robotics

MDPI

Article

Virtualization of Robotic Hands Using Mobile Devices †

Santiago T. Puente ‡, Lucía Más ‡, Fernando Torres ‡ and and Francisco A. Candelas *,‡

Physics, Systems, Engineering and Signal Theory Department, University of Alicante, 03690 Alicante, Spain; santiago.puente@ua.es (S.T.P.); lml44@alu.ua.es (L.M.); fernando.torres@ua.es (F.T.)

* Correspondence: francisco.candelas@ua.es; Tel.: +34-965-90-3400

† This paper is an extended version of our paper published in Puente, S.T.; Más, L.; Torres, F.; Candelas, C. Teleoperation of robotic hands through virtualization in Unity 3D for mobile device. In Proceedings of the Spanish Robotics Conference, Madrid, Spain, 26–29 November 2019.

‡ All of the authors contributed equally to this work.

Received: 13 July 2019; Accepted: 27 August 2019; Published: 16 September 2019

Abstract: This article presents a multiplatform application for the tele-operation of a robot hand using virtualization in Unity 3D. This approach grants usability to users that need to control a robotic hand, allowing supervision in a collaborative way. This paper focuses on a user application designed for the 3D virtualization of a robotic hand and the tele-operation architecture. The designed system allows for the simulation of any robotic hand. It has been tested with the virtualization of the four-fingered Allegro Hand of SimLab with 16 degrees of freedom, and the Shadow hand with 24 degrees of freedom. The system allows for the control of the position of each finger by means of joint and Cartesian co-ordinates. All user control interfaces are designed using Unity 3D, such that a multiplatform philosophy is achieved. The server side allows the user application to connect to a ROS (Robot Operating System) server through a TCP/IP socket, to control a real hand or to share a simulation of it among several users. If a real robot hand is used, real-time control and feedback of all the joints of the hand is communicated to the set of users. Finally, the system has been tested with a set of users with satisfactory results.

Keywords: virtual reality; robotic manipulation; human–robot interaction; telerobotics; Unity 3D; ROS; Allegro Hand; Shadow Dexterous Hand

1. Introduction

Tele-operation of robots has traditionally been performed with high costs and has been very important in dangerous environments, rescue tasks, robotic surgery, and so on [1]. The use of mobile devices, which are widely used in modern society, provides an interesting and cheap means to control a remote system [2]. Specifically, Android and iOS devices can be used to perform simulations and virtualizations of real systems. Furthermore, they can be used to control real remote systems using the concepts of tele-operation [3]. Some works have been focused on how robots and humans can collaboratively perform a task [4].

To control a robot hand, many studies have been carried out, such as [5–7]. We will focus on those who used the Allegro Hand [8], which is a four-fingered robotic hand. These works were focused on different aspects of use of the hand. In [9], object grasping was studied and tested with an Allegro Hand. The authors of [10] presented a framework to determine the grasping points of an object using the Allegro Hand. In [11], a tactile sensor to improve grasping tasks using the Allegro Hand has been described. The authors of [12] presented a study of the fingertip grasping characteristics of an Allegro Hand. In [13], a comparative study of grasping with a human hand and an Allegro

Hand has been performed. The authors of [14] undertook a study whereby a tactile sensor matrix was installed in an Allegro Hand. In [15], an Allegro Hand was installed into an Unmanned Aerial Vehicle (UAV) to perform grasping operations. In [16], a method to compute the minimum grasp forces of a multi-fingered hand with soft fingertips using an Allegro Hand was studied.

There was one main problem in all these works: there was no intuitive user interface [9]. For this reason, it is important to consider hand-held devices. They can provide intuitive user interfaces and have been greatly improved over the last decade: they are now more powerful, have a battery with greater autonomy, and have more storage, allowing for virtual interfaces and higher quality simulations [17]. In contrast with traditional desktop computers, they are lighter and do not need to be connected to electrical supplies. Hand-held devices can be used, in an easier way than desktop computers, in different environments. They provide portable solutions. There are already applications on the market which have been developed using mobile interfaces to control different kinds of robots [18].

Currently, a great number of hand-held devices with different resolutions, operative systems, and storage are available [19]. Hence, it is difficult to design an application which considers all possibilities.

Some control interfaces have been developed, to control robots using different techniques [20]. In [21], a JavaScript Object Notation (JSON) Application Programming Interface (API) for connecting a ROS [22] with another type of system was described. Unity 3D [23] has been used to simulate different environments [24]. One environment has been developed for controlling multiple UAVs [25]. Another is SARGE, a search and rescue game environment [26] which also simulates robot sensors. Furthermore, the virtual environment for a Brain–Computer Interface (BCI) was created, to control a virtual robot using the brain's electrical signals [27]. There are ways to connect Unity 3D and ROS for an immersive experience [28]. Furthermore, the models of the robots should be considered for correct inclusion in the devices [29]. In addition, ROS should be considered, as it is one of the main frameworks to program and control robots. Some works have been carried out to merge mobile devices and ROSs. For example, Faust [30] presented a connection between an Android device and a mobile robot in ROS using a Raspberry PI as the interface. HiBot [31] is an ROS-based, generic, configurable, and remote task management framework for robots, which implements task definition, remote task distribution, remote task monitoring, and remote task control of the robot. Another interesting tool to connect the ROS with Android devices is ROSBridge [32], which enables ROS to communicate with the web and application developers (for communication with robots), and allows robotics researchers to communicate with each other. The work of Codd, Downey, and Jenkin [33], which used an iOS device for human–robot interaction, should be highlighted.

The approach in this paper is a collaborative, user-friendly environment based on Unity 3D to connect a user's virtualization of a robot hand with the real hand through a remote ROS server. This paper extends a conference paper [34]. The paper is structured in the following manner. In Section 2, the system is described. Section 3 presents the experimental results of the system. Finally, in Section 4, the conclusions and current work are presented.

2. System Description

This section describes the architecture of the proposed system, from the points of view of the client and of the server, as well as the interconnection between them.

2.1. Architecture of the System

The proposed system is based on a bilateral tele-operation approach. The remote site contains the real robot hand, and the local side contains a virtual representation of the remote robot hand's state. It is a hierarchical architecture (see Figure 1) which allows the operator to send hand position commands to the remote system, as well as directly controlling the remote robot hand. Cartesian

control of each finger can be performed to define the position of it. The inverse kinematic is used to compute the joint values, which will be transmitted by the system.

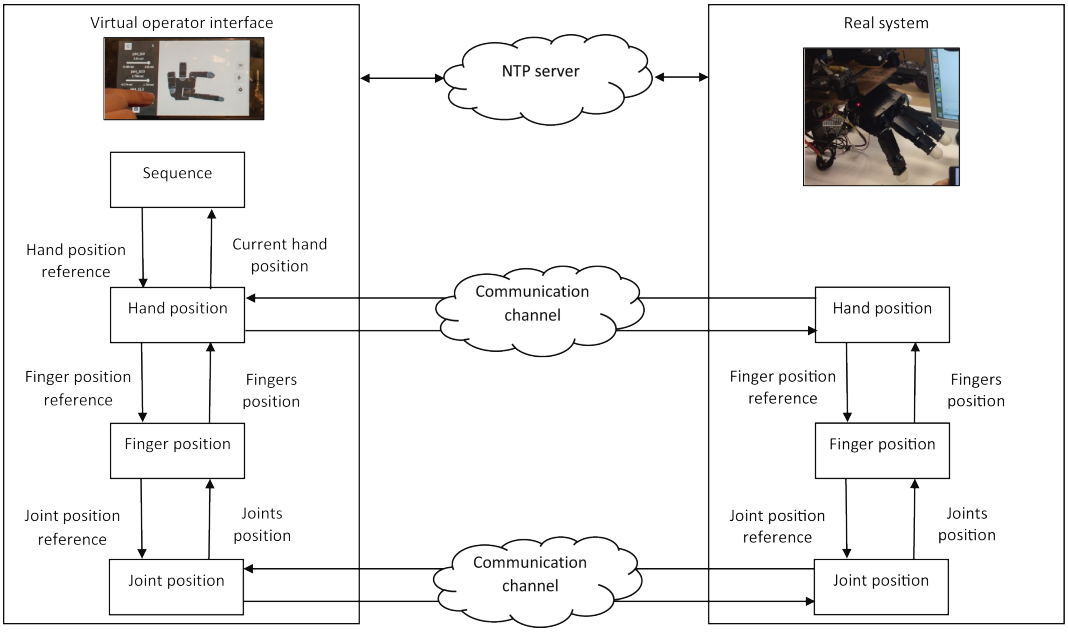

Figure 1. Architecture of the proposed system.

