Cloud Computing Services for Real Time Bilateral Communication, Applied to Robotic Arms

Cristian Gallardo[1] and Víctor Hugo Andaluz[1,2](✉)

[1] Universidad Técnica de Ambato, Ambato, Ecuador
cmgallardop@gmail.com
[2] Universidad de Las Fuerzas Armadas ESPE, Sangolquí, Ecuador
vhandaluzl@espe.edu.ec

Abstract. This work presents the design of a bilateral teleoperation system for a robotic arm. It proposes a new prototype communication protocol with Websockets for the communication and Json for data structuration on a cloud computing environment with OpenStack and Openshift Origin. The human operator receives visual and force feedback from the remote site, and it sends position commands to the slave. Additionally, in the tele-operation system it is proposed that the human operator is immersed in an augmented reality environment to have greater transparency of the remote site. The transparency of a tele-operation system indicates a measure of how the human feels the remote system. Finally, the experimental results are reported to verify the performance of the proposed system.

Keywords: Cloud computing · Bilateral teleoperation · Robotic arm · Path following · Virtual reality

1 Introduction

During the last decades, various models of field robots have been developed to take the place of humans in dangerous tasks, such as rescue missions, Mars exploration, airport patrols, and missions in war [1–3]. In order to operate in highly variable, unstructured, unknown, or dynamic environments, an advanced tele-robotic system with increased adaptability and human like manipulation capabilities is required. In general, a tele-operation system is composed of a local site, where a human operator drives a hand-controller named master; a remote site, where a robot named slave follows the motion of the master to execute a given task in interaction with the environment; and a communication channel that links both sites. The master is used to generate velocity or position commands which are sent to the remote site, while the force due to the interaction between the slave and the environment is back-fed to the human operator through the actuators of the master [4–7].

Cloud computing is a paradigm in the way that the computing resources and applications are used and delivered as a services. These main resources as a service are computing, storage and the network infrastructure. Cloud computing refers to providing these resources as a service over the Internet to the public or in an organization

© Springer International Publishing Switzerland 2016

that is of private use [8]. There are three types of cloud services and they differ in the approach on how the resources are made available. The first approach is to make the hardware infrastructure available as a service and is called Infrastructure as a Service (IaaS). The second approach is to provide a platform (the OS along with the necessary software, frameworks and tools) over the hardware infrastructure. This is called Platform as a Service (PaaS). The third approach is to provide the application as a service and is called Software as a Service (SaaS) [9–12].

Recently, many researches about cloud communication are taking an increasing interest in the field of robot services for manipulation and getting information from them, *e.g.,* Cloud-Based Robot services [13] that examining previous implementations RT-Middleware, ROS and make a new proposal (RSNP) that is based on web service technologies, similarly many researches have been made about cloud computing and how we can interact with it, but almost all of this interactions are by standard protocol http like http polling: that make a request to the server and the server answer this request, other protocols that give more interaction client-server are: push comet, long polling, flash, XMR, htmlfiles and others technologies like webservice. Cloud computing is a new wave in the field of technology information. Some see it as an emerging field in computing. They it consists of a set of resources and services offered by In-Internet. Therefore, "cloud computing" is also called "computer on the Internet. "The word" cloud "is a metaphor to describe the Web as a space where the computer was preinstalled and exists as a service. (Cloud Computing: Opportunities and Challenges) the main characteristic of cloud computing is the virtualization of network, storage and compute [14–17].

In such context, this work proposes a bilateral teleoperation system in order to allow the handling of objects. It comprises a robotic arm (slave) so that it can move and manipulate objects. The human operator receives visual and force signals and sends position commands generated by a haptic device (master) to the remote site; furthermore, the local site has proposed a virtual environment and augmented reality implemented in Unity3D. The augmented reality provides feedback of different signals from the remote site in real time, so that through different haptic devices stimulate the senses of sight, touch and hearing of the human operator so that it can "transmit" their skill, experience and expertise to the robot to perform a task. Whereas for the channel communication is proposed a new implementation of teleoperation-communication (Robot + Cloud) but in the communication using WebSockets and Json data Structure unlike in the works found in the literature. On the other hand, the design of the teleoperation system structure is mainly composed by two parts, each one being a controller itself. The first one is a minimum norm controller to solve the path following problem which considered the desired position of the person to move of the arm. The second one is a dynamic compensation controller, which receives as inputs the velocity references calculated by the kinematic controller. To validate the proposed teleoperation system, results are included and discussed.

The work is organized as follows: in Sect. 2 describes the bilateral teleoperation system proposed. Section 3 presents the implementation of communication channel through the cloud computing services; while that the kinematic and dynamic modeling of the robotic arm are shown in Sect. 4, also of the controllers design and the analysis of the system's stability. Next in Sect. 5 the design of the local site of the human operator developed, and finally the results are presented and discussed in Sects. 6 and 7, respectively.

2 Bilateral Teleoperation System

A bilateral tele-operation system for a robotic arm is proposed, in which forward and backward delay between local and remote sites are assumed to be irrelevant ($\zeta_1(t) = 0$ and $\zeta_2(t) = 0$). The proposed tele-operation system is shown in Fig. 1. In this system both the force back-fed to the human operator are considered.

