Mixed Reality Applied to the Teleoperation of a 7-DOF Manipulator in Rescue Missions

Percy W. Lovon-Ramos, Roger Ripas-Mamani, Yessica Rosas-Cuevas, Maria Tejada-Begazo,
Renato Marroquin Mogrovejo and Dennis Barrios-Aranibar
Research Group in Line of Industrial Automation, Robotics and Computer Vision (LARVIC)
Research Center in Computer Science
Universidad Católica San Pablo
Arequipa, Perú
{pwlovon, rdripas,yrosas, maria.tejada.begazo, renato.marroquin, dbarrios}@ucsp.edu.pe

***Abstract*—The need for complementing robot's autonomy behaviour with human reasoning has made robot teleoperation a research topic for more than fifty years. And while most of the research work usually deploys a master-slave architecture for controlling robots (whether using a joystick and/or desktop computer), recently Mixed Reality has come into play. This is because it is able to improve the teleoperator's view with less bandwidth consumption than traditional teleoperation architectures.**

In this paper a new approach for teleoperating a manipulator is presented. The virtual representation of the manipulator is made by using encoder sensors, and also, a camera image is displayed through an Android interface to complement the teleoperator's view.

We can show with our results that the real positions of the manipulator and the virtual model are closed to be the same with some differences due to the transmission time delay which is minimum.

Keywords*—teleoperation, human-robot interaction, mixed reality

I. INTRODUCTION

Since the beginning of robotics, teleoperation has been an important research topic. It allows human operators to work remotely with robots in several different environments and even in hazardous ones. Commonly, there is an operator located within a remote workstation. Thus robot teleoperation has many applications such as remote operation, teleoperation of prostheses, manufacturing, marine and space exploration, minimal medical surgeries, handling radioactive materials, telepresence, rescue missions and many others [1].

A very important application is related to victims search when natural disasters have occurred, such as earthquakes, and also when humans are in dangerous situations [2]. The search and rescue task is very complex due to its necessity to collect information from the possible victims, but also from the environment where they are. Thus, having a teleoperated robot to search for victims in these possibly dangerous places is much safer than sending a human being to put his life at risk.

Teleoperation has some clear disadvantages though such as data not being secure when sent through the Internet, being network bounded when transferring big files (e.g. video, images, sound, or others) which might cause a communication delay. In addition to that, other disadvantages are related to handling lose or incomplete data, training the operator, operator's fatigue [7], and others. But most importantly, the fact that in most cases a teleoperated robot will not be able to adapt to new circumstances [8] is the biggest concern.

Virtual Reality and Mixed Reality are recent approaches to deal with these problems with different approaches. For example, reducing of time delay by sending only relevant data, improving the operator's view with representation of 3D models made with real world data [9], and combining 3D representations and real camera views to allow the operator to have an egocentric and exocentric view of the context at the same time.

In this paper, a teleoperation approach with mixed reality is presented. This system is placed on an Android Tablet with a virtual model of the manipulator. This is operated using a touch screen which allows the operator to work directly on the moving joint. Also the operator has the view of the embedded camera which helps him to make decisions in real time on how to control the robot. The paper is organized as follows: Section II describes current techniques for teleoperation. In section III, the structure of our teleoperation process is explained, and also, the interaction between the device and the server is detailed; Section IV shows the modeling of the real and virtual manipulator. Experiments and results are shown in section V, and finally in section VI, conclusions and future work are described.

II. RELATED WORK

Teleoperation is the extension of a person's sensing and manipulation capability to a remote location. A teleoperation system includes at the minimum artificial sensors, arms and hands, a vehicle for carrying these, and the communication channels to and from the human operator [10]. Since the 1940s, teleoperation research has been undertaken when researchers built a robotic arm to replace human arms as prostheses. During all this time different kind of approaches has been taken, using different interfaces such as web platforms, mobile devices, minicomputers, and others. These interfaces combine different representations of the actual robot and its context. In the next paragraphs some teleoperation approaches are briefly explained.

