2024 Second International Conference on Emerging Trends in Information Technology and Engineering (ICETITE) | 979-8-3503-2820-2/24/$31.00 ©2024 IEEE | DOI: 10.1109/IC-ETITE58242.2024.10493515

Virtual reality-based training simulator for robot-assisted surgery using Unity and 3D slicer

Abhinav Azad*
Dept. of Transport & Planning
TU Delft
Delft, Netherlands
abhiazadz@gmail.com

Siddhant Panigrahi
Dept. of Eng. Design
IIT Madras
Chennai, India
sid.panigrahi1709@gmail.com

Asokan Thondiyath
Dept. of Eng. Design
IIT Madras
Chennai, India
asok@iitm.ac.in

Nirav Patel
Dept. of Eng. Design
IIT Madras
Chennai, India
niravpatel@iitm.ac.in

***Abstract*—Robotic interventions have become a pivotal component in minimally invasive neurosurgical procedures. While the use of robotic devices in medical operations is advancing, there is still a pressing need to train and equip medical professionals with emerging biomedical developments. Though Virtual reality (VR) has been proven to be a great tool for embodied learning, existing studies demonstrate a lack of interactive open-source platforms for medical interventions. Addressing this need for robot-assisted surgery (RAS) skills acquisition task, we propose a heuristic learning scheme in a simulator encompassing cross-platform registration tasks to train individuals for a typical image-guided surgery (IGS) procedure.**

The proposed training simulator offers an interactive, risk-free, cost-effective, and efficient learning platform for acquiring RAS skills both in Desktop VR and Head Mounted Display (HMD) VR settings. This training module provides a workflow for patient registration between the simulated robotic coordinate space in Unity 3D, and the IGS coordinate space in 3D slicer. The simulator's interface provides a user-friendly training environment with real-time instructions as well as visual and tactile feedback. A data collection scheme is also presented for a comparative user study of the learning experiences between HMD and desktop VR-based learning settings. This work showcases the efficacy of RAS training platform in performing cross-platform fiducial registration with an RMSE of 1.34 mm (in Desktop VR) and 1.49 mm (in HMD settings).

***Index Terms*—Robot-assisted surgery (RAS), Virtual reality (VR), Minimally Invasive surgery (MIS), Image Guided Surgery (IGS)**

I. Introduction

Recent advancements in robotics and biomedical engineering [1], [2], have enabled doctors to perform sophisticated surgical procedures with high confidence and reliability. Robot-assisted and IGS techniques are crucial in sophisticated procedures like neurosurgery involving intricate movements and prolonged operation time. These recent advancements have undoubtedly broadened the horizons for surgeons and doctors, enabling them to implement complicated and cutting-edge procedures. However, it has also introduced the challenge of enabling medical operatives with skills and expertise to use these biomedical instruments.

Supplementary Material: Multimedia link demonstrating the implementation of RAS simulator. https://www.youtube.com/watch?v=73omjcOy2lU

Only expert neurosurgeons with years of experience and honed skills through countless practice sessions can perform these complex brain surgeries with exceptional sensitivity. To reach such proficiency, surgeons require extensive technical exposure, hands-on experience with robotic systems, and several years of practice in actual surgeries. At the same time, required state-of-the-art robotic facilities are not universally accessible and affordable for training. Consequently, medical professionals face challenges in technical upskilling to utilize these latest technologies [3], [4]. This scarcity of RAS procedural skills in medical professionals is evident by its high demand, unaffordability, and inaccessibility to most patients.

