Safe Teleoperation of a dual hand-arm robotic system

Jan Rosell[1⋆], Raúl Suárez[1], and Alexander Pérez[2]

[1] Institute of Industrial and Control Engineering, Universitat Politècnica de Catalunya, Barcelona, Spain, {jan.rosell, raul.suarez}@upc.edu
[2] Escuela Colombiana de Ingeniería "Julio Garavito", Bogotá D.C., Colombia, alexander.perez@escuelaing.edu.co

Abstract. This paper introduces a supervised teleoperation system to cope with some of the main problems that arise in the teleoperation of hand-arm robotic systems. The set-up consists of two magnetic trackers and two sensorized gloves that command, respectively, two industrial robots and the mechanical hands with which they are equipped. The basic mapping aspects both at hand and arm level are discussed, as well as the communication issues related with the implementation done based on ROS (Robot Operating System). The risk of collisions and of reaching singular configurations, either internal or due to the workspace limits, is controlled by monitoring the state of the hand-arm system and stoping it when necessary. An automatic re-synchronization procedure allows to resume the teleoperation as soon as the risk has disappeared, and also permits the user to change the mapping when his/her posture becomes uncomfortable. The resulting system allows an intuitive, simple and safe teleoperation of a dual hand-arm robotic system.

1 Introduction

Despite the advances in the development of autonomous robots, with the robots taking decisions by themselves, the teleoperation of robots has a significant role in applications in which the decisions are still taken by an operator and the robots perform the actions following the movements of the operator. Advances in teleoperation of robots were based on the advances in the robot control systems, the improvements in the velocity and quality of the communications, and the development of devices to capture in a reasonable time and with a reasonable precision the gestures (or intentions) of the operator. A detailed discussion of the problems related with teleoperation as well as a description of typical applications was presented in [1].

One of the problems related with the teleoperation of anthropomorphic devices is the complexity generated by the large number of involved degrees of

⋆ Work partially supported by the Spanish Government through the projects DPI2010-15446 and DPI2011-22471. The authors want to thank Riccardo Rocchio, Armando Palumbo and Dario Davide for the implementation of the ROS nodes of the proposed system.

freedom (*dof*), e.g. up to 27 for the case of a 6 *dof* arm plus 21 of a hand with 5 fingers with 4 independent dof each one plus one additional *dof* in the palm. Therefore, the most intuitive way to command these devices is to capture the movements of a human operator and map them to the robotic system. But even in this case, the kinematic structure of the device is, in general, different from that of the operator. This means that the mapping of the sensorial information obtained from the operator movements onto the movement space of the devices is not an easy issue. This has lead to different mapping approaches, usually depending on the available hardware, but, in any case, quite frequently the operator has to arrive to non comfortable positions in order to command a robotic hand-arm system towards the proper configuration according to a visual feedback. One immediate consequence of this is that the operator has difficulties to properly execute a task with the teleoperated robot, even when the task looks like being trivial, and this easily ends with the teleoperated robot producing undesired collisions. One solution to this problem is the implementation of virtual constraints that do not allow the robot to move beyond some acceptable limits.

Three main approaches have been presented related with the mapping issues in teleoperation of anthropomorphic devices, particularly when anthropomorphic hands are involved [2][3][4]: *Joint-to-joint mapping*, in which each sensor of the operator movements is directly associated with a joint of the robot (e.g. [5]), *Pose mapping*, in which each pose of the operator is associated with a predefined pose of the robot (e.g. [6]), and *Point-to-point mapping*, in which the positions of particular points of the operators arms or hands are replicated by predefined points on the robot (e.g. [7]).

The identification of the pose of the operator to do teleoperation of anthropomorphic devices has been addressed using different types of sensorial information. The two most frequent approaches are based on vision systems, which is useful when information about the gesture of the operator is enough without needing precise positioning and the chance of occlusions is minimal (e.g. [8][9][10]), and based on the used of trackers and sensorized gloves, which quickly provides information about the joints of the operator and can be used for anthropomorphic and non anthropomorphic devices (e.g. [7][11][5]).

In this context, this paper deals with the problem of teleoperating a dual hand-arm robotic system and presents the implementation of a teleoperation approach, including the teleoperation of the arms (with a Point-to-point mapping) and the hands (with a Joint-to-joint mapping). The approach tries to be intuitive and robust at the same time and also cares about security aspects, limiting the movements of the robotic system to avoid collisions or undesired configurations.

After this introduction, the paper is structured as follows. Section 2 briefly presents a system overview and then Section 3 describes the hardware set-up. Sections 4, 5 and 6 deal, respectively, with the mapping, the communication and the supervision issues of the implementation. Finally, Section 7 summarizes the work.

2 System overview

The teleoperation system proposed in this work consists of two industrial robots equipped with mechanical hands that are to be teleoperated by an operator equipped with wrist trackers and sensorized gloves. The main goal of the proposal is that the system:

- Allows an intuitive teleoperation, both at the hand level (with respect to the mapping between the joints of the glove and those of the mechanical hand), and also at the robot level (assuming the possibility to use different cameras to feed back the image from the remote site).
- Allows a safe teleoperation stopping the robot when necessary to avoid collisions and singular configurations, and automatically and smoothly resuming the teleoperation to recover from these situations.