Furthermore, the system is prepared to synchronize several operators with the same robotic hand. In other words, it allows several local users to supervise the performance of the remote robot hand and interact with it. Figure 2 presents the basic concepts of the direct control architecture of the remote robot hand by a set of operators. It shows n local operators, labeled from 1 to n, each with a virtualization of the remote robot hand to perform direct control of it. To control the remote system, an operator sends a joint position to the remote system q_{oi}, where i indicates which operator is refereed, and t_i is a true-time obtained from a Network Time Server (NTP) for the operator i, which will allow for the synchronization of operator commands provided by different operators. In the proposed architecture, each command is delayed by the network communication, where the delay is represented by the remote time of each command t'_i. At the remote site, operator commands are sorted and merged according to their true time. The resulting joint parameter (q_d) is sent to the remote robot hand, which is an array with joint values for all fingers of the hand.

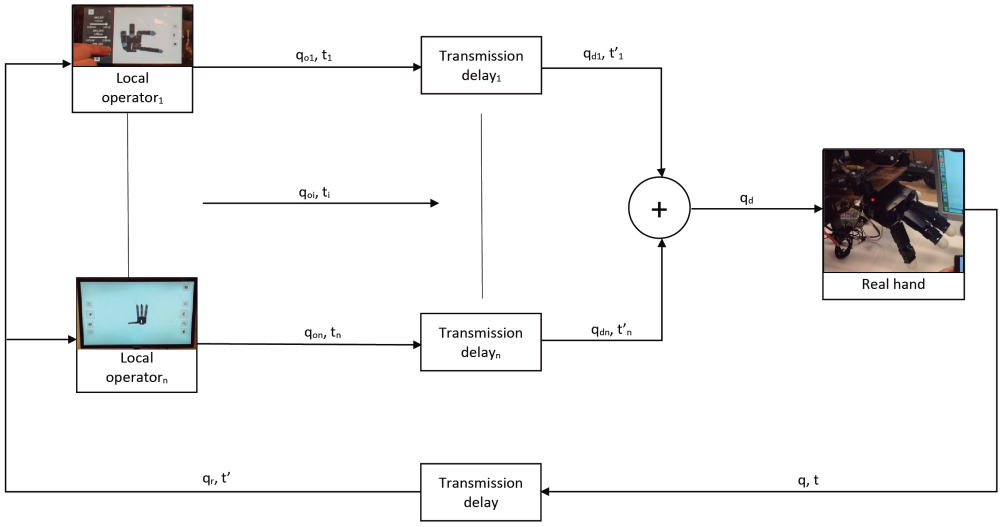

Figure 2. Supervision control architecture of the system.

The remote robot hand returns its current position (q) as well as a true-time information (t), in order to adjust the operator virtualizations according to the transmission delay, with (q_r) arriving at the local operator along with its true-time (t').

This approach is implemented using a Robot Operating System (ROS) user–server architecture, which is shown in Figure 3. Tele-operation of the remote system can be performed over the real one or over a simulation of the real system. The selection of one or another system only depends on server-side selection and is totally transparent to the user.

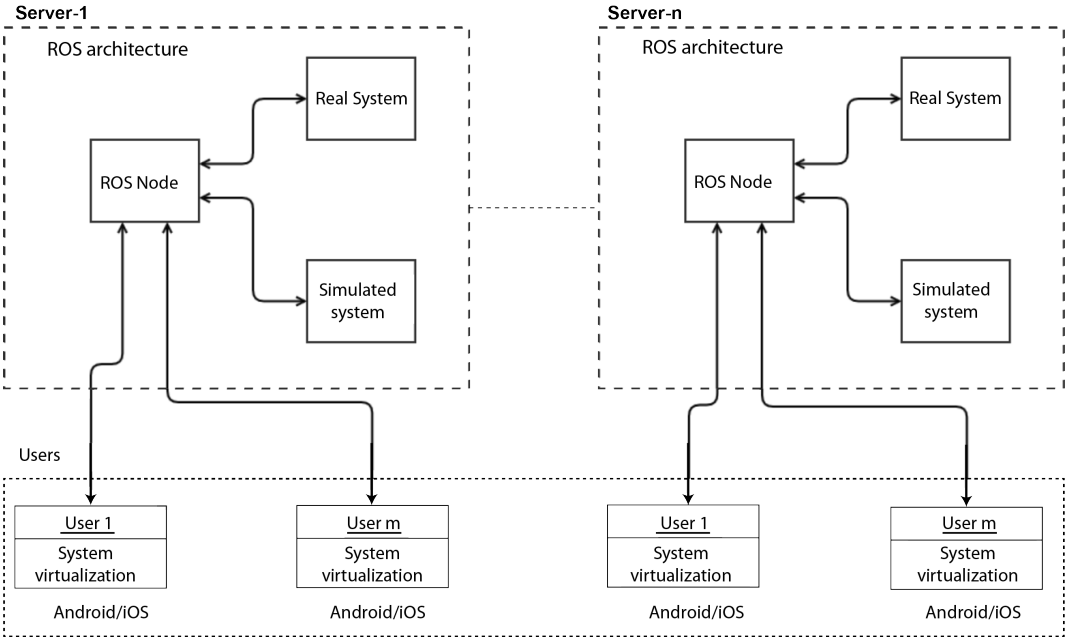

Figure 3. Scheme of the system architecture in the robot operating system (ROS).

The proposed architecture for the modules in the ROS has the following characteristics: On one hand, the servers include communication with the client of the system to be controlled. This performs as an interface between users and the ROS by using a set of topics to transmit information from the user to the system and vice versa. The server can be connected to either a real system or a simulated one. From the point of view of the user, it is totally transparent whether the server connects to a simulation or to the real system.

In the case of the Allegro robotic hand, the server is connected to the ROS nodes created in Linux by the SimLab application [8]. Thus, the server part contains a ROS node to publish and subscribe the SimLab nodes. In the case of the Shadow hand [35], the server is connected to the ROS nodes created by Shadow for the simulation and control of robotic hand.

Furthermore, as is shown in the architecture scheme, several servers can be active, with collaborative systems which can interact with a set of users. The information received by each server is independent of the rest of the servers. This is useful if several robotic hands must be controlled; for example, a pair of (right and left) robotic hands. This is done by starting with one server for each robot hand.

On the other hand, clients have a virtualization of the remote system. This virtualization is generated using Unity 3D. The virtualization of the system can be used in connection with one server or in a stand-alone manner. If it is connected to a server, the interaction performed over the system is shared, in a collaborative way, among all users connected to the same server. If the user is alone, simulation characteristics of the system are provided, as the user is connected to a server but without any network connection. This is useful for simulating the system and checking the behavior of it. Furthermore, the user can store actions in the local device for future use, when the user is connected to a server.

The main difference between the user alone and the server with a simulation of the system is that the server enables the collaborative features of the system as well as a connection to the real system, which the user alone does not have.