In the early days, teleoperation approaches were developed in order to make intelligent interfaces which complement user

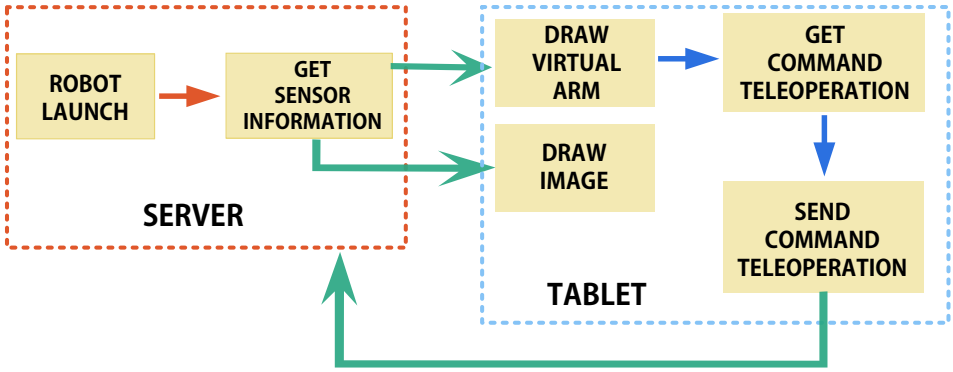


Fig. 1. System Overview.

inputs for controlling robots. Approaches such as making a Virtual Robot which represents and act as the real robot, robotic arms represented using a file RAFF (Robot Arm File Format) which was inspired in a tree structure [11]. One of the attempts to make an assisted interaction with the robot was developed using a VIA-SFX architecture in which the operator has an intelligent assistant that handles faulty scan jobs like losing data due to networks limitations or lose the perception data of one or more sensor [12].

On the other hand Mixed Reality has also been used as a support in Desktop Teleoperation. One of the first published research was ARGOS (Augmented Reality through Graphical Overlays on Stereovideo) which worked with stereo-vision to construct 3D models [13]. This approach was used also in rescue missions [14]. Other way to reconstruct the 3D environment is using Laser Range Finder [15] or using LIDAR data to reconstruct the 3D environment and color camera to get the color [16].

In the recent years, the usage of web architectures and virtual reality have provided the advantage to have multiple users, to train people with expensive equipment [17], to reduce the delay time in space teleoperations by making representations for the world [18], to teach students from other universities projects in robotics [19]. Other approaches use physical arm representations called master slave. In these cases the operator use the sensors of the master arm to teleoperate a slave arm [20]. Moreover different devices such as joysticks have been be used to perform repairing tasks in space satellites [9].

In all of those cases, the teleoperation always uses cameras to make representations of the world, but that increases the usage of bandwidth, therefore the delay time. In our proposal we use encoders that indicate the degree of rotation of the motor following the same approach as Turpel [20] but with different sensors. In addition to that, we make a representation on a Android tablet for teleoperation. Furthermore, to improve the task of teleoperation, we grant the operator a view of the embedded camera.

III. TELEOPERATION

A. System Overview

In this subsection our method for performing teleoperation will be described. Figure 1 shows the different components of our system: The part on the right hand side is the server (desktop computer) where the manipulator is connected to the server using a USB port; The left-hand side belongs to the Tablet where the actual teleoperation is carried on.

B. Server

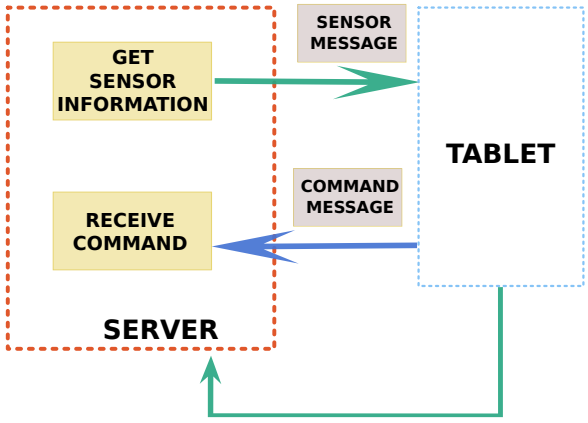


Fig. 2. Server Diagram

Figure 2 shows a diagram of the server where the manipulator is connected. It is working with a 7 DOF manipulator which operates under the Robot Operating System (ROS) framework. This framework allows our work to be used with any type of manipulator supported by ROS.