The logic schema of the system is shown in Fig. 1. The basic modules are:

- *Synchronization*: Required to synchronize the data provided independently by the wrist trackers and the sensorized gloves.
- *Mapping*: Required to transform the data provided by the trackers and gloves into joint values for the robots and the mechanical hands. This module has to verify wether the obtained configuration is kinematically correct, i.e. not singular and within limits.
- *Collision Validation*: Required to verify that the commanded configuration is collision-free.

To control all the devices and manage the communications between them and all the teleoperation modules, the Robot Operation System (ROS, www.ros.org) is used.

3 Hardware set-up

3.1 Magnetic trackers

Two different trackers of different brands were used in this work to capture the position and orientation of the operator wrist with respect to a global reference frame, allowing a mapping of the displacements of the user arm to the robot arm. One tracker is a *flock of birds* manufactured by Ascension Technology and the other a *fastrack* manufactured by Polhemus, both are shown in Figure 1 above the boxes with the corresponding names. Both trackers are prepared to be used together with sensorized gloves (described below). These devices are of magnetic type, measuring the position and orientation of a small sensor with respect to a base fixed in the workspace. The *flock of birds* has a static precision of 1.8 mm for the position and 0.5° for the orientation and allows 144 samples per second, and the *fastrack* has a static precision of 0.8 mm for the position and 0.15° for the orientation and allows 120 samples per second.

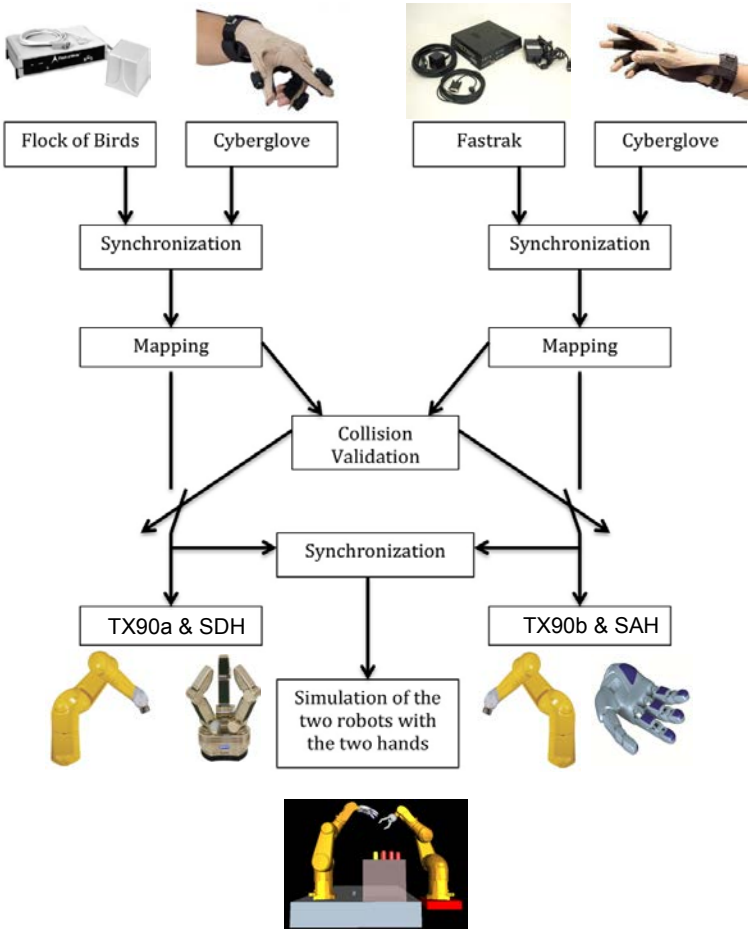

Fig. 1. System overview.

3.2 Industrial robots

The industrial robots are two Stäubli TX90, called TX90a and TX90b, are shown in Figure 1 below the boxes with the corresponding names. These are general purpose 6 *dof* serial robot arms with revolute joints. Each robot is equipped with a CS8C controller, that runs a real time operative system VxWorks, and over it the command interpreter VAL3 language facilitates the robot-level programming. The nominal capacity of this robotic arm is 7 kg and the workspace has a radius of approximately 1 m.

3.3 Sensorized gloves

Two sensorized gloves Cyberglove from Immersion Corporation were used to capture the hand configurations, i.e. the positions of the operator fingers. Two

pictures of the glove are shown in Figure 1 above the boxes with the corresponding name. Cyberglove is a fully instrumented glove that provides 22 joint-angle measurements using resistive bend-sensing technology, including three flexion sensors per finger, four abduction sensors between the fingers, a palm-arc sensor, and two sensors to measure the flexion and the abduction of the wrist.

3.4 Mechanical hands SAH and SDH

Two different mechanical hands were used in this work, the Schunk Anthropomorphic Hand (SAH) and the Schunk Dexterous Hand (SDH), which are shown in Figure 1 below the boxes with the corresponding names. Both hands can be attached to the industrial robots through an industrial standard interface EN ISO 9404-1-50.