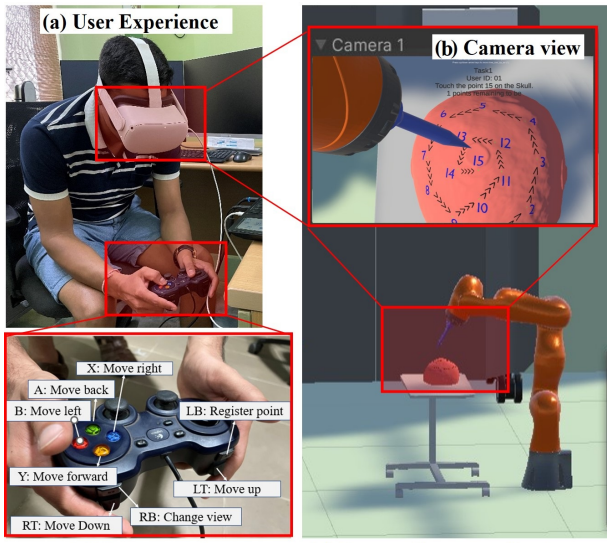

Fig. 1: Proposed VR simulator platform for training patient registration using KUKA LBR iiwa 7 on Oculus HMD VR. Here subfigure (a) illustrates the subject in HMD VR mode, (b) depicts the Camera view, (c) shows the user controls and (d) depicts the entire surgery simulator platform.

Inaccessibility in RAS training can be tackled by improving the availability and cost-effectiveness of training platforms to train surgeons with complex neurosurgical procedures while maintaining the accuracy and reliability of a real-life operational scenario. With the proposed training simulator, we aim

979-8-3503-2820-2/$31.00 ©2024 IEEE

to provide a Virtual learning environment (VLE) in neurosurgery where surgeons can cultivate their skills, evaluate their performance and reduce their errors with VR-enabled equipment, as illustrated in Fig.1. These simulators can provide a controlled environment replicating patient-specific data (from MRI or CT scans) so that surgeons can plan complex surgeries, like oncological tumour removal, on a virtual model that closely resembles the patient's anatomy. Hence the primary contributions of this work are listed as follows:

(a) Design of a training simulator for robotic-surgery skill acquisition performing cross-platform linkage between Unity and 3D Slicer using OpenIGTLink.
(b) Development of a learning module for patient registration task in an IGS scenario.
(c) Methodological user interface design for an effective immersive learning experience inducing cognitive learning.

The remainder of the paper is structured as follows: Section II provides a schematic of the heuristic training module, whereas Section III discusses the implementation of the training simulator platform using Unity and KUKA LBR iiwa 7. The entire procedure of cross-platform fiducial registration in between RAS simulator platform on Unity and 3D slicer-based IGS platform is explained in Section IV. Finally, the preliminary results from the training platform are provided in Section V, followed by future scope and conclusions in Section VI.

II. Proposed Schematic of Training Module

This section provides an overview of task planning and workflow schematic, focusing on the fundamental framework behind our training module. The decision to choose the Unity game engine-based VR RAS simulator platform aligns with the findings of Shin et al. [5]. This study highlights that Virtual Learning Environments (VLEs) can enhance spatial awareness, performance fidelity, and confidence in RAS skills through heuristic learning.

Existing simulators such as Da-Vinci skills simulator [6], Mimic dV-Trainer [7], Robotic Surgical Simulator [8], and Sim-Surgery Educational Platform, have a steeper learning curve leaving it to the trainees to acquire the skills whenever required. Hence, our work encompasses the design of gamification elements in UX and UI while preserving the realism of manoeuvring a Robot with seven degrees of freedom, working as a slave manipulator for patient registration. Existing platforms can provide hyper-realistic simulations of the surgery but lack the methodological procedure, limiting their acceptability [9]. In existing surgical setups, there is a need to translate 2D representations of pre-operative scans into 3D anatomical structures to visualise the point of entry, plan surgical procedures and analyse its impact on the patient tissue. Fu et al.'s study [10] provided an overview of manipulators for needle insertion and surgical task planning while showcasing the applicability of KUKA manipulators in these scenarios. This led to the selection of KUKA LBR iiwa 7 as the teleoperated manipulator. Hence, this entire task can be divided into three key steps: (a) Skull model preparation using Blender,

