Remote Lab of Robotic Manipulators through an Open Access ROS-based Platform

Bruno Stefanuto*‡, Luis Piardi*, Alexandre Oliveira Junior*, Marcos Vallim‡, and Paulo Leitão*†
* Research Centre in Digitalization and Intelligent Robotics (CeDRI), Instituto Politécnico de Bragança,
Campus de Santa Apolónia, 5300-253 Bragança, Portugal,
Email: {brunostefanuto, piardi, alexandrejunior, pleitao}@ipb.pt
† Laboratório Associado para a Sustentabilidade e Tecnologia em Regiões de Montanha (SusTEC),
Instituto Politécnico de Bragança, Campus de Santa Apolónia, 5300-253 Bragança, Portugal
‡ Universidade Tecnológica Federal do Paraná (UTFPR), Cornélio Procópio, Paraná, Brasil
Email: mvalim@utfpr.edu.br

2023 IEEE 21st International Conference on Industrial Informatics (INDIN) | 978-1-6654-9313-0/23/$31.00 ©2023 IEEE | DOI: 10.1109/INDIN51400.2023.10218202

***Abstract*—The research, training, and learning in robotic systems is a difficult task for institutions that do not have an appropriate equipment infrastructure, mainly due to the high investment required to acquire these systems. Possible alternatives are the use of robotic simulation platforms and the creation of remote robotic environments available for different users. The use of the last option surpasses the former one in terms of the possibility to handle real robotic systems during the training process. However, technical challenges appear in the management of the supporting infrastructure to use the robotic systems, namely in terms of access, safety, security, communication, and programming aspects. Having this in mind, this paper presents an approach for the remote operation of real robotic manipulators under a virtual robotics laboratory. To this end, an open access and safe web-based platform was developed for the remote control of robotic manipulators, being validated through the remote control of a real UR3 manipulator. This platform contributes to the research and training in robotic systems among different research centers and educational institutions that have limited access to these technologies. Furthermore, students and researchers can use this educational tool that differs from traditional robotic simulators through a virtual experience that connects real manipulators worldwide through the Internet.**

***Index Terms*—Virtual and remote laboratories, robotic manipulators, remote operation.**

I. Introduction

Robotics has been a fast-growing field in recent years, with manipulator robots increasingly taking a more significant place within industries [1]. The advent of Industry 4.0 and the emergence of cyber-physical systems bring the collaborative robots to the market, increasing the demand for professionals exhibiting the necessary skills to operate these systems and the need to re-qualify the existing workforce [2], [3]. However, educational institutions and research centers usually have scarce resources to acquire these systems and do not always have a real robot available to practice the hands-on operation of these robotic manipulators, develop teaching-oriented tasks and test experimentally innovative robotic approaches.

This challenge can be solved by using simulated environments for robotics teaching and research. According to Carrasquinho [4], robotic simulation environments can bring many benefits to students at a beginner level, such as avoiding accidents that can occur in the first interactions with a robot, enabling faster and more effective learning and distancing from losses by repair costs and waste of resources. However, for students and researchers with more advanced levels of knowledge in the area, simulations may not represent a system that is faithful to reality or depend on high computational processing, being necessary to explore a new alternative.

With the advance in Information and Communication Technologies (ICT), as well as the robustness and higher speed of communication infrastructures, which enabled the connection between devices, and especially between robots and operators in a virtual way, the current research is directed toward the use of the remote operation of manipulator robots [5], [6]. For example, the use of remotely operated robots in medical surgeries is a reality [7], being able to faithfully replicate the movements desired by the surgeon in charge of the operation.

The remote operation consists in using a real robot (getting rid of the problem of the lack of fidelity with the reality of simulated environments) without necessarily being presential (solving the challenge of the impossibility of the local operation due to the lack of availability of a real robot for testing). However, it is essential to notice that the remote operation can go beyond the simple teleoperation of the robot, e.g., using a real robot that is available remotely, for testing automated programs within the training and research perspectives.

The existing software programs that carry out these operations, e.g., RobotStudio and RoboDK, are advanced platforms for robot development and simulation that demand pre-existing knowledge. These programs often entail a steep learning curve, and even require specialized courses to cover advanced topics. Moreover, they are typically commercial software packages that demands local installation and possess operational prerequisites that make them unattainable for some students and researchers.

