The Development of Robot Control via Virtual Reality for Safe Human-Robot Interaction

M. A. Naeem*, Armughan Sarwar, Abdullah Humayun, M. Waseem

Department of Mechatronics & Control Engineering, University of Engineering and Technology, Lahore

* **Corresponding Author:** Email: mahsan.naeem@uet.edu.pk

Abstract

In Human-Robot Interaction (HRI) domain the prime focus is laid on the safety of the involved humans. High inertias, velocities and extensive work envelop, particularly of most of the currently used industrial robots, are reasons for the hazards associated with the interaction of humans with robots. Several strategies are being investigated for ensuring safety which includes lightweight robots (LWR), force-limiting control, start-stop monitoring, speed limiting mode, etc. These approaches often require a range of hardware peripherals to be incorporated within and around the robotic system. Alternatively, the use of Virtual Reality (VR) systems to interact with otherwise dangerous robots allows for contactless interaction. In this paper we present a Virtual Reality based arrangement that allows the interaction with the robot from distance, in a virtually created environment, to ensure human safety. Keeping in view the frequent requirements of changes in robot's movement plan, the proposed scheme demonstrates the capability of programming these plans in the virtual environment as well. We have shown the results of controlling a real robot in VR and the root mean square error (RMSE) between the two joint angles of the real and the virtual robot is 2.61 degrees and 3.08 degrees. This leads to remote/tele-control of the robot which is one of the future demands considering the social impact of COVID-19.

Key Words: Human-Robot Interaction, Robot Programming, Virtual Reality

1. Introduction

Robotic systems have penetrated all aspects of modern-day life. Be it a social domain or industrial, one can easily find some form of a robotic system involved. These systems are mostly utilized for repetitive, dirty, dangerous, or dull tasks. However, the dexterity of humans in certain situations cannot be matched by these automated systems. This leads to collaborative robots (cobots) where both humans and robots share the same workplace and work in collaboration. The most challenging factor in cobots is the risk of accidents that can cause damage to humans, mainly due to high-speed moving parts of the robots [1-2]. To ensure safe human-robot collaboration, limits are imposed on the robot's apparent inertia, speed, and payload [3].

Another important aspect, whenever robotic systems are involved, is the effort and skill required for programming such systems for a specific task. The whole process of programming and subsequent testing loses its worth if frequent changes in the robot's path are required. To address these issues, we are presenting the use of Virtual Reality (VR) [4] to control and program the robot from a distance. VR creates a virtual environment as a digital twin of the real-world environment along with the virtual objects inside it. A user may immerse in the VR environment and interact with the object there. Although there has been significant research and development in VR in the last decade [5-8], the applications within the robotics field are still not up to the industrial standards.

VR provides a cost-effective and safe method to test and verify various planning scenarios before their deployment. In this work, we have used VR to control a real-world robot by interacting with a virtual robot in a virtual environment. The technique employs motion tracking of a hand-held controller, which is used to grab the end effector of the robot and guide it along the desired path. This process is also known as Programming by Demonstration (PbD). This approach ensures safety and eliminates the requirement of a skilled programmer to update path and task planning. The new planned path or task can be observed offline and if it meets the requirements, it can be uploaded to the robot for execution in the real world.

The rest of the paper is organized as follows: In Section 2, the detailed methodology along with the required hardware and software is discussed. Section 3 describes the forward and inverse kinematics of the two robots that we used in the experimentation. Section 4 presents the detail of different working modes of our system, followed by Section 5 which describes the results. Section 6 discusses discusses the conclusion and possible future development.

2. Methodology

This section describes the detail of hardware and software tools used in our work and their integration and interaction mechanism.

2.1 VR Headset

The hardware we used for the VR experience is Oculus Rift which renders the headset's positional tracking system with near-zero latency along with two hand-held controllers that simulate the hand gestures of the user and two cameras as a motion tracking system. The Oculus Rift headset uses an OLED panel for both eyes, each having a resolution of 1080×1200 pixels at 90 Hz, and has lenses with a wide field of view. The Rift has a full 6 degrees of freedom rotational and positional tracking. A standard desktop computer with NVIDIA GeForce GTX 1050 graphics card was used for the computation of the robot's kinematics. To ensure safety and workspace awareness for the user, we included a live camera feed from the real world into the virtual workspace. We have used Unity3D, a multi-platform game development environment, as the main platform to integrate VR headset with a virtual environment and interact with the robot inside that. The user's hand motion and gestures are well detected by integrating the motion capture system, Oculus Rift, hand controllers, and Unity3D as shown in Figure 1.

2.2 Robotic Manipulator

We have tested our work on two different robots. The first one is an industrial-grade 3-DOF SCARA (Serpent-I) and will be referred to as Robot-I for the rest of the paper. The movements of SCARA are simple but entirely adequate for a vast number of assembly and pick-and-place applications. The second robot we used is a 5-DOF manipulator and will be referred to as Robot-II. This robot also has an articulated configuration with a spherical wrist and it uses six servo motors covering 180° each. The end effector of the robot can be used to clamp or pick and place objects with ease. The servo motors (MG996R) used in the revolute joints possess metallic gears for maximum torque requirements and can provide a torque of 9.4 kg-cm at 4.8 V and 11 kg-cm at 6.0 V. We used MATLAB to run several simulations of the robot's dynamics to verify the inverse kinematics equations. The robot is shown in Figure 2.