Every motor of the manipulator has an encoder which gives us their exact position. This data is used to construct the virtual model that will be explained in section IV.

ROSJAVA package was used,to extract the information and to move the motors. This package is also used to extract information from the camera embedded in the last joint of the manipulator.

Java Sockets are used to establish communication between the server and the client. Also, two different process are shown in the figure 2, they are in charge of retrieving the information and sending commands, but a more detail description is given in the following subsections.

1) Get Sensor Information: In the first process perform as a client, this client connects to the tablet and always sends information (Sensor Message) from the arm sensors and the embedded image of the camera.

2) Receive Command: The second process acts as a server. The Android client is always trying to send commands in order to move the manipulator by message passing (Command Message). Both processes are launched simultaneously because the image needs to be always showing the latest available information in order to keep the virtual world (Sensor Message) updated. Also because we need to get the teleoperation commands at all the times. This task is performed using Java primitives for parallelism (Thread Class).

C. Android Client

Figure 1 shows the entire process in the Android client. This process will be described step by step, messages used in Android Client are the same as in Server (figure 2).

1) Draw Virtual Arm: In this process a connection via sockets is created for sending the virtual model created, and also for sending a *Sensor Message*. This message contains the positions sent by the server, thus this is the first process that is performed when launching the virtual manipulator.
The *Sensor Message* contains a message which is in different units (radians) than virtual manipulator, which unit is in angles in degrees). Then a transformation from encoder values to the hexadecimal degrees is performed. The transformation function used is the following:

$$F(e) = \begin{cases} \mid e * 60 \mid & \text{if } -3,0 < e < 0, \quad (1a) \\ 90 - e * 60 & \text{if } 0 < e < 3,0 \quad (1b) \end{cases}$$

Where "*e*" is the encoder value obtained from server (*Sensor Message*) and "*F(e)*" the angle to be sent to the Unity virtualization.

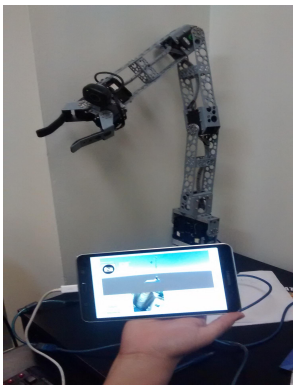
2) Draw Image: This process is in charge of drawing within the Android view the RGB Matrix *Sensor Message*, and every time this message is received the image is redrawn in the view. It should be noted that this process as all of other client processes should be launched in parallel. Other processes use the predefined Java threads, but this redrawing thread especially uses Android parallelism, because it is used to modify graphical components of the application.

3) Get Command Teleoperation : Each time a change in any of the joints within the virtualization is done, a command is received in the Android application. This command is sent to the server via Sockets and the message *Command Message*. An important difference between *Sensor Message* and *Command Message* is that former one contains the RGB image, so it is heavier than the other. The size of the image we use 240x320 pixels to get a transmission that it is continuous.

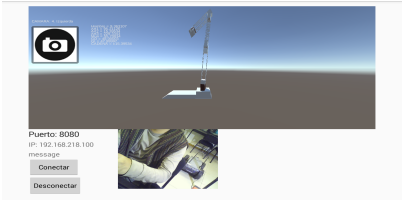
4) Send Command Teleoperation: The virtual representation every time a change occurs, sends a *Command Message* to Android and this is retransmitted to the server, which performs the movement of the manipulator.

D. Teleoperation Interface

In this section, the final interface is shown. Figure 3 (a) is an image of the tablet with the manipulator. And part (b) of the image is the interface took from tablet.