2.2. Allegro Robotic Hand Characteristics

This section describes the main characteristics of the Allegro robotic hand used in the system. It is an anthropomorphic hand with four fingers; one of which is the thumb, with four joints in each finger, for a total of 16 joints. This hand uses humanoid robotic hand technology developed by the Korea Institute of Industrial Technology (KITECH). More detailed technical specifications of the hand are shown in Table 1, and the size of the hand is shown in Figure 4.

Table 1. Technical specifications of the Allegro Hand.

Feature	Specification
Number of fingers	4 fingers, including the thumb
Degrees of freedom	4 fingers × 4 = 16 (Active)
Actuation	Type: DC motor
	Gear ratio: 1:369
	Maximum torque: 0.70 (Nm)
	Maximum joint speed: 0.11 (s/degree)
Weight	Finger: 0.17 (kg)
	Thumb: 0.19 (kg)
	Total: 1.08 (kg)
Joint resolution	0.002 (deg)
Communication	CAN 333 (Hz)
Payload	5 (kg)
Power requirement	7.4 VDC (7.0–8.1 V), 5 A minimum

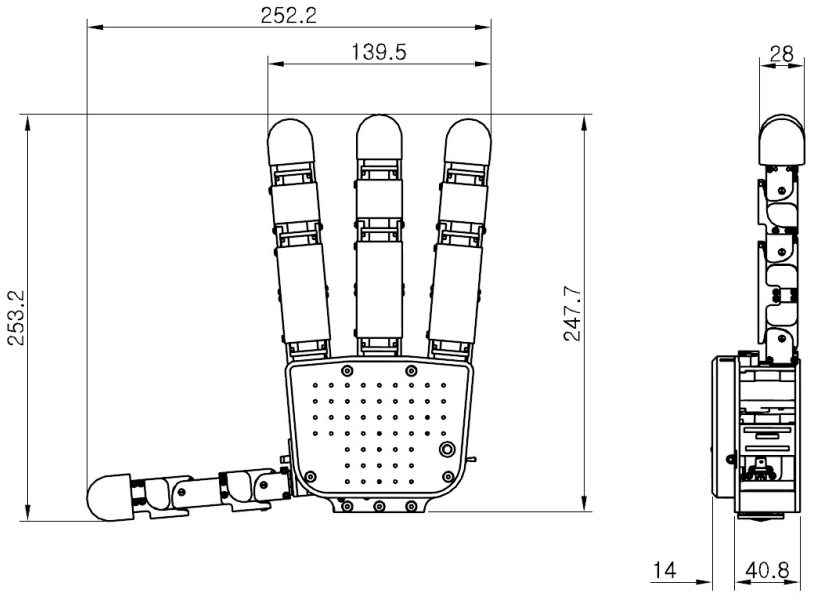

Figure 4. Size of the allegro robot hand (Source: [36]).

Taking these characteristics into consideration, virtualization of the hand can be performed. Additionally, the type of hand (right or left) must be considered, due to the rotational differences between them. These directions of rotation are important, to perform correct simulation of both hands, and to allow for an intelligent automatic transformation by the user interface of the movements defined by one hand to be used by the other one. These characteristics allow for the programming of one hand and, in an automatic way, transforms the commands to the other hand without user interaction.

In addition, it is possible to tele-operate the real hand using the Rviz simulator, to which the system server is connected.

2.3. Shadow Dexterous Hand Characteristics

This section describes the main characteristics of the Shadow dexterous robotic hand used in the system. It is a five-fingered anthropomorphic hand developed by the Shadow company. This hand is designed to be able to move in the same manner as a real hand. It has a total of 24 joints. It is capable of precise and complex movements. More detailed technical specifications are shown in Table 2. In Figure 5, the size of the hand is shown.

Table 2. Technical specifications of the Shadow dexterous Hand.

Feature	Specification
Number of fingers	5 fingers, including the thumb
Degrees of freedom	5 fingers × 4 + 4 = 24 (Active)
Actuation	Type: Smart motor
Weight	4.3 (kg)
Resolution	±0.2 (deg)
Communication	EtherCAT
Payload	5 (kg)
Power requirement	48 VDC, 2.5 A

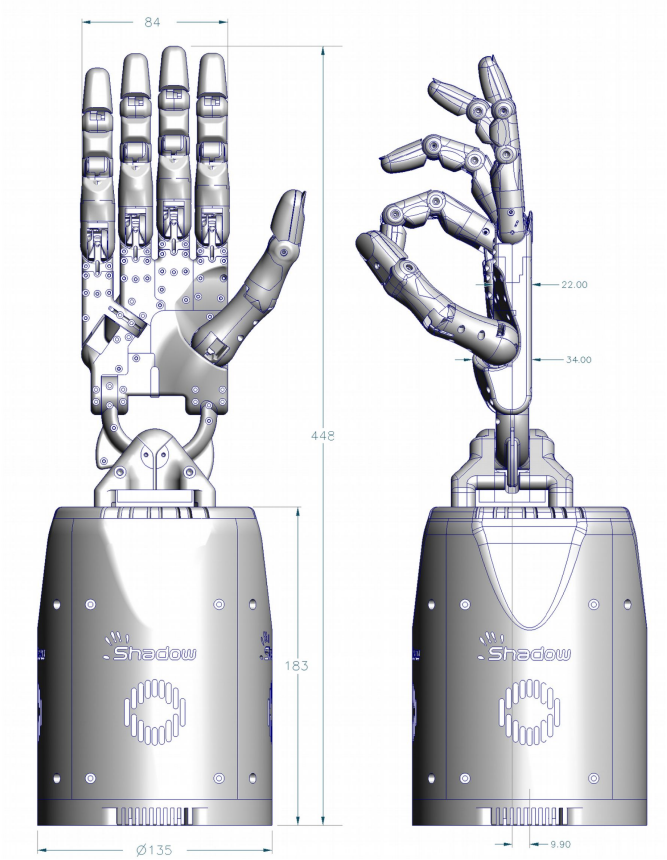

Figure 5. Size of the allegro robot hand (Source [35]).

It is possible to tele-operate either the real hand or a simulated hand using the Gazebo-based simulator [37], to which the system server is connected.

2.4. User Interface

This section describes the user interface, which was designed, for the client side, to accomplish a collaborative multi-device interface with high usability. The interface was developed using Unity3D. It allows the user to sense and supervise the remote robotic hand. The proposed system has a set of functionalities in the user interface, which were designed to improve the usability experience.

The user interface is divided into two main scenes; their main functionalities are shown in Figure 6. Furthermore, a map of each of the scene options is shown.