2.3 3D Model of Robot-I and Robot-II

The robot models were developed in SolidWorks and then were imported into Unity3D using SolidWorks Visualizer. Then we used hinge joints in Unity3D to reassemble the robot model. The joints were placed at carefully calculated positions in Uinty3D so that the specific motors can rotate accurately with high precision. The two models inside Unity3D are shown in Figure 3.

Fig. 1: Human Hand Gestures vs Hand Controller Simulations

2.4 Virtual Environment and GUI

To interact with the robot, it is important to give a realistic experience of the surrounding environment when a user wears the VR headset. For that, we created a lab environment with different robots and a workstation. The user can interact with the virtual environment using hand controllers and gaze. The GUI designed is simple enough to make things easier for the user to understand and verstalie enough to include all the features that we have developed. It is shown in figure 4.

2.5 Interaction Method

The method described here is aimed to make indirect virtual robot programming just as intuitive as real robot movements by gripping and guiding the robot. The interaction experience should be as intuitive and easy to learn as possible. One natural way for a human to select and move an object is through grasping it by hand. Considering this

intuitive mechanism, we have used a method that is referred to as Dummy Method. We have a dummy tip embedded at the end effector and a dummy target. When the user grabs the dummy target through the controller and moves it, the dummy tip will follow it. The concept is depicted in Figure 5.