Having this in mind, this work proposes an open, accessible and safe solution for the remote operation of robotic manipulators in education, allowing the remote access and control of robots at run-time via a web interface with a faster learning curve. This enables users to operate robotic manipulators remotely from anywhere with an internet connection, giving greater accessibility and convenience. Furthermore, the user-friendly interface of this front-end makes it more appealing to certain users in contrast to the complex and technical interfaces of RobotStudio and RoboDK, rendering it a valuable addition to the existing tools and platforms in this field. Thereby this platform contributes to the democratization of the access to robotics education, due to the ease of access, intuitive commands, free to use and operate, and by having a generic architecture to easily replicate and consequently increase the availability of the application worldwide.

The proposed approach was experimentally tested using a UR3 collaborative robot, that can be successfully accessed remotely in a safe manner. Furthermore, the experimental tests using this platform were operational and promising, performing this exchange of information with low delay (in the order of milliseconds) and in run-time, as initially proposed.

The remaining of paper is organized as follows: Section II analyses the related work in the remote operation of manipulator robots, and Section III presents the proposed system architecture to operate remotely manipulator robots in a safe manner. Section IV describes the experimental implementation and testing using an UR3 manipulator robot and discusses the achieved results. Finally, Section V rounds up the paper with the conclusions and points out the future work.

II. Related Work

The robotic remote operation can be defined as performing a particular task or handling a robot remotely, i.e. operating over long distances, e.g., an operator controlling a robot or a doctor performing a remote robot operation for emergency surgery in other city [8]. For Boboc et al. [9], dangerous and inhospitable environments are the most focused areas for remote operations, such as underwater exploration, mining, agriculture, rescue, among others. From this analysis, one realizes the possibility of using the remote operation to increase the access of research centers and universities worldwide to robots wherever they are installed for better professional training and research development on different robots and applications. Ponce et al. [10] highlight that the students' learning process was improved through a robotic platform due to the interaction with the robot and the stimulus of using new technologies.

There is currently a great deal of research into developing robotic operation in several areas, including remote operations allied with web applications. Toquica et al. [11] developed a web application for the remote operation of an ABB industrial robotic arm focusing on the advancement of Industry 4.0 and the possibility of everything be connected due to the Internet of Things (IoT) technologies. It also highlights the importance of the TCP/IP protocol for the teleoperation and remote operation from the web and some challenges, such as latency when the operation is done over long distances.

As a potential technology for increasing the connection quality and speed, the 5G network brought speedy response in the remote operation. Yaovaja et al. [12] had the goal that an user could make any drawing by teleoperating an ABB robotic arm that would replicate the same design. 5G solved possible latency problems for long-distance operations, however, since the speed of the robot movements had to respect technical limitations, it led to some errors in the application.

The Internet for the remote robot operation based on the WebSocket protocol was tested by Kapic et al. [13] and achieved high speeds and full-duplex communication, using the Django framework with Robot Operating System (ROS) to integrate and control a robot. Peppoloni et al. [14] worked on developing a ROS-integrated interface for the remote operation from the user's hand movements. They used a Kinect, Head Mounted Display and a Leap Motion device to track movements and gestures, and the MoveIt! software for the calculation and trajectory planning of the robotic manipulator.

In healthcare, Yang et al. [15] designed a telerobotic system using an ABB YuMi dual-arm capable of assisting (delivering medicines, food, and other essential items) and performing necessary diagnostics on COVID-19 patients with replaceable connectors such as ultrasound and disinfection equipment, allowing the doctor to operate remotely, respecting the social distance and lowering the risk of virus contagion. There are also telesurgeries that join the remote operation of robots with emergency surgical procedures, allowing the execution of the medical operation remotely and in hard-to-reach areas that do not have a surgeon, e.g., on ships or spaceships [7].

In the area of customer service, there are also applications for the robotic remote operation. As example, Song et al. [16] proposed a remotely operated robot to sell products to public in a shopping mall that uses autonomous robots; however the remote operation brings more dynamism to unusual situations that are not foreseen or known by autonomous operation since the operator can respond in different ways.

