A Unity based Da Vinci Robot Simulator for Surgical Training

Ke Fan, Aldo Marzullo, Nicolò Pasini, Alberto Rota, Matteo Pecorella, Giancarlo Ferrigno and Elena De Momi

***Abstract*— The development of the Robot-Assisted Minimally Invasive Surgery (RAMIS) imposes an increasing demand of surgical training platforms, especially low cost simulation-based surgical training through the creation of new open-source modules. For this goal, a da Vinci Surgical robot simulator based on Unity Physics Engine is developed. The simulator is integrated with da Vinci Research Kit (dVRK), robot kinematic models and multiple sensors. The Robot Operating System (ROS) interface is embedded for better integration with ROS based software components. The simulator can provide interactive information such as haptic feedback with master input devices. Realistic scenes can help users get more presence. An application of a virtual fixture is implemented to test and verify the performance of the simulator. The results show that the simulator have high expansibility and support interactive training tasks well. The simulator is an open-source virtual platform that allows for extension of external algorithms.**

I. INTRODUCTION

The da Vinci surgical robot is designed for Robot-Assisted Minimally Invasive Surgery (RAMIS). The system is a master-slave teleoperation system. It allows surgeons to perform complex minimally invasive surgical procedures with precision and accuracy. The system is an advanced robotic platform designed to expand surgeons' capabilities and provides surgeons with precision, dexterity and control during surgery [1]. The robot system is dexterous with a set of advanced instruments. However, the high cost of using the da Vinci system hinders its widespread adoption and sets a higher request to the operation level of surgeons. Therefore, simulation-based surgical training through lower cost, validated, virtual surgical training modules is helpful to teaching new surgeons surgical procedures and techniques[2].

The da Vinci Research Kit (dVRK) is an open-source system, composed of hardware and software that is wildly adopted for research based on the first-generation da Vinci system[3]. Many alternative master/slave devices such as the Phantom Omni and the Raven II research robot [4] can be used to assemble research platforms that explore innovative applications in minimally-invasive surgery. Researchers have developed various innovative applications and algorithms[5], [6] based on this shared research platform. Yan[7] proposed a dynamic simulation based on dVRK for modelling of the parallelograms, springs and counterweight. Adnan[8] presented a simulation for manipulation, visualization, and interaction of soft bodies and robots. Fontanelli[9] proposed a simulation for the identification of the Patient Side Manipulators (PSMs) and the Master Tool Manipulators (MTMs) based on dVRK. These simulations all aimed at single applications on the da Vinci system. A surgical training platform of the da Vinci system needs to simulate all the functions, also require significant system integration and a unified control framework for customized algorithm development. One alternative is to develop a virtual simulation platform for the complete da Vinci surgical system. The dVRK can provide a programming interface for software development.

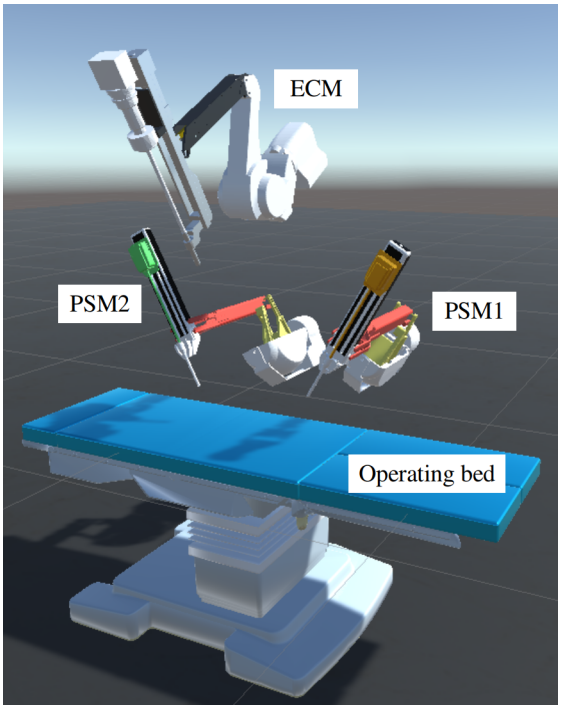

Fig. 1. The da Vinci surgical simulator in Unity scene

This paper proposes the development of a virtual simulation platform composed of full da Vinci surgical system components for surgical training, as shown in Fig.1. The simulator is a telesurgical system, which requires master input devices, preferably with haptic feedback. Currently, there are several haptic input devices with open interfaces, ranging from low-cost systems such as the Phantom Omni (now Geomagic Touch) and Novint Falcon, to more costly alternatives such as Sigma7 and MTMs of dVRK. In this context, MTMs are used as the haptic input device. The simulator can provide dynamic simulations of 2 PSMs and the Endoscopic Camera

