kinematics prevent the straightforward use of standard planning algorithms. In this work, we present a novel implementation of a motion planning framework for minimally invasive surgical robots, composed of two subsystems: An arm-camera registration method only requiring the endoscopic camera and a graspable device, compatible with a 12mm trocar port, and a specialized trajectory planning algorithm, designed to generate smooth, non straight trajectories. The approach is tested on a DaVinci Research Kit obtaining an accuracy of 2.71 ± 0.89 cm in the arm-camera registration and of 1.30 ± 0.39 cm during trajectory execution. The code is organised into STORM Motion Library (STOR-MoLib), an open source library, publicly available for the research community.

***Index Terms*—** Da Vinci Research Kit (dVRK), Trajectory planning, ROS

1. INTRODUCTION

Trajectory planning lies at the heart of most robotic manipulation tasks and is crucial to enable high levels of autonomy [1]. While tasks usually define a set of different poses to be achieved, how the robot should move in between these poses is often left to motion planning algorithms. Common motion planners integrate a plethora of robot models, but surgical minimally invasive surgical systems are not well represented. This may attributed to their complex kinematic structures, often including parallel chains that are not supported by most inverse kinematics solvers and can be numerically challenging. Moreover, the software frameworks used to control surgical robots such as the Collaborative Robot Toolkit (CRTK) [2] and the DaVinci Research Kit (dVRK) [3] only provide

* these two authors contributed equally. Research reported in this article was supported by Intuitive Surgical Inc. under the Technology Research Grants program 2019, by the Royal Society, by the Engineering and Physical Sciences Research Council (EPSRC) under grant number EP/R045291/1, and by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 818045).

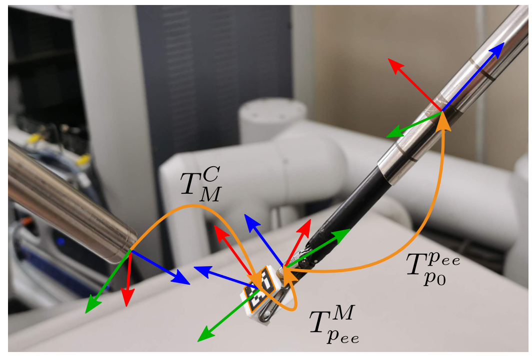

Fig. 1. Transformations of the different frames considered for the registration of the arm to the camera frame.

the ability to reach a final pose with zero velocity, thus not supporting the execution of complex trajectories.

In the particular case of the dVRK, one of the most popular surgical robotics research platform [4], a point to point trajectory in the joint space is generated from the current end effector pose to the goal by means of the Reflexxes RML II [5] library. The resulting trajectory might be optimized in joint space but is generally neither smooth nor optimal in cartesian space. The available literature on motion planning for surgical robots is scarce. In [6] the problem is addressed for the dVRK platform using the MoveIt![7] motion platform. However, the extended abstract is silent on how the problem of parallel kinematics is solved, nor is their code publicly available to the community. Recent works have focused on employing machine learning techniques, such as Pyramid Stereo Matching Network (PSMNet) [8] and reinforcement learning [9]. While these methods show impressive results on specific tasks, they are not generally applicable and easily adaptable. Moreover, they are highly dependent on large amounts of labeled data, obtained via computationally and time-intensive simulations. Another common problem limiting the development of autonomous tasks in MIS robotics platforms is the coregistration between the camera and the robotic arms, since the two subsystems are usually connected to different bases. This issue is commonly solved for generic manipulators using external optical trackers [10]. This approach has been adopted for surgical robots [8, 11] by attaching markers on the tip of the surgical instruments. Although accurate, this method requires the use of an external camera, which is a major limitation in a small and delicate environment such as the abdominal cavity, and is prone to inaccuracies due to the presence blood or debris in the surgical scene. In this work, we: (1) Present a software framework aimed at solving the problem of co-registration for robotic platforms specific to MIS, focused on the ease of use and the feasibility of the application in a clinical environment. (2) Present an approach to the planning and execution of complex trajectories on surgical robots, integrated with ROS and easily adaptable to any platform. (3) Provide public and documented code in a web repository to benefit the surgical robotics research community.