The SAH has four identical fingers with four joints each one (abduction, proximal flexion, medium flexion and distal flexion), being one of them prepared to work as the opposing thumb, which is equipped with an additional joint (thumbbase joint) that moves the whole thumb with respect to the palm. In all the fingers the distal and medium flexion joints are mechanically coupled, thus there is a total of 17 joints with only 13 independent *dof*. The weigh of the hand is 2.2 kg and the size is around 1.4 times the average human hand. The maximum regular force at the fingertips is approximately 5 N. The hand requires a power supply of 24 V with a regular power of 1 A when it is unload and a maximum power of 2.7 A when it is applying grasping forces.

The SDH has identical fingers with two joints each one (proximal flexion and distal flexion) and two of them have an additional *dof* that makes the fingers rotate around their bases to put them exactly one in front of the other, thus, the whole hand has 7 independent *dof*. Although it is not anthropomorphic it is suitable for many different grasping actions. The weigh of the hand is 1.95 kg and the size is also around 1.4 times the average human hand. The hand requires a power supply of 24 V with a maximum power of 5 A when it is applying grasping forces. It is also equipped with two arrays of tactile sensors per finger, but these are not used in this work.

4 Mapping issues

4.1 Tracker-robot

Let H be the reference frame of the workspace of the operator, $\mathcal{W}_H$, and W that of the robot workspace, $\mathcal{W}_W$. Let also HIP be the reference frame of the tracker located at the wrist of the operator, and TCP be the reference frame at the end-effector of the robot. Then, the mapping between the HIP and the TCP is based on the procedure proposed in [12] that maps $\mathcal{W}_H$ into $\mathcal{W}_W$ to obtain the virtual operator workspace ($\mathcal{W}_{\mathsf{vH}}$, with reference frame vH), following two guidelines:

– The orientation of vH must be the same as that of the camera used to feed back the video (Fig. 2 left).

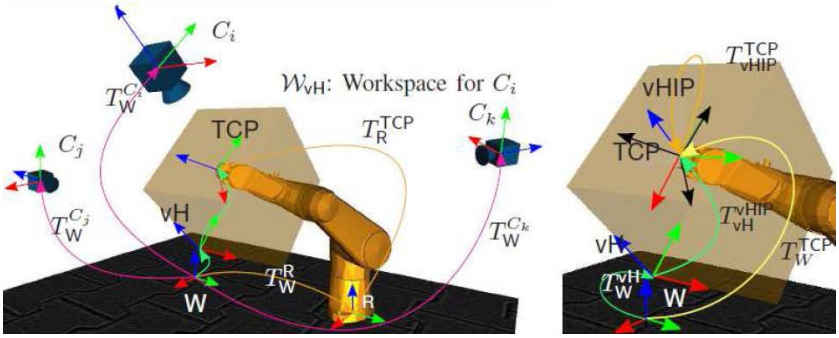

Fig. 2. (left) Model of the remote site showing the virtual operator workspace aligned with camera i; (right) Virtual coupling attained by making the origins of frames vHIP and TCP coincident. Figures reproduced from [12].

– The origin of the HIP within vH, called vHIP, must coincide with that of the TCP (Fig. 2 right).

While teleoperating, the location of the HIP with respect to H, i.e. the transform $T_{\mathsf{H}}^{\mathsf{HIP}}$, is continually read from the magnetic tracker, and the transform $T_{\mathsf{vH}}^{\mathsf{vHIP}}$ is made coincident with it (or modified by a simple scale factor) and used to obtain the location of the TCP with respect to W (Fig. 2 right):

$$T_W^{\mathsf{TCP}} = T_{\mathsf{W}}^{\mathsf{vH}} \cdot T_{\mathsf{vH}}^{\mathsf{vHIP}} \cdot T_{\mathsf{vHIP}}^{\mathsf{TCP}}, \tag{1}$$

where the transforms $T_{\mathsf{W}}^{\mathsf{vH}}$ and $T_{\mathsf{vHIP}}^{\mathsf{TCP}}$ are fixed and set when teleoperation starts:

$$\begin{aligned} T_{\mathsf{vHIP}}^{\mathsf{TCP}} &= \left(\begin{array}{c|c} \mathcal{R}_{\mathsf{vHIP}}^{\mathsf{TCP}} & \mathbf{0} \\ \hline 0 & 1 \end{array}\right) \\ \mathcal{R}_{\mathsf{vHIP}}^{\mathsf{TCP}} &= (\mathcal{R}_{\mathsf{vH}}^{\mathsf{vHIP}})^{-1} \cdot (\mathcal{R}_{\mathsf{W}}^{\mathsf{Ci}})^{-1} \cdot \mathcal{R}_{\mathsf{W}}^{\mathsf{TCP}} \\ T_{\mathsf{W}}^{\mathsf{vH}} &= T_W^{\mathsf{TCP}} \cdot (T_{\mathsf{vHIP}}^{\mathsf{TCP}})^{-1} \cdot (T_{\mathsf{vH}}^{\mathsf{vHIP}})^{-1} \end{aligned} \tag{2}$$

being Ci the reference system of the camera that feeds back the image from the remote site.