Ke Fan, Aldo Marzullo, Nicolò Pasini, Alberto Rota, Matteo Pecorella, Giancarlo Ferrigno and Elena De Momi are with Department of Electronics, Information and Bioengineering, Politecnico di Milano, Milan, Italy ke.fan@polimi.it

Manipulator (ECM). Various surgical training scenarios can be established so that the simulator provides low-cost access to the development of innovative control strategies. It can reduce the risk of breaking the real da Vinci robot, especially when testing some artificial intelligence applications such as reinforcement learning algorithms. We build the simulator based on the Unity physic engine. The engine is a powerful platform for interactive simulations. Users can develop their own algorithms towards every component of the robot through Unity embedded scripts, ROS nodes or customized remote clients. The communication between the simulator and ROS is realized by ROSSHARP based on the rosbridge suite. New surgical components or scenarios can be easily imported into the simulator. In order to validate the potentiality for surgical training of the simulator, we develop a scenario of the virtual fixture to supply haptic guidance towards specific surgical tasks. In robot-assisted surgical systems, if the surgeon's inputs are not perfectly aligned to the goal behavior, a proactive robot behavior is needed to assist in the task execution [10]. The virtual fixture can assist the surgeon in carrying out the operation under the guidance of the desired trajectory[11].

the paper is organized as follows: Section II describes the details of the setup of the simulator such as framework and communication methods; Section III illustrates the virtual fixture implemented in the simulator. The experimental results and discussion are presented; Section IV is the conclusion and future work.

II. Setup of the Simulator

The first-generation da Vinci Surgical System is composed of two or three PSMs, one ECM, and two MTMs. The PSMs are mounted on the Setup Joints (SUJ). The position of each manipulator can be manually adjusted through SUJ. We do not import the SUJ into our simulator, since we can easily adjust the relative position in the simulator. As shown in Fig. 1, the Unity simulator consists of PSM1, PSM2, ECM and the operating bed.

A. Software Architecture

The standard da Vinci surgical system is composed of mechanical hardware and control software that supplies directly bilateral control for receiving and sending messages such as joint state and Cartesian position to the manipulators on both the patient side and the surgeon side.

The software architecture of the simulator is shown in Fig.2. The simulator consists of dynamic simulation part, teleoperation part, force feedback (wrench) generation part, Graphical User Interface (GUI) and rodsbridge suite for communication between Unity and dVRK. The dynamic simulation part provides the graphics and physics of each virtual component with kinematics and collision configuration. The manipulators are modelled in Solidworks and imported as the Unified Robot Description Format (URDF) files. In Unity, the kinematic chain of each manipulator is realized by the hinge joint component for revolute joints and the configurable joint component for prismatic joints. The dynamic simulation runs at a computer with an i7-9750H and a GeForce GTX 1650. The teleoperation is realized by the da Vinci console with MTMs and the stereo display. The da Vinci surgeon's console consists of a stereo display, three foot pedals, two MTMS with 7 active joints and a passive gripper. The MTMs can provide force feedback. The teleoperation scaling is 0.6. The controller hardware is connected by the firewire with a safety relay check. On the Unity side, two cameras (resolution of 640*480) are mounted on the end of the ECM link to simulate the binocular vision of the real da Vinci robot endoscope. The view of the simulated endoscope is displayed in the real console to realize the stereo display. The force feedback generation part aims to improve the surgical training performance of the simulator. The generation algorithm is a high-level controller which can be developed using C++, Python outside of Unity or inside of Unity with C-sharp scripts.

B. Communication Between Components

The dVRK is a hybrid of the low-level controllers and hardware, which has a component-based software architecture [3]. Different modules and functions can be tailor-made with configuration files (JSON files). In this context, we customize three kinematic configuration files for simulating the state of PSMs and the ECM. For example, the PSM kinematic file leverages the PSM device interface embedded in the dVRK to trick dVRK to regard the simulated PSM as the real PSM connected to the system. The sawSocketStreamer component is used to connect controllers and manipulators by sending JSON-based messages over User Datagram Protocol (UDP). The rosbridge suite is used to connect dVRK and the Unity environment after pointing the IP address and port number of the ROS master. Joint state messages can be subscribed by the scripts in Unity through the rosbridge suite. The wrench message generated in Unity can also be published to MTMs. In the Unity environment, the surgeon can set the task states and parameters. Furthermore, the performance of the surgeon can be collected by the simulator for computing the intensity of force feedback to provide haptic guidance.