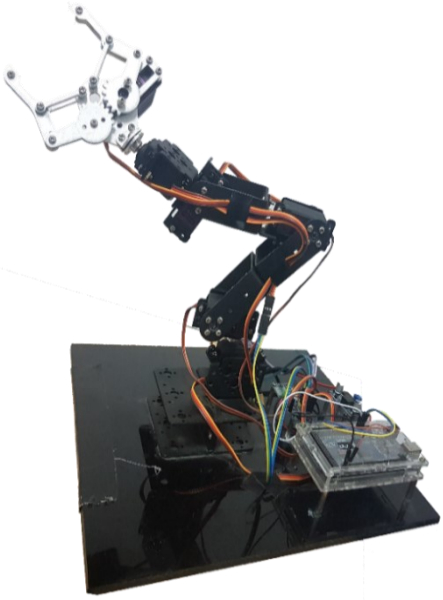


Fig. 2: Robot-II

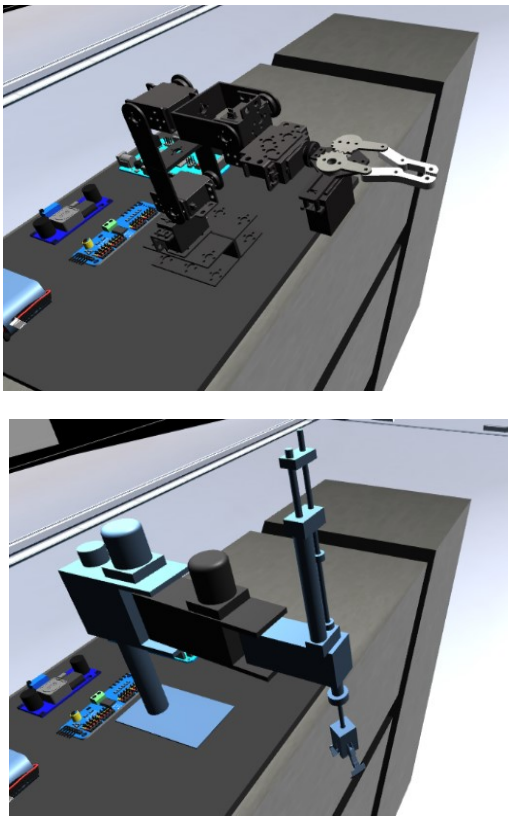


Fig. 3: 3D Models of Robots a) Robot-I b) Robot-II

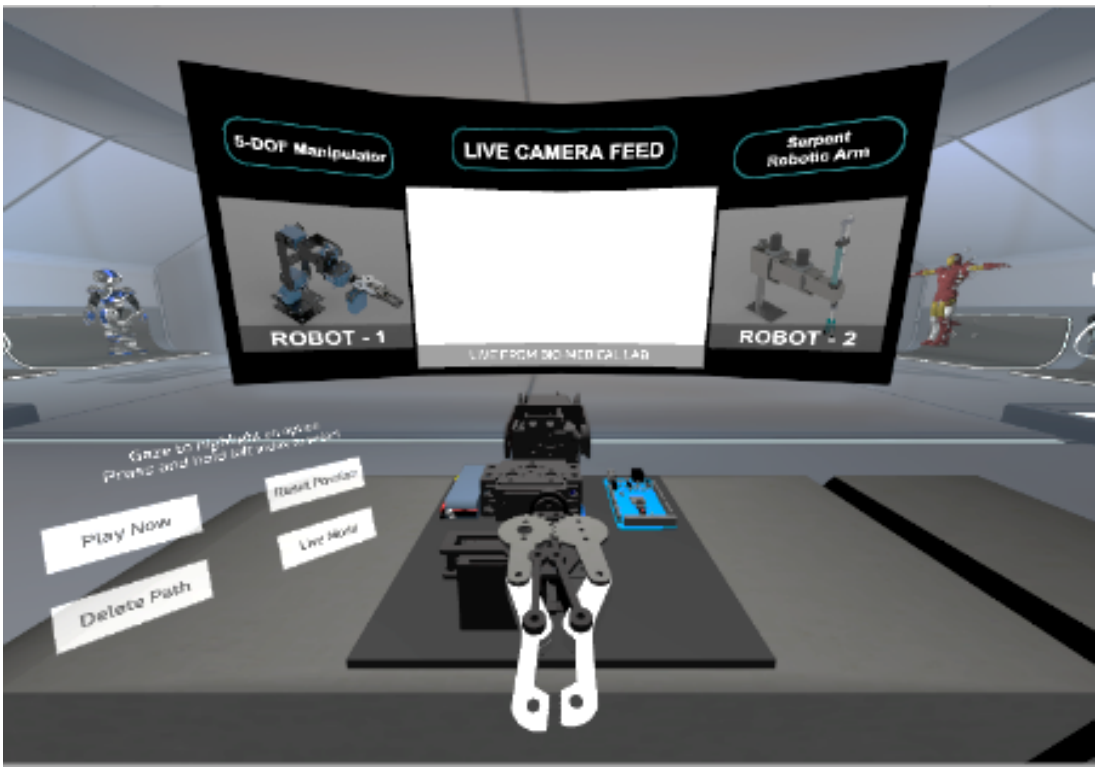


Fig. 4: Workstation in VR Environment with GUI

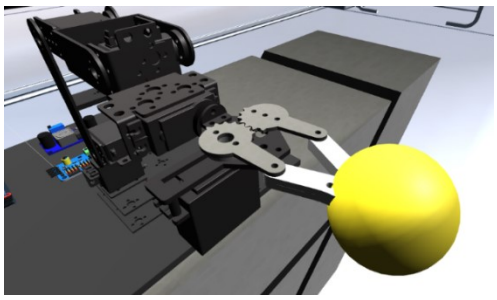


Fig. 5: Concept of Dummy Tip

The movement of the hand controller is tracked inside the virtual environment and its collision with the interactable object is checked just like a human touch to the physical robot. When the controller is colliding with the invisible sphere placed at the end effector and the fingers grasp it, the sphere gets coupled to the controller and will start following it, generating the movement of the manipulator model and its associated links using the physics engine of Unity3D. By releasing the fingers, the sphere gets decoupled. The sphere is not rigidly attached to the model and instead, there is a spring joint giving the user a natural feel. Angles at each configuration are calculated (detailed in Section 3) and sent to the microcontroller via serial communication and it moves the real robot in the same way. The buttons on the controller are configured to open and close the jaws of the end effector and the information is sent to the hardware the same way as described above. The sequence for the workflow is shown in Figure 6.

3. Robot Kinematics

3.1 Forward and Inverse Kinematics of Robot-I

The forward and inverse kinematics of the Robot-I can be found in [9] using the configuration shown in Figure 7.

The coordinates of the point B are given as:

$$x_b = r_1 \cos(\theta_1) + r_2 \cos(\theta_1 + \theta_2) \quad (1)$$

$$y_b = r_1 \sin(\theta_1) + r_2 \sin(\theta_1 + \theta_2) \quad (2)$$

The joint angles can be calculated as:

$$\cos(\theta_2) = \frac{(x_b^2 + y_b^2) - (r_1^2 + r_2^2)}{2 r_1 r_2} \quad (3)$$

$$\tan(\theta_1) = \frac{-(r_2 \sin\theta_2)\, x_b + (r_1 + r_2 \cos\theta_2)\, y_b}{(r_2 \sin\theta_2)\, y_b + (r_1 + r_2 \cos\theta_2)\, x_b} \quad (4)$$