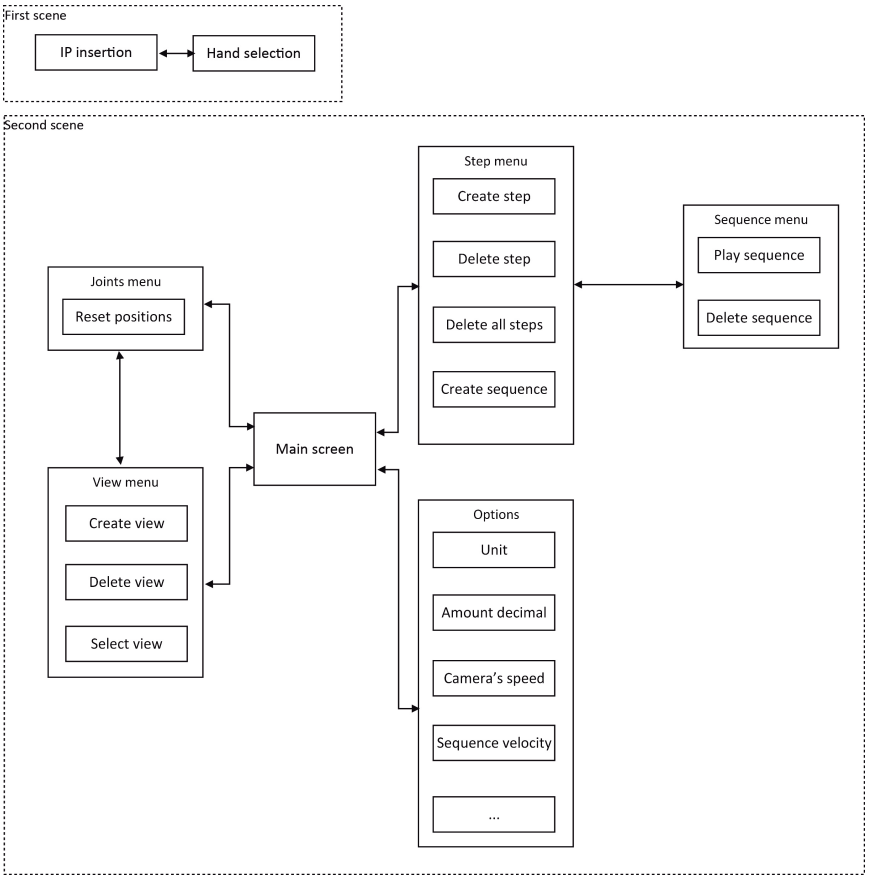

Figure 6. Scheme with client application options.

The models of the hands are stored in the system as a URDF file. The client translates the model to Unity3D and, so, it was necessary to develop a translator for URDF files, the format in the ROS for the robot modes, into one readable in Unity3D. This file contains all the necessary information for simulation of the robot.

In Figure 7, a set of hand configurations is shown. Currently, the system does not allow the combination of right and left hands of different types.

If the real system is connected, the virtual environment will sense the movements of the remote robot hand, performing and updating them in the user's virtual hand. Furthermore, if some movements are performed in the virtual environment of the robotic hand, they will be performed in the real one and updated for the rest of the users connected to the same server.

The usability of the user interfaces is dependent on the location of each option in the user screen. Some options cannot be visualized simultaneously, to avoid overlapping. Furthermore, the menus which control the joint positions are semi-transparent, to allow the user to visualize the virtual hand and control it simultaneously. These menus are automatically moved to the background when they are focused. Hand-held devices provide a tactile screen that facilitates human–robot interaction in the virtual environment. Furthermore, the user can configure the visual characteristics of the system, as well as the notation in which the joint positions are represented.

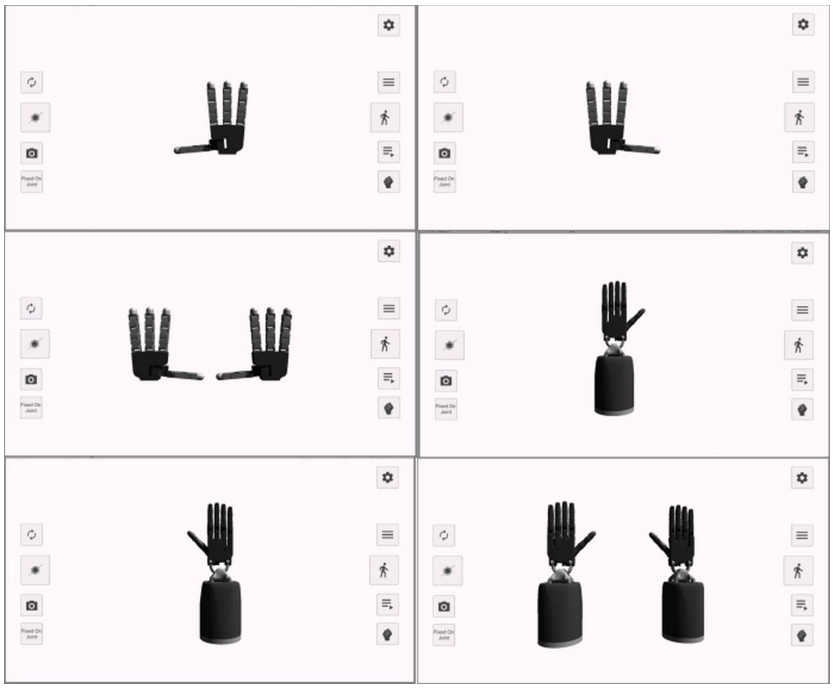

Figure 7. Combinations of robotic hands in the simulator: Allegro Hand left hand, Allegro Hand right hand, Allegro Hand both hands, Shadow hand left, Shadow hand right, and Shadow hand both hands.

The user interfaces allow for the execution of groups of movements as sequences. This provides a flexibility to control the robot hand through several movements, instead of requiring them to be performed one at a time. It is possible to exchange a sequence defined for the right hand to the left hand, and the system will automatically correct the joint positions to take into account the differences between the hands.

Furthermore, if several users are connected to the same server, they will simultaneously update their virtual environment with the current position of the hand. If any of them sends a command to the server, all user interfaces will update their simulations to consider the modifications. If one user executes a sequence in the robot hand, the other users will view it in their devices, and interaction with the server will be restricted until the sequence is completed, to avoid inconsistent commands.

In addition, the user interface has a button to open or close the hand, based on predefined sequences for each hand to allow opening or closure with a single direct indication by the user.

2.5. Server Specifications

The server was designed to control the virtual or real robot hand in a way which is transparent for the user. From the user's point of view, whether the server uses a simulation or a real hand is totally transparent. It uses the ROS nodes through the topics to connect with the hand controller provided by the manufacturer. First, it is necessary to execute the SimLab application (in the case of the Allegro Hand) or the Gazebo module (in the case of the Shadow hand).

The server contains a ROS node for publishing and subscribing SimLab's nodes (which control the Allegro Hand) or for publishing and subscribing the Gazebo module (to control the Shadow hand).

The server is designed to work with one hand at the same time. However, several servers in different machines can be started to control a set of hands, as was described in Section 2. If two hands need to be controlled by the same server (e.g., both right and left hands), it must be taken into consideration that the hand controller, in the case of the Shadow hand, does not allow several real hands to be controlled the same computer; however, the Allegro Hand allows several hands to be controlled the same computer. Taking this into consideration, if two Shadow hands must be controlled,

then two servers must be launched in different computers, one connected to each hand. Once this is performed, the main server will connect to the other server to control the other hand as a client. From the point of view of the user, the main server will control both hands in a transparent way.

Furthermore, existing libraries to perform grasping tasks, such as Graspit! [38], can be used to show, in the user devices, the performed grasping motion.

2.5.1. Real Hand Server Specifications

To connect with the real robotic hand, the Allegro Hand server requires that SimLab's server be run in the ROS. Starting the server for a real hand starts a simulation of the hand in the ROS (Rviz) in the same server computer. To modify the current position of the hand, the joint values can be changed using the topic *allegroHand_0joint_cmd*. Figure 8 shows the relations among the nodes and topics of the ROS on the server side. The movements of the real hand will be updated both in Rviz and in the remote user interfaces.

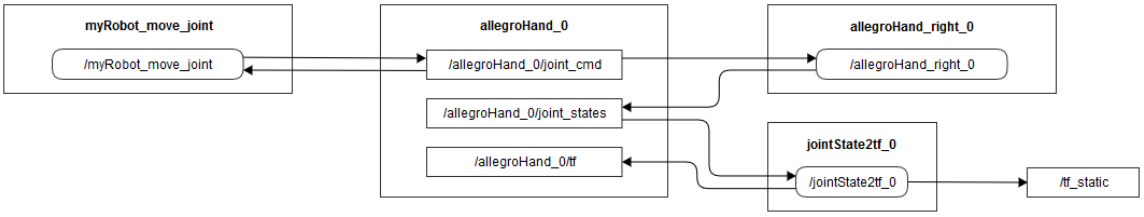

Figure 8. Nodes and topics schema after launching the Allegro server application with a real hand.