III. Example Application for Surgical Training

In order to demonstrate the potentiality and performance of the simulator in surgical training, an application of the virtual fixture for providing haptic guidance during surgical tasks is shown in this section.

A. Virtual Fixture

Robot-assisted minimally invasive surgery can perform many kinds of operations with higher accuracy and flexibility. When performing a surgical training task through the teleoperated robotic system, the surgeon should enter orders as the input to achieve specific task requirements. Although applying the surgeon's input commands directly can achieve maximum control over the system, the operator's input signal may not align to the operator's intention. The concept of virtual fixtures provides the possibility of eliminating

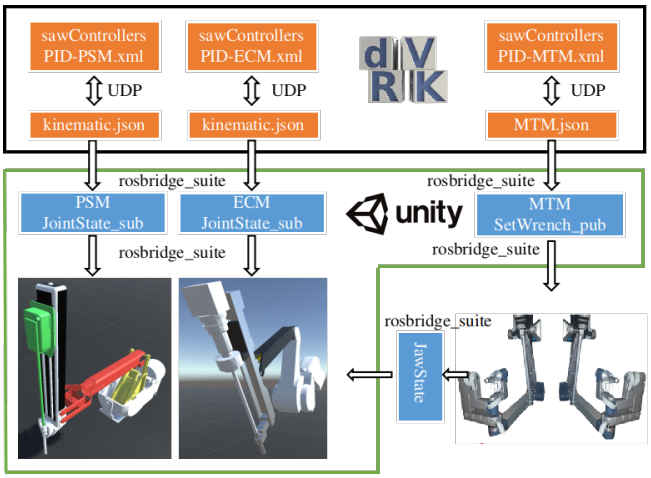

Fig. 2. The software architecture

the adverse effect of biased commands from surgeons by applying the superiority of robots and including the initiative of surgeons in the meantime.

The guidance force computed for the virtual fixture is of viscous type, as its magnitude depends on the velocity change of the end-effector in space: a more detailed discussion can be found in the previous work [12] of our lab. The force magnitude $\vec{G}(k)$ at time step k is, in this case, presented as Fig. 3. $v(k)$ is the velocity of the tool tip. $\varphi(k)$ is a normalized vector that represents the direction from the tool tip to the reference path.

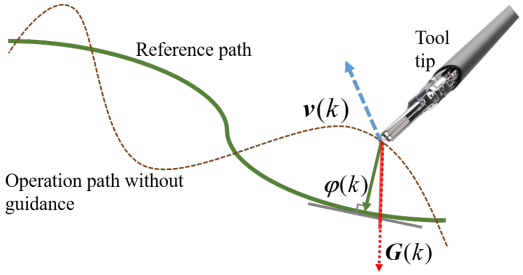

Fig. 3. The concept of the viscous-based virtual fixture

The direction of the guidance force is continually changing with the tool tip motion state as shown in Fig. 4.

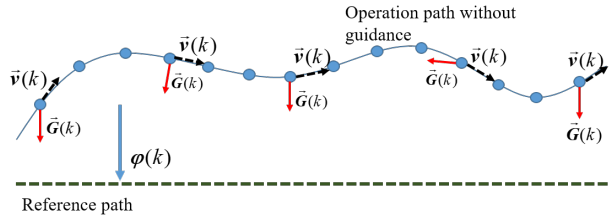

Fig. 4. The direction of the guidance force.

The $\vec{G}(k)$ represents the direction vector of the guidance force, the $\vec{v}(k)$represents the direction vector of the tool tip velocity.