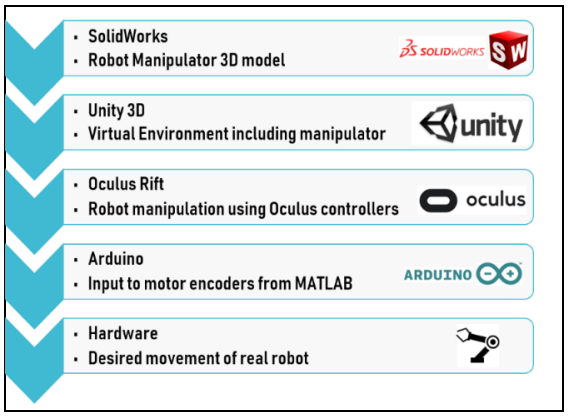


Fig. 6: Sequence of the Work

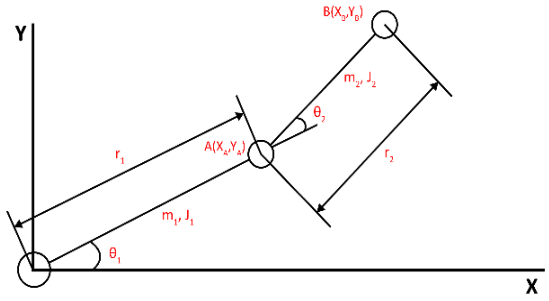


Fig. 7: Serpent-I Configuration

The forward and inverse kinematics of the Robot-II is given in the following subsections.

3.2 Forward Kinematics of Robot-II

Robot-II is of 5-DOF and has an articulated configuration. To find its forward kinematics equations, we assigned coordinate frames at its joints as shown in Figure 8. The D-H parameters of the robot are given in Table 1.

Using the D-H table, homogeneous matrices and the transformation matrices for the transformation of the third link with respect to the base frame and the transformation of the end effector frame with respect to the base frame are calculated as given in Eq (5-6).

Table 1: D-H Table of link parameters of Robot-II

Links	Link length (a_i)	Link twist (d_i)	Link offset (α_i)	Joint angle (θ_i)
1	a_1	d_1	90°	θ_1
2	a_2	0	0	θ_2
3	a_3	0	0	θ_3
4	0	0	90°	θ_4
5	0	d_5	0	θ_5

The x, y and z positions of the end effector frame correspond to the last column of the $T_{(0-5)}$ matrix and are used in inverse kinematics equations to find the joint angles.

3.3 Inverse Kinematics of Robot-II

For inverse kinematics, we used the principle of kinematic decoupling which will allow us to consider the position and orientation of the end effector frames independently. We used the analytical approach and calculated the joint variables of the robot from the position and orientation of the end effector frame as presented in Eq (7-10).

Where r_{13}, r_{23}, r_{33}, are the elements of the third column of rotation matrix R_5^3. Moreover, θ_5 was not calculated because its movement was made through the Rift controller.

3.4 Verification and Calculation of Robot's Workspace

To test the equations, we used MATLAB's RVC tool and created the DH table in which we inserted the joint angles and link lengths, and MATLAB provided the position and the orientation as a transformation matrix. With those position and orientation values, the equations (7-10) computed the same joint angle. The simulation results can be seen in Figure 9.

3.5 Robot Path Algorithm

To make the robot's end effector frame reach the position and orientation provided in VR, we developed an algorithm that calculates all possible solutions and gives the joint angles to reach the specific position and orientation. The flow chart of the algorithm is shown in Figure 10.

4. Robot Operation Modes in VR

We can interact with the robot in three modes, described below:

i. **Live Mode:** It can also be called as online mode. It allows the user to control the robot in real-time to perform a specific task. This mode is also used for the calibration of the real robot with the virtual one. Thus, whatever movements the user makes on the virtual robot are precisely followed by the real robot in real-time, with almost near-zero latency. The delay was even less than a second. We calculated it by conducting several experiments and taking the mean of the observed readings. It was never more than 1 second and 80% of the readings were below 0.5 seconds.