Therefore, disruptive technologies, such as, 5G, IoT, and cloud computing, integrated by ROS, contributed to the advancement of remote robotic operation, reaching several areas and developing the most diverse tasks. However, the access to robotic manipulators in education is not yet universal. Not all research centers and universities have enough resources to provide these tools. Therefore, it is important to develop an architecture to build a network of virtualized real robots models available for remote operation for testing, training and research, that is agile and easy to access (i.e. no need to proceed with complex software procedures) for users who wish to have contact with different kinds of real robots (due to the interoperability of the architecture that can be replicated by several research centers to contribute with this network).

III. Remote Operation of Manipulator Robots

The proposed system architecture for the safe remote operation of robotic manipulators is presented in Fig. 1, where the developed algorithms, procedures or tasks are executed simultaneously in the virtualized robot and the corresponding real robot (that is accessed remotely through the Internet).

A. User

The user represents a student, researcher, or robot enthusiast who needs to execute some procedures (e.g., move the robot from the starting point to the end point) or tasks (i.e. perform

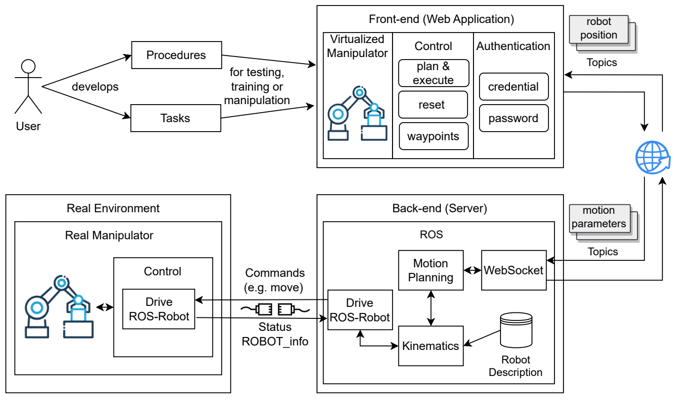


Fig. 1. Architecture for the remote operation of a manipulator robot.

a set of movements to carry out a function) to be performed in a real robotic manipulator, looking for an alternative to the available simulation environments. This interaction with the robot can be proposed by a robotics professor in his classes, by robotics-specific hackathons, or freely by the user himself.

The proposed architecture aims to accelerate the learning curve in robotics from the user's perspective, since the robot is ready without additional configuration. The user does not need to install the ROS platform, or to have a prior knowledge of this software package, which is displaced from the user's computer due to the web application that performs the communication with the server computer (that already has the ROS and the robot configured). For this purpose, and to ensure the proper remote access to the robotic system, only a stable and fast Internet access and a connected device, e.g., computer, tablet, or smartphone, are required from the user's perspective.

B. Front-end

The front-end consists of an user-friendly web application with 3D visualization of the robotic model (i.e., the virtualized representation of the robot's real state), interactive buttons and commands for the manipulation of the virtualized robot, executing on the real robot, and an user authentication home page for security reasons. For the graphics to be displayed, an browser with Web Graphics Library (WebGL) installed is required, which is nothing more than an application that helps to render interactive 3D graphics, natively presented in most browsers and running on most operating systems.

After visualizing the initial robot pose, the user can make desired changes in that pose to achieve a certain goal on the virtualized robot through the interactive marker by moving the x, y, and z axes, or by selecting the value of each joint. The interface also includes the following command buttons: "reset" to return to the initial position, "plan & execute" to perform the movement to the desired pose by sending the motion planning topics from the server computer to the real robot, and finally, "add waypoint" option to save the desired points in a script to resume them in the future (an option commonly used for pick and place operations).

C. Back-end

The back-end is running on a server, that is programmed and configured by an expert in ROS and robotics, is responsible for the heavy operation behind the web application (i.e., using ROS for system control and performing motion planning processing), preventing the user's computer from suffering such demand. Therefore, the server also includes the exchange of information, with this module being responsible for subscribing and publishing topics, i.e. messages on ROS, through the WebSocket protocol that allows the communication between the back and front-end.

