Design and Validation of a Multi-Arm Robot Platform for Scientific Exploration

Murilo Marques Marinho, Member, IEEE, Juan José Quiroz-Omaña, and Kanako Harada, Member, IEEE

Abstract—There are a large number of robotic platforms with two or more arms targeting surgical applications. Despite that, very few groups have employed such platforms for scientific exploration. Possible applications of a multi-arm platform in scientific exploration involve the study of the mechanisms of intractable diseases by using organoids (i.e., miniature human organs). The study of organoids requires the preparation of a cranial window which is done by carefully removing an 8 mm patch of the mouse skull. In this work, we present the first prototype of the AI robot science platform for scientific experimentation, its digital twins, and perform validation experiments under teleoperation. The experiments showcase the dexterity of the platform by performing peg transfer, gauze cutting, mock experiments using eggs, and the world’s first teleoperated drilling for a cranial window. The digital twins and related control software are freely available for noncommercial use at https://AISciencePlatform.github.io.

I. INTRODUCTION

The collaboration among humans and robots is a reality in many fields of industry, science, and medicine. The robotics community has made much progress in the direction of making that collaboration safe and meaningful. We often rely on robots for tasks that might be over the limit of human accuracy or endurance, and humans for decision making and higher-level recognition.

This development has been further empowered by the recent re-rise of data-driven deep learning [1] (usually referred to as artificial intelligence—AI), which has been undoubtedly influential in robotics research since at least 2015 [2]. The success of these technologies has sparked the interest of society in the impact of future AI-embedded robots (henceforth AI-robots), further rekindled by the COVID-19 pandemic [3]. This includes ethical concerns regarding the accountability of these intelligent systems, clear definitions of levels of autonomy [4], and so on.

In line with the views of other groups regarding the next frontiers in robotics [2], [3] and the talks of influential roboticists in the ICRA’22 Workshop “Toward AI-embedded robots in 2050”, we see the future of humans and AI-robots not only for industry and medicine. Adding to those, we seem us as part of a co-evolution scheme, to expand science frontiers1. Our long-term goal, part of the ambitious Moonshot Research & Development Program of the Cabinet Office of Japan2, is, by 2050 and beyond, have AI-robots that autonomously learn, adapt to their environment, evolve in intelligence and act alongside human beings.

To achieve that long-term goal, our project is divided into many fronts. In one of those, discussed herein, we are interested in developing an experimental platform that will serve to conduct robotic experiments and scientific discovery. Indeed, one can think of the AI as the brain, whereas the robotic platform will be the body.

In order to have a concrete aim, our initial prototype has been inspired by scientific discovery related to intravital imaging [5]. Intravital imaging refers to images taken while the organism is alive, through a type of window. One of the applications of this imaging technique is to study the growth of human cells [6].

In this work we present our first iteration of the AI Science Platform (AISP) we envision. In this prototype, this platform has several novel elements regarding hardware and software. We also developed a matching digital twin which we make openly available. As a proof-of-concept we perform experiments under teleoperation, namely peg transfer, gauze cutting, mock experiments using eggs, and the world’s first teleoperated drilling for a cranial window.

II. RELATED WORKS

In recent years, teleoperated robotic platforms have gained the scientific community’s attention thanks to the emergence of new applications in several fields, especially in surgical robotics [7], where surgeons exploit the robot’s dexterity to perform challenging medical procedures.

Despite the high level of accuracy of the robotic platforms, the quality of teleoperated robot-assisted surgery (RAS) often relies on the surgeon’s skills and experience, which are subject to human imprecision [8].

Aiming for greater robot autonomy, several platforms have been developed for surgical robotics research, for instance the dVRK [9] and the RAVEN [10]. The former is based on the first-generation da Vinci robot hardware and the latter is based on custom open hardware. Both systems are based on cable-driven mechanisms, and, different from serial manipulators, they often require additional strategies to compensate for the inaccuracies related to slack and stretch of the cables [11], [12]. Because of that, several platforms based on industrial serial manipulators have been proposed [13]–[15].