2. CO-REGISTRATION ALGORITHM

This section describes the approach adopted to determine the transformation between the endoscopic camera and the surgical instrument held by the robot. This step is crucial to plan and execute autonomous tasks based on visual servoing in scenarios where the endoscope and the robotic arm do not share the same reference frame. This is the case with robots such as the dVRK, the Raven [12] and modular robots like CMR Versyus or Medtronic's Hugo RAS. The goal is to compute the transformation from the camera frame to the origin of the robotic arm. This can be solved by evaluating a sequence of transformations that start from the pose of the robot end-effector with respect to the camera. In robots equipped with cameras, this can be achieved by adopting a computer vision algorithm to detect one or more visual markers mounted on the end-effector. To this end, we adopt the ArUco markers [13] and mount them on a custom 3D printed pick-up device, designed to be held by standard surgical instruments and be inserted through standard 12mm trocar ports. Once the pick-up device with ArUco marker is grasped by the robotic instrument (Fenestrated Bipolar Forceps), exposed to the camera and recognized by the vision algorithm, the transformation T_C^{p0} between the PSM's base frame T_{p0} and the endoscope's base frame T_C is calculated as follows:

$$T_C^{p0} = T_C^M T_M^{p_{ee}} T_{p_{ee}}^{p_0} \quad (1)$$

where T_C^M is the transformation between camera and a visual marker held by the end-effector, $T_M^{p_{ee}}$ is the transformation between the marker and the end-effector reference frame, and finally $T_{p_{ee}}^{p_0}$ is the pose of the end-effector with respect to the robot base frame. The transformations are shown in Figure 1 on a DaVinci Patient Side Manipulator (PSM), in which the base frame is placed in the remote centre of motion, on the trocar. Assuming that $T_{p_{ee}}^{p_0}$ can be extracted from the robot kinematics and that $T_M^{p_{ee}}$ is known by design of the marker holder, T_C^M can be estimated by using the endoscope in conjunction with software packages like tuw_marker_detection [14]. Finally, the transformation $T_M^{p_{ee}}$ is applied to align the marker frame with the tool tip frame of the robot. To increase robustness of the results, we combine both detected transformations from the left and right endoscopic camera and average the results over 100 frames, each 100ms apart.

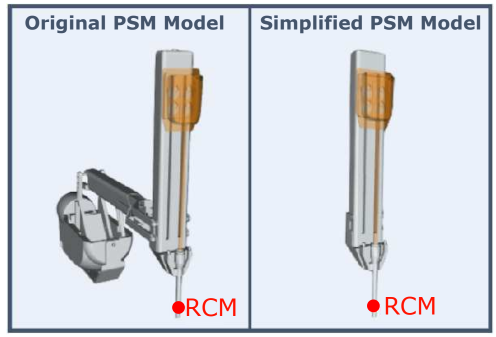

Fig. 2. Original PSM model and the simplified model used in this work. In our simplified model, the base of the robot is omitted, thus removing the parallel kinematic chain and allowing the usage of the MoveIt! package without any loss of generality in the trajectory planning.