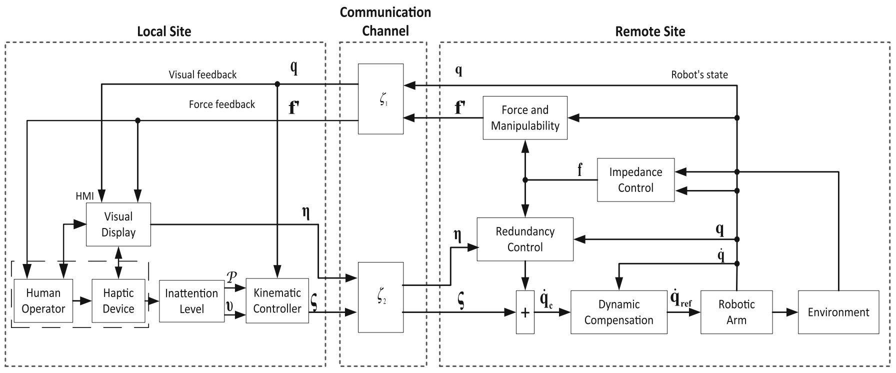

Fig. 1. Block diagram of the bilateral tele-operation system

For communication channel of the bilateral tele-operation system this work proposed a new implementation of teleoperation-communication (Robot + Cloud) with a communication using websockets and Json data Structure as shown in Fig. 2. Websockets is a standardized communication protocol that give a full-duplex, bidirectional channel communication and help to make scalable applications giving a high performance communication through Internet in this case on a cloud computing infrastructure; while that Json Data Structure (JavaScript Object Notation) is a light weight format for data interchange.

On other hand, a given mission is generally composed of several operation processes, in this context the human operator controls the robotic arm by sending position commands to the end-effector of the robot: h_l, h_m, and h_n, one for each axis in respect to the inertial frame $\mathcal{R}(\mathcal{X}, \mathcal{Y}, \mathcal{Z})$, using a haptic device $\mathbf{h_d} = [h_l \quad h_m \quad h_n]^T$. The human operator commands are generated with the use of the FALCON^{TM} from Novint Technologies Incorporated as indicated in Fig. 3. Its positions P_x, P_y, and P_z are translated into position commands h_l, h_m and h_n for the manipulation mode, through the following rotation matrix,

$$\begin{bmatrix} h_l \\ h_m \\ h_n \end{bmatrix} = \begin{bmatrix} \cos(q_1) & -\sin(q_1) & 0 \\ \sin(q_1) & \cos(q_1) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} \tag{1}$$

where q_1 represents the first joint of the robotic arm that rotates about the axis Z.

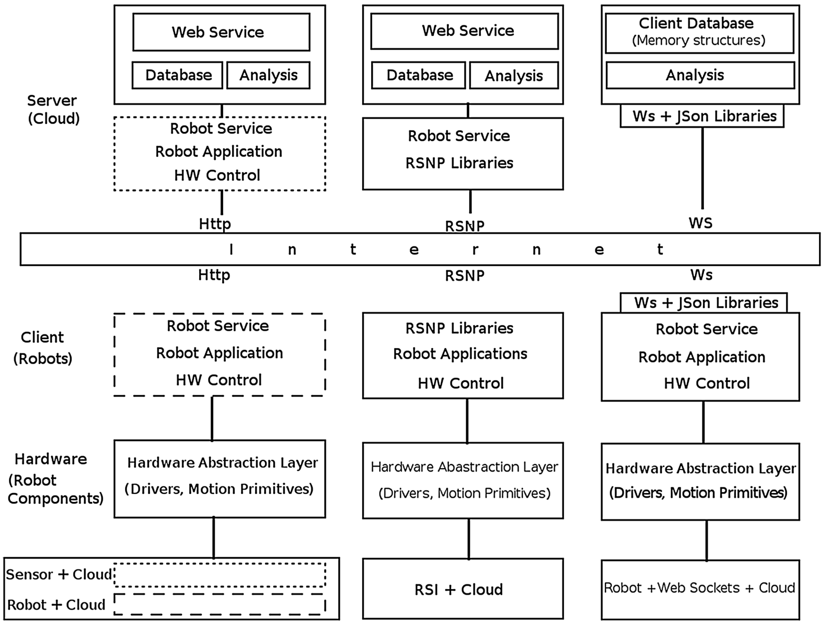

Fig. 2. Block diagram of the bilateral tele-operation system

In the case of robotic manipulators, the force feedback is the result of a physical interaction with the environment.

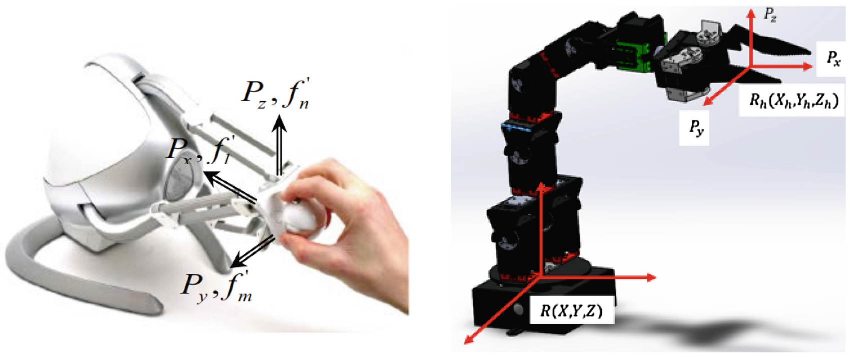

Fig. 3. FALCON^{TM} Novint Technologies Incorporated

On the other hand, the human operator has the possibility of changing the desired internal configuration of the robotic arm at any time during the execution of a mission, through the visual display (HMI), *i.e.*, $\boldsymbol{\eta} = [\, q_{1d} \quad q_{2d} \quad \cdots \quad q_{nd} \,]^T$. This desired configuration vector may or may not have the values that maximize manipulability of the robotic arm.