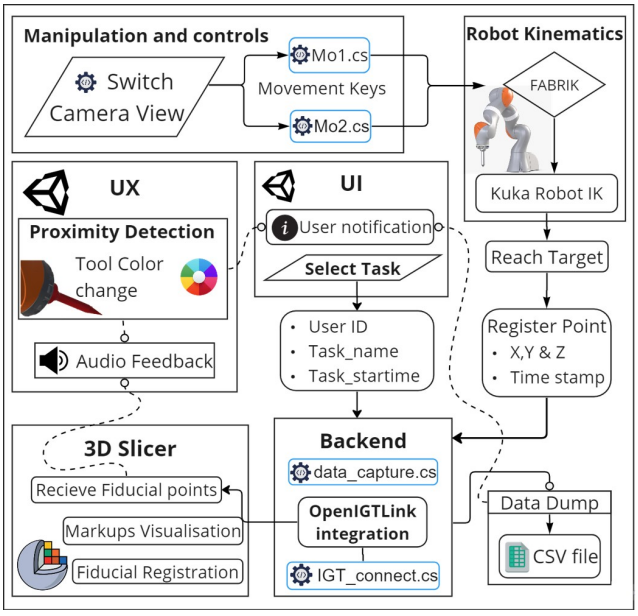

Fig. 2: Proposed pipeline of the RAS simulator platform. This diagram demonstrates the schematic workflow of the event manager and their interdependence in our VR Training programme.

(b) Environment setup for the proposed RAS platform, and (c) Simulating the manipulator teleoperation on Unity 3D, which is schematically explained in Fig. 2.

A. Skull model preparation using Blender

Three-dimensional STL model of the skull is extracted by segmenting the cranial region in the "MRBrainTumor1", a sample MRI scan data available on 3D slicer [11]. The segmentation was performed semi-automatically by the tools available in 3-D Slicer. Our training module comprises a trial run and five registration tasks utilising six different fiducial point sequences on a skull incorporating variability in the learning tasks. The registration sequence pattern of the tasks is designed with the complexity of the tasks increasing with each successive task (i.e. 1, 2, 3, 4, and 5). A well-defined registration sequence is designed for each task the trainee can follow.

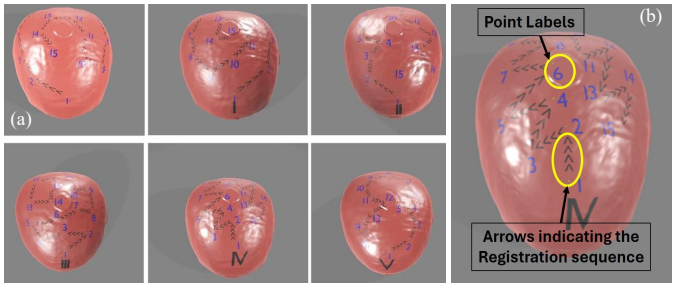

Fig. 3: Six different task scenarios generated using the skull STL model in blender and (b) the enlarged view of annotated points and their sequence

Here, we used utilised Blender's "Shrinkwrap" modifier for text wrapping over the skull's 3D surfaces. Fig. 3 illustrates the

six different task sequences generated over the skull models. Additionally, directional arrows are added to help the user navigate in the correct sequence to the next point where these points are placed in sequence to increase the task complexity with each progressive iteration, as shown in Fig.3.

B. Environment setup for the proposed RAS platform

The procedure of setting up a complete surgical simulation involves modelling the interactions in Unity, where the user commands are mapped to the simulator using joysticks and the medical data is analysed in 3D slicer using the dedicated hardware setup of Oculus Quest 2 HMD and Desktop VR. This section introduces the individual components and discusses their implementation along the pipeline.

1) Unity 3D: The simulation environment is developed on Unity 3D with various plugins and packages to fulfil all the required functionalities. The environment had Oculus integration installed with OpenVRLoader and Oculus enabled in Extended Reality (XR) plugin management. All the standard cameras are deleted, and VR CameraRig was imported from the prefabs. The Unified Robot Description Format (URDF) importer package is also included to handle Kuka iiwa URDF model description files. Customized features for user interaction, collision detection, data capture, and user management are implemented using additional C# scripts to enhance user experience, which is further elaborated in the next section.

2) 3D slicer: Medical imaging data, such as MRI scans, is typically acquired as a series of 2D images taken at different depths (slices) throughout the body. Here, 3D Slicer is utilised as a visualization and analysis software for medical data imaging. For our use case, "MRBrainTumor1" MRI scan data of the cranial region is taken from the sample datasets available in 3D Slicer. 3D Slicer takes the 2D images from the MRI scan and reconstructs a 3D representation of the anatomical structures. Such visualisation can help physicians and radiologists better understand complex anatomical structures, identify abnormalities, and plan surgical interventions more precisely. The objective here is to establish seamless integration between Unity's robotic environment and the 3D slicer's medical imaging and guidance platform. This is achieved using OpenIGTLink [12], which enables server connectivity and real-time data transfer capabilities amongst these cross-platforms.