(a)



(b)

Fig. 3. Teleoperation Interface

IV. VIRTUAL MANIPULATOR

The virtual manipulator was developed using the Unity3D game engine. This tool allows the development of applications for several platforms like mobiles devices, PCs, Web, and many other platforms. Therefore, this virtual model could be used in all of those platforms.

A. Geometric Model

The manipulator which was used in this paper, is a robotic arm with 80 cm long, with capacity of payload of 0.4Kg, aditionally it has a web camera. The manipulator has seven rotational joints with Dynamixel motors. The top four rotational joints are MX-106R model and the remaining are MX-28R. Because of weight distribution, the shoulder joint uses two motors (master-slave).
For the modeling of the manipulator arm, Denavit Hartenberg (D-H) representation was used. We assign the base as a reference system and then start getting the referential systems of the posterior joints of the kinematic chain. This notation needs to first have the length of the links of the joints. The parameter θ refers to the angle of the joint that moves on the axis z and the parameter α to the angle of the joint that moves in the axis x. Having into account these observations, the parameters shown in the table I are proposed. In addition, we had to make changes in some parameters for the structure of the manipulator arm (Fig.4).

B. Design of the Virtual Model

The model manipulator was embedded into an Android application, which is responsible for the data exchange between the 3D model developed in Unity and the readings recorded by ROS (Robotic Operating System). We took as reference for the simulated model the 7 degrees of freedom that the real manipulator has. Moreover, we took into consideration

TABLE I. DENAVIT HARTENBERG PARAMETERS FOR THE KOMODO'S ARM

Joint	a (cm)	α (degrees)	d (cm)	θ (degrees)
1	0	0	$L_0 + L_1$	θ_1
2	0	90	0	$90 + \theta_2$
3	L_2	90	0	$180 + \theta_3$
4	$-L_3$	-90	0	$\theta_4 - 90$
5	0	90	$L_4 + L_6$	$\theta_5 - 90$
6	L_5	90	0	$90 + \theta_6$
7	$-L_5$	-90	0	$\theta_7 - 90$

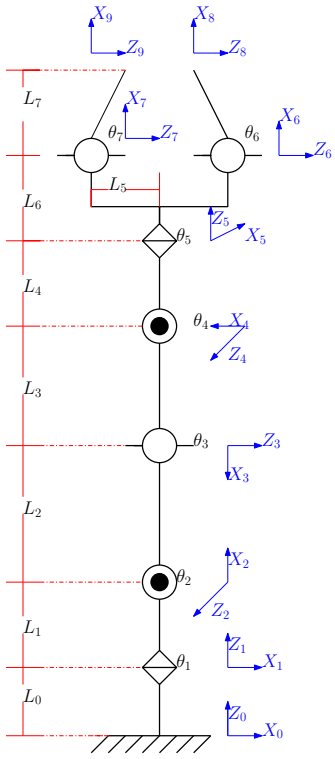


Fig. 4. D-H Notation for the Komodo's arm

the proportional relationship of each component's size (figure 5). The assembly of all components can be seen in figure 6.

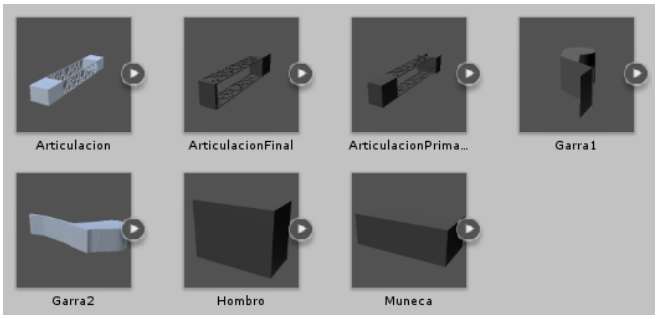


Fig. 5. Components of Manipulator Arm

After assembling all components in the correct location, the rotations of all joints were tested according to their respective rotation axis. It is worth noting that joints have an independent rotational movement and also a rotational movement associated with the joint to which they are assembled. These tests were performed manually by inputting the necessary commands for

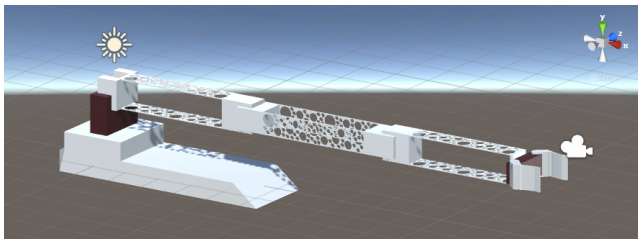


Fig. 6. Assembly Manipulator

handling each joint (Fig 7).