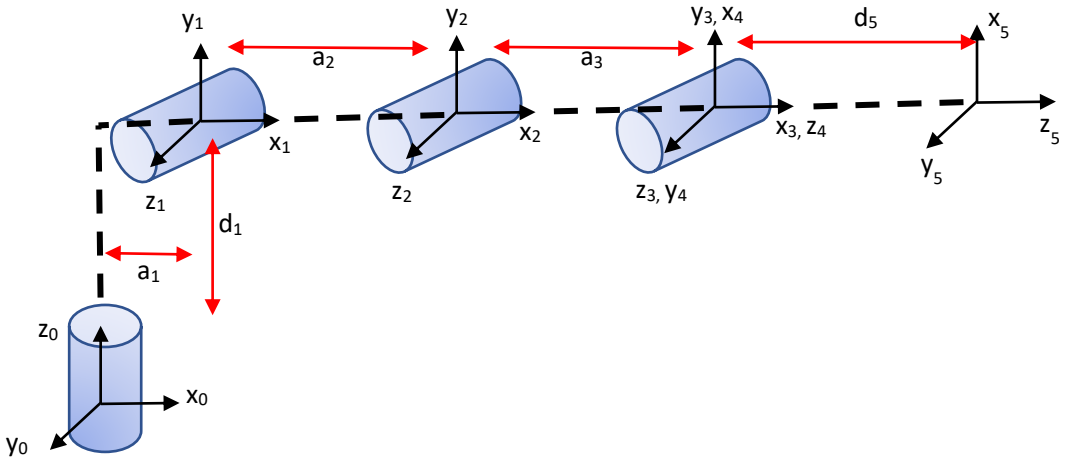


Fig. 8: Articulated Configuration of Robot-II

$$T(0-3)=\begin{bmatrix} c\theta_1 c\theta_2 c\theta_3 - c\theta_1 s\theta_2 s\theta_3 & -c\theta_1 c\theta_2 s\theta_3 - c\theta_1 s\theta_2 c\theta_3 & s\theta_1 & a_3 c\theta_1 c(\theta_2+\theta_3)+a_2 c\theta_1 c\theta_2+a_1 c\theta_1 \\ s\theta_1 c(\theta_2+\theta_3) & -s\theta_1 s(\theta_2+\theta_3) & -c\theta_1 & a_3 s\theta_1 c(\theta_2+\theta_3)+a_2 c\theta_1 c\theta_2+a_1 c\theta_1 \\ s(\theta_2+\theta_3) & c(\theta_2+\theta_3) & 0 & a_3 s(\theta_2+\theta_3)+a_2 s\theta_2+d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (5)$$

$$T(3-5)=\begin{bmatrix} \mathrm{c}(\theta_4+\theta_5) & 0 & \mathrm{c}(\theta_4+\theta_5) & 0 \\ \mathrm{s}(\theta_4+\theta_5) & 0 & \mathrm{c}(\theta_4-\theta_5) & 0 \\ 0 & 1 & 0 & d_5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6}$$

$$\theta_1=\tan^{-1}\left(\frac{y_c}{x_c}\right) \tag{7}$$

$$\theta_2=\cos^{-1}\left(\frac{(x_c-a_1\cos\theta_1)(a_3\cos\theta_3+a_2)+a_3\cos\theta_1\sin\theta_3(z_c-d_1)}{(a_3\cos\theta_1\cos\theta_3+a_2\cos\theta_1)(a_3\cos\theta_3+a_2)+a_3\cos\theta_1\sin\theta_3 a_3\sin\theta_3}\right) \tag{8}$$

$$\theta_3=\cos^{-1}\left(\frac{\left(\sqrt{x^2+y^2}-a_1\right)^2+(z_c-d_1)^2-a_3^2-a_2^2}{2a_2a_3}\right) \tag{9}$$

$$\theta_4=\sin^{-1}\left(r_{13}\cos\theta_1\cos(\theta_2+\theta_3)+r_{23}\sin\theta_1\cos(\theta_2+\theta_3)+r_{33}\sin(\theta_2+\theta_3)\right) \tag{10}$$

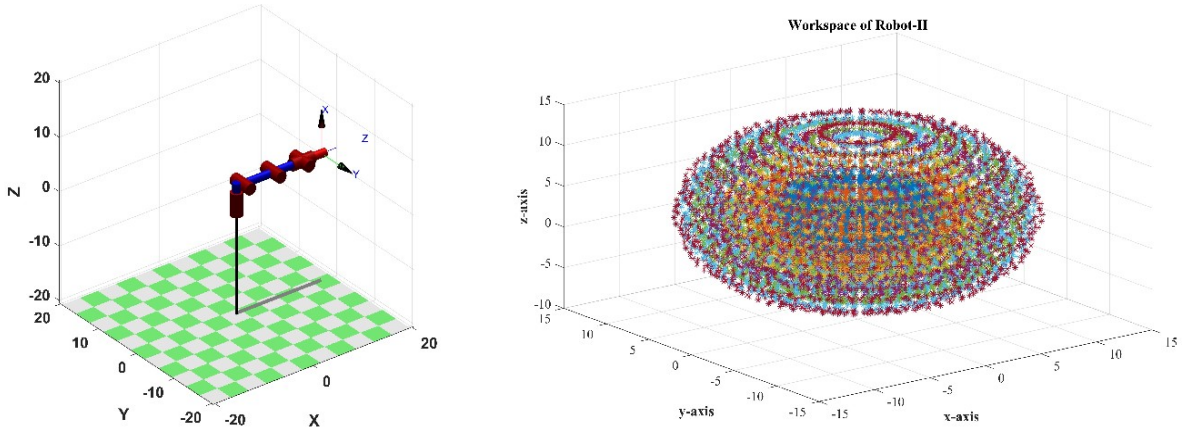


Fig. 9: Robt Model in RVC Toolbox (left) and Robot Workspace (right)

ii. **Teach Mode:** It can also be called as offline mode. It works similar to the live mode but this mode gives an additional option to store the programmed path and use it later for the repetitive task. The program stores the joints angles after every 0.0167 seconds and thus generates a matrix of all joint angles. This matrix is then fed back to the program and all the joints' angle values are incremented every 0.0167 seconds thus allowing the robot to follow and repeat the last task, which generally is the requirement for industrial robots.

iii. **Inverse Kinematics Mode:** With regular human control in live mode and teach mode, it becomes a bit difficult to achieve a straight-line path between two points since it requires good hand-eye coordination and a tremor-free hand movement. In most cases, it is not possible to support the hand in mid-air. This makes it difficult to hold the hand steadily. So, for easy operation, we provided the option of picking up any point in the 3D workspace of the robot, and the end effector moves to that point from its initial position, in a straight line. This shortest path between two points can also be taught to the robot for later use.