3. TRAJECTORY PLANNING

The co-registration algorithm enables to evaluate and control the position of the robot end-effector in the camera workspace. This feature facilitates the definition of points of interest based on computer vision or deep-learning algorithms and to relate them to the position of the end-effector. In many autonomous tasks, it is required to generate a trajectory based on the points identified in this step, and to execute it smoothly. One goal of this paper is to provide a framework for planning and smoothing of the trajectory dedicated of surgical robotic tools. For this purpose, the MoveIt! [15] framework has been used, due to the wide adoption in the research community. MoveIt! is based on the widely used Open Motion Planning Library (OMPL) [16] that includes state-of-the-art algorithms for trajectory planning, manipulation and navigation and is integrated into ROS [17]. In order to plan a trajectory for a specific robot, and therefore produce a feasible trajectory in joint and Cartesian spaces, MoveIt! gathers information about the robot layout from two files: the Unified Robot Description Format file (URDF), used in the ROS ecosystem to define robots kinematics, and the Semantic Robot Description Format file (SRDF), which includes additional information to the URDF such as default robot configuration and collision checking. The trajectory planning is carried out in four steps: (1) The robot URDF and SRDF are loaded onto Moveit!. (2) The robot starting position, way-points and goal of the trajectory are defined. (3) The MoveIt! function computeCartesianPath() is used to evaluate a sequence of points on straight lines from the starting position, through the way-points, to the final goal. (4) The Stochastic Trajectory Optimization for Motion Planning

(STOMP) [18] is used to plan trajectory using the previously generated points as seeds and produce the final trajectory, represented as a set of points in the 3D workspace. STOMP is adopted for its capability of avoiding local minima while allowing a faster convergence to the solution if compared to other planners such as Covariant Hamiltonian Optimization for Motion Planning (CHOMP) [19]. Additionally, given its stochastic nature, the STOMP planner can generate a smooth path even in the presence of obstacles.

A C++ library, STORM Motion Library (STOR-MoLib) is developed to provide the code to the community. The library requires minimal user input and can be utilized by means of the following methods: *compileMotionPlanRequest(waypoints_constraint, trajectory_seed)* and *transformTrajectory(trajectory, base_frame)*. The first populates the MoveIt! motion request constraining the passage through the desired way-points. The trajectory seeds are the output of the computeCartesianPath function included in MoveIt!. The second function transforms the trajectory points from the robot frame to the user-defined base frame, in our case the camera frame. The MoveIt! motion request is then solved by the STOMP Planner which returns a smoothed trajectory.

4. EXPERIMENTAL VALIDATION

The validation of our approach is composed of two steps: the evaluation of the accuracy for the camera-arm registration and the assessment of the trajectories planning and execution. Although the application of the framework could be generalized to any robot, in this work we focus on the dVRK due to its ubiquity and the availability of an open source simulation software, thus circumventing the need for a physical platform, to replicate the results described here. In particular, we adopt a subset of the full DaVinci system composed of one PSM and one stereoscopic endoscope mounted on an independent base. A Linux (Ubuntu 18.04) machine equipped with an Intel Xeon Gold 6140 (2.30GHz) CPU, an Nvidia Quadro 5000 RTX GPU and 128 GB DDR4 2666MHz RAM was adopted to carry out the planning. While the use of a specific robot is transparent to the co-registration algorithm, the trajectory planning depends on the features of each robotic arm through the URDF and SRDF files. Initially, the PSM description files provided with the dVRK library [3] are used. However, the PSM adopts a parallel mechanism to ensure a fixed remote centre of motion. This type of kinematics is not supported in MoveIt!. In order to overcome this issue, a modified version of the PSM excluding the parallel link is developed (Figure 2). Despite the different physical layout, the kinematics of the robot is correctly reproduced by maintaining the Remote Centre of Mass fixed and eliminating the parallel link and the preceding links in the kinematic chain.

To quantify the registration error, a 3D-printed calibration body attachable to the endoscope's tip was designed. The calibration body contains nine landmark points (p_C^1 - p_C^9) with

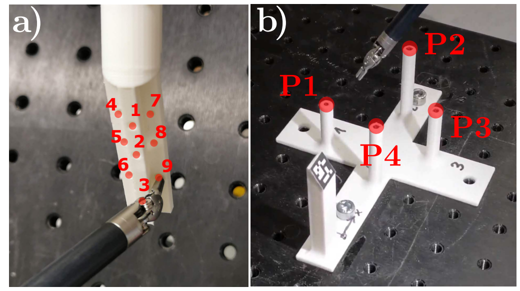

Fig. 3. 3D-printed rigid body used for the validation of the marker-based co-registration (a). 3D-printed rigid body used to validate the precision during the trajectory execution (b). A marker has been attached to the body to allow the registration of the points via the camera.