The back-end is also responsible to ensure the communication with the real robot. For this purpose, upon receiving the user's trigger to process the robot's target pose, a program is activated inside the ROS on the server to perform the non-trivial motion planning calculation and send the kinematic parameters to the real robot. For optimal performance, it is recommended that both the back-end and the robot be connected to the same network via cable. This exchange of information between the server and the robot is performed via the ROS-Robot driver (i.e., a package containing calibration information, scripts, and drivers for the robot via ROS). The server also has the robot description packages provided by the manufacturer stored in a local database, which contain parameters needed for the motion planning and graphical visualization.

D. Environment

The environment represents the physical elements that make up the environment where the manipulator robot is inserted (e.g., bench or table) and the robot itself (which can be of various models and from different manufacturers, e.g., ABB, Universal Robots, and Fanuc). These elements are essential to be described at the moment of the motion planning for a safe operation, avoiding collisions with these environment objects or with the robot itself, and respecting the manipulator's limits.

The ROS-Robot driver is also installed on the robotic manipulator's controller, which will receive the kinematic command topics calculated by the motion planning module on the server and must send the updated status at each period of the pose and positions of each joint along with basic robot information (e.g., joint nomenclature, initial angulation). Thus, the manipulator robot is able to reach the position requested by the user through topics that have been processed by the server, closing the connection between the end user and the remotely connected object of study.

IV. Implementation and Experimental Results

The proposed approach was implemented on a UR3 robot from Universal Robots, with the developed system being available at [17]. By following the steps of this repository, users will be able to reproduce the system and operate robotic manipulators remotely via web. The operating system chosen for the server computer was Ubuntu 18.04 LTS (Bionic Beaver) due to its good integration and support with ROS Melodic, avoiding incompatibilities with the used tools (e.g., robot drivers) and the adoption in the robot developer community.

A. Implementation

The Unified Robotic Description Format (URDF) was used for the 3D visualization of the robot by the web application. It is a XML file that contains the robot description, with default parameters such as the kinematics, visual and physical design, and joint limits. This XML file can be found on the web provided by the robot manufacturer.

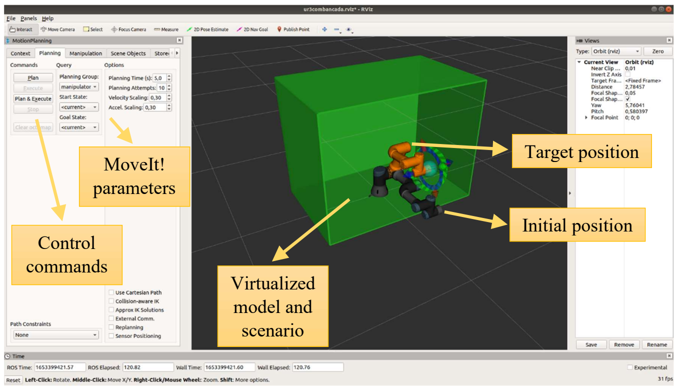
The Moveit! is the software that solves the motion planning, chosen due to its ease of initial configuration, providing the trajectories for each robot joint. The motion planning is a non-trivial task, generating a sequence of values for each joint of the robotic manipulator to follow in coordination with the others. Therefore, the use of MoveIt! facilitates this task by publishing topics in ROS with the trajectory plan for each joint to leave the initial position and reach the target position requested by the user.

The configuration of MoveIt! for the motion planning was done through the URDF file, which generates a self-collision matrix, contributing to a safe operation that avoids the user to overpass the robot's limits or cause crashes due to an unreachable input. The communication between the UR3 and the computer was made through the TCP/IP protocol, in which the robot was connected to the local network via cable, and the computer was also kept on the same network, both with fixed IP addresses for the constant exchange of topics without interruptions.

The motion planning validation was done through testing using the RViz software package, a ROS visualization tool, as shown in the Fig. 2. This test, which was extended throughout the web application, consisted in operating the robotic manipulator through this visualization tool, i.e. changing the target position, and executing the motion planning through MoveIt!. A limit of 5 seconds was imposed as the maximum time for the motion planning process and motion initiation on the real robot, which was able to reach the target position every time it was within its range. The only motion planning failures that occurred were when a collision was predicted or when the target position was beyond the range limits of this robotic model, so always due to a safety reason.