5. Results

The virtual and the real robots must collaborate to perform some tasks. We selected two tasks for Robot-I and one task for Robot-II. These tasks were decided based on the relative

applications in the manufacturing, automotive and packaging industries, and are described below.

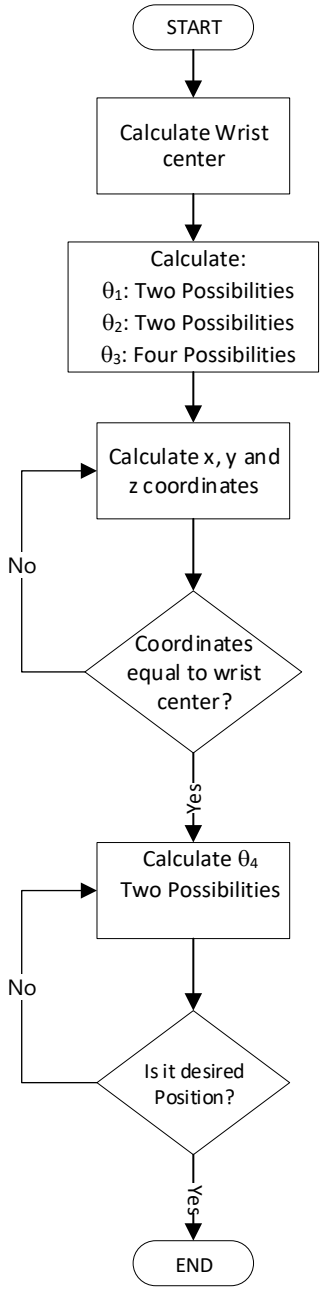


Fig. 10: Robot Path Plan Algorithm

5.1 Pick and Place Task on Robot-II

The large bulk of robotic manipulators in industries perform the tedious activity of picking and placing objects from one place to another. To demonstrate the same application we made a square box as a workpiece and a square cavity inside which the robot will place the workpiece. These can be seen in Figure 11. The task is to be performed by an operator wearing the VR headset who can see the real robot through a camera in the VR environment. To complete this task, the user must pick the workpiece from the table and place it inside the cavity. The opening and closing of the robot's end effector are done through the buttons in the hand controller. The user can also change the orientation of the workpiece.

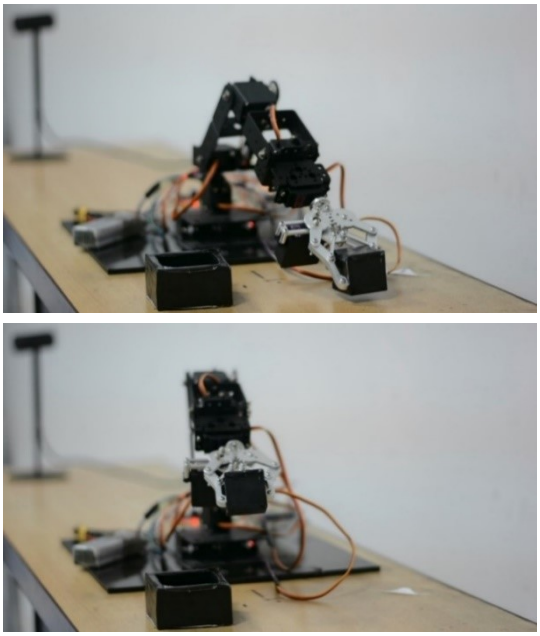


Fig. 11: Pick and Place Task on Robot-II

5.2 Drawing Task on Robot-II

The second task was drawing some random patterns in the virtual environment. To perform this task, the robot's end effector was gripped with a marker. This task verifies that the system can perform relatively complex tasks for industrial

The user would grab the virtual robot and move it in the workspace and draw any random pattern on the paper placed on the table. To get an idea of the accuracy of the repeated path of the robot, we taught the robot using the teach mode and repeated the movement multiple times. This repeated drawing was compared with the original drawing points and the results are shown in Figure 12 and Figure 13.

5.3 Drawing Task on Robot-I

A similar drawing task was performed on Robot-I and for better comparison, we stored the angles (θ_1 and θ_2) generated in VR and that of the real robot. The time delay between the virtual robot and the Robot-1 is about 0.55 sec which is quite nominal for most of the applications. Moreover, after ignoring this time delay, the root mean square error (RMSE) of the angles of the real robot and the virtual robot is 2.61 degrees for θ_1 and 3.08 degrees for θ_2. The comparison of both angles is shown in figure 14.