To sense the joint positions, a message of the sensor type, *joint_State*, is published. This contains the 16 positions of the joints and their respective names. The values sent by the user, which move the hand in its device, are published on the topic modifying the joint positions. The server should send this information to all connected users, except for the one which modifies the current joint values of the hand, to accomplish a collaborative update of the virtual robotic hand.

In the case of running two real Allegro hands on the same server, the ROS node scheme would be as shown in Figure 9.

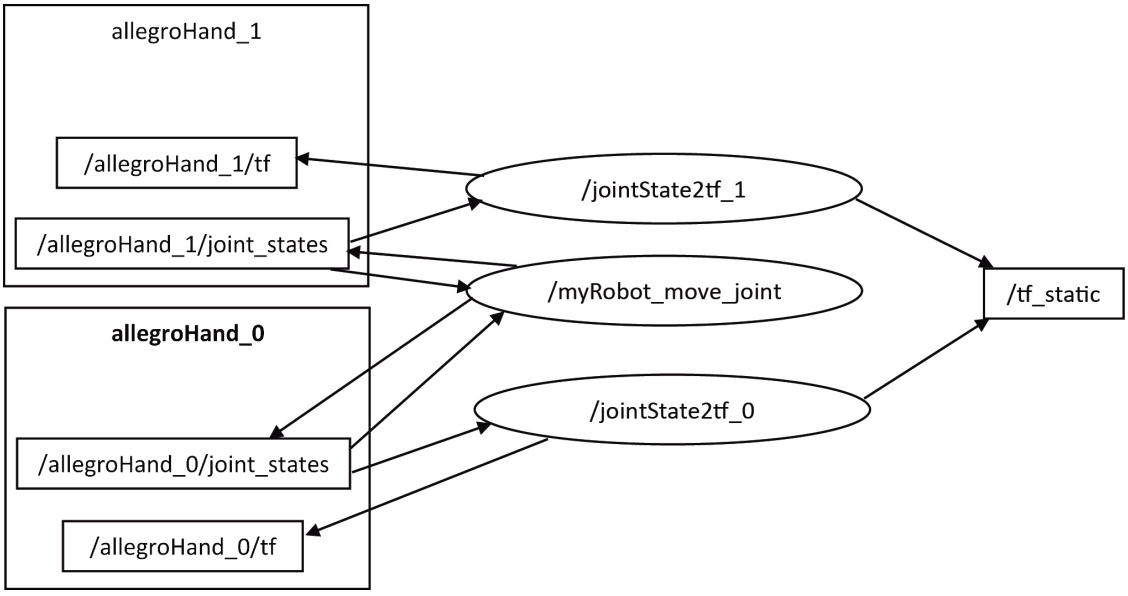

Figure 9. Nodes and topics schema after launching the server application for two Allegro hands.

Figure 10 shows the interconnection of the nodes with the connected server for the Shadow hand case. As can be seen, when the server is running, the server node *myRobot_move_joint* publishes in two topics, one for the right hand and one for the left, as they are used.

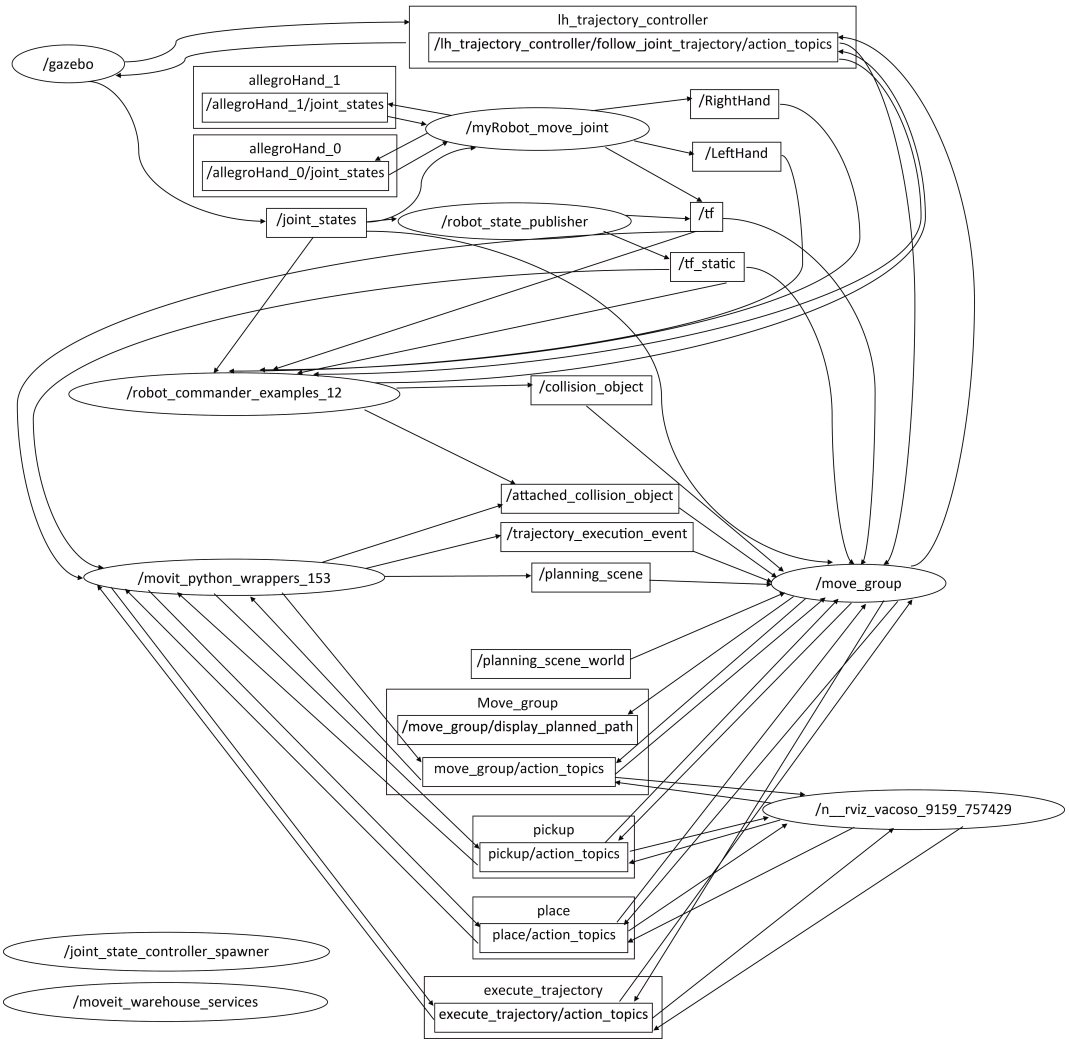

Figure 10. Nodes and topics schema after launching the Shadow server application with a real hand.

2.5.2. Virtual Hand Server Specifications

It is possible to use a server with a simulation of a hand, instead of using a real hand. From the point of view of the client, the virtual server has the same collaborative functionalities of the system as if a real robotic hand was available. With this option, the topic with the joint values is *allegroHand_0joint_states*. This mode of the server provides a Rviz simulation of the real Allegro robotic hand, which will move in the same way as the real one does.

Figure 11 shows the relationships among the nodes and topics of the ROS on the server side when a simulation of the hand is used instead of using the real one. As can be seen in Figures 8 and 11, the differences between both nodes are in the topics *allegroHand_0_joint_cmd* and *allegroHand_right_0* (for the right hand, if the control is for the left hand the topic will be *allegroHand_left_0*). These differences are presented in the controller of the real hand and exist in the controller of the simulation.

Figure 11. Nodes and topics schema after launching the Allegro server application with a virtual hand.

3. Experimentation and Discussion

Several experiments were performed to check the performance and usability of the system. For the user experience, a test was done by users after using the system. With the results of the test, improvements were made, and the new version of the system was tested again by the same users. The test was done for both users connected to a virtual hand server and for users connected to a real hand.