C. Simulating Kuka iiwa on Unity 3D

An immersive RAS experience is provided by enabling the teleoperation of the robotic manipulator with inverse kinematics configured for all the robot links. The entire Kuka iiwa URDF is generated on ROS using the Xacros provided in the official repository [13]. After generating the URDF for the robot, the kinematics were set up on Unity to enable manipulation of Kuka's tool GameObject, utilizing all seven degrees of freedom (DOFs). This is done by utilising the Forward and Backward Reaching Inverse Kinematic (FABRIK) solver [14], [15] to achieve a real-time RAS teleoperation. In the implementation, the robot's links are sequenced from "iiwa_link_1" to "iiwa_link_7", followed by the custom end-effector "iiwa_link_tool", and finally, the end effector tooltip as "iiwa_tool_tip_ee". The FABRIK solver sequentially processes various gameObjects representing the robot links and the end effector as a separate invisible sphere or the target. It is essential to note that FABRIK approximates reaching the target position, not orientation. Precision and smooth joint angle changes are ensured by optimising the maximum number of iterations the FABRIK solver performs to reach the target position.

III. RAS Training simulator on Unity

This section discusses the proposed RAS learning workflow and the primary components used to build the presented simulator platform. The key components depicted are wireframed to each other to build the open-source RAS simulator platform, as illustrated in Fig. 2.

A. Task workflow and User journey

The designed training module can be used in both HMD VR and desktop VR modes to provide end-to-end comparative user study schematics for the RAS training simulator. Currently, the training is setup at an experimental level where the user journey of the training simulator is depicted in the UI wireframe, as shown in Fig. 4. Before entering the simulator, an introduction to the training platform, HMD (Head Mounted Devices) devices, controls and a walkthrough of the UI and UX elements of the platform are provided to the user.

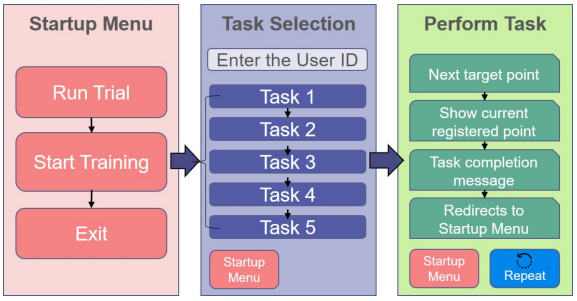

Fig. 4: UI wire-frame of our training module. Here, each UI panel is illustrated with arrows depicting the flow of events through UX design.

After the wireframe introduction, the user is given five minutes to get familiarised with the registration tasks by running a trial session, where the user can experience all the simulator elements independently. Following the trial, the actual training begins where the user performs five independent registration tasks in order from Task I to Task V. In each task, the user needs to spatially navigate the tool tip of the Kuka iiwa robot to reach the fiducial point markups in a given sequence on the skull model, as shown in Fig. 3.

B. UX design and Controls

Once the robot kinematics have been configured to align with the user's movements, we introduce essential game control elements. These elements enhance the user experience

by providing additional functionalities and feedback systems, creating a more interactive, responsive, and appealing training environment for the users.

1) Controls: For manoeuvring the slave robot and using other user functionalities like change in camera view, the Logitech Gamepad F310 joystick is used in both the Desktop VR as well as HMD VR settings to keep homogeneity in controls, as depicted in Fig. 1. A toggle switch key is dedicated to switching between two camera views, a close-up and a far viewpoint. For Kuka robot tool manipulation in three degrees of freedom, i.e. translation in X, Y, and Z using the joystick keys. When the user reaches a target fiducial point, a "send" key can be pressed, triggering sound feedback to reassure users that the point is recorded successfully. The multiple user functionalities are also shown in Fig. 2.