Besides the initialization at the start time, the establishment of the virtual connection between the tracker and the robot, as stated by Eq. (2), is also done in the following situations encountered while teleoperation is executed:

– The operator changes the reference camera to have a better view of the robot workspace, or any other configuration parameter like the scaling.
– The operator stops the teleoperation and resumes it in another (more comfortable) position (this is done by pressing the clutch button on the tracker).
– The inverse kinematics cannot be calculated because the robot is placed at a singular configuration.
– The teleoperation was stopped because an imminent collision or a singular configuration was detected, and it is resumed from a different safe commanding position.

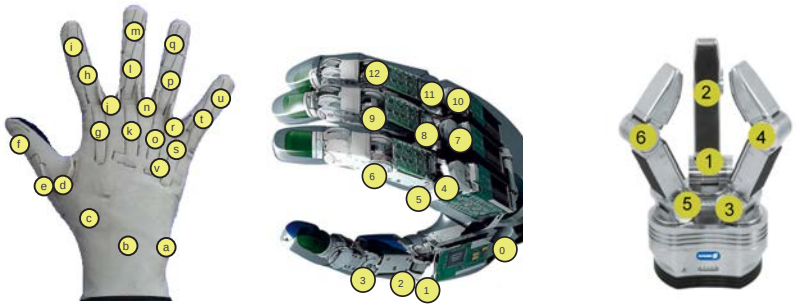

Fig. 3. Cyberglove with the sensor labels and SAH (middle) and SDH (right) with the joint labels.

4.2 Glove-mechanical hands

A particular mapping was implemented for each of the mechanical hands. The mapping from the glove to the SAH is more direct since it is anthropomorphic, while the mapping to the SDH was thought to allow a simple way of commanding of the hand.

Mapping Glove-SAH. The mapping from the glove sensors to the SAH movements is based on a previous work [13], where different types of mapping are presented for this glove and this hand. Basically, the following issues are considered (the sensors of the glove and the joints of the hand are labeled as shown in Fig. 3):

- The palm of the mechanical hand is rigid and therefore the palm arc sensor v and the wrist flexion and abduction sensors a and b are ignored.
- The mechanical hand lacks the little finger, thus the sensors u, t, s and r are ignored.
- The mechanical hand has a coupling between the medium and distal joints of each finger, therefore the distal sensors i, m and q are ignored.
- The use of sensor c to command the hand joint 1 produces, in practice, a more natural motion of the SAH than using sensor d, therefore sensor c is used for both joints 0 and 1.
- The abduction movements are measured as relative movements in the glove, with sensors j and n providing, respectively, the angle between the index and the middle fingers and the angle between the middle and the ring fingers, while the abduction is measured as the absolute movement of each finger in the mechanical hand. Then, the mapping is done using the middle finger of SAH as a fixed reference (joint 7 is fixed to 0) and then sensors j and n are directly associated to joints 4 and 10 respectively.

Then, with this mapping only 11 out of the 22 available sensor in the glove are actually used to command the hand, even when the hand has 13 *dof*. The complete mapping is shown in Table 1.

Cyberglove sensors		SAH joints	
label	name	label	name
c	thumb base	0	thumb base
c	thumb base	1	thumb abduction
e	thumb medium-flexion	2	thumb proximal-flexion
f	thumb distal-flexion	3	thumb medium-flexion
j	index-middle abduction	4	index abduction
g	index proximal-flexion	5	index proximal-flexion
h	index medium-flexion	6	index medium-flexion
-	middle abduction	7	middle abduction
k	middle proximal-flexion	8	middle proximal-flexion
l	middle medium-flexion	9	middle medium-flexion
n	middle-ring abduction	10	ring abduction
o	ring proximal-flexion	11	ring proximal-flexion
p	ring medium-flexion	12	ring medium-flexion

Table 1. Mapping from the Cyberglove to the SAH.

Mapping Glove-SDH. The mapping from the glove sensors to the SDH movements is also a joint-to-joint mapping, but it is not as evident as the previous one because the hand SDH is not anthropomorphic and both the number and the structure of the *dof* are different. The direct association of one finger of the glove with one finger of the mechanical hand was ruled out after some initial experiments, since it is not easy for the human operator to intuitively move the hand to make the mechanical hand doing the desired actions. Instead, for grasping applications, a simple mapping associating the three fingers of the SDH to only one single finger of the glove produced acceptable results. This mapping ignore some possible manipulation configurations of the SDH, but makes it really easy for the operation to exploit some others, the mechanical hand is opened or closed in a predefined grasp configuration when the operator makes an extension or flexion of the index finger. The complete mapping is shown in the Table 2 (see Fig. 3 for the labels of the sensors of the glove and the joints of the hand). The joint 0 of the SDH is not associated with any joint of the glove, instead it is fixed a priori depending on the desired grasp configuration of the hand.

Cyberglove sensors		SDH joints	
label	name	label	name
g	index proximal-flexion	1	finger 1 - proximal
h	index medium-flexion	2	finger 1 - distal
g	index proximal-flexion	3	finger 2 - proximal
h	index medium-flexion	4	finger 2 - distal
g	index proximal-flexion	5	finger 3 - proximal
h	index medium-flexion	6	finger 3 - distal

Table 2. Mapping from the Cyberglove to the SDH.