3 Communication Channel: Cloud Computing Services

The classification of communication infrastructure of the cloud computing is [13] *(i) Peer-based model* where each virtual machine in the cloud is considered as a computing unit and the robots and Vms form a fully distributed computing mesh; (*ii) Clone-based model* where each robot corresponds to a system level clone in the cloud. A task can be executed in the robot or in its clone. This model allows for sporadic outage in the physical (Machine to machine) M2 M network; and *(iii) Proxy-Based Model* in the group of networked robots, one unit functions as a group leader, the communication with a proxy VM in the cloud infrastructure. For this work the roxy-Based Model is implemented because this model has a best performance in the communication speed. In our proposed model considers an internal cloud infrastructure composed of three nodes, as shown Fig. 4., where, *(a) Controller & Block Storage node* this node mainly contain: Controller administrate all the openstack resources, administration of services, api end points, projects, users and roles. Block storage service provides block storage devices to guest instances (Cinder service). Image Storage service enables users to discover, register and retrieve virtual machine images (Glance). Identity service perform tracking users and their permissions and providing a catalog of available services with their API endpoints (Keystone). Dashboard service is a web interface that enables cloud administrator and users to manage OpenStack resources and services (Horizon); *(b) Network & Object storage node* this node mainly contain: Network service allows to create and attach interface devices managed by other OpenStack services to network (Neutron). Object Storage Service work together to provide object storage and retrieval through a REST API (Swift); and finally *(c) Compute Node* this node contain Compute Service allow to host and manage cloud computing systems this host contain the Hypervisor (Nova).

4 Remote Site: Robotic Arm

The manipulator configuration is defined by a vector $\mathbf{q}(t)$ of n independent coordinates, called *generalized coordinates of the manipulator*, where $\mathbf{q} = [\, q_1 \quad q_2 \quad \cdots \quad q_n \,]^T$ represents the generalized coordinates of the robotic arm. The configuration $\mathbf{q}$ is an element of the manipulator *configuration space*; denoted by $\mathcal{N}$. The location of the end-effector of the mobile manipulator is given by the m–dimensional vector $\mathbf{h} = [\, h_1 \quad h_2 \quad \cdots \quad h_m \,]^T$ which defines the position and the orientation of the end-effector of the manipulator in $\mathcal{R}$. Its m coordinates are the *operational coordinates of the manipulator*. The set of all locations constitutes the *manipulator operational space*, denoted by $\mathcal{M}$.

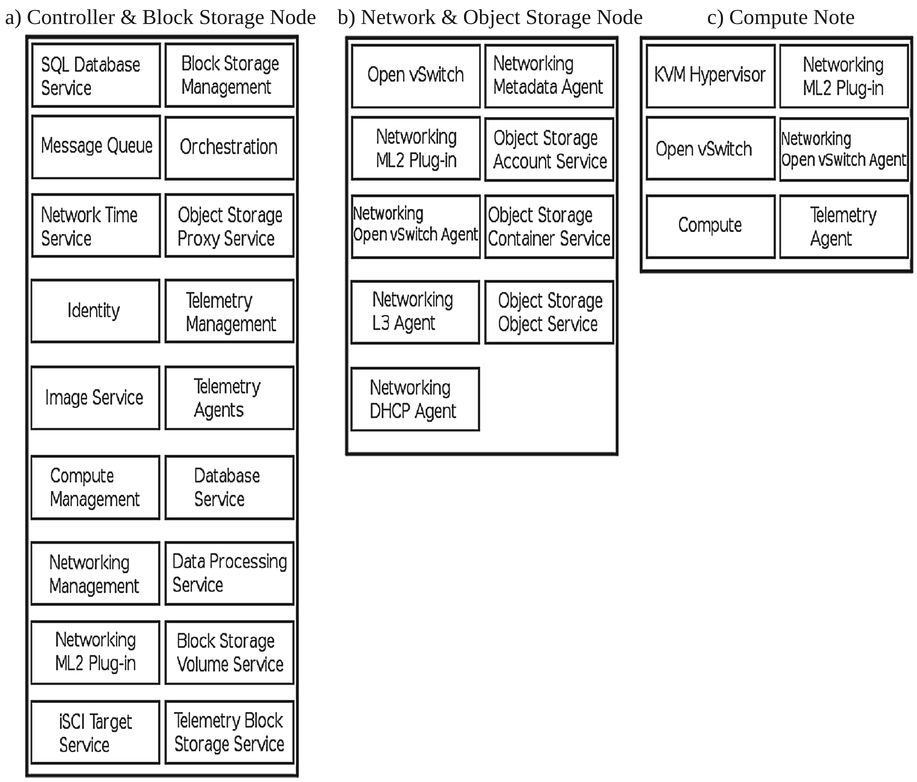

Fig. 4. OpenStack: Infrastructure

4.1 Kinematic and Dynamic Model

The kinematic model of a robotic arm gives the derivative of its end-effector location as a function of the derivatives of both the robotic arm configuration and the location of the mobile platform,

$$\dot{\mathbf{h}}(t) = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}}(t) \tag{2}$$

where $\mathbf{J}(\mathbf{q})$ is the Jacobian matrix that defines a linear mapping between the vector of the robotic arm velocities $\dot{\mathbf{q}}(t)$ and the vector of the end-effector velocity $\dot{\mathbf{h}}(t)$.