known distance with respect to the camera's base frame T^C (Figure 3a). By touching the landmarks with the tip of the surgical instrument, we acquired the location of these positions in the PSM's base frame T_{p0}.By performing several registrations ($n = 5$) and averaging the position of each of the nine points over all runs we obtain p_{p0}^1 - p_{p0}^9. With a confidence interval of 0.0734 mm ($c = 0.95$), we assume the robot's positional accuracy to be fairly high and consistent compared to the camera. In order to assess the accuracy of the co-registration approach on our surgical setup, five registrations are performed using the ArUco marker with differing tool positions and thus different placements of the marker with respect to the camera. With the acquired transformations $T_C^{p_0}$ from the visual marker registrations, we transform the points p_C^1 - p_C^9 on the calibration body from the camera's base frame T_C to the PSM's base frame T_{p0} and calculate the euclidean distance to the respective points obtained via landmark registration. Our results indicate a mean positional error of 2.71 ± 0.89 cm ($c = 0.95$) over all registered points and registration runs compared to the position obtained via the camera calibration body. We believe the main source of inaccuracy to be the camera distortion. Despite a thorough calibration, the fish-eye lenses of the endoscope produce a significant distortion that negatively affects the accuracy of the marker detection, particular when the marker is not place directly at the center of the image. Additionally, the small distance between the two cameras limits the usage of further information from the 3D scene via stereo matching or similar techniques.

In order to evaluate the accuracy of the trajectory planning and execution, a 3D-printed reference body with four vertical pegs was designed. The tip of each peg represents either a way-point or the goal of the trajectory (Figure 3b). The reference body also integrates an ArUco marker, added to obtain a transformation from its local reference frame to the camera frame T_C^{RB}. The coordinates of each way-point are transformed into the PSM's base frame T_{p0} by combining the two previously obtained transformations ($T_{p0}^{RB} = T_{p0}^{C} T_{C}^{RB}$). The planner evaluates a trajectory starting from the current position of the instrument, passing along the way-points and ending in the goal position. Two different trajectory scenarios have been considered with three and four way-points, respectively. Each trajectory has been repeated 8 times and, for each repetition, the surgical instrument was initially manually placed in a varying position around the starting point. Although the planner can consider variable instrument orientations, we maintained a constant, randomly selected, orientation during the whole trajectory.

The planner's output consists of a trajectory defined as an array of joint values, one set for every trajectory point. These are converted to the Cartesian space by means of forward kinematics and eventually organised in a vector of poses sent to the dVRK software. The dVRK only allows a point to point trajectory, constraining the initial and goal velocity to zero. To perform a smooth trajectory, we published the new poses at a rate of 20Hz, sending a new command before the robot had reached the previous goal and thus avoiding the condition of zero velocity. Before executing each trajectory, the position of each way-point with respect to the robot's base frame T_{p0} was collected by manually positioning the surgical instrument (large needle driver) onto a landmark on each peg's tip and recording its position. Figure 4 shows the 8 trajectories for both the three and four point case. The start and end point of the trajectory are represented in blue and green, respectively. The way-points are represented in red. It must be pointed out that the sequence of the way-points is different for the two trajectories. The sequence chosen in the four point case is aimed at demonstrating the ability of the planner to find a solution in the even in the case of more involved trajectories, containing a indirect path with back and forth motion. The evaluation of the trajectories is carried out by considering the minimum distance between the path executed by the robot and each way-point measured before the trajectory execution via the robots tool tip. With this reference, the average error amounts to 1.09 ± 0.59 cm ($c = 0.95$) for the three point and 1.30 ± 0.39 cm ($c = 0.95$) in the four point case.