As described above, the user can use the simulation environment without connecting to any server. This option was not tested, as it is an independent simulation without any collaborative aspect. The usability of the application is similar to when the system is connected to any of the servers.

Experiments were performed with a set of phones and tablets with the iOS and Android operation systems, to test a wide range of devices.

This section is divided into four subsections. Section 3.1 shows the results of the user experience test. Section 3.2 shows the results with the users connected to a simulation Allegro server of the robot hand. Section 3.3 shows the results of connecting the users with the real Allegro Hand server, which included a simulation of the hand in the server, as was described in previous sections. Section 3.4 shows the results of connecting several users with the simulated Shadow dexterous hand.

3.1. Usability of the User Interface Test

The application passed two review processes by the users. Those who tested the system were students aged between 18–23 years. Overall, 16% of the students were left-handed. Taking into consideration the results of the test, the system was improved and re-tested with the same population of students. The questions given to the students were as follows:

1. Is the application easy to start?
2. Is the interface intuitive?
3. Does the virtual hand respond as expected?
4. Do the functionalities of the interface fulfill necessities?
5. Do the functionalities of the interface work well?
6. Is the connection with the real hand smooth?
7. Does the collaborative management of real the hand work well?
8. Can you use the application in your device?

Students had to answers each question using a scale between 1 and 5. On this scale, 1 represented the worst experience and 5 represented a wonderful experience. The last question was an exception, as it was a direct yes/no question. This question was designed to learn about usability in different devices. The results obtained from questions 1–7 of the test are presented in Table 3. The last question obtained the response "yes" in 100% of cases, which means that the system developed worked in a perfect way on a wide range of devices.

Table 3. Results of the questionnaire.

Round	Question 1	Question 2	Question 3	Question 4	Question 5	Question 6	Question 7
1	67.32%	71.30%	62.07%	75.53%	51.02%	55.33%	43.38%
2	92.83%	89.44%	69.28%	76.25%	75.44%	82.14%	79.14%

3.2. Test of the System Connected to a Simulated Allegro Robot Hand Server

The results of using the system with a virtual simulation of the robot hand are described in this section. All devices were tested in a collaborative way, providing a coherent refresh of the simulation. Furthermore, the control and sensing of the server virtual robot hand were performed without noticeable problems. On the server side, the Rviz simulation was used to observe the current positions of the joints. The time was supervised using the true timestamp sent by each device,

which recorded the time (t_l) spent between sending a command and obtaining the confirmation of reception by the server. Another value measured in the server was the time (t_s) required to spread a movement across the devices connected to it; that is, how much time was required once a new movement was performed in the real hand to be updated for all users connected to that server. Table 4 shows the results of these values when a virtual server was used. It should be noted that all clients were connected to the same Local Area Network (LAN) as the server. This was an IEEE802.11b network.

Table 4. Time delays using a virtual server for the robot hand.

Number of Users	Median t_l (ms)	Median t_s (ms)
1	0.7	0.0%
2	0.7	0.1%
5	0.7	0.5%
10	0.7	1.0%

As Table 4 shows, on one hand, the local delay (t_l) remained fixed because it did not depend on the number of users connected to the server, but only on the network. On the other hand, the delay of spread time (t_s) grew proportionally to the number of users connected to the server, because the server needed to send the movement information to a greater number of users.

A practical example of the application is shown in Figure 12, which shows a time sequence of execution actions in the virtual system using an Android device. This sequence shows the joint control toolbar, which was used to control the hand. In the lower-left side of each image, the Android device with the virtual hand interface is shown. In the center of the image, the server-side Rviz simulation of the robot hand is shown.

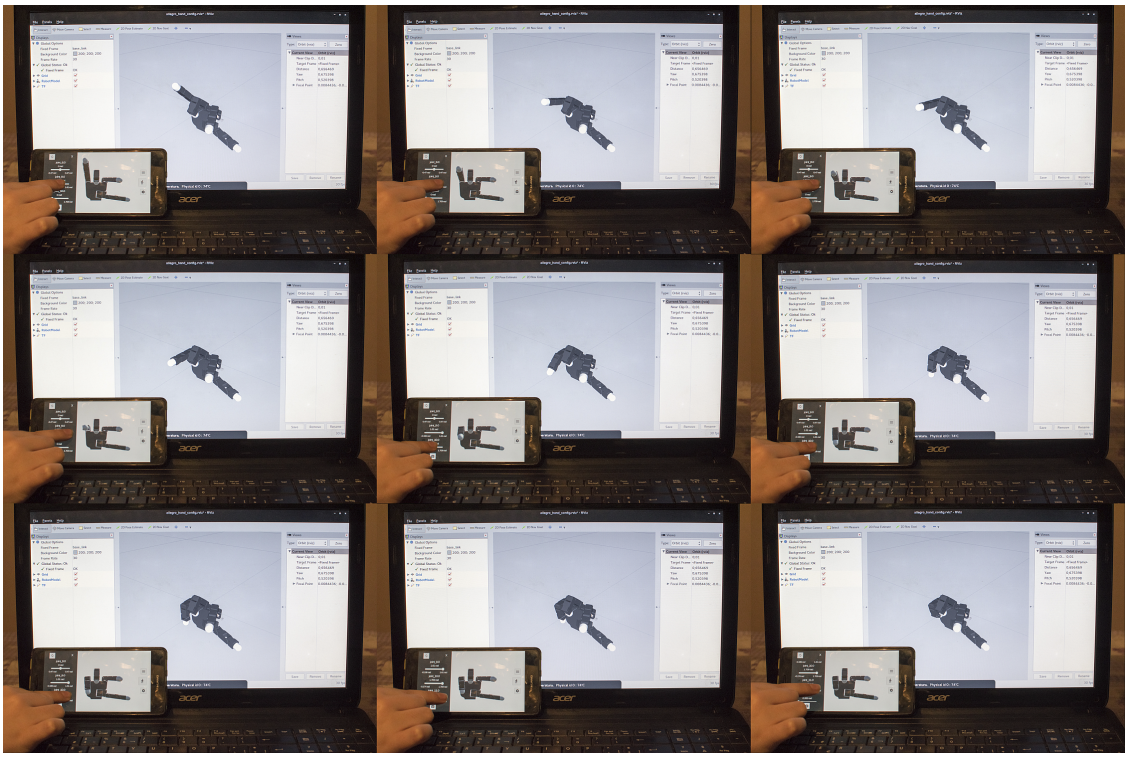

Figure 12. User interface check with the virtual server and one Android device.

The system performed the movements, as was expected. All joints and fingers performed the commands according to the user references. All users obtained feedback of the hand movements performed by any user connected to the same server.

3.3. Test of the System Connected to a Real Allegro Hand

The results for the case in which a real robot hand was connected to the server side are presented in this section. Furthermore, as was described before, the use of the real robot hand included the Rviz simulation of the robot hand by the server. All tested devices were given a concurrent refresh of the simulation, sensing the real hand without problems. Furthermore, if any user sent a new command to the server, the real hand updated its position as well, such that all users connected to the same server sensed the new position of the hand. The same parameters as those for the virtual server were tested.

3.4. Test of the System Connected to a Simulated Shadow Robot Hand Server

This section does not enter into the specification of communication delays, but instead presents the simulation test when the system was connected to the Shadow Robot Hand. Figure 13 shows the system tested, with two users interacting, in a collaborative way, with the server controlling the real robot hand. The corresponding sequence was executed by one user, to be followed by the real hand. As expected, the other user received feedback of the new position in real time. This representation was similar to the one addressed in Figure 13. In one of Figures, the simulation performed with the Allegro Hand is shown and, in the other, the simulation with the Shadow Robot Hand is shown. Furthermore, in the Allegro example, only one user was using the hand server. On the other hand, with the Shadow Robot Hand, two users were working simultaneously with the remote robot hand.