The Robot Web Tools, a collection of open-source libraries for building web applications with ROS, were used as the basis for the web application. The *ros3djs* and *roslibjs* libraries were used for the 3D visualization of ROS in web through JavaScript (i.e. visualization of URDF, interactive markers, and maps), and the communication through WebSockets with *rosbridge* to exchange publishing and subscribing messages between the web application and the server computer.

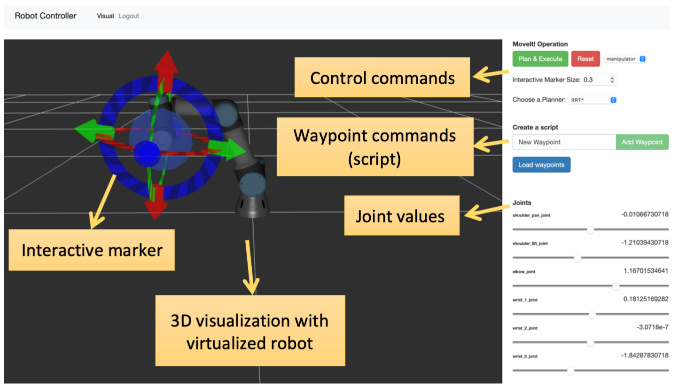
Then, from the libraries mentioned above and adding the URDF of the UR3 instance, the 3D viewer along with the virtualized robot was generated in the web application, as illustrated in Fig. 3. The information of the values of each joint is automatically presented with the initialization of the visualization and is updated according to the movement of the robot or with the user's command, through the communication provided by the rosbridge in ROS topics.


ur3combancada.rviz* - RViz
File Panels Help
Interact Move Camera Select Focus Camera Measure 2D Pose Estimate 2D Nav Goal Publish Point
MotionPlanning
Context Planning Manipulation Scene Objects Stored
Commands Query Options
Plan Execute Plan & Execute Stop Clear octomap
MoveIt! parameters
Control commands
Target position
Initial position
Virtualized model and scenario
Views

Planning Group: manipulator
Start State: <current>
Goal State: <current>
Planning Time (s): 5,0
Planning Attempts: 10
Velocity Scaling: 0,30
Accel. Scaling: 0,30

Fig. 2. MoveIt! parameters used for the motion planning defined in the RViz visualization interface.

According to Fig. 3, it is possible to notice that the developed web application is an intuitive tool for the remote operation with all the planning and execution controls clear to the user and that it fulfills the goal of being user-friendly.



manipulator
Interactive Marker Size: 0.3
Choose a Planner: RRT*

Joints

shoulder_pan_joint	-0.01066730718
shoulder_lift_joint	-1.21039430718
elbow_joint	1.16701534641
wrist_1_joint	0.18125169282
wrist_2_joint	-3.0718e-7
wrist_3_joint	-1.84287830718

Fig. 3. Web application user interface.

The control of the robotic manipulator through the remote operation based on web application proved to be efficient, just like the Moveit! test previously done, since, instead of the server computer generating the 3D visualization, this now occurs in the web application itself, but the motion planning processing continues to run on the server, not altering the operation of this already validated system. Again, it was possible to replicate all the movements inserted virtually in the real plant, according to the Fig. 4.

B. Experimental Results

For testing the run-time execution, the UR3 robotic manipulator was remotely operated in three different ways for run-time comparison: i) through only the server computer using the RViz and MoveIt! (direct connection), ii) through the user's web application running on the local network (local network connection), and iii) through the user's web application with the VPN to allow direct access to the computer server.