2) Proximity and collision awareness: To enhance the immersive experience and promote awareness, providing visual or tactile sound feedback is helpful as the user navigates around the scene. Our feedback system aims to minimize training errors and enhance the user's attention to the critical moments in the registration task. Unity provides collision detection between convex collider meshes with considerable accuracy. Hence an invisible sphere with a radius of 2 centimetres is used at the robot tooltip to define the critical operating proximity zone. Rigid body kinematics are only turned on to reduce computational load when the collision is detected. Visual and alert sound and vibration feedback were triggered upon detecting collisions using custom-made functions. As part of the visual feedback, the tooltip colour was changed from blue to red, as shown in Fig. 5. The alert sound and tactile vibration were triggered whenever the proximity sphere GameObject enters into a collision state, prompting users to focus intensely on achieving precise positioning. Further, a warning sound is triggered when the tool collides with the skull. The tool colour returns back to blue when the sphere leaves the state of collision.

C. Training module and User interface

To turn the simulator into a user-friendly platform, the UI scheme as shown in Fig. 4 is implemented. The core objective behind the UI development is to couple the user experience with a simple, accessible and informed workflow.

The "Startup menu" is the default menu panel where the training module begins when the simulator starts. "Task Selection" is the second UI panel under the "Start Training" option of the "Startup menu". Under the "Experiment menu", the users can provide their User ID and choose a training task. The input in the User ID tab will be used to generate personalised game instructions and task data annotation. Each task selection will reveal a unique skull registration pattern. For this experimental training and validation, the user needs to follow the order from Task 1 to Task 5.

When the user selects a task, the menu UI disappears, and only the simulated training platform with the default camera view can be seen with a menu button on the side. Now, the user can perform the registration task over the given skull model. The user must navigate around the environment, approach the point fiducial marking on the skull, and register these points. As the users progress in the registration task, dynamic instructions are prompted on the top central part of the view. These instructions include the current target point, notifications when a point is registered, and prompt errors when the user deviates from the desired path.

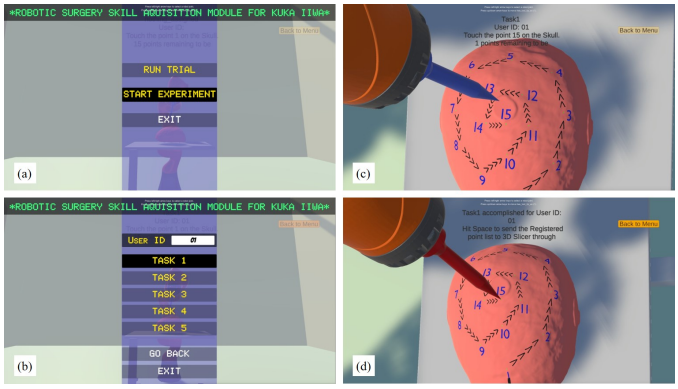

Fig. 5: The subfigure (a) shows the Startup menu, (b) shows the Task menu, (c) and (d) depicts Registration task in a close view without and without collision state respectively

D. Data capture and management

The registration tasks begin with instructions to reach fifteen points marked on the skull in ascending order as accurately as possible, following the fiducial point sequence imprinted on the skull models. As the user progresses to capture the fiducial markers, a success sound is triggered. The data frame is updated with X, Y and Z coordinates in the Unity environment and the time stamps. Once the final 15^{th} point is captured, the user is instructed to register, sending all recorded points to 3D Slicer, which is elaborated in the next section.

IV. Unity to 3D Slicer linkage

An essential contribution of our work involves achieving cross-platform registration between Unity and 3D Slicer to establish a robust cross-platform RAS simulator. This final step is crucial for seamless integration and functionality across different platforms. This section also discusses the role of the 3D slicer in our RAS training simulator.

A. Cross platform linkage using OpenIGTLink

The cross-platform connection was implemented using OpenIGTLink [12], a standardization protocol for enabling communication and data transfer among devices. Following the communication protocols, an IGT server was established on 3D Slicer using the OpenIGTLinkIF extension and an OpenIGTLink client at the Unity end. The registered point list is collected by the data capture system on Unity, and is prepared following a consistent OpenIGTLink message structure with a 58-byte header and a body section. The header message consisting of all the fiducial point lists is sent via the Unity client which is visualised on the 3D slicer server instantly, as shown in Fig. 6.