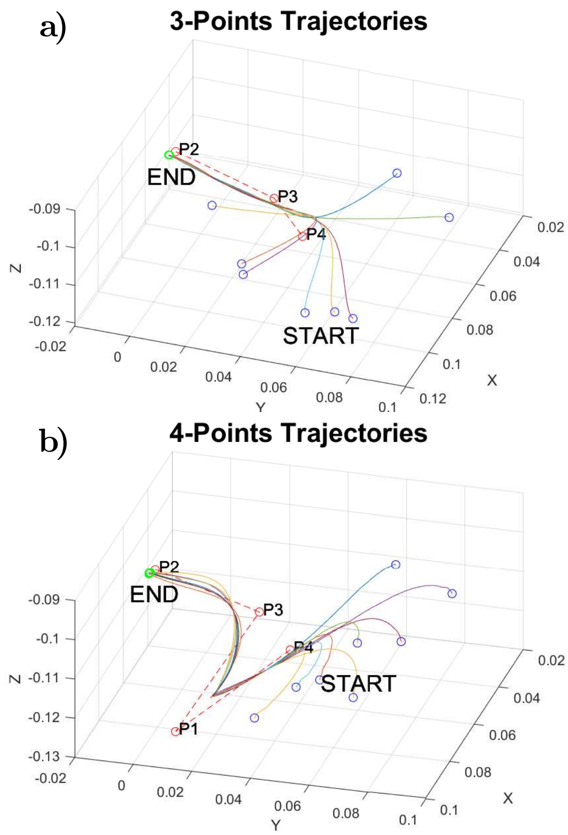

Fig. 4. Repetitions for the trajectory planning and execution for three point (a) and four point (b) case. The initial point is shown in blue, the goal point in green and the way-points in red. The red dashed lines depict the seeds used by the STOMP planner.

5. CONCLUSIONS

In this paper, we presented a comprehensive library to manage the trajectory planning of surgical robots with the specific aim of developing a method that does not require dedicated hardware such as optical trackers or external cameras, thus applicable in the context of minimally invasive surgery. Initially, we presented a method for arm-to-camera registration based on the ArUco markers. We showed the method to be a feasible approach in robotic systems where the arms and the camera do not share the same kinematic base. Subsequently, we demonstrated an approach for planning and executing trajectories based on Moveit! and integrated with ROS. For our evaluation, we applied our framework and approach to the dVRK platform. The registration makes it possible to plan trajectories with respect to the camera frame, thus supporting the execution of vision-based autonomous surgical gestures. Moreover, the registration algorithm can be useful in setups, such as the dVRK, in which teleoperation is challenging due to the lack of a simple built-in co-registration protocol. Although the dVRK Setup Joints controller will be available in the future, not all the research groups have access to the full platform. We believe that this library could significantly benefit the research community. STOR-MoLib code is open source and publicly available [1].

Further development of this library, currently under investigation, include the implementation of a collision avoidance algorithm, useful in collaboration scenarios in which a human operator is controlling one arm, while the other arm is autonomously operated. Other improvements, particularly regarding the registration accuracy, might be obtained by further investigations on the distortion of the cameras' lenses.

[1] https://github.com/Stormlabuk/dvrk_stormolib

6. REFERENCES

[1] Aleks Attanasio, Bruno Scaglioni, Elena De Momi, Paolo Fiorini, and Pietro Valdastri, "Autonomy in surgical robotics," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 4, 2020.

[2] Anton Deguet Peter Kazanzides, "CRTK - Collaborative Toolkit," 2021, [Online; accessed 21-April-2021].

[3] Peter Kazanzides, Zihan Chen, Anton Deguet, Gregory S. Fischer, Russell H. Taylor, and Simon P. Dimaio, "An open-source research kit for the da Vinci® Surgical System," in *IEEE International Conference on Robotics and Automation (ICRA)*. 2014, pp. 6434–6439, IEEE.