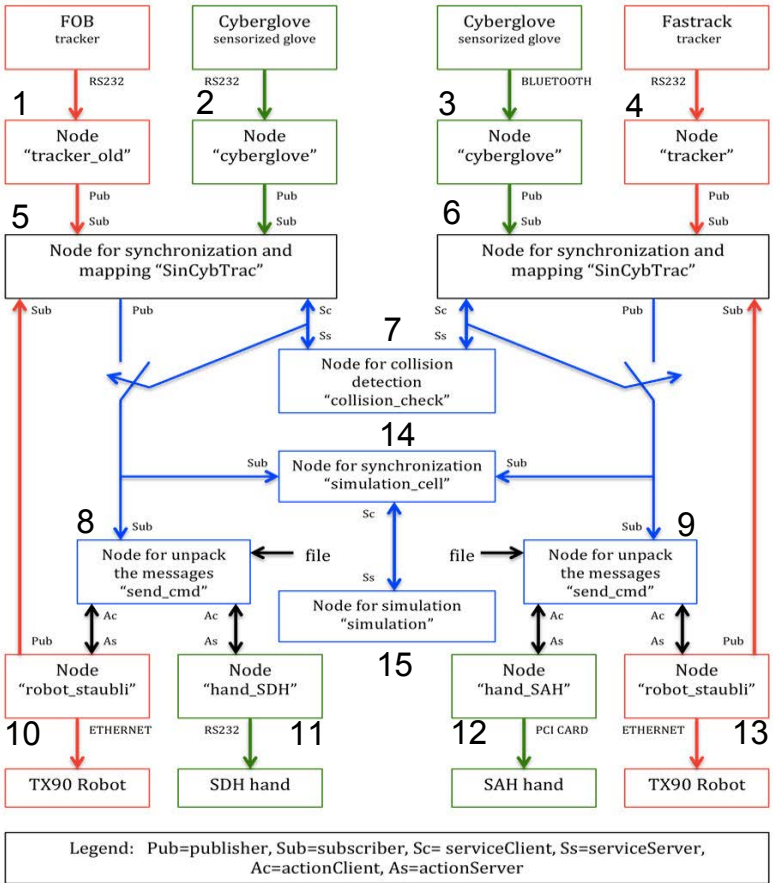

Fig. 4. ROS nodes.

5 Communication issues

5.1 The Robot Operating System (ROS)

The Robot Operating System (ROS) is a meta-operating system for robotic applications that includes hardware abstraction, low-level device control, implementation of commonly-used functionalities, message-passing between processes, package management, and provides tools and libraries for obtaining, building, writing, and running code across multiple computers.

Robotic applications are composed of several processes loosely coupled, connected as nodes of a peer-to-peer network (the ROS runtime graph) using different types of communications:

– **Topic:** Communication based on the Publisher/Subscriber mechanism. It is an asynchronous communication of strongly typed data (the *messages*)

#	Name	Purpose	Communication
1	tracker_old	broadcasts the data of the FOB tracker	Publish Topic
2	cyberglove	broadcasts the data of the Cyberglove (left hand)	Publish Topic
3	cyberglove	broadcasts data of the Cyberglove (right hand)	Publish Topic
4	tracker	broadcasts data of the Fastrack tracker	Publish Topic
5	SinCybTrac	synchronizes (left) glove and FOB data maps to SDH and TX90a configuration queries collision-check broadcasts a correct configuration	Subscribe Topic – – – – Service Client Publish Topic
6	SinCybTrac	synchronizes (right) glove and Fastrack data maps to SAH and TX90b configuration queries collision-check broadcasts a correct configuration	Subscribe Topic – – – – Service Client Publish Topic
7	collision_check	performs a collision-check test	Service server
8	send_cmd	reads valid hand-arm data commands the SDH hand and the TX90a robot	Subscribe Topic Action clients
9	send_cmd	reads valid hand-arm data commands the SAH hand and the TX90b robot	Subscribe Topic Action clients
10	robot_staubli	moves the TX90a robot broadcasts the TX90a robot position	Action server Publish Topic
11	hand_SDH	moves the SDH hand	Action server
12	hand_SAH	moves the SAH hand	Action server
13	robot_staubli	moves the TX90b robot broadcasts the TX90b robot position	Action server Publish Topic
14	simulation_cell	gets the two sets of hand-arm data sends a move command to the simulator	Subscribe Topics Service Client
15	simulation	moves the simulated system	Service Server

Table 3. ROS nodes, purpose and type of communication used.

between a node that produces the information and publishes it under a *topic*, and the node(s) that subscribe to that topic to consume this information.

- **Service:** Communication based on the Client/Server mechanism. It is a synchronous communication between a node (the client) that sends a request message to another node (the server) and awaits the reply, i.e. allows to perform a remote procedure call.
- **Action:** Communication based on the Client/Server mechanism with feedback. In some cases the service takes a long time to execute and therefore, in this case, the client needs to get periodic feedback about how the request is progressing and, may be, cancel the request.