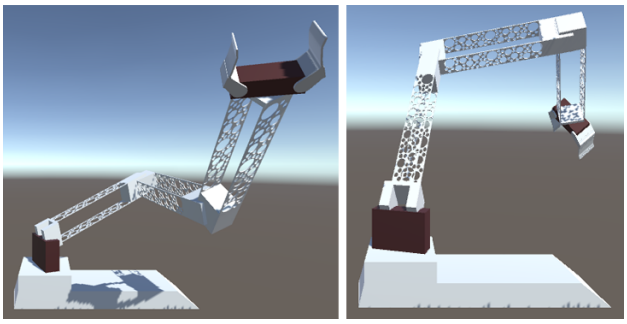


Fig. 7. Examples of the joint rotations made by PC

The manipulator joints are rotational. This means that each joint revolves around an axis which is located in the previous joint. These rotations can be of the following types: Yaw type (rotation in the Z axis), Pitch type (rotation on the Y axis) or Roll type (rotation on the X axis). The table II details the rotations of each joint.

TABLE II. AXIS OF ROTATION OF EACH JOINT

Joint	Axis of rotation
Base	Axis Z (Yaw)
Shoulder	Axis Y (Pitch)
Elbow 1	Axis Z (Yaw)
Elbow 2	Axis Y (Pitch)
Wrist	Axis X (Roll)
Left Claw	Axis Z (Yaw)
Right Claw	Axis Z (Yaw)

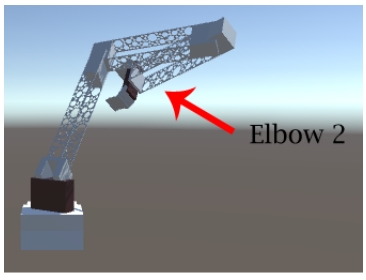


Fig. 8. Location of Elbow 2 in rear view

In order to enable virtual manipulation by dragging the joints on to the screen of mobile application, various events were programmed. This allowed us to make that a drag on a particular area of the screen would produce some joint rotation. These drag operations over the screen can be made horizontally and vertically which in the first instance (having only a view

in the application that by default is the rear view) would not allow to make turns in some joints as is the case of the joint Elbow 2 (Fig. 8). This is the reason why we implemented different views as shown in fig. 9.

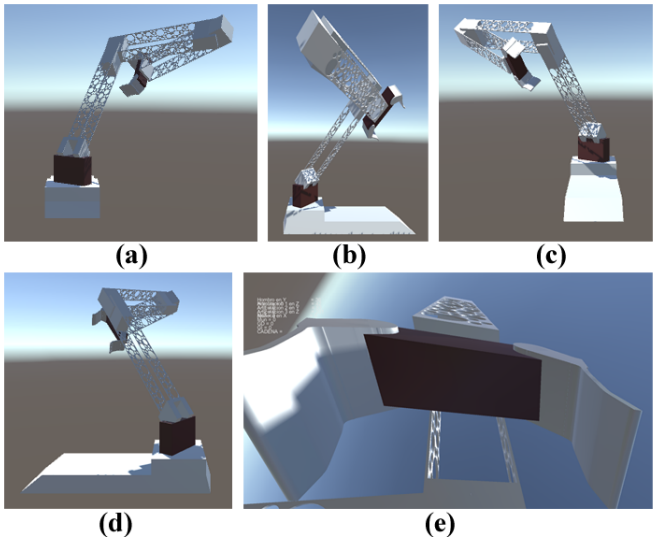


Fig. 9. Views of the virtual manipulator: (a) rear view, (b) right side view, (c) front view, (d) left side view, (e) view of the claw.