1 https://sites.google.com/g.ecc.u-tokyo.ac.jp/moonshot-ai-science-robot/
2 https://www8.cao.go.jp/cstp/english/moonshot/top.html
Schwaner et al. [16] presented MOPS, an open platform for surgical robotics research based on serial manipulators. MOPS has high modularity, uses commercially available robot arms, and allows scalability with custom tools. In addition to all these features that are also present in AISP, our platform includes a novel hardware design, which is oriented to dexterous cooperative task manipulations, and targets general experiments for scientific exploration.

III. ROBOTIC SYSTEM OVERVIEW

The proposed robotic system is shown in Fig. 1. The whole system is closed within a frame. At the base, there is a circular rail to which each branch of the four branches is attached. Each branch has a servomotor (283867, MAXON, Switzerland) to independently rotate about the rail. There is no mechanical limitation for the motion of each branch about the rail.\(^3\) Serially, each branch has a linear actuator that moves back-and-forth in the direction of the center of the rail (RCP6, AIA, Japan). Two branches are composed by a collaborative-type robot (CVR038, Densowave, Japan) attached in series. The remaining two branches are composed by the serial connection of an industrial-type robot (VS050, Densowave, Japan).

Aiming at a large set of possible applications including the cranial window, we have four customized end-effectors. One is a customized micro-drill based on a commercial micro-drill (MD1200, BrainTree Scientific, USA). Another is a customized grasper for handling regular-sized cotton swabs. The last two are customized actuators based on a rotary gripper module (EHMD-40-RE-GE, FESTO, Germany) to operate tweezers (15-416, BRC, Japan) and scissors (S7, Tsubasa Kougyo, Japan) that commonly used in cranial window procedures.

The vision of the workspace is provided through six 4K cameras (STC-HD853HDMI, Omron-Sentech, Japan). Four of these cameras are distributed around the workspace directed at the center to provide an overall understanding of the state of the robotic system using wide field-of-view lenses (VS-LDA4, VS Technology, Japan). Another camera is used to provide vision for the cranial window task using another small field-of-view lenses (VS-LDA75, VS Technology, Japan). The last camera uses lenses to provide a more balanced field-of-view, for tasks such as those included in the fundamentals of laparoscopic surgery\(^4\) (FLS).

IV. SYSTEM INTEGRATION & SOFTWARE

Our system has been designed to have flexibility on the control modes. This means that any of the four robots can be controlled through teleoperation or autonomously. Each robot can be independently controlled by its respective operator side, which can be anywhere in the world with a properly configured internet connection. In this light, our system can be divided into \textit{operator sides} (OS) and a \textit{follower side} (FS). One example configuration is shown in Fig. 2.

\(^3\)In effect, the rotation of each branch about the central reference frame is limited by the length of the cables connecting each branch.

\(^4\)https://www.flprogram.org

| Controller | Qt. | Protocol        | Software package         |
|------------|-----|-----------------|--------------------------|
| ESCON 50/5 | 4   | Analog voltage  | sas_robot_driver_escon   |
| CMMO-ST   | 4   | ModBus-TC      | sas_robot_driver_festo   |
| PCON-CB   | 4   | Modbus-Serial   | sas_robot_driver_aia     |
| RC8       | 4   | bCAP-UDP       | sas_robot_driver_denso   |

This integration has been achieved through the development of a large number of custom ROS Noetic packages\(^5\), all free for noncommercial use\(^6\). The 	exttt{sas_robot_driver_denso} and 	exttt{sas_robot_driver_escon} are highly upgraded versions of our earlier software development [15]. The software packages 	exttt{sas_robot_driver_festo} and 	exttt{sas_robot_driver_aia} were newly developed using pymodbus\(^7\).

Because of the continuous changes in ROS it is important to isolate all ROS behavior into specific modules to facilitate future upgrades. Each device can be used directly through C++

\(^5\)https://github.com/SmartArmStack/smart_arm_master_windows/releases/latest

\(^6\)https://www.3dsystems.com/haptics

\(^7\)https://www.medicareoid.com/en/product/hinotori/

\(^8\)A ROS2 version is expected in the future.

\(^9\)https://github.com/SmartArmStack

\(^10\)https://github.com/riptideio/pymodbus
or Python without ROS, through their sas_robot_driver interfaces. Then, using inheritance and subclass polymorphism, each device is individually managed through ROS by an instance of sas_robot_driver_ros. In general each robotic device will have different configuration requirements and those are handled in a modular way through configuration files.