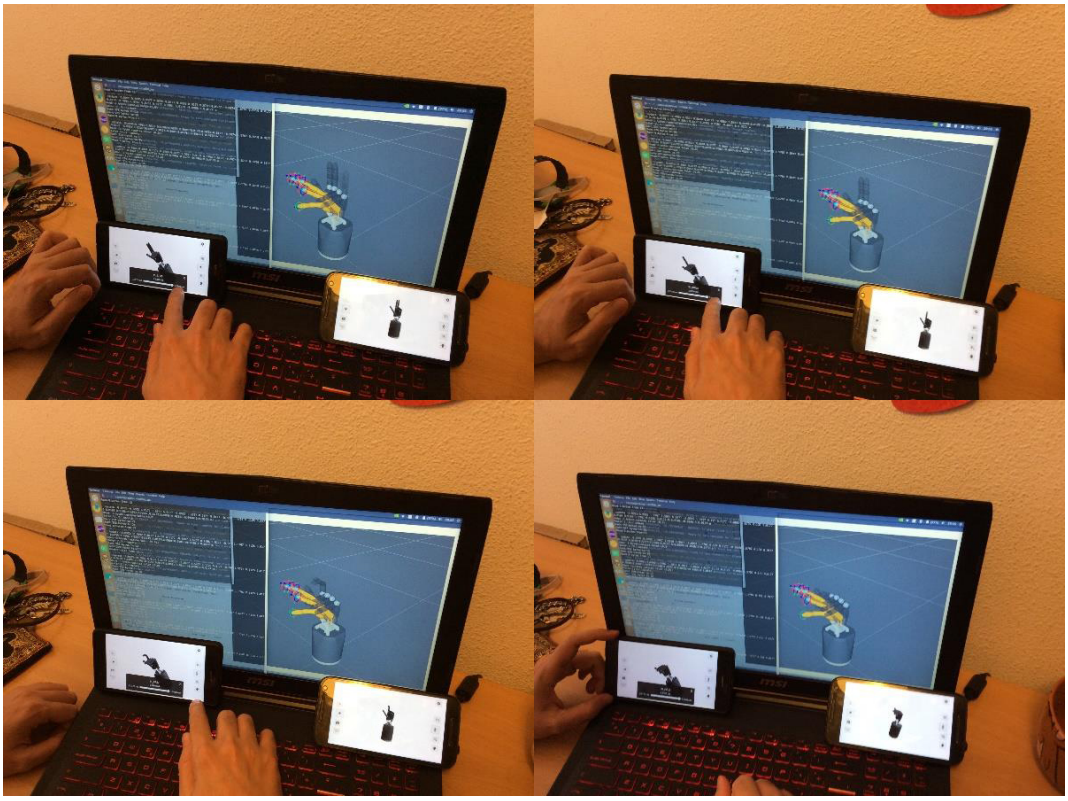

Figure 13. Two users connected simultaneously to the virtual Shadow hand server.

4. Conclusions

An architecture for a multi-device collaborative tele-operation system to control the movements of a robotic hand through a network was developed. Furthermore, the design architecture was tested with a specific robot hand. The user interface system provides a multi-device architecture and 3D simulation. The server side is based on the ROS, which allows for a proper connection among devices and the real system, providing the possibility of expanding the designed system in order to sense a wide range of robotic systems.

The system presented allows for the connection with a shared virtual remote hand or with a real one with any number of users, providing collaborative interoperability among them. The set of users can interact with the same server and, consequently, with the same robot hand, obtaining real-time feedback from the real system to all user interfaces.

A timing test showed that t_l is independent of the server used. However, t_s was found to be dependent on the number of users and was greater for the real system than for the virtual one. This increase in time was due the necessity for sending the data to the real system.

Furthermore, as control of the robot hand was performed in the ROS, any application that generates grasping movements in the ROS can be connected to the server in order to control the robot hand; the user will obtain feedback through the virtual environment.

Integration of the designed system with other robot hands, as well as connecting the system with other tools to generate control commands for the fingers are works in progress. Furthermore, the possibility of having more than one hand in the server is another point which is being studied to reduce the number of different servers required for each hand to be shared among users.

As a future work, alternative ways of reducing communication delays will be addressed, to optimize the delay when a set of users is connected to the same hand server.

To summarize, the main novelty of the system presented is that it provides a usable interface for the remote sensing of a robot hand from a collaborative environment, allowing for scaling of the system with more functionalities, according to the desired necessities, and the connection of portable devices to the ROS.

Author Contributions: Conceptualization, S.T.P.; methodology, F.T.; software, L.M.; writing—original draft, S.T.P.; writing—review and editing, F.A.C.

Funding: This research was funded by Ministerio de Ciencia, Innovación y Universidades grant number RTI2018-094279-B-100.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Casper, J.; Murphy, R.R. Human-robot interactions during the robot-assisted urban search and rescue response at the world trade center. *IEEE Trans. Syst. Man Cybern. Part B (Cybern.)* **2003**, *33*, 367–385. [CrossRef] [PubMed]
2. Jonggil, A.; Gerard Jounghyun, K. SPRinT: A Mixed Approach to a Hand-Held Robot Interface for Telepresence. *Int. J. Soc. Robot.* **2018**, *10*, 537–552.
3. Alepis, E.; Sakelliou, A. Augmented car: A low-cost augmented reality RC car using the capabilities of a smartphone. In Proceedings of the 7th International Conference on Information, Intelligence, Systems & Applications (IISA), Chalkidiki, Greece, 13–15 July 2016.
4. Stoll, B.; Reig, S.; He, L.; Kaplan, I.; Jung, M.F.; Fussell, S.R. Wait, Can You Move the Robot? Examining Telepresence Robot Use in Collaborative Teams. In Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction, Chicago, IL, USA, 5–8 March 2018; pp. 14–22.
5. Xiang, C.; Guo, J.; Rossiter, J. Soft-smart robotic end effectors with sensing, actuation, and gripping capabilities. *Smart Mater. Struct.* **2019**, *28*, 055034. [CrossRef]
6. Li, Y.; Chen, Y.; Li, Y. Pre-Charged Pneumatic Soft Gripper with Closed-Loop Control. *IEEE Robot. Autom. Lett.* **2019**, *4*, 1402–1408. [CrossRef]
7. Khan, A.; Li, J.; Malik, A.; Yusuf, K.M. Vision-Based Inceptive Integration for Robotic Control. In *Soft Computing and Signal Processing*; Springer: Singapore, 2019; pp. 95–105.
8. SimLab. Allegro Hand Overview. Available online: http://wiki.wonikrobotics.com/AllegroHandWiki/index.php/Allegro_Hand_Overview (accessed on 30 April 2019).
9. Li, M.; Yin, H.; Tahara, K.; Billard, A. Learning object-level impedance control for robust grasping and dexterous manipulation. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China, 31 May–7 June 2014; pp. 6784–6791.