When performing these rotations, we obtain hexadecimal angles from each joint. These angles are subsequently provided to the general Android application. Finally, from the mobile application one can teleoperate the manipulator arm.

C. Information exchange into Android application

An important detail when making the application is embedding the model made in Unity within an Android application and then extract the data provided Unity (angles of each joint according to the rotation presented) and use within the application general.
The embedded Unity model works as a plugin within the Android application and to enable communication of this plugin with the general application (send data from the Unity plugin to Android) there is no specific function in Unity to do so. If you want to communicate with the general application plugin (send data from Android to the Unity plugin) there is a function called UnitySendMessage(). This is the reason why we used an internal connection via sockets, so we could send data from the Unity plugin to our Android application. The data is sent through Java sockets. This data is the angle that the joint was actually moved. To avoid saturating the internal network, only one socket is opened when a joint is moved; and to detect which joint was moved, we use an initial identifier which is concatenated in a string with the recorded angle. For example, "c36" means that the joint "Elbow 1" moved to 36 degrees in Yaw.

V. EXPERIMENTS AND RESULTS

Test were performed in order to check whether or not the manipulator arm's state is the same as of the real manipulator. In rescue operations, a common task is to pick an object and to move it to another place, this is the test we also used.

Figure 10 shows the process of moving two different types of objects, the distance was 1 meter, in the first case is picking a metal bottle with 150 gr. of weight, and in the other case was a box of 300 gr.

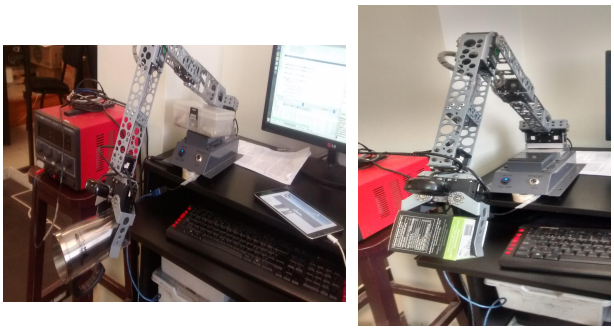
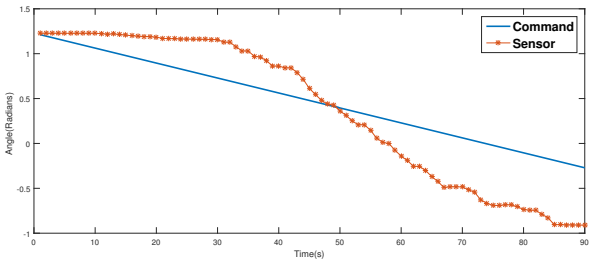


Fig. 10. Different types of objects that were picked

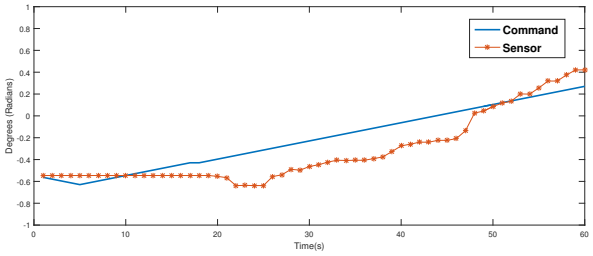
After doing the task of picking and moving, the manipulator positions by encoders and the commands positions were sent and received by the Android tablet. Then, those measurements were compared to each other to measure the accuracy of the approach. Figure 11 shows a difference between such positions.
In some cases such as Figure 11.c , the manipulator keeps its position due to some delay of the *Command Message* reaching the robot. This is because of some network delay.
In all the other cases results show that current positions are close to the virtual manipulator. This means the teleoperation is reliable because the position in the virtual model will be similar to the real manipulator. The comparisons in figure 11 show the positions during a period of time. The line tagged as *Command* is the position sent by the Android Tablet, and the line tagged as *Sensor* is the positions taken by the encoder of each motor from the manipulator.