2) Composition of robots: The composition of several devices into a single serial robotic system is enabled by an instance of sas_robot_driver_ros_composer. Using a configuration file, we specify which sas_robot_driver_ros are to be composed and in what order. In addition, we can add CoppeliaSim information for each composed robot, such as CoppeliaSim IP address and the joint names on the simulator. The abstraction of any number of systems into a single serial robot facilitates and modularizes the higher-level control modules which is impervious to changes in the lower-level devices. In this implementation, we have a total of four branches, each abstracted by a sas_robot_driver_ros_composer.

3) Centralized control: The centralized control module is the main contribution in terms of software. It is the implementation of a centralized kinematic control strategy based on quadratic optimization and inequality constraints [17], [18]. The centralized control can be configured to simultaneously handle any number of robot branches, in this case four.

\[
\mathbf{u} \in \arg\min_{\mathbf{q}} \sum_{i=1}^{4} F_i(q_i, x_{d,i})
\]

subject to \[
\begin{align*}
\mathbf{W}_s \mathbf{q} &\leq \mathbf{w}_s \\
\mathbf{W}_p \mathbf{q} &\leq \mathbf{w}_p
\end{align*}
\]

where \( F_i(q_i, x_{d,i}) \) is a convex objective function related to the task of each robotic branch [17], \( q_i \) are the joint values of the \( i \)th robot, \( q \) is the vector containing all \( q_i \) in order, \( x_{d,i} \) is the task-space target for the \( i \)th robot, and \( \mathbf{u} \) is the vector of joint values output by the solver that will be send to the robotic system.

With respect to the constraints, the first set of constraints contains the block-diagonal matrix \( \mathbf{W}_s \) that encodes constraints related to a single robot branch. Those can be, for instance, joint limits, joint velocity limits, and collision avoidance with static objects in the workspace using VFIs. Those constraints can be written in general terms as

\[
\mathbf{W}_s = \begin{bmatrix} W_1 & 0 & 0 & 0 \\ 0 & W_2 & 0 & 0 \\ 0 & 0 & W_3 & 0 \\ 0 & 0 & 0 & W_4 \end{bmatrix},
\]

\[
\mathbf{w}_s = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \end{bmatrix}.
\]

Lastly, the second set of constraints is related to pairwise constraints to prevent collisions between robotic branches. Those can be written as follows when using dynamic VFIs
Fig. 2. An example of the system architecture when two robots are teleoperated and two robots are autonomously controlled. The local devices are controlled from the centralized control computer. Each robotic branch is encapsulated in a `sas_robot_driver_ros_composer`. There are four branches, but only two are shown in the figure for simplicity. Those four robotic branches are controlled in joint-space by the `aisp_inverse_kinematics` node which takes into account joint limits, environmental constraints, and collision avoidance among robots. The `aisp_inverse_kinematics` node receives desired pose signals for each branch individually. The desired pose signals can be sent through a master device, a custom program, or a ROS-based program.

\[ W_p = \begin{bmatrix} W_{1,2} & W_{2,1} & 0 & 0 \\ W_{1,3} & 0 & W_{3,1} & 0 \\ W_{1,4} & 0 & 0 & W_{4,1} \\ 0 & W_{2,3} & W_{3,2} & 0 \\ 0 & W_{2,4} & 0 & W_{4,2} \\ 0 & 0 & W_{3,4} & W_{4,3} \end{bmatrix}, \]

\[ w_p = \begin{bmatrix} w_{1,2} \\ w_{1,3} \\ w_{1,4} \\ w_{2,3} \\ w_{2,4} \\ w_{3,4} \end{bmatrix}. \]

The inequality constraints are used to provide (self) collision-avoidance, that is, a sense of self-awareness to the robotic system. That is very important in such a complex system, because a naive control strategy certainly would cause the robot to break. The self-awareness is provided through our VFI methodology. Each VFI is based on the signed distance between geometric primitives. One of the biggest challenges we had so far with our strategy was related to the cumbersome process of adding several primitive pairs.