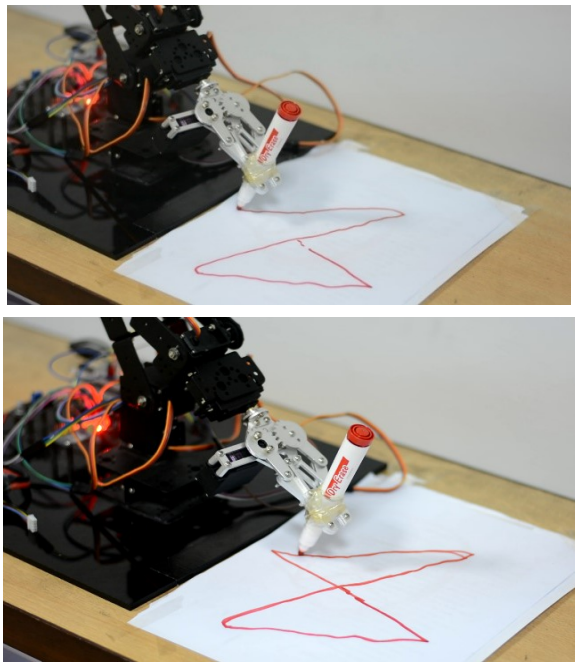


Fig. 12: Drawing Task Assisted by Human in VR

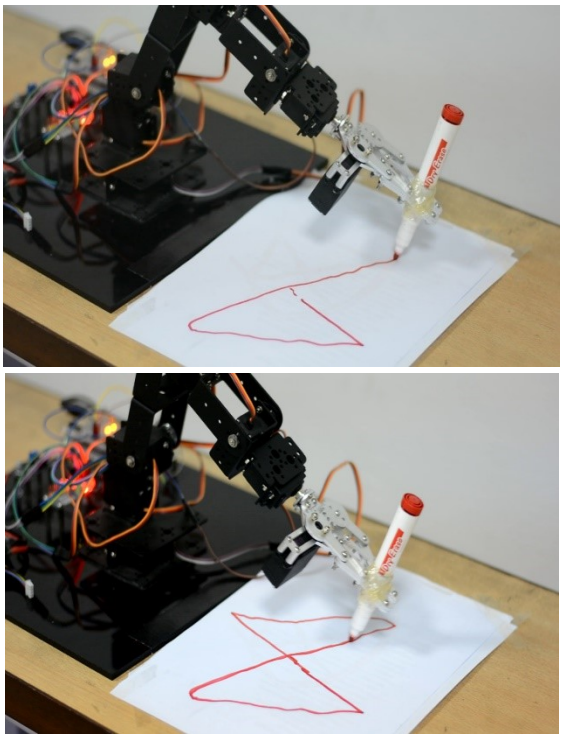
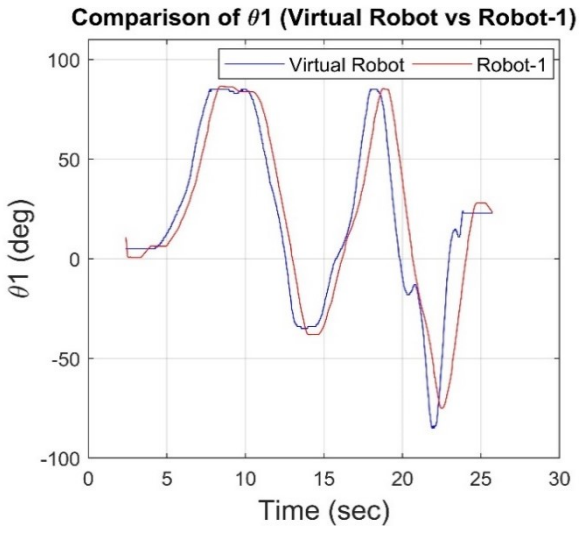


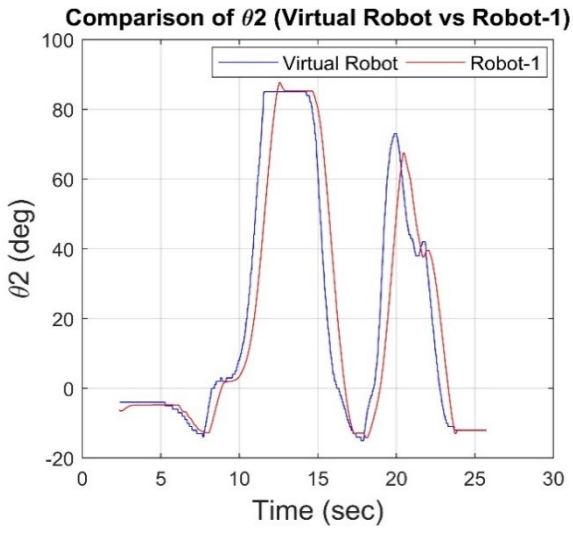
Fig. 13: Drawing Task performed by Robot 1

6. Conclusion and Future Work

We have investigated the effectiveness of using Virtual Reality for improved human-robot collaboration. Accurate results are achieved but the user requires a few test trials before he can comfortably interact with the system. Such technologies can be used in industry to train the staff by creating complex scenarios in the virtual environment [10]. Moreover, before deploying any complex scheme in the industry and transferring that to the system, it can be first verified in a virtual environment. To further improve the completion of the task, the optimization of the trained path plan can also be included. Incorporating Augmented Reality (AR) along with VR can further improve the collaboration, where a layer is added on top of the user's field of vision. Furthermore, indications about malfunctioning and machine faults can be augmented to facilitate the user and save the repair time.