On the other hand, the mathematic model that represents the dynamics of a robotic arm can be obtained from Lagrange's dynamic equations, which are based on the difference between the kinetic and potential energy of each of the joints of the robot (energy balance) [18]. The dynamic model of the robotic arm, having as control signals the reference velocities of the system, can be represented as follows [19],

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \dot{\mathbf{q}}_{\mathbf{ref}} \tag{3}$$

where, $\mathbf{M}(\mathbf{q}) = \mathbf{H}^{-1}(\bar{\mathbf{M}} + \mathbf{D})$, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{H}^{-1}(\bar{\mathbf{C}} + \mathbf{P})$, $\mathbf{g}(\mathbf{q}) = \mathbf{H}^{-1}\bar{\mathbf{g}}(\mathbf{q})$. Thus, $\mathbf{M}(\mathbf{q}) \in \Re^{\delta_n \, \mathrm{x} \, \delta_n}$ is a positive definite matrix, $\bar{\mathbf{C}}(\mathbf{q}, \mathbf{v})\mathbf{v} \in \Re^{\delta_n}$, $\bar{\mathbf{G}}(\mathbf{q}) \in \Re^{\delta_n}$ and $\dot{\mathbf{q}}_{\mathbf{ref}} \in \Re^{\delta_n}$ is the vector of velocity control signals, $\mathbf{H} \in \Re^{\delta_n \, \mathrm{x} \, \delta_n}$, $\mathbf{D} \in \Re^{\delta_n \, \mathrm{x} \, \delta_n}$ and $\mathbf{P} \in \Re^{\delta_n \, \mathrm{x} \, \delta_n}$ are constant symmetrical diagonal matrices, positive definite, that contain the physical parameters of the robotic arm, *e.g.*, motors, velocity controllers. More details about the dynamic model (5) can be found in [19].

4.2 Control Design

From the viewpoint of control theory, the end-effector of the robotic arm must follow the path desired by a person and generated through brain signals, as shown in Fig. 5. As represented in Fig. 5, the path to be followed is denoted as $\mathcal{P}(s)$, where $\mathcal{P}(s) = (x_{\mathcal{P}}(s), y_{\mathcal{P}}(s), z_{\mathcal{P}}(s))$; the actual desired location $P_d(s_D) = (x_P(s_D), y_P(s_D) z_P(s_D))$ is defined as desired point of the person, *i.e.*, the closest point on $\mathcal{P}(s)$, with s_D being the curvilinear abscissa defining the point P_d; $\tilde{x} = x_P(s_D) - x$ is the position error in the $\mathcal{X}$ direction; $\tilde{y} = y_P(s_D) - y$ is the position error in the $\mathcal{Y}$ direction; $\tilde{z} = z_P(s_D) - z$ is the position error in the $\mathcal{Z}$ direction; $\boldsymbol{\rho}$ represents the distance between the end-effector position of the robotic arm $h(x, y, z)$ and the desired point for the person P_d, where the position error in the $\boldsymbol{\rho}$ direction is $\tilde{\rho} = 0 - \rho = -\rho$, *i.e.*, the desired distance between the end-effector position $h(x, y, z)$ and the desired point P_d must be zero.

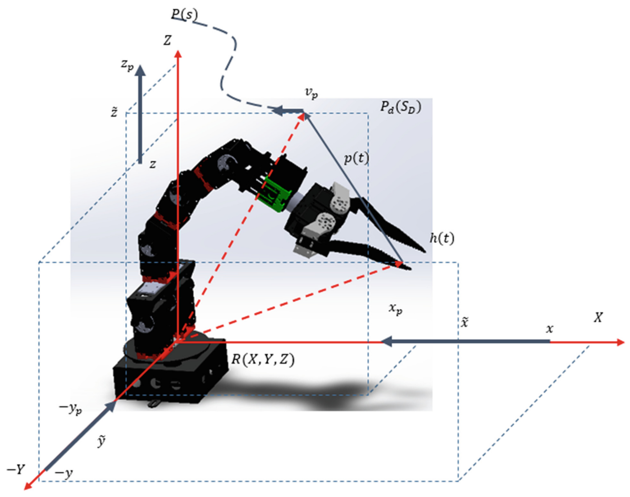

Fig. 5. The orthogonal projection of the point of interest over the path

Hence, the path following problem is to find the control law for the robotic arm as a function of the control errors (position and orientation of the end-effector) and the desired velocities of the end-effector

$$\dot{\mathbf{q}}_{\mathbf{ref}}(s_D, h) = f(\rho(t,s), \boldsymbol{\upsilon}_P(s_D, h)) \tag{4}$$

such that $\tilde{x} = 0$, $\tilde{y} = 0$ and $\tilde{z} = 0$. Therefore, if $\lim_{t\to\infty} \tilde{x}(t) = 0$, $\lim_{t\to\infty} \tilde{y}(t) = 0$ and $\lim_{t\to\infty} \tilde{z}(t) = 0$ then $\lim_{t\to\infty} \rho(t) = 0$.

4.2.1 Kinematic Controller

The design of the kinematic controller of the robotic arm is based on the kinematic model of the arm (4). The following control law is proposed

$$\dot{\mathbf{q}}_{\mathbf{c}} = \mathbf{J}^{\#}\left(\boldsymbol{\upsilon}_{\mathbf{d}} + \mathbf{L_K}\mathbf{tanh}\left(\mathbf{L_K^{-1}K}\,\tilde{\mathbf{h}}\right)\right) + \left(\mathbf{I} - \mathbf{J}^{\#}\mathbf{J}\right)\mathbf{L_B}\mathbf{tanh}\left(\mathbf{L_B^{-1}B}\,\Lambda\right) \tag{5}$$

where $\mathbf{J}^{\#} = \mathbf{W}^{-1}\mathbf{J}^{T}\left(\mathbf{J}\mathbf{W}^{-1}\mathbf{J}^{T}\right)^{-1}$, being $\mathbf{W}$ a definite positive matrix that weighs the control actions of the system, $\boldsymbol{\upsilon}_{\mathbf{d}}$ is the desired velocities vector of the end-effector $\mathbf{h}$, $\tilde{\mathbf{h}}$ is the vector of control errors, defined as $\tilde{\mathbf{h}} = \mathbf{h_d} - \mathbf{h}$, $\mathbf{B}$ and $\mathbf{L_B}$ are definite positive diagonal matrices that weigh the vector Λ. In order to include an analytical saturation of velocities in the robotic arm the use of the **tanh** (.) function is proposed, which limits the error in $\tilde{\mathbf{h}}$ and the magnitude of the vector Λ. The second term of (5) represents the projection on the null space of $\mathbf{J}$, where Λ is an arbitrary vector which contains the velocities associated to the robotic arm. Therefore, any value given to Λ will have effects only on the internal structure of the arm, and will not affect the final control of the end-effector at all. By using this term, different secondary control objectives can be achieved effectively [20].