In this work, we partially solve this issue by providing a way to add each VFI through a configuration file (see Fig. 3). The system designer must input the necessary information about each VFI, including the type of each primitive, which robotic branch it is related to, and where the primitive for the VFI calculation is attached. The user must also place each primitive on CoppeliaSim. The strategy proposed in this work will read through the configuration file and retrieve the necessary information between primitives during start-up and build the constraint-related matrices \( W_s, W_p \) and vectors \( w_s, w_p \) automatically for the user.

V. DIGITAL TWINS

The simulation of our robotic platform is addressed using two different environments, namely CoppeliaSim, and Isaac Sim. The former one is mainly used to define geometric primitives, useful to prevent (self)-collisions by means of the VFIs framework. The latter implements a digital twin, and it is used to train operators before using the real platform. Both environments run on the Windows computer, which is equipped with an Intel i9-12900K with 64GB RAM and a GPU Nvidia RTX A6000.

Figure 4 summarizes the proposed strategy. The haptic interface and both simulators are managed by the Windows system. The kinematic controllers, which are running on Ubuntu and ROS, receive the teleoperation commands of the haptic interface and set the joint positions of the CoppeliaSim simulation. Finally, Isaac Sim receives the joint position commands from CoppeliaSim by means of the DQ Robotics library and the Isaac Sim Python API.

A. CoppeliaSim

We define geometric primitives on CoppeliaSim, as shown in Fig. 5. These primitives, in addition to a configuration 11https://www.coppeliarobotics.com/ 12https://developer.nvidia.com/isaac-sim
Fig. 3. Example of two VFIs described using a yaml configuration file. The first object describes a forbidden zone between a point in the forth robotic branch, related to the second joint, with respect to a line in the workspace. The second object describes a forbidden zone between two robots. The first entity is a sphere related to the second robotic branch’s first joint and second entity is related to the first robot branch’s first joint. Using a total of XXX VFIs, we can guarantee that the system is highly reactive and has collision-avoidance guarantees.

Fig. 4. Two computers manage the overall control strategy of the robotic platform. The Windows system addresses GPU-intensive tasks, such as dynamics simulations on CoppeliaSim, and renders tasks on Isaac Sim. The GNU/Linux system runs ROS and the kinematic controllers.

Fig. 5. Snapshot of the CoppeliaSim scene. The blue and red spheres are geometric primitives defined by the user. The kinematic controllers use these primitives to impose VFIs that prevent (self)-collisions.

VI. EXPERIMENTAL SHOWCASE

A. Digital Twin: Isaac Sim

To validate the proposed approach, we performed two challenging task simulations. The first one is the peg transfer task, and the second one is cranial microsurgery on a mouse. Both tasks are performed via teleoperation. The main goal is to develop a training platform to improve the operator’s skills before using the real robotic system.

Peg Transfer Task: Since the grasping of general objects is a challenging task, which usually relies on specialized simulation environments [19], we implemented a fake-grasping strategy useful to practice peg transfer experiments. We used the kinematic branch equipped with the forceps. The idea is to constrain the relative pose between the forceps and the block when they are in contact below a minimum distance. Once the teleoperator opens the forceps, the block is released.

The block is composed of two types of bodies, namely rigid and deformable ones, as shown in Fig. 7. The rigid bodies are static, and therefore they are not affected by external forces. Furthermore, the rigid bodies interact exclusively with the forceps of the end-effector, which are equipped with a contact sensor. On the other hand, the dynamic deformable bodies are attached to the rigid ones. This enables interactions between the block and the pegs. Each peg is linked to a fixed base using a spherical joint set with damping and stiffness, as shown in Fig. 8. This approach allows small movements of the block without affecting the rigid bodies.

13The collision meshes of the rigid objects that compose the block do not interact with other objects of the scene, including the pegs. This is done by defining collision groups.

14The contact sensor does not work with deformable bodies in the current version of Isaac Sim.

B. Isaac Sim

Isaac Sim allows different development workflows such as UI, Python API, and ROS. We use the Python API standalone workflow since enables more features than the others and avoids the ROS dependency. Fig. 6 shows the render of the scene using the path tracing algorithm.
Fig. 6. Path tracing render on Isaac Sim. On the left, the safety cage of the platform is displayed. On the right, the cage was removed for the sake of the visibility of the inner components.