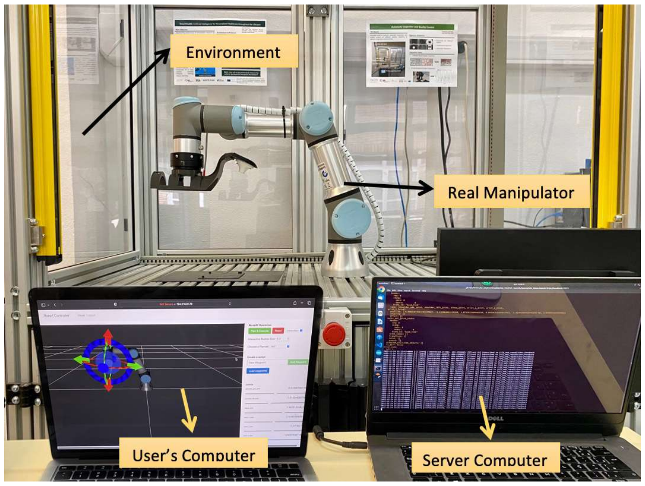


Fig. 4. Real robotic manipulator reached the same position as the virtualized one from remote operation.

The initial and final positions of the robot were the same for all tests, as illustrated in Fig. 5. The trajectory calculated by the motion planning was also the same, since MoveIt! always looks for the most optimal trajectory. The trajectory from the initial to the final position was repeated 30 times, and the average time was obtained for each scenario, according to the Table I. The achieved results clearly show that the type of connection made for remote operation has low impact on the performance of the developed architecture, the difference of 0.28 and 0.79 seconds from the local and VPN connection, respectively, between the direct connection is hardly felt by the end user. The standard deviation behaved as expected since the most stable connection is the direct connection (lower standard deviation of the time in the trajectory) and the most impaired connection is the VPN for depending on more external servers.

TABLE I
AVERAGE TRAJECTORY EXECUTION TIME COMPARING CONNECTIONS

	Direct	Local Network	VPN
Avg. Movement Time	4.13 s	4.41 s	4.92 s
Standard Deviation	0.00681445	0.01422318	0.02176415

This means that the expected delay, due to the communication time between the user computer, the server computer, and the real robot, is low and does not interfere with the usability. Additionally, an user who is located far from the real robot will have the opportunity to use the platform and implement trajectories without noticing significant delays.

It was also tested the run-time execution in 3 popular browsers, namely Google Chrome, Mozilla Firefox and Microsoft Edge. The robot trajectory was the same as in the previous test and repeated 20 times for each browser. The results are shown in the Table II validating that the proposed architecture supports multi-browsers. The achieved results also show that using different browsers have close values for the movement execution time, with the difference of at most 0.03 seconds between them being imperceptible at the end. This contributes to the universalization of the application, since it suffers practically no interference from the browser selection or the device, e.g., computer, tablet or smartphone, as the user does not have to provide high computational processing due to the system architecture that imposes this demand to the back-end and the user can therefore choose his or her preference.

TABLE II
AVERAGE TRAJECTORY EXECUTION TIME COMPARING BROWSERS

	Chrome	Firefox	Edge
Avg. Movement Time	4.42 s	4.44 s	4.41 s
Standard Deviation	0.01031095	0.00759155	0.00887041

Fig. 5 illustrates the movement of the robot simultaneously in the real and virtual spaces. It is possible to notice that the 3D visualization of the robot state presented in the web application also has a low delay, less than 0.1 second when connected to the local network and 0.5 second when connected via VPN. When connected by VPN there is a small drop in the frame refresh rate of the visualization, but this does not compromise the proper functioning of the tool.

Lastly, the motion planning process takes place once the server receives the user-provided parameters, and its processing time varies depending on the selected trajectory optimization method. However, to avoid causing a long waiting time to the user for the starting of the movement, a maximum time limit of 10 seconds has been imposed on this process.

From these experimental tests, it was possible to notice that the system brings acceptable range times to keep the good quality in the remote operation of robotic manipulators. Furthermore, this system ensured, in relation to safety, that all movements were performed without collision with the bench and the robot itself and that the limits were respected (e.g., speed, angulation, reach). Also as a form of safety, the simultaneous access of the application by different users does not put the manipulator at risk since the application screen is mirrored to everyone, making it impossible to have two different inputs at the same time or during the execution of a movement.