4.2.2 Controller with Dynamic Compensation

The proposed kinematic controllers presented in Subsect. 4.2.1 assume perfect velocity tracking; nevertheless this is not true in real contexts, *i.e.*, $\dot{\mathbf{q}}(t) \neq \dot{\mathbf{q}}_{\mathbf{c}}(t)$, mainly when high-speed movements or heavy load transportation are required. Therefore, it becomes essential to consider the robot's dynamics, in addition to its kinematics. Then, the objective of the dynamic compensation controller is to compensate the dynamics of the system, thus reducing the velocity tracking error. This controller receives as inputs the desired velocities calculated by the kinematic controllers, and generates velocity references for the mobile platform and robotic arm. Hence, relaxing the perfect velocity tracking assumption, there will be a velocity error defined as, $\dot{\tilde{\mathbf{q}}}(t) = \dot{\mathbf{q}}_{\mathbf{c}}(t) - \dot{\mathbf{q}}(t)$. This velocity error motivates the dynamic compensation process, which will be performed based on the inverse dynamics of the system. With this aim, the exact model of the system without disturbances is considered, thus the following control is proposed,

$$\dot{\mathbf{q}}_{\mathbf{ref}} = \mathbf{M}(\mathbf{q})\sigma + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + g(\mathbf{q}) \tag{6}$$

where $\boldsymbol{\sigma} = \ddot{\mathbf{q}}_{\mathbf{c}} + \mathbf{L_v}\mathbf{tanh}\left(\mathbf{L_v^{-1}K_v}\dot{\tilde{\mathbf{q}}}\right)$. Now, Replacing (6) in (5) it results $\ddot{\tilde{\mathbf{q}}} + \mathbf{L_v}\mathbf{tanh}\left(\mathbf{L_v^{-1}K_v}\dot{\tilde{\mathbf{q}}}\right) = \mathbf{0}$. For the stability analysis the following Lyapunov's candidate function is considered $V\left(\tilde{\dot{\mathbf{q}}}\right) = \frac{1}{2}\tilde{\dot{\mathbf{q}}}^{\mathbf{T}}\tilde{\dot{\mathbf{q}}}$, its time derivative is $\dot{V}\left(\tilde{\dot{\mathbf{q}}}\right) = -\tilde{\dot{\mathbf{q}}}^{\mathbf{T}}\mathbf{L_v}\mathbf{tanh}\left(\mathbf{L_v}^{-1}\mathbf{K_v}\tilde{\dot{\mathbf{q}}}\right) < 0$. Thus it can be immediately concluded that the error vector $\lim_{t\to\infty}\tilde{\dot{\mathbf{q}}}(t) = 0$ asymptotically, provided that $\mathbf{K_v}$ and $\mathbf{L_v}$ are symmetrical positive definite matrices.

5 Local Site

The local site consists of a virtual reality and augmented to allow the human operator to have greater transparency in the remote site reality, as illustrated by the Fig. 6. The Fig. 6 can be described in two stages, the first stage is *(A) Inputs,* These let read all the variables considered in the bilateral tele-operation of both the robot and the environment in which develops the task, entries that are processed in the script are: *(i) Output states of the robot,* represents the variables that describe the robot kinematics $\mathbf{q} = [\,q_1 \quad q_2 \quad \cdots \quad q_n\,]^T$. In the script, the data acquired from the platform $\mathbf{q}$ are