10. Hang, K.; Li, M.; Stork, J.A.; Bekiroglu, Y.; Pokorny, F.T.; Billard, A.; Kragic, D. Hierarchical Fingertip Space: A Unified Framework for Grasp Planning and In-Hand Grasp Adaptation. *IEEE Trans. Robot.* **2016**, *32*, 960–972. [CrossRef]
11. Tomo, T.P.; Schmitz, A.; Wong, W.K.; Kristanto, H.; Somlor, S.; Hwang, J.; Jamone, L.; Sugano, S. Covering a Robot Fingertip with uSkin: A Soft Electronic Skin with Distributed 3-Axis Force Sensitive Elements for Robot Hands. *IEEE Robot. Autom. Lett.* **2018**, *3*, 124–131. [CrossRef]
12. Or, K.; Schmitz, A.; Funabashi, S.; Tomura, M.; Sugano, S. Development of robotic fingertip morphology for enhanced manipulation stability. In Proceedings of the IEEE International Conference on Advanced Intelligent Mechatronics (AIM), Banff, AB, Canada, 12–15 July 2016; pp. 25–30.
13. Or, K.; Tomura, M.; Schmitz, A.; Funabashi, S.; Sugano, S. Interpolation control posture design for in-hand manipulation. In Proceedings of the IEEE/SICE International Symposium on System Integration (SII), Nagoya, Japan, 11–13 December 2015; pp. 187–192.
14. Jara, C.A.; Pomares, J.; Candelas, F.A.; Torres, F. Control Framework for Dexterous Manipulation Using Dynamic Visual Servoing and Tactile Sensors' Feedback. *Sensors* **2014**, *14*, 1787–1804. [CrossRef]
15. Torres, A.; Candelas Herías, F.A.; Mira, D.; Torres Medina, F. DM-UAV: Dexterous Manipulation Unmanned Aerial Vehicle. In Proceedings of the ICAART, Porto, Portugal, 24–26 February 2017; pp. 153–158.
16. Liu, F.; Kim, Y.B.; Yee, G.K.; You, W.S.; Kang, G.T.; Kim, A.N.; Lee, Y.H.; Moon, H.; Koo, J.C.; Choi, H.R. Computation of minimum contact forces of multifingered robot hand with soft fingertips. *Intell. Serv. Robot.* **2015**, *8*, 225–232. [CrossRef]
17. Saxena, S.; Sanchez, G.; Pecht, M. Batteries in Portable Electronic Devices: A User's Perspective. *IEEE Ind. Electron. Mag.* **2017**, *11*, 35–44. [CrossRef]
18. Sánchez-Alonso, R.E.; Ortega-Moody, J.; González-Barbosa, J.; Reyes-Morales, G. Use of Platforms for the Development of Virtual Applications in the Modeling of Robot Manipulators. *Rev. Iberoam. Autom. Inf. Ind. RIAI* **2017**, *14*, 279–287. [CrossRef]
19. Berg, L.; Vance, J. Industry use of virtual reality in product design and manufacturing: A survey. *Virtual Real.* **2017**, *21*, 1–17. [CrossRef]
20. Nilsson, N.C.; Serafin, S.; Laursen, M.H.; Pedersen, K.S.; Sikstrom, E.; Nordahl, R. Tapping-in-place: Increasing the naturalness of immersive walking-in-place locomotion through novel gestural input. In Proceedings of the IEEE Symposium on 3D User Interfaces, Orlando, FL, USA, 16–17 March 2013; pp. 31–38.
21. Andaluz, V.H.; Chicaiza, F.A.; Gallardo, C.; Quevedo, W.X.; Varela, J.; Sánchez, J.S.; Arteaga, O. Unity3D-MatLab Simulator in Real Time for Robotics Applications. In Proceedings of the Augmented Reality, Virtual Reality, and Computer Graphics: 3rd International Conference, AVR 2016, Lecce, Italy, 15–18 June 2016; Volume 9768, pp. 246–263.
22. Quigley, M.; Conley, K.; Gerkey, B.P.; Faust, J.; Foote, T.; Leibs, J.; Wheeler, R.; Ng, A.Y. ROS: An open-source Robot Operating System. In Proceedings of the ICRA Workshop on Open Source Software, Kobe, Japan, 17 May 2009.
23. Unity. Available online: https://unity.com/ (accessed on 30 April 2019).
24. Craighead, J.; Murphy, R.; Burke, J.; Goldiez, B. A Survey of Commercial & Open Source Unmanned Vehicle Simulators. In Proceedings of the IEEE International Conference on Robotics and Automation, Roma, Italy, 10–14 April 2007; pp. 852–857.
25. Meng, W.; Hu, Y.; Lin, J.; Lin, F.; Teo, R. ROS + Unity: An Efficient High-Fidelity 3D Multi-UAV Navigation and Control Simulator in GPS-Denied Environments. In Proceedings of the 41st Annual Conference of the IEEE Industrial Electronics Society, IECON 2015, Yokohama, Japan, 9–12 November 2015; pp. 2562–2567.
26. Craighead, J.; Burke, J.; Murphy, R. Using the Unity Game Engine to Develop SARGE: A Case Study. In Proceedings of the 2008 Simulation Workshop at the International Conference on Intelligent Robots and Systems (IROS 2008), Nice, France, 22–26 September 2008; Volume 4552.
27. Salvietti, G.; Gioioso, G.; Malvezzi, M.; Prattichizzo, D.; Serio, A.; Farnioli, E.; Gabiccini, M.; Bicchi, A.; Santello, M.; Bianchi, M.; et al. Hand synergies: Integration of robotics and neuroscience for understanding the control of biological and artificial hands. *Phys. Life Rev.* **2016**, *17*, 1–23.
28. Codd-Downey, R.; Mojiri Forooshani, P.; Speers, A.; Wang, H.; Jenkin, M. From ROS to Unity: Leveraging robot and virtual environment middleware for immersive teleoperation. In Proceedings of the IEEE International Conference on Information and Automation (ICIA), Hailar, China, 28–30 July 2014; pp. 932–936.

29. Koga, M.; Yunoki, T. 3D Visualization Tool for Robot in URDF on Android Devices. In Proceedings of the Japan Joint Automatic Control Conference, Hamburg, Germany, 28 September–2 October 2015; Volume 59; pp. 339–342.
30. Faust, O. Android based teleoperation for the finch robot. *ICTACT J. Commun. Technol.* **2016**, *7*, 1334–1340.
31. Yan, B.; Shi, D.; Wei, J.; Pan, C. HiBot: A generic ROS-based robot-remote-control framework. In Proceedings of the 2nd Asia-Pacific Conference on Intelligent Robot Systems (ACIRS), Wuhan, China, 16–18 June 2017; pp. 221–226.
32. Crick, C.; Jay, G.; Osentoski, S.; Pitzer, B.; Jenkins, O.C. Rosbridge: ROS for Non-ROS Users. In Proceedings of the Robotics Research: The 15th International Symposium ISRR, Flagstaff, AZ, USA, 9–12 December 2011; Christensen, H.I., Khatib, O., Eds.; Springer: Berlin, Germany, 2017; pp. 493–504.
33. Codd-Downey, R.; Jenkin, M. RCON: Dynamic Mobile Interfaces for Command and Control of ROS-enabled Robots. In Proceedings of the 12th International Conference on Informatics in Control, Automation and Robotics (ICINCO), Colmar, France, 21–23 July 2015; Volume 2, pp. 66–73.
34. Puente, S.T.; Más, L.; Torres, F.; Candelas, C. Teleoperation of robotic hands through virtualization in Unity 3D for mobile device. In Proceedings of the Spanish Robotics Conference, Madrid, Spain, 26–29 November 2019; pp. 169–174.
35. Shadow Dexterous Hand. Available online: https://www.shadowrobot.com/products/dexterous-hand/ (accessed on 30 April 2019).
36. Bae, J.H.; Park, S.W.; Kim, D.; Baeg, M.H.; Oh, S.R. A Grasp Strategy with the Geometric Centroid of a Groped Object Shape Derived from Contact Spots. In Proceedings of the 2012 IEEE International Conference on Robotics and Automation (ICRA2012), Saint Paul, MN, USA, 14–18 May 2012; pp. 3798–3804.
37. Aguero, C.; Koenig, N.; Chen I.; Boyer, H.; Peters, S.; Hsu, J.; Gerkey, B.; Paepcke, S.; Rivero, J.; Manzo, J.; et al. Inside the virtual robotics challenge: Simulating real-time robotic disaster response. *IEEE Trans. Autom. Sci. Eng.* **2015**, *12*, 494–506. [CrossRef]
38. Miller, A.; Allen, P.K. Graspit! A Versatile Simulator for Robotic Grasping. *IEEE Robot. Autom. Mag.* **2004**, *11*, 110–122. [CrossRef]