V. Conclusions and Future Works

Due to the lack of access to robotic manipulators in many research centers and universities, many students must seek alternatives to perform tests, validations, and tasks in robotics. One well-known alternative is the robotic simulation, but this usually either do not faithfully reproduce reality or rely on a lot of computational processing. Thus, the idea is combine the remote operation, a field widely used in surgical medicine, space or underwater exploration, with education and research.

Developing a system capable of joining different virtualized robots available for the remote operation that can be objects of study for people anywhere means expanding the access to robotics and contributing for the users be able to test and validate their developed algorithms or procedures and execute tasks with real robots remotely at run-time. This circumventing the problem of lack of fidelity with the reality

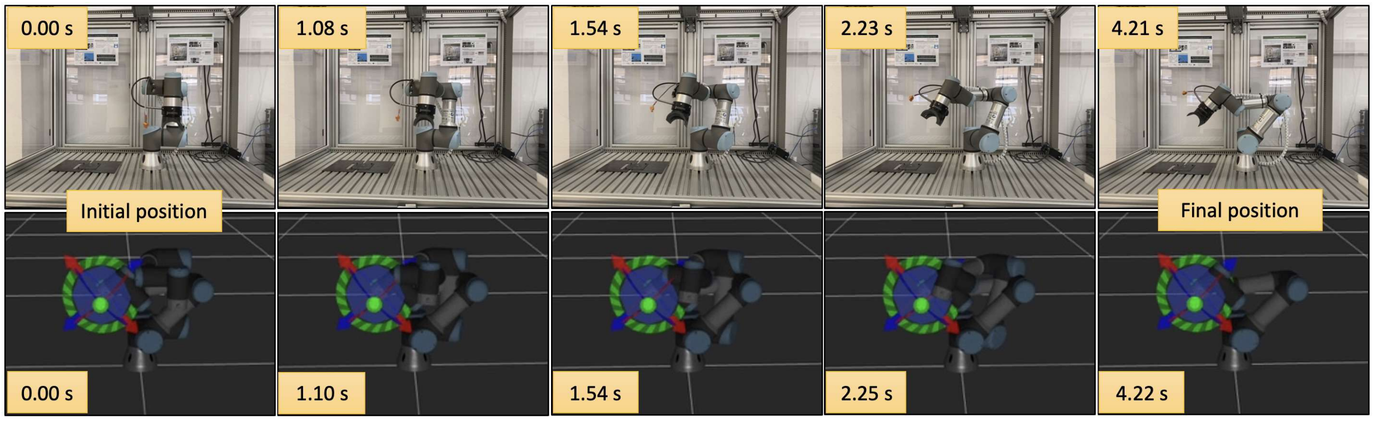


Fig. 5. Robot trajectory performed during the run-time execution test.

of some simulators, and even more, through a web application, recognizing the problem of the need for heavy programs and many computational resources. Moreover, with a generic system architecture that can be used with different robots supported by the ROS platform, the system can be easily replicated by several centers that want to compose a virtual laboratory network and have a robotic manipulator available to be remotely operated by other students and researchers around the world.

Preliminary results show that it was possible to remotely control and operate, in run-time, the UR3 robotic manipulator by using the MoveIt! software package for the kinematic motion planning, and libraries developed by Robots Web Tools to host the web application. It also became clear the potential of the developed system to contribute to the teaching of robotics, an area in great growth but often expensive to maintain in research centers and universities.

Future work is dedicated on implementing sensor data acquisition for processing, e.g., inputs from a joystick to control interactive markers, and the possibility for users to insert Python algorithms into the browser, enabling more skilled students and researchers to program robotic manipulator actions and tasks. Additionally, security concerns must be further addressed to host the back-end on an Internet-accessible server, allowing the access without a VPN.

Acknowledgment

The authors are grateful to the Foundation for Science and Technology (FCT, Portugal) for financial support through national funds FCT/MCTES (PIDDAC) to CeDRI (UIDB/05757/2020 and UIDP/05757/2020), and SusTEC (LA/P/0007/2021).

References

[1] C. Yang, J. Luo, C. Liu, M. Li, and S.-L. Dai, “Haptics electromyography perception and learning enhanced intelligence for teleoperated robot,” *IEEE Transactions on Automation Science and Engineering*, vol. 16, no. 4, pp. 1512–1521, 2018.