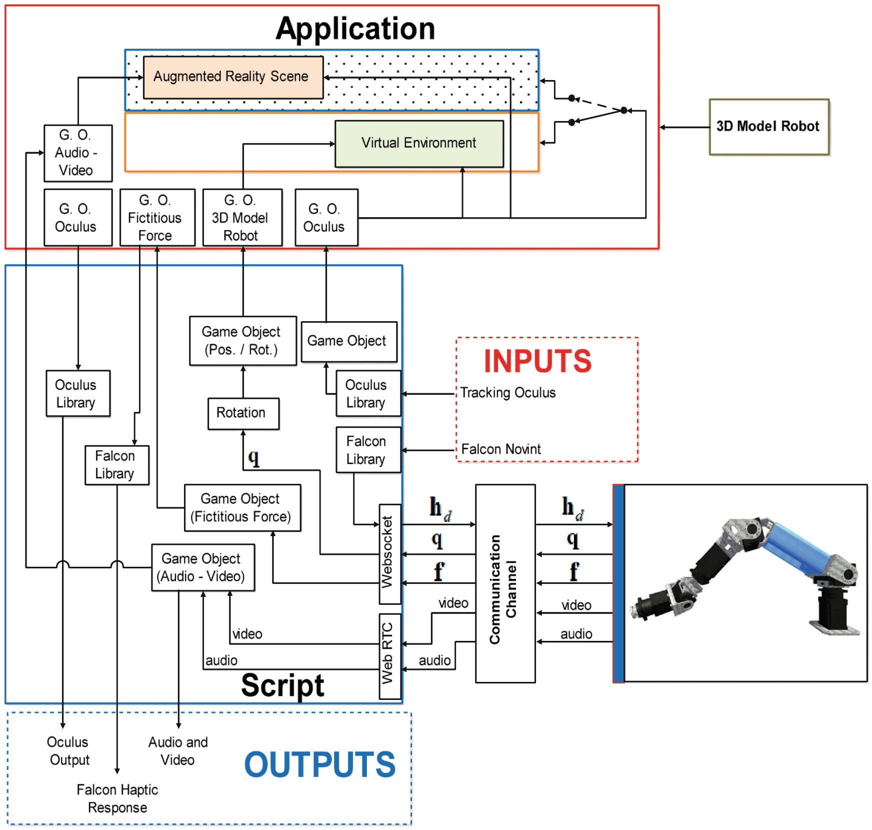

Fig. 6. Block diagram of the virtual environment

assigned to the characteristic of *Rotation* for each joint of the simulated arm Game Object; *(ii) Desired positions* through the FALCON^{MT} haptic device emit the desired location $\mathbf{h_d} = [\,h_l \quad h_m \quad h_n\,]^T$ for manipulation mode. The script obtains the variation of position in world coordinates by a Game Object that simulates the movement of the end-effector of FALCON^{MT} haptic device in space. *(iii) Video and audio* transmits the image and audio captured by the camera located at the end-effector of the robot; *(iv) Fictitious force* is produced by a sensor that maps the area in which the robot can be moved freely; y finally *(v) Oculus Positional Tracking* modifies the rotation of the camera in the virtual and / or augmented environment, corresponding to the movement performed by the human operator when using the Head-mounted Display, HMD, in order to have a greater immersion in the experience of using the application.

Furthermore, the second stage of Fig. 6 is *(B) Outputs Script* that recreates the sensations that the human operator should feel if he were at the remote site, this paper considers three ways that help the human operator to "transmit" their ability and experience to the robot to perform a task.

6 Experimental Results

In order to illustrate the performance of the proposed tele-operation structure, several simulation experiments are carried out for control of a robotic arm, most representative results are presented in this section. The experiments were carried with the kinematic and dynamic models of a robotic arm 6 DOF which was developed at the University of the Armed Forces ESPE (only 3 DOF of the 6 available DOFs are used in the experiments, see Fig. 7). The links of the robotic arm are controlled by a network of servomotors Dinamixel communicated via a RS-485 interface. Therefore to give greater scalability for future functions of the cloud computing environment are used three computers with the same characteristics, intel core i7-3770 Processor, 6 GB ram memory, two Hard disk sata connection of 750 GB for each computer, CentOS Linux 7 X64 with minimal install for the base system, the network connection are using Cat6a devices, for The IaaS infrastructure of the Cloud-Computing environment was used

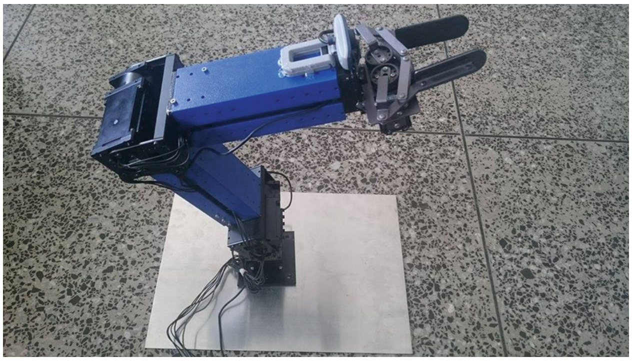

Fig. 7. Rootic arm 6DOF developed at the University of the Armed Forces ESPE

OpenStack Kilo, then created a OpenShift project with php5.4 support on the cloud infrastructure for the communication service and presentation of the data.

The local site has an Oculus Rift HMD and a haptic device FALCON^{TM} Novint. To transmit the comands generated by humans with was used a FALCON^{TM} haptic device from Novint Technologies Incorporated as indicated in Fig. 8; while the Fig. 9 shown a robotic arm 6DOF developed in SolidWorks.