Fig. 7. Block meshes used in the peg transfer task. From left to right: Three rigid static bodies compose the shape of the block; three deformable dynamic bodies are placed overlapping the rigid bodies; the attachments between the rigid bodies and the deformable bodies; both rigid and deformable bodies compose the model of the block.

Fig. 8. Interaction between the block and the peg. From left to right: A spherical joint links the peg with a static base; different behaviors of the block and the peg when subject to an external force. The blue arrow represents the external force vector applied.

peg when subjected to external forces, and prevents numerical instabilities when the robot forceps touch the peg.

Cranial Microsurgery: We implemented a second simulation aiming to practice cranial microsurgery experiments using the kinematic branch equipped with the drill. Instead of drilling specific meshes, which is not a trivial solution for Isaac Sim, we adopted a draw-based approach. The goal is to draw small spheres on the target object when the contact force applied by the end-effector exceeds a threshold. By doing so, we include different behaviors for different force values. This feature allows setting different colors for the drawn spheres or different textures for the drilled object, as shown in Fig. 9.

Results and Discussion: Fig. 10 shows the snapshots of the peg transfer simulation task. The block is transferred from its initial peg to a different one successfully. However, the lack of 3D vision makes it difficult for the operator to locate the end-effector in the workspace. To circumvent this one could add more cameras in different positions in the scene. Nevertheless, each camera increases the GPU consumption considerably. Our scene, with one camera, runs at 30 FPS using a 1920x1080 render resolution. When two cameras are used, the overall performance drops to about 18 FPS. Another solution could be the use of a VR headset, which is currently supported by Isaac Sim. However, that is out of the scope of this work.

Figure 11 shows the snapshots of the cranial microsurgery simulation task. The goal is to draw a small spherical trajectory on the skull surface of a mouse. This trajectory represents a small piece of skull tissue that is to be removed. In this case, the robot is not teleoperated. Instead, we define a position task to follow a closed trajectory. The height of the end-effector is independently controlled to keep a constant force applied on the skull. The task is performed successfully, as shown in Fig. 11. However, our current simulation does not take into account the deformable behavior of the mouse skull, which is a challenging feature in the real experiments.
B. Physical Platform

Fig. 9. Egg-drilling task. From left to right: The drill is near to the egg, but there is no contact force; the drill is applying on the egg a contact force, and small black spheres are drawed on the egg surface. The applied force is above a threshold and the egg texture is modified to imitate a broken egg.

Fig. 10. Snapshots of the peg transfer task simulation. The goal is to transfer the block from its initial peg to a different one while avoiding collisions between the robot’s forceps and the pegs.

Fig. 11. Snapshots of the cranial microsurgery simulation. The desired trajectory is denoted by the green dashed line. On top, the same snapshot is shown using an isometric view.

Fig. 12. Peg transfer on the real system operated in teleoperation mode. The peg transfer experiment is ubiquitous in surgical robotics automation and a common benchmark for human and robot dexterity. This feasibility experiment serves to show that our system is capable of both incredibly fine tasks and dexterous tasks such as peg transfer.

1) Peg transfer: The peg transfer task, part of the fundamentals of laparoscopic surgery (FLS) certification, is ubiquitous in surgical robotics automation and a common benchmark for human and robot dexterity. In this showcase experiment, our intention is to show the feasibility of using the robotic arm with tweezers to perform the peg transfer. The procedure was performed through teleoperation using the large field-of-view lenses on top and left, front, and right cameras.

With only the top view it was extremely challenging to properly perform the task because of the difficulties in perceiving depth from a single camera. With the many simultaneous views, the user can naturally compensate for that difficulty. In this context, the user was able to perform the task without difficulties as shown in Fig. 12. As a reference, the task was done within 3 minutes.

Depending on usage, the system could also be retrofitted with two tweezer actuation unit, allowing for two-arm peg transfer execution.

2) Two-person teleoperated cranial window drilling: In this experiment our objective was to use the robotic platform for drilling a 8 mm patch on the skull of an euthanized mouse without damaging the tissue below. The mouse used in this experiment was otherwise going to be discarded and was provided by a neighboring research laboratory.