VI. CONCLUSIONS

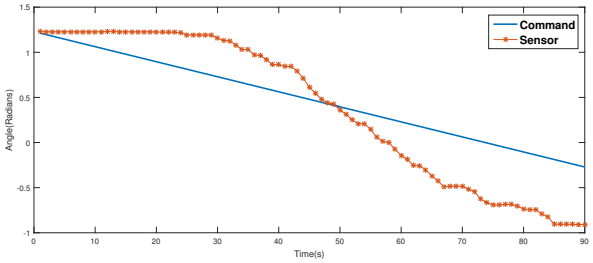
In this paper a new approach for teleoperation in rescue missions is presented. Virtual Reality is used to represent a 3D manipulator in an Android device; such representation is made with positions of the real manipulator. In order to help the operator, a camera is embedded in the last joint of the manipulator which is also shown in the Android device.
For checking the accuracy of this approach the operator had to pick and move objects. Our tests showed that positions of both representations were always very similar to each other.
This means that the virtual model and the real model were representing the same position. Although there are differences they are mainly because of the time delay when processing commands in the manipulator. This means that the operator in a rescue mission can rely in this interface, because the virtual model is always similar to the real manipulator, and the camera view can help showing a more complete view of the world around the manipulator.
In real rescue mission this approach could help the operator of a manipulator which could be embedded into a mobile robot. This is more efficient because it also takes less time transmitting positions of encoders than sending complete RGB or RGBD images.



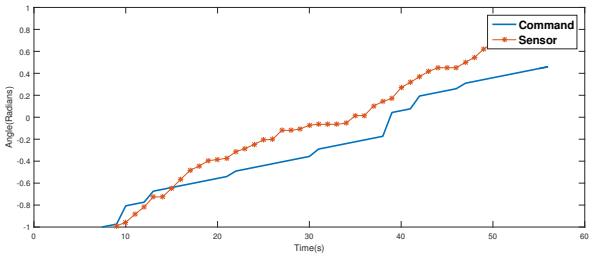
(a)



(b)



(c)



(d)

Fig. 11. **(a)** Comparison between Virtual position (command) and Real Position (sensor) of Base Joint, **(b)** Comparison of Shoulder Joint, **(c)** Comparison of Elbow 1 Joint and **(d)** Elbow 2 Joint

ACKNOWLEDGMENT

This work was supported by the *Fondo para la Innovación, Ciencia y Tecnología* (FINCyT), Perú, under the contract 216-FINCyT-IA-2013.

REFERENCES

[1] Goodrich, Michael A. and Schultz, Alan C. *Human-robot Interaction: A Survey*. ,Found. Trends Hum.-Comput. Interact.: Now Publishers Inc., 2007.

[2] Mizumoto, Hisashi and Sato, Noritaka and Kon, Kazuyuk and Mano, Hayato and Shin, Hayato and Chatterjee, Ranajit and Matsuno, Fumitosh*Flexible interface for multiple autonomous and teleoperated rescue robots*. Bangkok, Thai, 2008 IEEE International Conference on Robotics and Biomimetics, ROBIO 2008: IEEE, 2008.

[3] Yepes, Juan C. and Yepes, Juan J. and Martinez, Jose R. and Perez, Vera Z. *Implementation of an Android based teleoperation application for controlling a KUKA-KR6 robot by using sensor fusion*. Medellin, Colombia , Pan American Health Care Exchanges, PAHCE: IEEE, 2013.

[4] Elhajj, Imad and Xi, Ning and Fung, Wai Keung and Liu, Yun Hui and Li, Wen J. and Kaga, Tomoyuki and Fukuda, Toshio *Haptic information in internet-based teleoperation*. ,IEEE/ASME Transactions on Mechatronics: IEEE, 2013.

[5] Song, Weibo and Guo, Xianjiu and Jiang, Fengjiao and Yang, Song and Jiang, Guoxing and Shi, Yunfeng *Teleoperation humanoid robot control system based on kinect sensor*. ,Proceedings of the 2012 4th International Conference on Intelligent Human-Machine Systems and Cybernetics, IHMSC 2012, 2012.