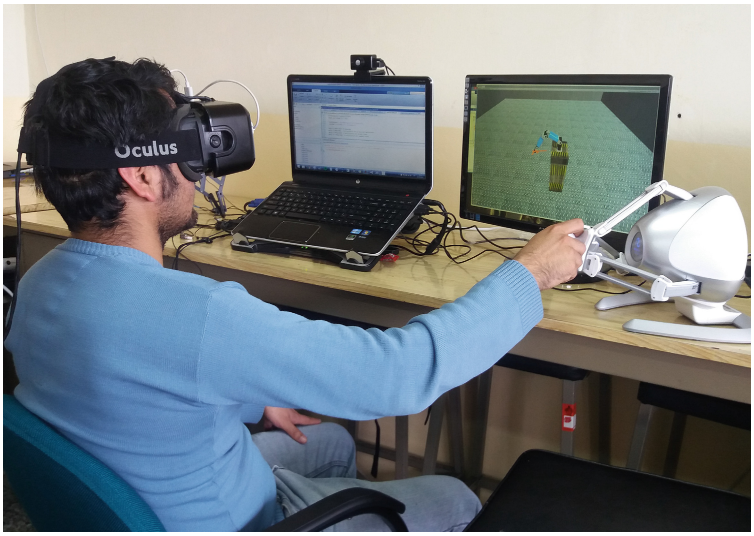

Fig. 8. Local site of the tele-operation scheme

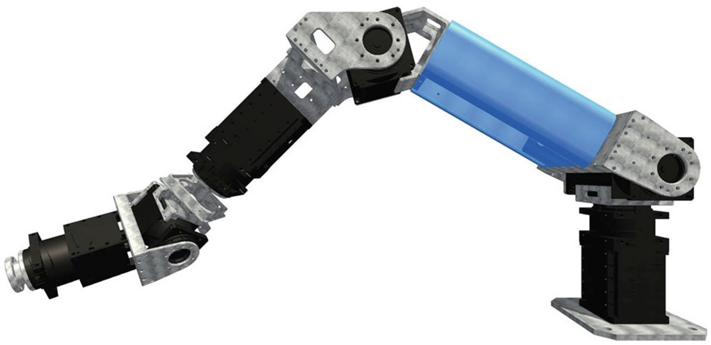

Fig. 9. Rootic Arm 6DOF developed in SolidWorks

The FALCON^{TM} Novint communication functions were developed in C ++ language to create dynamic link libraries to use on LabView environment to get FALCON^{TM} signals, for testing of FALCON^{TM} hosts were used windows 8.1 X64 and windows 7 X86, for test of C ++ dll compatibility on each Operative System. On labView was created some operations and functions, the data that will be send are set in Json data structure, to encapsulate and send to the cloud was used websocket protocol, for keeping the connection on the cloud with all the clients, each client must set a id and store the connection in a structure list in memory where the data will be decoded and will show information of the origin, the destination and the message, on the server for manage all the connections thread, the service on the host were developed with php + javascript + html languages with phpwebsockets, the fancywebsocket libraries and codeigniter framework. The connection to the cloud of the arm and the functions are developed with C ++ language on GNU/Linux Operative System.

Hence, Figs. 10–12 show the results of the experiment of bilateral tele-operation. Figure 10 shows the stroboscopic movement on the *X*-*Y*-*Z* space of Unity3D. Figure 11 shows a comparison between the reference generated by the human operator and the

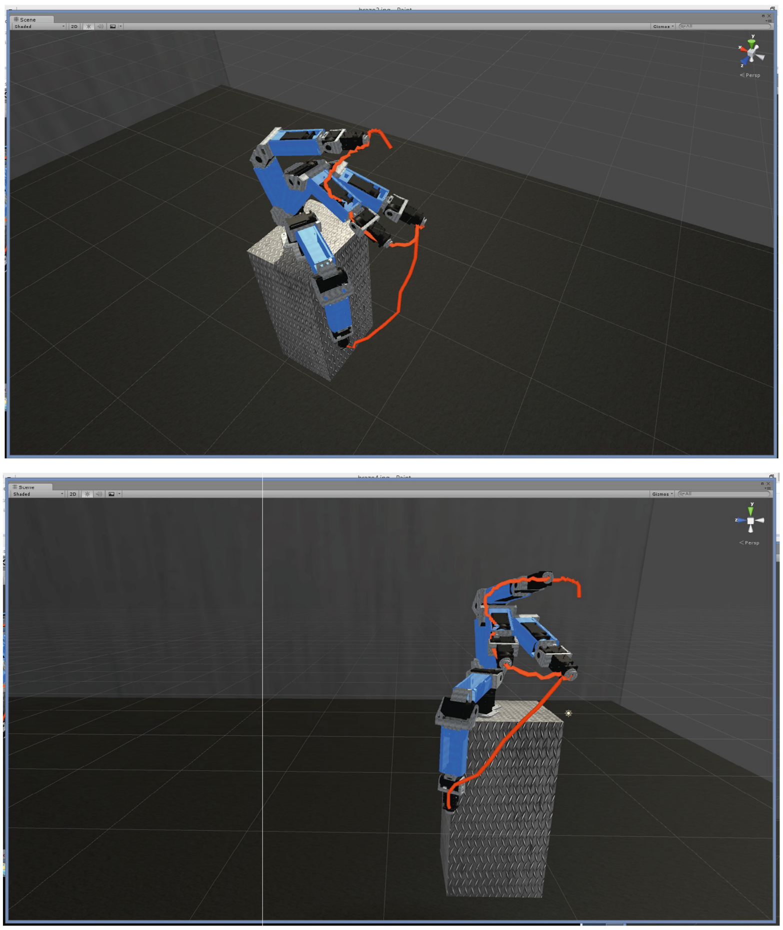

Fig. 10. Stroboscopic movement of the robotic arm in Unity 3D.

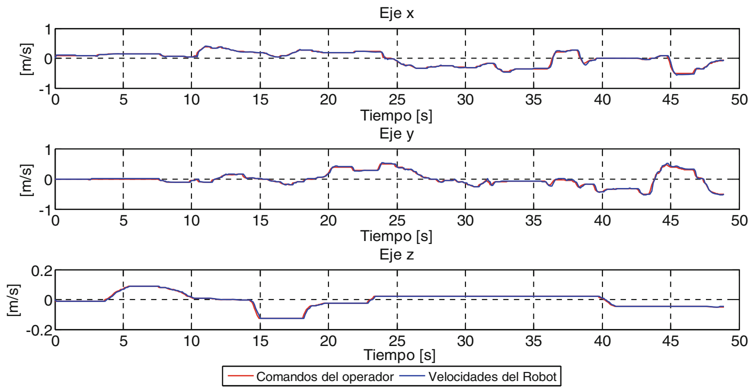

Fig. 11. Comparison between the references generated by the human operator and the actual velocities of the end effector (Color figure online)