The preparation of the mouse for the experiments was done manually by removing the skin on top of the cranium and fixing the mouse under camera vision. One operator was tasked with handling the microdrill and the cotton swab. When the mouse is alive, the cotton swab and other tools are used to control and stop bleeding. In our case, the cotton swab was used to keep the cranium wet using tap water inside a small plastic container. The user operated the robot to wet the cotton swab and then applied it to the cranial surface. After that, the user performed the drilling of the skull. Those steps were repeated for about 5 minutes. After that, the second user, seated in another computer and having access to their
own master interface, operated the tweezers. The operator could successfully remove the cranial patch. Snapshots of this procedure are shown in Fig. 13.

Conclusions

In this work, we have introduced a multi-arm robot platform for scientific exploration. We described its design, based on a centralized rail system and customized end effectors. After showing the scalable system integration, we showed two digital twins. One digital twin is based on CoppeliaSim and ready to run on most computers. Another is based on IsaacSim and, while more computationally demanding, provides more realistic graphics and physical accuracy. To evaluate this platform, we have shown experiments involving the real platform and the digital twins. We have performed peg transfer, mock experiments using egg models, and the drilling toward cranial window generation using three robot arms controlled by two operators. The digital twins and related control software are available free for noncommercial use and can be used by any group interested in validating their algorithms in a state-of-the-art multiarm platform. These initial results show the dexterity and usability of such a complex system in many different settings. There is ongoing work in using this platform for artificial intelligence research toward scientific exploration. The teleoperation of the system is paramount in this case for the understanding of the tasks to be performed and for the generation of data to understand the features that are relevant to the task. We expect this platform to be the first step towards collaborative AI robots that will autonomously perform scientific experiments that otherwise could not be done by human scientists alone.

References

[1] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” Nature, vol. 521, no. 7553, pp. 436–444, may 2015.
[2] N. Sunderhauf, O. Brock, W. Scheirer, R. Hadsell, D. Fox, J. Leitner, B. Upcroft, P. Abbeel, W. Burgard, M. Milford, and P. Corke, “The limits and potentials of deep learning for robotics,” The International Journal of Robotics Research, vol. 37, no. 4-5, pp. 405–420, apr 2018.
[3] A. D. Lallo, R. Murphy, A. Krieger, J. Zhu, R. H. Taylor, and H. Su, “Medical robots for infectious diseases: Lessons and challenges from the COVID-19 pandemic,” IEEE Robotics and Automation Magazine, vol. 28, no. 1, pp. 18–27, mar 2021.
[4] G.-Z. Yang, J. Cambias, K. Cleary, E. Daimler, J. Drake, P. E. Dupont, N. Hata, P. Kazanzides, S. Martel, R. V. Patel, V. J. Santos, and R. H. Taylor, “Medical robotics-regulatory, ethical, and legal considerations for increasing levels of autonomy,” Science Robotics, vol. 2, no. 4, mar 2017.
[5] M. J. Pittet and R. Weissleder, “Intravital imaging,” Cell, vol. 147, no. 5, pp. 983–991, nov 2011.
[6] T. Takebe, K. Sekine, M. Enomura, H. Koike, M. Kimura, T. Ogarei, R.-R. Zhang, Y. Leno, Y.-W. Zheng, N. Koike, S. Anyama, Y. Adachi, and H. Taniguchi, “Vascularized and functional human liver from an iPSC-derived organ bud transplant,” Nature, vol. 499, no. 7459, pp. 481–484, jul 2013.
[7] P. E. Dupont, B. J. Nelson, M. Goldfarb, B. Hannaford, A. Menciassi, M. K. O’Malley, N. Smaia, P. Valdastri, and G.-Z. Yang, “A decade retrospective of medical robotics research from 2010 to 2020,” Science Robotics, vol. 6, no. 60, Nov. 2021. [Online]. Available: https://www.science.org/doi/10.1126/scirobotics.abi8017
[8] H. Saeidi, J. D. Opfermann, M. Kam, S. Wei, S. Leonard, M. H. Hsieh, J. U. Kang, and A. Krieger, “Autonomous robotic laparoscopic surgery for intestinal anastomosis,” Science Robotics, vol. 7, no. 62, Jan. 2022. [Online]. Available: https://www.science.org/doi/10.1126/scirobots.abj2908
[9] P. Kazanzides, Z. Chen, A. Dugue, G. S. Fischer, R. H. Taylor, and S. P. DiMaio, “An open-source research kit for the da Vinci® Surgical System,” in 2014 IEEE International Conference on Robotics and Automation (ICRA), May 2014, pp. 6434–6439, ISSN: 1050-4729.
[10] B. Hannaford, J. Rosen, D. W. Friedman, H. King, P. Roan, Lei Cheng, D. Glozman, Ji Ma, S. N. Kosari, and L. White, “Raven-II: An Open Platform for Surgical Robotics Research,” IEEE Transactions on Biomedical Engineering, vol. 60, no. 4, pp. 954–959, Apr. 2013. [Online]. Available: http://ieeexplore.ieee.org/document/6363574/
[11] H. Peng, X. Yang, Y.-H. Su, and B. Hannaford, “Real-time Data Driven Precision Estimator for RAVEN-II Surgical Robot End Effector Position,” in 2020 IEEE International Conference on Robotics and Automation (ICRA). Paris, France: IEEE, May 2020, pp. 350–356. [Online]. Available: https://ieeexplore.ieee.org/document/9196915/
[12] M. Hwang, B. Thananjeyan, S. Paradis, D. Seita, J. Ichinowski, D. Fer, T. Low, and K. Goldberg, “Efficiently Calibrating Cable-Driven Surgical Robots With RGBD Fiducial Sensing and Recurrent Neural Networks,” IEEE Robotics and Automation Letters, vol. 5,
[13] H. Mayer, I. Nagy, A. Knoll, E. Schirmbeck, and R. Bauernschmitt, “The Endo[PA]R system for minimally invasive robotic surgery,” in 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566), vol. 4. Sendai, Japan: IEEE, 2004, pp. 3637–3642. [Online]. Available: http://ieeexplore.ieee.org/document/1389980/