[4] Claudia D'Ettorre, Andrea Mariani, Agostino Stilli, Ferdinando Rodriguez y Baena, Pietro Valdastri, Anton Deguet, Peter Kazanzides, Russell H. Taylor, Gregory S. Fischer, Simon P. DiMaio, Arianna Menciassi, and Danail Stoyanov, "Accelerating Surgical Robotics Research: Reviewing 10 Years of Research with the dVRK," apr 2021.

[5] Torsten Kroeger, "Opening the door to new sensor-based robot applications - The reflexxes motion libraries," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 6–9, 2011.

[6] Zhixian Zhang, Adnan Munawar, and Gregory S Fischer, "Implementation of a motion planning framework for the davinci surgical system research kit," in *The Hamlyn Symposium on Medical Robotics*, 2014, p. 43.

[7] Sachin Chitta, Ioan Sucan, and Steve Cousins, "Moveit![ros topics]," *IEEE Robotics & Automation Magazine*, vol. 19, no. 1, pp. 18–19, 2012.

[8] Florian Richter, Shihao Shen, Fei Liu, Jingbin Huang, Emily K Funk, Ryan K Orosco, and Michael C Yip, "Autonomous robotic suction to clear the surgical field for hemostasis using image-based blood flow detection," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1383–1390, 2021.

[9] Florian Richter, Ryan K Orosco, and Michael C Yip, "Open-sourced reinforcement learning environments for surgical robotics," *arXiv preprint arXiv:1903.02090*, 2019.

[10] Christoff M. Heunis, Beatriz Farola Barata, Guilherme Phillips Furtado, and Sarthak Misra, "Collaborative Surgical Robots: Optical Tracking During Endovascular Operations," *IEEE Robotics & Automation Magazine*, vol. 27, no. 3, pp. 29–44, sep 2020.

[11] Caitlin Schneider, Christopher Nguan, Robert Rohling, and Septimiu Salcudean, "Tracked "pick-Up" ultrasound for robot-assisted minimally invasive surgery," *IEEE Transactions on Biomedical Engineering*, vol. 63, no. 2, pp. 260–268, 2016.

[12] Blake Hannaford, Jacob Rosen, Diana W Friedman, Hawkeye King, Phillip Roan, Lei Cheng, Daniel Glozman, Ji Ma, Sina Nia Kosari, and Lee White, "Raven-ii: an open platform for surgical robotics research," *IEEE Transactions on Biomedical Engineering*, vol. 60, no. 4, pp. 954–959, 2012.

[13] S. Garrido-Jurado, R. Muñoz-Salinas, F.J. Madrid-Cuevas, and M.J. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognition*, vol. 47, no. 6, pp. 2280–2292, jun 2014.

[14] Markus Bader Lukas Pfeifhofer, "ROS package tuw_aruco," 2019, [Online; accessed 14-April-2021].

[15] Sachin Chitta, Ioan Sucan, and Steve Cousins, "MoveIt!," *IEEE Robotics and Automation Magazine*, vol. 19, no. 1, pp. 18–19, 2012.

[16] Ioan A Sucan, Mark Moll, and Lydia E Kavraki, "The open motion planning library," *IEEE Robotics & Automation Magazine*, vol. 19, no. 4, pp. 72–82, 2012.

[17] Hideki Yoshida, Hiroshi Fujimoto, Daisuke Kawano, Yuichi Goto, Misaki Tsuchimoto, and Koji Sato, "Range extension autonomous driving for electric vehicles based on optimal velocity trajectory and driving braking force distribution considering road gradient information," *IECON 2015 - 41st Annual Conference of the IEEE Industrial Electronics Society*, pp. 4754–4759, 2015.

[18] Mrinal Kalakrishnan, Sachin Chitta, Evangelos Theodorou, Peter Pastor, and Stefan Schaal, "STOMP: Stochastic trajectory optimization for motion planning," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 4569–4574, 2011.

[19] Nathan Ratliff, Matt Zucker, J. Andrew Bagnell, and Siddhartha Srinivasa, "CHOMP: Gradient optimization techniques for efficient motion planning," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2009, pp. 489–494.