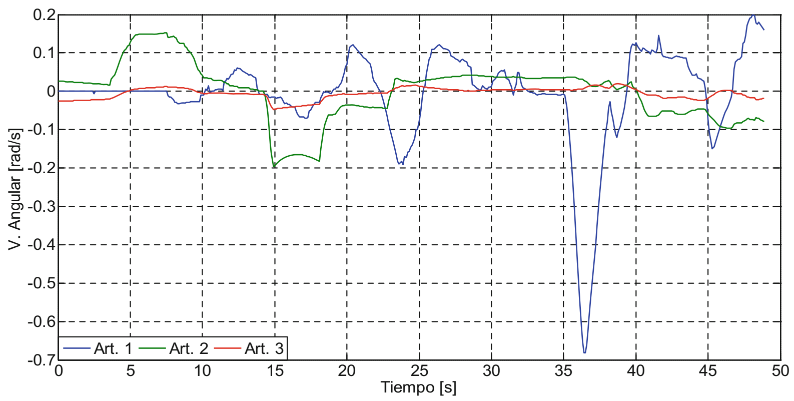

Fig. 12. Evolution of the control actions of the robotic arm (Color figure online)

actual velocities of the end effector; while the Fig. 12 depicts the time evolution of the control actions for the joints of the robotic arm.

7 Conclusions

In this paper a bilateral tele-operation system is proposed, in this system is considers that human operator has greater transparency of the remote. This system allows a human operator to perform complex tasks in a remote environment with a robotic arm. The teleoperation system use a communication channel that implements a new prototype of communication protocol with WebSockets and Json data structures on a cloud computing environment with OpenStack. Therefore, experimental results were also presented showing the feasibility and the good performance of the proposed teleoperation structure.

Acknowledgment. The authors would like to thanks to the Consorcio Ecuatoriano para el Desarrollo de Internet Avanzado -CEDIA-, Universidad de las Fuerzas Armadas ESPE and the Universidad Técnica de Ambato for financing the project *Tele-operación bilateral cooperativo de múltiples manipuladores móviles – CEPRAIX-2015-05*, for the support to develop this paper.

References

1. Andaluz, V.H., Canseco, P., Varela, J., Ortiz, J.S., Pérez, M.G., Morales, V., Robertí, F., Carelli, R.: Modeling and control of a wheelchair considering center of mass lateral displacements. In: Liu, H., Kubota, N., Zhu, X., Dillmann, R. (eds.) ICIRA 2015. LNCS, vol. 9246, pp. 254–270. Springer, Heidelberg (2015)
2. Slawinski, E., Mut, V.: PD-like controllers for delayed bilateral teleoperation of manipulators robots. Int. J. Robust Nonlinear Control **25**(12), 1801–1815 (2015)
3. Voth, D.: A new generation of military robots. Intell. Syst. **19**(4), 2–3 (2004)
4. Sheridan, T.B.: Telerobotics, Automation, and Human Supervisory Control. The MIT Press, Cambrige (1992)
5. Stramigioli, S., Mahony, R., Corke, P.: A novel approach to haptic tele-operation of aerial robot vehicles. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2010), pp. 5302–5308, May 2010
6. Niculescu, S.-I., Taoutaou, D., Lozano, R.: Bilateral teleoperation with communication delays. Int. J. Robust Nonlinear Control **13**(9), 873–883 (2003)
7. Yin, S., Ding, S.X., Xie, X., Luo, H.: A review on basic datadriven approaches for industrial process monitoring. IEEE Trans. Ind. Electron. **61**(11), 6414–6428 (2014)
8. Griffith, R., Joseph, A.D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I., Armbrust, M., Fox, A., Zaharia, M.: Above the clouds: A berkeley view of cloud computing [white paper] (2009). http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.pdf
9. Amazon ec2. amazon elastic compute cloud (2009). http://aws.amazon.com/ec2/
10. Google app engine (2009). http://code.google.com/appengine/
11. Wurm, K.M., Stachniss, C., Burgard, W.: Coordinated multi-robot exploration using a segmentation of the environment. In: International Conference on Intelligent Robots and Systems (2008)
12. Fox, D.: Distributed multi-robot exploration and mapping. In CRV 2005: Proceedings of the 2nd Canadian Conference on Computer and Robot Vision, pp. 15–xv. IEEE Computer Society, Washington, DC (2005)
13. Dhiyanesh, B.: Dynamic Resource Allocation for Machine to Cloud Communications Robotics Cloud. In: IEEE International Conference on Emerging Trends in Electrical Engineering and Energy Management, pp. 451–454 (2012)
14. Guizzo, E.: Robots with their head in the clouds. IEEE Spectrum, Robotics, March 2011
15. Asaro, P.: Remote-control crimes: roboethics and legal jurisdictions of tele-agency, Special issue on Roboethics, Veruggio, G., Van der Loos, M., Solis, J. (eds.) IEEE Robot. Autom. Mag. **18**(1), 68–71 (2011)
16. Shiraz, M., Sookhak, M., Gani, A., Shah, S.A.A.: A Study on the critical analysis of computational offloading frameworks for mobile cloud computing. J. Netw. Comput. Appl. **47**, 47–60 (2015)
17. Riazuelo, L., Civera, J., Montiel, J.M.M.: C2TAM: a cloud framework for cooperative tracking and mapping. Robot. Auton. Syst. **62**(4), 401–413 (2014)

18. Siciliano, B., Sciavicco, L., Villani, L., Oriolo, G.: Robotics: Modeling, Planning, and Control. Springer, London (2009). ISBN 978-1-84628-641-4
19. Andaluz, V., Roberti, F., Carelli, R.: Robust control with redundancy resolution and dynamic compensation for mobilemanipulators. In Proc. of the IEEE-ICIT International Conference on Industrial Technology, pp. 1449–1454 (2010)
20. Bayle, B., Fourquet, J.-Y.: Manipulability analysis for mobile manipulators. In: Proceedings of the IEEE International Conference on Robots & Automation, pp. 1251–1256 (2001)