[14] U. Hagn, M. Nickl, S. Jorg, G. Passig, T. Bahls, A. Nothhelfer, F. Hacker, L. Le Tien, A. Albu Schaffer, R. Konietzschke, M. Grebenstein, R. Warup, R. Haslinger, M. Frommberger, and G. Hirzinger, “The DLR MIRO: a versatile lightweight robot for surgical applications,” Industrial Robot: An International Journal, vol. 35, no. 4, pp. 324–336, Jun. 2008. [Online]. Available: https://www.emerald.com/insight/content/doi/10.1108/01439910810876427/full/html

[15] M. M. Marinho, K. Harada, A. Morita, and M. Mitsuishi, “Smartarm: Integration and validation of a versatile surgical robotic system for constrained workspaces,” The International Journal of Medical Robotics and Computer Assisted Surgery, (in press) 2019.

[16] K. L. Schwaner, I. Iturrate, J. K. Holm Andersen, C. Rosendahl Dam, P. T. Jensen, and T. Rajeeth Savarimuthu, “MOPS: A Modular and Open Platform for Surgical Robotics Research,” in 2021 International Symposium on Medical Robotics (ISMR), Atlanta, GA, USA: IEEE, Nov. 2021, pp. 1–8. [Online]. Available: https://ieeexplore.ieee.org/document/9661539/

[17] M. M. Marinho, B. V. Adorno, K. Harada, K. Deie, A. Deguet, P. Kazanzides, R. H. Taylor, and M. Mitsuishi, “A unified framework for the teleoperation of surgical robots in constrained workspaces,” in 2019 International Conference on Robotics and Automation (ICRA). IEEE, May 2019.

[18] M. M. Marinho, B. V. Adorno, K. Harada, and M. Mitsuishi, “Dynamic Active Constraints for Surgical Robots Using Vector-Field Inequalities,” IEEE Transactions on Robotics, vol. 35, no. 5, pp. 1166–1185, Oct 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8742769/

[19] A. Miller and P. Allen, “Graspit! a versatile simulator for robotic grasping,” IEEE Robotics & Automation Magazine, vol. 11, pp. 110–122, 12 2004. [Online]. Available: http://ieeexplore.ieee.org/document/1371616/