[2] J. A. García-Esteban, L. Piardi, P. Leitão, B. Curto, and V. Moreno, “An interaction strategy for safe human co-working with industrial collaborative robots,” in *4th IEEE Int'l. Conf. on Industrial Cyber-Physical Systems (ICPS)*, 2021, pp. 585–590.

[3] L. Variz, L. Piardi, P. J. Rodrigues, and P. Leitão, “Machine learning applied to an intelligent and adaptive robotic inspection station,” in *2019 IEEE 17th International Conference on Industrial Informatics (INDIN)*, vol. 1. IEEE, 2019, pp. 290–295.

[4] F. C. Carrasquinho, “Ferramenta de simulação para robô industrial,” Ph.D. dissertation, Universidade Nova de Lisboa, 2015.

[5] A. W. Barbosa, R. E. Bianchi, and R. A. F. Romero, “Control of mobile robots via internet,” in *Proceedings of COBEM–18th Int'l. Congress of Mechanical Engineering, November, (cd-rom), Ouro Preto, Minas Gerais, Brazil*, 2005.

[6] S. Sirouspour, “Modeling and control of cooperative teleoperation systems,” *IEEE Trans. on Robotics*, vol. 21, no. 6, pp. 1220–1225, 2005.

[7] N. Shahzad, T. Chawla, and T. Gala, “Telesurgery prospects in delivering healthcare in remote areas,” *The Journal of the Pakistan Medical Association*, vol. 69, no. Supl. 1, p. S69, 2019.

[8] S. Opiyo, J. Zhou, E. Mwangi, W. Kai, and I. Sunusi, “A review on teleoperation of mobile ground robots: Architecture and situation awareness,” *International Journal of Control, Automation and Systems*, vol. 19, no. 3, pp. 1384–1407, 2021.

[9] R. Boboc, H. Moga, and D. Talaba, “A review of current applications in teleoperation of mobile robots,” *Bulletin of the Transilvania University of Brasov. Engineering Sciences. Series I*, vol. 5, no. 2, p. 9, 2012.

[10] P. Ponce, A. Molina, E. O. L. Caudana, G. B. Reyes, and N. M. Parra, “Improving education in developing countries using robotic platforms,” *International Journal on Interactive Design and Manufacturing (IJIDeM)*, vol. 13, no. 4, pp. 1401–1422, 2019.

[11] J. S. Toquica, D. Benavides, and J. M. S. Motta, “Web compliant open architecture for teleoperation of industrial robots,” in *IEEE 15th Int'l. Conf. on Automation Science and Engineering (CASE)*, 2019, pp. 1408–1414.

[12] K. Yaovaja and J. Klunngien, “Teleoperation of an industrial robot using a non-standalone 5g mobile network,” in *Eurasia Conf. on IOT, Communication and Engineering (ECICE)*. IEEE, 2019, pp. 320–323.

[13] Z. Kapić, A. Crnkić, E. Mujčić, and J. Hamzabegović, “A web application for remote control of ros robot based on websocket protocol and django development environment,” in *IOP Conf. Series: Materials Science and Engineering*, vol. 1208, no. 1, 2021, p. 012035.

[14] L. Peppoloni, F. Brizzi, C. A. Avizzano, and E. Ruffaldi, “Immersive ros-integrated framework for robot teleoperation,” in *IEEE Symposium on 3D User Interfaces (3DUI)*, 2015, pp. 177–178.

[15] G. Yang, *et al.*, “Keep healthcare workers safe: application of teleoperated robot in isolation ward for COVID-19 prevention and control,” *Chinese J. of Mechanical Engineering*, vol. 33, no. 1, pp. 1–4, 2020.

[16] S. Song, J. Baba, J. Nakanishi, Y. Yoshikawa, and H. Ishiguro, “Teleoperated robot sells toothbrush in a shopping mall: A field study,” in *Extended Abstracts of the CHI Conf. on Human Factors in Computing Systems*, 2021, pp. 1–6.

[17] “Remote Lab for Operation of Robotic Manipulators through ROS,” https://github.com/mmsbruno/remote-operation-app.