B. Fiducial Registration on 3D slicer

After receiving the moving point list in 3D Slicer, the registration is performed between the fiducial markups of the same point sequence on two different platforms, which are 3-D Slicer and Unity 3D. The fiducial registration can be done for a rigid, similar or affine transformation between different models point lists. Following the registration, the registration error is computed along with the 3D transformation matrix that can transform any point in one coordinate space to another. Once this is achieved, one can plan a surgical procedure on medical imaging software like 3-D Slicer and use the transform computed in registration to perform the procedure on a simulator platform, and they can compute the corresponding points in the robot coordinate space in Unity 3D. This concludes the final step of our training workflow for an RAS image-guided surgery scenario on our proposed training simulator.

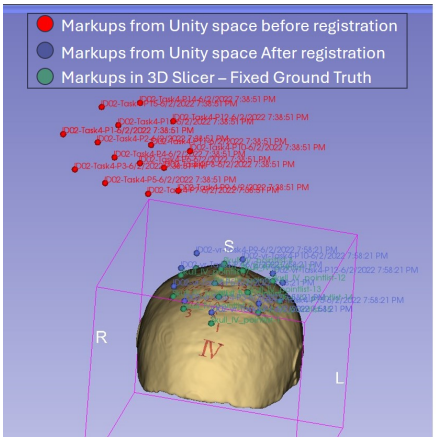

Fig. 6: Fiducial registration in the 3D Slicer

V. Experimental Protocol

The RAS simulator design incorporating all the functionalities mentioned in the above sections, is put together to formulate an easy-to-follow experiment protocol for the participants to finish the RAS tasks in Desktop VR and HMD VR mode. We selected five male participants aged 20-25 years with no prior experience with RAS and VR simulators. The experiment consisted of two training sessions conducted once for each Desktop and HMD VR mode. Each training session consists of a trial followed by five tasks, clearly depicted in Fig. 4. Following the data acquisition protocol mentioned in section III, the completion time and the inaccuracy are extracted for each registration task as performed by each user.

Completion time (CT) and RMSE (Root mean square error) are two primary elements extracted from the temporal trajectory analysis of the registration tasks. To gain deeper insights and differentiate the registration performance of individual users across different tasks, we introduce a novel performance evaluation metric called the Combined Task Performance Score (CTPS). The CTPS comprises two essential elements: the Completion Time (CT) Score and the registration RMSE

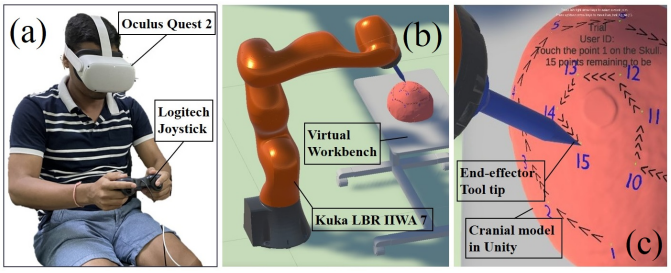

Fig. 7: Experiment setup for the RAS training simulator (a) shows a teleoperator in Physical space, (b) shows Kuka in medical VR workspace, and (c) Precision manipulation for skull Registration

penalty. The CT Score quantifies the efficiency of participants in completing the task and is expressed as shown in Eqn. 1:

$$\text{CT Score} = \frac{\text{Completion Time}}{\text{Average Completion Time for all tasks}} \quad (1)$$

In contrast, the registration RMSE penalty quantifies the reduction in CTPS score for relative compromise in the registration accuracy, calculated as the difference between the best achievable (minimum) RMSE and the participant's RMSE, divided by the best achievable RMSE within their respective setting, as expressed in Eqn. 2:

$$\text{RMSE Penalty} = \frac{\text{Achieved RMSE} - \text{Best RMSE}}{\text{Best RMSE}} \quad (2)$$

Cumulative CTPS is derived by averaging the CT Score and the registration RMSE penalty, scaled by a factor of 100 for better comparison. CTPS within each task sample can be denoted as shown in Eqn. 3:

$$\text{CTPS} = \frac{(\text{CT score - RMSE penalty})}{2} \times 100 \quad (3)$$

This CTPS score can serve as a valuable tool for understanding the capabilities and performance variations of the participant in the patient registration task. Here we computed the CTPS metric for each task performed by each subject and derived task and user performance along with the mean performance score as reported in Table I and II.