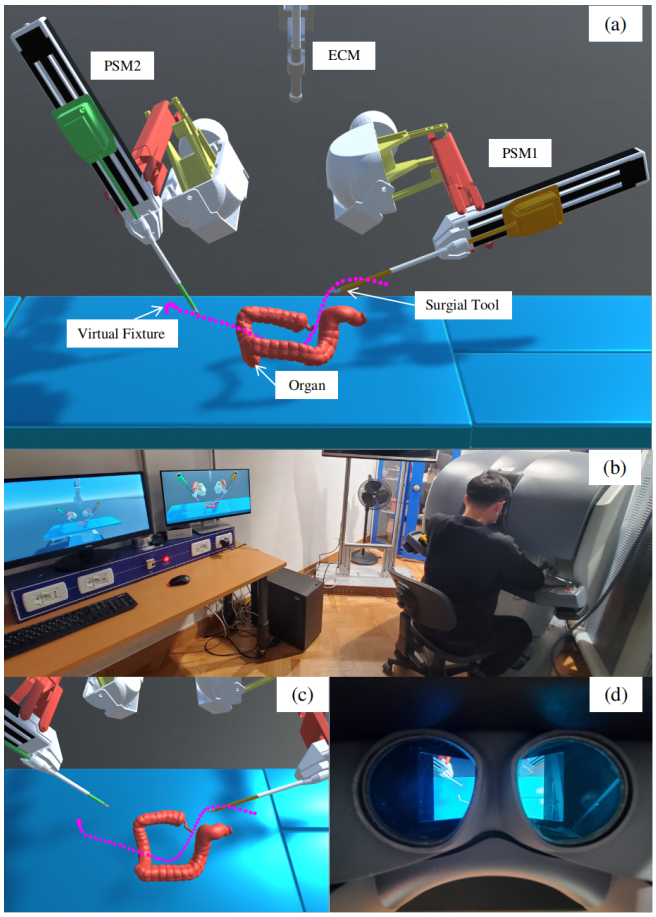

Fig. 5. The virtual fixture in the Unity scene. Figure (a) is the overview of the training task. Figure (b) shows the surgeon operating MTMs of the da Vinci console. Figure (c) shows the view of surgeons on the console with the stereo display. Figure (d) shows the stereo display of the console

B. Implementation of Virtual Fixture

The above virtual fixture is implemented in our simulator as shown in Fig. 5. The surgeon moves the surgical tool through MTMs along a curved reference path and attempts to avoid the surgical tool contacting the organ, as shown in (a). Figure. 5 (b) shows that the operator is operating the task at the console. The simulator can support different camera positions for external monitoring of the tasks. The operator's view is a stereo view including the depth information, as shown in (c). Figure. 5 (d) is the stereo viewer on the console. The training task requires hand-eye coordination and manual dexterity. The curved reference path is designed to go along the three Cartesian dimensions. The depth information is important for completing the task. Once the operator deviates from the reference path, the MTMs will pull the hands back by generated guidance force.

C. Results and discussion

After the implementation of the methods described above, we collected data coming from the simulation both for force, velocity and deviation estimation, as shown in Fig. 6. Please notice that, since the three variables are grouped together, on the y-axis, no unit of measurement is specified, while on

the common x-axis we can find the timestamps, according to the DVRK acquisition frequency. The data acquisition process starts as soon as the MTMs are enabled for user control [ts=1]. Force computation takes into account both the velocity magnitude and direction, and it can be seen analysing the force's parabolic trends with respect to the deviation derivative, in these four particular patterns repeated multiple times.

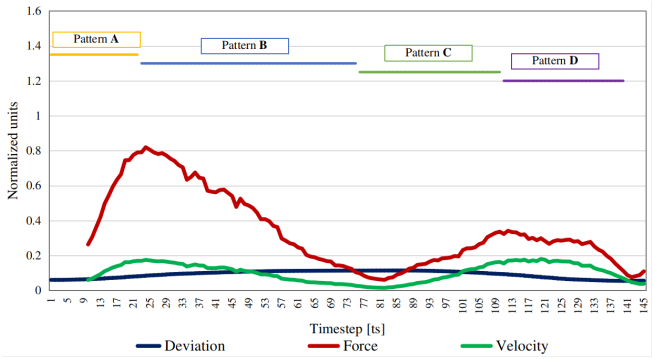

Fig. 6. Magnitude of the force, deviation and velocity vectors during movements deviating (Patterns A and B) and approaching (Patternss C and D) the predefined path

- Pattern A: Increasing deviation and velocity: the tool tip is moving away from the path, both the variables have a positive contribution in the increase of force magnitude. Force and velocity increase till they both reach a local maximum when deviation's derivative reaches its peak.
- Pattern B: Increasing deviation, decreasing velocity: the tool tip is still moving away from the path but is decelerating, hence deviation is the only positive contribution. Force starts decreasing and follows velocity's trend till they both reach a local minimum when deviation's derivative is zero: the tool tip is temporary still.
- Pattern C: Decreasing deviation, increasing velocity: the tool tip has reversed course and started approaching the path at an increasing velocity, which is the main positive contribution to the computations. Force starts increasing again, but the decreasing deviation tends to lower its magnitude, resulting in a parabolic trend, as for the ascending part, with reduced values. Peaks reached inside this pattern are the lowest local maximum encountered during experiments.
- Pattern D: Decreasing deviation, decreasing velocity: the tool tip has almost reached the path, so the user decreases its velocity. Both deviation and velocity have a negative contribution to force's magnitude, which reaches again a local minimum.