[6] Weibo Song and Shuping Zhao and Fengjiao Jiang and Kaiyan Zhu and Lijie Cao and Yunfeng Shi *Teleoperation Robot Control System Based on Mindband Sensor*. ,Computational Intelligence and Design (ISCID), 2012 Fifth International Symposium on, 2012.

[7] Grange, S. and Fong, T. and Baur, C. *Effective vehicle teleoperation on the World Wide Web*. San Francisco, CA, Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065): IEEE, 2000.

[8] Malysz, Pawel and Sirouspour, Shahin *Trilateral Teleoperation Control of Kinematically Redundant Robotic Manipulators*. ,Int. J. Rob. Res.,Sage Publications, Inc, 2011.

[9] Jiang Zainan, Liu Hong, Wang Jie, Huang Jianbin *Virtual Reality-based Teleoperation with Robustness Against Modeling Errors*. , Chinese Journal of Aeronautics, Volume 22, Issue 3, June 2009, Pages 325-333, ISSN 1000-9361,

[10] Sheridan, T. B. *Telerobotics*. ,Automatica : Pergamon Press, Inc., 1989.

[11] Flueckiger, L and Piguet, Laurent and Baur, Charles *Generic robotic kinematic generator for virtual environment interfaces*. Proc. SPIE 2901, Telemanipulator and Telepresence Technologies III, 1996.

[12] Murphy, Robin R and Brawley, James P *Estimating Time Available for Sensor Fusion Exception Handling* . SPIE Sensor Fusion VIII, Volume 2589, October 1, 1995, pages 242-253

[13] P. Milgram, A. Rastogi and J. J. Grodski *v*. Robot and Human Communication, 1995. RO-MAN'95 TOKYO, Proceedings., 4th IEEE International Workshop on, Tokyo, 1995, pp. 21-29.

[14] Frauke Driewer, Markus Sauer, Klaus Schilling *MIXED REALITY FOR TELEOPERATION OF MOBILE ROBOTS IN SEARCH AND RESCUE SCENARIOS*. IFAC Proceedings Volumes, Volume 39, Issue 3, 2006, Pages 267-272, ISSN 1474-6670.

[15] Markus Sauer, Florian Zeiger, Klaus Schilling *Mixed-Reality User Interface for Mobile Robot Teleoperation in Ad-Hoc Networks*. IFAC Proceedings Volumes, Volume 43, Issue 23, 2010, Pages 77-82, ISSN 1474-6670.

[16] Alonzo Kelly, Erin Capstick, Daniel Huber, Herman Herman, Pete Rander, Randy Warner *Real-Time Photorealistic Virtualized Reality Interface for Remote Mobile Robot Control*. Robotics Research: The 14th International Symposium ISRR, 2011, Springer Berlin Heidelberg, Pages 211–226, ISSN 1474-6670.

[17] R. Safaric, S. Sinjur, B. Zalik and R. M. Parkin *Control of robot arm with virtual environment via the Internet*. Proceedings of the IEEE, vol. 91, no. 3, pp. 422-429, Mar 2003. doi: 10.1109/JPROC.2003.809205.

[18] R. Marin, P. J. Sanz and J. S. Sanchez *A very high level interface to teleoperate a robot via Web including augmented reality*. Robotics and Automation, 2002. Proceedings. ICRA '02. IEEE International Conference on, Washington, DC, 2002, pp. 2725-2730.

[19] S.H. Yang X. Zuo L. Yang *Controlling an Internet-enabled arm robot in an open control laboratory*. Assembly Automation, Vol. 24 Iss 3 pp. 280 - 288.

[20] P. Turpel, B. Xia, X. Ge, S. Mo and S. Vozar *Balance-arm tablet computer stand for robotic camera control*. 2013 8th ACM/IEEE International Conference on Human-Robot Interaction (HRI), Tokyo, 2013, pp. 241-242.