In summary, the interaction scheme presented in this paper has the potential to reduce the cost and save time. This opens new horizons for safer and efficient human-robot interaction. Considering the increasing demands of robotic operations in industry and the implementation of Industry 4.0, incorporating such technologies is of dire need.





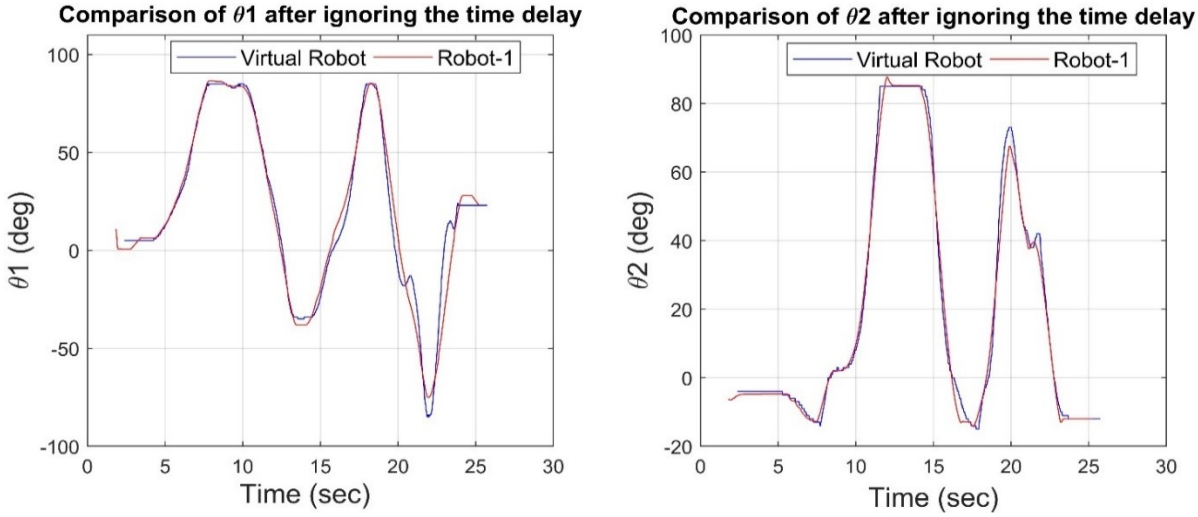


Fig. 14: Comparison of the Virtual and Real Robot angles

7. References

[1] Jerčić, P., Hagelbäck, J., & Lindley, C. (2019). An affective serious game for collaboration between humans and robots. *Entertainment Computing*, *32*, 100319.

[2] Villani, V., Pini, F., Leali, F., & Secchi, C. (2018). Survey on human–robot collaboration in industrial settings: Safety, intuitive interfaces and applications. *Mechatronics*, *55*, 248-266.

[3] Oyekan, J. O., Hutabarat, W., Tiwari, A., Grech, R., Aung, M. H., Mariani, M. P., & Dupuis, C. (2019). The effectiveness of virtual environments in developing collaborative strategies between industrial robots and humans. Robotics and Computer-Integrated Manufacturing, 55, 41-54.

[4] El Beheiry, M., Doutreligne, S., Caporal, C., Ostertag, C., Dahan, M., & Masson, J. B. (2019). Virtual reality: beyond visualization. Journal of molecular biology, 431(7), 1315-1321.

[5] Ohtani, H., Tamura, Y., Kageyama, A., & Ishiguro, S. (2011). Scientific visualization of plasma simulation results and device data in virtual-reality space. IEEE Transactions on plasma Science, 39(11), 2472-2473.

[6] Li, J., Liu, Q., & Su, H. (2013). Virtual reality method of portal slewing crane based on WPF. Advances in Mechanical Engineering, 5, 320757.

[7] Xu, X., Wang, L., Li, Z., Yao, S., & Fang, X. (2018). Modeling of submarine initial pipe-laying process and its real-time semi-physical virtual reality system. Advances in Mechanical Engineering, 10(1), 1687814017747734.

[8] Usher, W., Klacansky, P., Federer, F., Bremer, P. T., Knoll, A., Yarch, J., & Pascucci, V. (2017). A virtual reality visualization tool for neuron tracing. IEEE transactions on visualization and computer graphics, 24 (1), 994-1003.

[9] Das, M. T., & Dülger, L. C. (2005). Mathematical modelling, simulation and experimental verification of a scara robot. Simulation Modelling Practice and Theory, 13 (3), 257-271.

[10] Wolfartsberger, J. (2019). Analyzing the potential of Virtual Reality for engineering design review. Automation in Construction, 104, 27-37.