IV. CONCLUSIONS AND FUTURE WORK

In this work, a Unity physic engine based simulator of da Vinci surgical robot is proposed. The dVRK is integrated with this simulator to provide communication between the software and the hardware. External algorithms can also be developed and connected to the simulator via ROS. The simulator is an interactive simulator that can benefit surgical training. The application of a virtual fixture shows the potentialities of developing different control algorithms and application scenes.

In the future, we will develop more machine learning algorithms based on this simulation framework, and try to extend the function of simulation for transferring to real robots.

ACKNOWLEDGMENT

We received suggestions and help on the dVRK from Anton Deguet. Thank to our subjects for participating in the experiment.

REFERENCES

[1] N. Enayati, E. De Momi, and G. Ferrigno, "Haptics in robot-assisted surgery: Challenges and benefits," *IEEE Reviews in Biomedical Engineering*, vol. 9, pp. 49–65, 2016.

[2] A. E. Abdelaal, M. Sakr, A. Avinash, S. K. Mohammed, A. K. Bajwa, M. Sahni, S. Hor, S. Fels, and S. E. Salcudean, "Play me back: A unified training platform for robotic and laparoscopic surgery," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 554–561, 2019.

[3] 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 *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 6434–6439.

[4] B. Hannaford, J. Rosen, D. W. Friedman, H. King, P. Roan, L. Cheng, D. Glozman, J. 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, 2013.

[5] X. Ma, C. Song, P. W. Chiu, and Z. Li, "Visual servo of a 6-dof robotic stereo flexible endoscope based on da vinci research kit (dvrk) system," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 820–827, 2020.

[6] J. Xu, B. Li, B. Lu, Y.-H. Liu, Q. Dou, and P.-A. Heng, "Surrol: An open-source reinforcement learning centered and dvrk compatible platform for surgical robot learning," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 1821–1828.

[7] Y. Wang, R. Gondokaryono, A. Munawar, and G. S. Fischer, "A convex optimization-based dynamic model identification package for the da vinci research kit," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3657–3664, 2019.

[8] A. Munawar, N. Srishankar, and G. S. Fischer, "An open-source framework for rapid development of interactive soft-body simulations for real-time training," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 6544–6550.

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

[10] J. Abbott and A. Okamura, "Virtual fixture architectures for telemanipulation," in *2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*, vol. 2, 2003, pp. 2798–2805 vol.2.

[11] Z. Chen, A. Malpani, P. Chalasani, A. Deguet, S. S. Vedula, P. Kazanzides, and R. H. Taylor, "Virtual fixture assistance for needle passing and knot tying," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 2343–2350.

[12] N. Enayati, E. C. Alves Costa, G. Ferrigno, and E. De Momi, "A dynamic non-energy-storing guidance constraint with motion redirection for robot-assisted surgery," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 4311–4316.

[13] M. Shahbazi, S. F. Atashzar, C. Ward, H. A. Talebi, and R. V. Patel, "Multimodal sensorimotor integration for expert-in-the-loop telerobotic surgical training," *IEEE Transactions on Robotics*, vol. 34, no. 6, pp. 1549–1564, 2018.

[14] R. Geoghegan, J. Song, A. Singh, T. Le, A. Abiri, and A. H. Mendelsohn, "Development of a transoral robotic surgery training platform¡sup¿*¡/sup¿," in *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2019, pp. 5851–5854.

[15] J. Luo, P. Kania, P. P. Banerjee, S. Sikder, C. J. Luciano, and W. G. Myers, "A part-task haptic simulator for ophthalmic surgical training," in *2016 IEEE Symposium on 3D User Interfaces (3DUI)*, 2016, pp. 259–260.

[16] M. A. Schill, C. Wagner, M. Hennen, H.-J. Bender, and R. Männer, "Eyesi–a simulator for intra-ocular surgery," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 1999, pp. 1166–1174.

[17] C. K. Lam, K. Sundaraj, and M. N. Sulaiman, "Virtual reality simulator for phacoemulsification cataract surgery education and training," *Procedia Computer Science*, vol. 18, pp. 742–748, 2013.

[18] H. Wang, S. Jiang, and J. Wu, "A virtual reality based simulator for training surgical skills in procedure of catheter ablation," in *2018 IEEE International Conference on Artificial Intelligence and Virtual Reality (AIVR)*, 2018, pp. 247–248.

[19] M. Bischoff, "Rossharp," *https://github.com/siemens/ros-sharp*.

[20] S. Park, R. D. Howe, and D. F. Torchiana, "Virtual fixtures for robotic cardiac surgery," in *International conference on medical image computing and computer-assisted intervention*. Springer, 2001, pp. 1419–1420.