The conceptual schema shown in Fig. 1 is redrawn in Fig. 4 showing those ROS nodes that encapsulate each of the hardware devices and those that implement synchronization and supervision functions. The type of communication used by these nodes varies with the communication needs as explained below and summarized in Table 3:

- The glove and tracker nodes (#1 to #4) simply broadcast their data, i.e. the communication between these nodes and the synchronization and mapping nodes (#5 and #6) is done using Topics.
- The synchronization and mapping nodes (#5 and #6) send queries to the collision-check node (#7) using Services, once they have computed the configuration of the robotic system, in order to receive information of whether they are collision-free or not.
- The nodes (#8 and #9) send the motion commands to the nodes that control the mechanical hands and the robots (#10 to #13) using Actions in order to get feedback of their execution to guarantee the correct performance.
- The data computed by the synchronization and mapping nodes (#5 and #6) is broadcasted using Topics; the nodes responsible for sending the commands to the hands and robots (#8 and #9) subscribe to them and orderly manage the queues that receive the information.

5.2 ROS nodes for trackers and gloves

The ROS node of each tracker performs the following steps:

1. Opens a RS232 connection and configures the tracker (the Flock of Birds tracker use the *FOBLIB* library and the Fastrack tracker the *Polhemus* library).
2. Reads data from the tracker (position and orientation).
3. Publishes the data on a Topic, named /tracker_old/tracker_states for the FOB and /tracker/tracker_states for the Fastrack, formatted using the standard ROS messages of type geometry_msgs/TransformStamped, that basically include a time stamp and the information of position and orientation (as a quaternion).

Besides broadcasting the position and orientation data, this node provides two services, one to manage the start and another one to control the publication frequency.

The ROS nodes for the sensorized gloves follow a similar procedure. In this case the data, read through a RS232 connection (for the left-hand glove) or a bluetooth connection (for the right-hand glove), is the set of hand joint values, either raw or calibrated. The data are broadcasted through two topics, named /cyberglove/raw/joint_states and /cyberglove/calibrated/joint_states, formatted using the standard ROS messages of type geometry_msgs/JointState.msg, that basically include a time stamp and a vector of floats to store the joint positions. Besides broadcasting the joint data, these nodes provide three services, one to manage the start, another one to control the publication frequency of the glove, and the last one to change the calibration file, in order to correct the data according to the particularities of the hand of each operator.

5.3 ROS nodes for robots and hands

The ROS node of each robot provides an Action service to control de robot. The connection to the robot controller is done through Ethernet using the CS8Soap

library that allows to set the power on and off, to login and logout, to get the positions of the joints, to move the robot to a desired position, using either joint variables or the robot Cartesian position in the workspace.

Action clients and Action servers communicate following the ROS Action Protocol that define three messages, the goal, the result and the feedback. The Action service is started at the server side and each time the Action client sends a request (a goal), a function named execute is called. This function sends the desired position to the robot controller using the CS8Soap library. When the motion terminates the server responds with a boolean variable (the result) informing wether the motion has been executed successfully or not. The Action client may send a feedback request and the Action server responds with a message indicating the state of the joints (the feedback).

Besides the Action service, these nodes periodically publish their configurations through topics, named /robot_pose, formatted using the standard ROS messages of type sensor_msgs/JointState.msg, that basically include a time stamp and a vector of floats to store the joint positions.

The ROS node of each hand works in a similar way with respect to the Action services although now no topic communication is issued by this node. The communication with both the SDH hand and the SAH hand is done through RS232 using in each case the API of the corresponding drivers. Since the SDH hand has different operation modes, a service has been included in order to change between them.

6 Supervision issues

6.1 ROS nodes for synchronization and supervision

As shown in Table 3, each of the ROS nodes for the synchronization and supervision, named SinCybTrac (nodes #5 and #6), is responsible for:

- Getting the data from the glove and the tracker by subscribing to the corresponding topics with the information of the values of the joints of the operator hand and of the position and orientation of his/her wrist (since the data from each topic include a time stamp, this is used to synchronize the information of the glove with that of the tracker).
- Mapping the joint values of the operators hand to those of the mechanical hand, as detailed in Section 4.2.
- Mapping the position and orientation of the tracker to the position and orientation of the robot wrist, as detailed in Section 4.1.
- Performing the inverse kinematics, as detailed in Section 6.2, to obtain the joint values of the robot or to detect that the teleoperated goal is out of bounds or at a singular configuration.
- Verifying, if the inverse kinematics ended successfully, whether the mapped hand-arm configuration is in collision or not by sending a query to the collision-check node (#7), that tests for collisions as detailed in Section 6.3.

– Broadcasting a correct configuration using a topic, i.e. the last encountered configuration that was within bounds and collision-free.

In summary, nodes #5 and #6 are the key nodes in the ROS graph, i.e. they are at the core of the teleopeation process and are responsible for having an intuitive, simple and safe teleoperation of the dual hand-arm robotic system.

6.2 Management of singularities

Inverse kinematics is needed to map the motions commanded in Cartesian space to the motions executed in the robot joint space. We implemented the Jacobian-based method available with the Kinematics and Dynamics Library (KDL). Problems arise at the singularities, those configuration of the manipulator that make the Jacobian matrix of the manipulator to loose rank. Some approaches minimize this problem by taking advantage of redundancy to assure large convex workspaces for singularity-free operation [14]; others try to find approximate solutions to the inverse Jacobian by means of complex robust control schemes based on damped least squares and dynamic weighting [15].