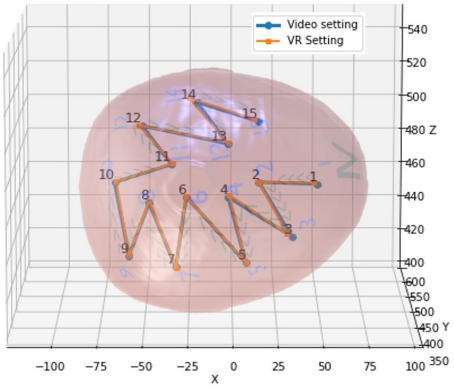

Fig. 8: Comparative plots from preliminary experiments of HMD VR and Desktop VR settings

TABLE I. PERFORMANCE METRICS FROM SIMULATOR TRAINING USING HMD VR

Task	Subject 1		Subject 2		Subject 3		Subject 4		Task Perf.
	CT	RMSE	CT	RMSE	CT	RMSE	CT	RMSE	
1	114.30	1.10	101.40	1.16	102.00	1.15	105.20	1.17	79.70
2	134.00	1.34	106.10	1.26	122.20	1.29	109.00	1.30	75.43
3	159.40	1.38	127.20	1.39	127.80	1.51	114.60	1.42	75.27
4	156.60	1.89	134.90	1.82	116.80	1.92	131.10	1.84	37.10
5	127.00	2.31	133.40	1.48	156.50	1.50	144.00	1.51	56.59
User Perf.	63.76		66.31		65.12		64.09		64.81
Overall performance in HMD VR (CTPS): 64.81 ±25.47									

TABLE II. PERFORMANCE METRICS FROM SIMULATOR TRAINING USING DESKTOP VR

Task	Subject 1		Subject 2		Subject 3		Subject 4		Task Perf.
	CT	RMSE	CT	RMSE	CT	RMSE	CT	RMSE	
1	113.20	1.05	101.00	1.06	102.90	1.12	110.30	1.01	77.61
2	147.70	1.15	114.80	1.13	132.80	1.15	125.10	1.18	86.42
3	167.90	1.29	99.80	1.25	127.20	1.32	114.60	1.38	68.72
4	197.30	1.67	104.90	1.76	144.10	1.74	109.30	1.69	37.52
5	152.60	1.84	147.30	1.31	145.60	1.36	130.30	1.35	66.18
User Perf.	81.72		58.83		68.37		60.22		67.29
Overall performance in Desktop VR (CTPS) : 67.29 ±25.96									

VI. Results and Discussion

The proposed RAS simulator presents a comprehensive patient registration workflow facilitating the collection of user performance data. These collected metrics obtained from our analysis indicate the following outcomes:

(a) CT is an important measure to assess task complexity, as it is observed that the CT is directly correlated to the task complexity with correlation (R^2) of 0.96 and 0.91 for HMD and Desktop VR settings.

(b) Analysis of CT trends indicated that participants can complete the tasks more quickly in the HMD VR environment (CT:126.18 sec) but with a slight decrease in registration accuracy (higher RMSE:1.49 mm).

(c) Performance trends reveal that Desktop VR has higher registration accuracy in comparison to HMD VR (lower RMSE: 1.34 mm) but has a marginal increase in the CT (CT:129.42 sec). However, the CTPS (67.29, Desktop VR $>$ 64.81, HMD VR) is greater in Desktop VR, demonstrating this as a better learning medium.

(d) Trends in task performance, as represented by the colour code, show that both Desktop and HMD VR modes exhibit comparable trends with increasing task complexity. This means that a simple task has a higher score in both modes, highlighted in green, and a tough task has a lower score, indicating the generalisability of our training platform.

Hence, Desktop VR is a more effective training medium as the CTPS of the participants is more due to lesser RMSE and comparable CT. This distinction in results is due to the higher resolution in Desktop VR (15.6" Dell G5 120Hz 1080p) in comparison to the HMD display (720p), which allows users to identify the markups with higher precision. It can also be concluded that the users are more familiar with Desktop VR than Oculus HMD settings, which can induce dizziness in longer durations [16].