In this work we opted to limit the values of the joint velocities that become very large (which happens when the robot is in a pose very close to a singular configuration and when the desired direction requires the robot to approach the singular configuration). The pose obtained by these joint values differ from the commanded pose. This error is captured by the algorithm that blocks the communication (i.e. keeps publishing the same configuration). When the operator does the appropriate movement(one that does not cause large values of the inverse Jacobian), a re-synchronization is performed and the robot smoothly follows the operator, moving away from the singular configuration.

As an example, Fig 5(top) shows some snapshots of the execution of this supervision. When the commanded configuration makes the robot to reach the limits of the workspace, this is detected and the last configuration within limits is being kept published, thus the robot is stopped there (snapshot b). While the operator moves away from that configuration trying to move the robot further out of limits, the robot remains stopped (snapshot c). When the operator returns to a configuration within the workspace limits then the robot follows him/her again (snapshot d) after an automatic re-synchronization is performed.

6.3 Collision detection

The remote cell has been modelled using The Kautham Project, the simulation and planning toolkit developed at our lab (sir.upc.edu/projects/kautham/ [16]). This software uses the PQP library for the collision detection. The modules for the modelling of the cell and for collision detection have been encapsulated as a ROS node that, as a service server, offers responses to collision-check queries. Collisions between a robot and the environment, or between the two robots, or possible autocollisions are detected in a very fast manner due to the simplicity and efficiency of the collision-check library used.

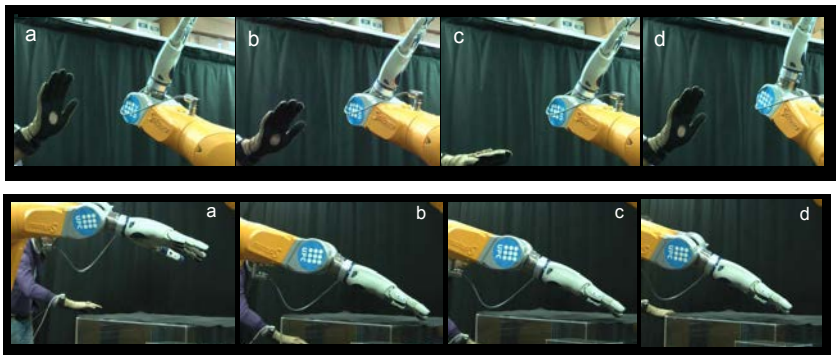

Fig. 5. Example of the supervision of the workspace limits (top) and of the collision detection supervision (bottom). The last commanded valid configuration (b) is being continually published while the operator tries unfruitfully to command an out-of-bounds or a collision configuration (c). When the commanded configuration is valid again (d) a re-synchronization is executed and the teleoperation resumes.

When the commanded configuration makes the robot to collide with the environment, it is detected and the last free configuration is being kept published, thus stoping the robot in a safe place. When the operator detects that the robot is not following him/her, he/she should return approximately to the position where the robot stopped. Once the commanded configuration is again in the free space, a re-synchronization is performed. This is done to assure smooth motions because this new free configuration may not exactly coincide with the last free configuration where the robot stopped.

As an example, Fig 5(bottom) shows some snapshots of an example of this supervision. When the commanded configuration makes the robot to collide with the table, this is detected and the last free configuration is being kept published, thus the robot is stopped there (snapshot b). While the operator keeps moving toward forbidden configurations, the robot remains stopped (snapshot c). When the operator returns to a configuration within the free space the robot follows him/her again (snapshot d) after an automatic re-synchronization is performed.

6.4 Simulation

As an additional support to the operator the visualization of the cell is optionally shown to the user (Fig. 1, bottom). The viewer of the The Kautham Project simulation and planning toolkit has been encapsulated as a ROS node that provides a service called move_robot that moves the robots and the hands to the configurations requested by the client, the Simulation_cell node that synchronizes the data from the two hand-arm sets, i.e. it builds a single message as the concatenation of the joint data of each of the hand-arm systems.

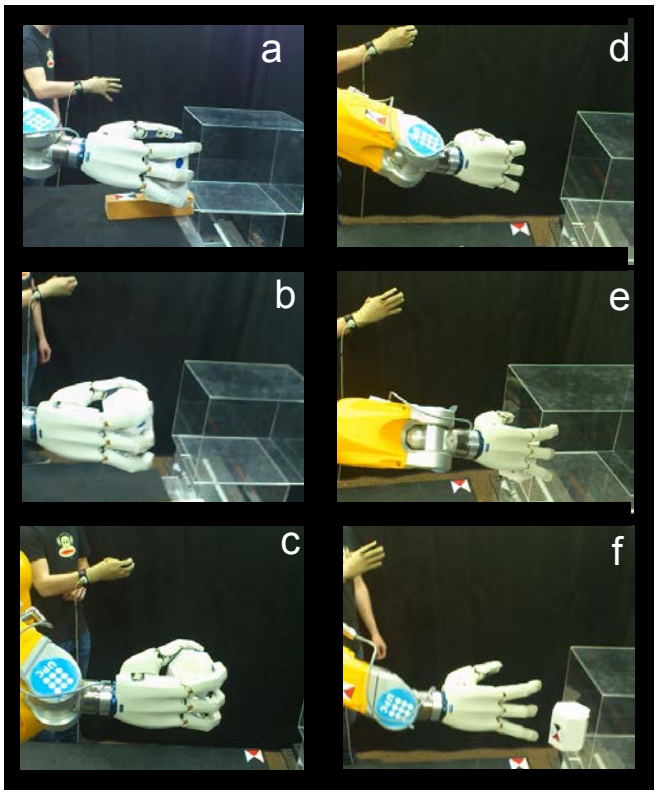

Fig. 6. Pick-and-place real task. A manual re-synchronization is performed at step c in order to command the robot in a more comfortable way.

6.5 Manual re-synchronization

Fig 6 shows several snapshots of a pick-and-place teleoperated task. The operator must take a prism and place it inside a box. The execution includes a manual re-synchronization at step **c**, done to command the robot in a more comfortable way. This can be seen observing that the transformation between the human arm and the robot changes from the snapshots of the left column and those of the right one.

In order to do a manual re-synchronization the operator stops the teleoperation by pressing the clutch button on the tracker and then, using the same button, resumes the teleoperation with a more comfortable posture. The manual re-synchronization, as the automatic one, is done as described in Subsection 4.1.

7 Summary

The paper has presented the implementation of a teleoperation system for a dual hand-arm robotic system that includes two industrial robots and two mechanical

hands, one of them with anthropomorphic features, commanded with two sensorized gloves and two trackers attached to the wrists of the operator. The paper includes the description of: the mappings used to transform the information captured from the operator movements to the movements of the robotic system, the functional schema of the whole teleoperation system, and the particular modules developed for the communication and supervision issues. Future work includes the application of the approach to an anthropomorphic robotic system.

References

1. Basañez, L., Suárez, R.: Teleoperation. In Nof, S., ed.: Springer Handbook of Automation. Springer-Verlag (2009) 449–468
2. Speeter, T.: Transforming human hand motion for telemanipulation. Presence **1**(1) (1992) 63–78
3. Rohling, R., Hollerbach, J., JAcobsen, S.: Optimized fingertip mapping: a general algorithm for robotic hand teleoperation. Presence **2**(3) (1993) 203–220
4. Peer, A., Einenkel, S., Buss, M.: Multi-fingered telemanipulation - mapping of a human hand to a three finger gripper. In: Proc. 17th IEEE Int. Symp. on Robot and Human Interactive Communication. (2008) 465–470
5. Rosell, J., Suárez, R., Rosales, C., Pérez, A.: Autonomous motion planning of a hand-arm robotic system based on captured human-like hand postures. Autonomous Robots **31**(1) (2011) 87–102
6. Pao, L., Speeter, T.H.: Transformation of human hand positions for robotic hand control. In: Proc. IEEE Int. Conf. on Robotics and Automation. (1989) 1758–1763
7. Hong, J., Tan, X.: Calibrating a vpl dataglove for teleoperating the utah/mit hand. In: Proceedings of IEEE Int. Conf. on Robotics and Automation. (1989) 1752–1757
8. Abe, K., Saito, H., Ozawa, S.: Virtual 3-D interface system via hand motion recognition from two cameras. IEEE Trans. on Systems, Man, and Cybernetics - Part A: Systems and Humans **32**(4) (2002) 536–540
9. Wachs, J.P., Stern, H., Edan, Y.: Cluster labeling and parameter estimation for the automated setup of a hand-gesture recognition system. IEEE Trans. on Systems, Man, and Cybernetics - Part A: Systems and Humans **35**(6) (2005) 932–944
10. Infantino, I., Chella, A., Dindo, H., Macaluso, I.: Cognitive architecture for robotic hand posture learning. IEEE Trans. on Systems, Man, and Cybernetics - Part C: Applications and Reviews **35**(1) (2005) 42–52
11. Tao Geng, M.L., Hülse, M.: Transferring human grasping synergies to a robot. Mechatronics **21**(1) (2011) 272–284
12. Pérez, A., Rosell, J.: An assisted re-synchronization method for robotic teleoperated tasks. In: Proc. IEEE Int. Conf. on Robotics and Automation. (2011) 886–891
13. Colasanto, L., Suárez, R., Rosell, J.: Hybrid mapping for the assistance of teleoperated grasping tasks. IEEE Transactions on Systems, Man, and Cybernetics: Systems **43**(2) (2013) 651–660
14. Peer, A., Stanczyk, B., Buss, M.: Haptic Telemanipulation with Dissimilar Kinematics. In: Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems. (2005) 3493 – 3498
15. Schinstock, D.: Approximate solutions to unreachable commands in teleoperation of a robot. Robotics and CIM **14**(3) (1998) 219 – 227
16. Pérez, A., Rosell, J.: A roadmap to robot motion planning software development. Computer Applications in Engineering Education **18**(4) (2010) 651–660