VII. Conclusion

This paper presented the development of RAS simulator platform incorporating a 7 DOF KUKA iiwa 7 manipulator in the Unity engine. The presented work combines a simple and user-friendly UX designed in Unity to allow professionals to interact with robotic manipulators. The primary findings derived from the proposed RAS training simulator platform indicate that Desktop VR is a more effective medium for training when compared to HMD, which is also consistent with other studies [16]. This immersive nature of VR platforms can enable surgeons to learn from patient-specific medical data, providing a scalable solution for surgical education. In future works, we plan to evaluate and classify user clusters based on user performance with our proposed framework.

References

[1] H. J. Marcus, A. Hughes-Hallett, R. M. Kwasnicki, A. Darzi, G.-Z. Yang, and D. Nandi, "Technological innovation in neurosurgery: a quantitative study," *Journal of neurosurgery*, vol. 123, no. 1, pp. 174–181, 2015.

[2] I. Muskens, S. Diederen, J. Senders, A. Zamanipoor Najafabadi, W. van Furth, A. May, T. Smith, A. Bredenoord, and M. Broekman, "Innovation in neurosurgery: less than ideal? a systematic review," *Acta neurochirurgica*, vol. 159, pp. 1957–1966, 2017.

[3] D. Gerhardus, "Robot-assisted surgery: the future is here," *Journal of Healthcare Management*, vol. 48, no. 4, p. 242, 2003.

[4] P. Palani, S. Panigrahi, S. A. Jammi, and A. Thondiyath, "Real-time joint angle estimation using mediapipe framework and inertial sensors," in *2022 IEEE 22nd International Conference on Bioinformatics and Bioengineering (BIBE)*, pp. 128–133, 2022.

[5] D.-H. Shin, "The role of affordance in the experience of virtual reality learning: Technological and affective affordances in virtual reality," *Telematics and Informatics*, vol. 34, no. 8, pp. 1826–1836, 2017.

[6] W. S. Khor, B. Baker, K. Amin, A. Chan, K. Patel, and J. Wong, "Augmented and virtual reality in surgery—the digital surgical environment," *Annals of translational medicine*, vol. 4, no. 23, 2016.

[7] J. D. Bric, D. C. Lumbard, M. J. Frelich, and J. C. Gould, "Current state of virtual reality simulation in robotic surgery training: a review," *Surgical endoscopy*, vol. 30, pp. 2169–2178, 2016.

[8] J. M. Albani and D. I. Lee, "Virtual reality-assisted robotic surgery simulation," *Journal of Endourology*, vol. 21, no. 3, pp. 285–287, 2007.

[9] R. Eagleson, D. Kikinov, L. Bilbie, and S. de Ribaupierre, "Clinical trainee performance on task-based ar/vr-guided surgical simulation is correlated with their 3d image spatial reasoning scores," *Healthcare Technology Letters*, 2024.

[10] J. Fu, A. Rota, S. Li, J. Zhao, Q. Liu, E. Iovene, G. Ferrigno, and E. De Momi, "Recent advancements in augmented reality for robotic applications: A survey," *Actuators*, vol. 12, no. 8, 2023.

[11] "3d slicer sample data." Online Link Accessed on Dec 2022, https://www.slicer.org/wiki/SampleData.

[12] "Openigtlink." Online Link Accessed on Dec 2022, http://openigtlink.org/.

[13] "Kuka lbr iiwa 4, udrf importer." Online Link Accessed on Dec 2022, GitHub-Unity-Technologies/URDF-Importer:URDFimporter.

[14] A. Aristidou and J. Lasenby, "Fabrik: A fast, iterative solver for the inverse kinematics problem," *Graphical Models*, vol. 73, no. 5, pp. 243–260, 2011.

[15] S. Panigrahi, S. Ganesan, P. Palani, P. Jadhav, A. Kuncolienkar, and A. Thondiyath, "Swarm-based exploration in unknown environments: A case study of mobile-robots using ros framework," (New York, NY, USA), Association for Computing Machinery, 2023.

[16] Y. Feng, D. C. Duives, and S. P. Hoogendoorn, "Wayfinding behaviour in a multi-level building: A comparative study of hmd vr and desktop vr," *Advanced Engineering Informatics*, vol. 51, p. 101475, 2022.