Telepresence Using the Kinect Sensor and the NAO Robot

Jose Avalos*, *Student Member, IEEE*, Sergio Cortez*, *Student Member, IEEE*, Karina Vasquez*, *Student Member, IEEE*, Victor Murray*†, *Senior Member, IEEE*, and Oscar E. Ramos*, *Member, IEEE*
*Department of Electrical Engineering, Universidad de Ingenieria y Tecnologia - UTEC, Lima, Peru.
†Department of Electrical and Computer Engineering, University of New Mexico, Albuquerque, New Mexico, USA.

***Abstract*—Several applications require that a telepresence system does not only transmit images, audio and video but also real-time motion. This work presents the implementation of a system that allows for telepresence using a humanoid robot and a motion capture device. The proposed methodology for motion imitation is fast and uses a low-cost motion acquisition sensor. The objective is to make the robot reproduce the motion of a person. In this way, remote actions can be executed through the robot sending also images and audio from the environment. The method has been applied to the humanoid robot called NAO, which has been able to reproduce human motions. This framework can also be used for different education purposes.**

***Keywords*—NAO robot, kinect sensor, robotic kinematics, telepresence**

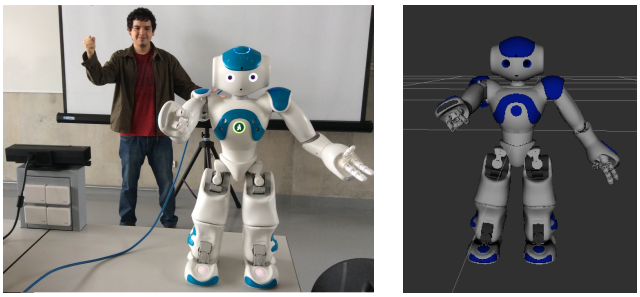

Fig. 1: Example of robot teleoperation. The real and simulated NAO robot follows the movements of the person.

I. Introduction

Telepresence is a resource that allows people to be connected from any place in order to virtually accomplish a specific task. An example is a videoconference in which both visual and audio are enabled to recreate a certain scenario. However, the real target of telepresence is to get the user have the control of all his faculties in any place at any time. This will improve communication and, as a result, it can be applied on any educational field.

Different methods have been applied and analyzed to achieve telepresence through the usage of robots. Suleiman *et al.* [1] present human motion imitation by a human-size humanoid robot considering inverse dynamics, where the physical properties of the robot constitute the main problem. The method by Cela *et al.* [2] is a low-cost system that consists in the use of accelerometers attached to the human body to detect the human joint motion; however, this method is limited to simple robots due to imprecise data. On the other hand, It is also possible to use visual sensors to acquire the human motion without needing to attach any special device to the person as in [3] where a robot with "eyes" capable of capturing motion is presented. In this front, the Kinect sensor has also been used to directly detect human body joints [4] and to use the point cloud data with some further processing in a virtual environment [5]. In all these cases, the data recovered from the motion capture system is processed to recover the motion and is then sent to the robot. An important issue yet to be solved is the time needed to process the motion and to determine the most adequate joint configuration for the robot, which is usually not generated in real time due to computational complexity.

In this work, we propose a methodology to achieve real-time telepresence using a geometric approach for planar motion and an inverse kinematics approach for a complete 3D motion of a humanoid robot. The methods have been validated using both Kinect sensor v1 and v2 [6] controlling the NAO robot [7]. An example of the proposed teleoperation method is shown in Fig. 1 where the robot imitates the motion shown by the demonstrator in real time. The remainder of this paper is organized as follows. In Section II we present the methods to achieve motion imitation, which is crucial for teleoperation. Section III shows the experimental results of two methods.

II. Methods

To apply the teleoperation to any robot-sensor system, we propose to divide the task in three main parts. The first one consists in moving the robot joints through inverse kinematic operations in the operational space or in the joint configuration space. The second part obtains compatible data from the motion capture device and processes it to make it suitable for motion imitation. The last part consists in creating a bridge between the robot and the motion capture device, which will allow the transmission of data in real time, achieving teleoperation. The process can be geared to work under a two or a three dimensional model.

A. Two dimensional teleoperation motion

1) Robot Motion Control: In this case, a joint-space based control system is more efficient and easier to implement since the motion constrained to a plane represented using joint angles. Therefore, the model will directly work in the joint

ISBN 978-1-4673-7835-2/16/$31.00©2016 IEEE

IEEE Catalog Number CFP16LAS-ART

space rather than in the Cartesian space. In two dimensions, specific angles relative to the motion of certain limbs will be the data sent to the robot. The program will receive joint angles which will constitute the motion targets.

2) Data acquisition: There are different ways and devices that act as motion capture systems to recover data from the human body [2][5]. Another example is a depth sensor such as the Kinect which can capture specific joints of the human body, in Cartesian coordinates, related to the arms, legs, head and torso. These sensors usually come with an SDK (Software Development Kit) which can be used to transform Cartesian into spherical coordinates to obtain the desired joint angles. The propose method is based on vector operations. This method can be extrapolated for other limbs as long as the motion is constrained to a plane.

First, we analyze both arms of the person who will tele-operate the robot. Note that the inertial reference frame is a motionless torso, and local reference frames lie on each shoulder; that is, the position of each arm is represented with respect to its shoulder. The third component of these positions is not important, since we are using a two dimensional model (it is only included for mathematical purposes).

Let the sensor output describing the i^{th} joint position with respect to the sensor reference frame be represented in Cartesian coordinates as:

$$\mathbf{x}_i = (x_i, y_i, z_i). \tag{1}$$

For instance, let the shoulder position be $\mathbf{x}_{s_j}$, the elbow position be $\mathbf{x}_{e_j}$ and the hand position be $\mathbf{x}_{h_j}$, where j refers to either the left or right limb. Using these positions, let the vector referred to the forearm, $\mathbf{v}_{f_j}$ and the vector referred to the arm $\mathbf{v}_{a_j}$ be defined as:

$$\mathbf{v}_{f_j} = \mathbf{x}_{e_j} - \mathbf{x}_{s_j} \tag{2}$$

and

$$\mathbf{v}_{a_j} = \mathbf{x}_{h_j} - \mathbf{x}_{e_j}. \tag{3}$$

In the two-dimensional case, let the angle between the torso and the forearm be represented by ϕ as Fig. 2 shows. This angle can range from 0° to 180° where 0° corresponds to the forearm close to the torso and 180° corresponds to the forearm fully raised. The angle at the elbow, between the forearm and the arm is $\phi + \beta$, where β is obtained in a similar way as ϕ. Note that the angle $\phi + \beta$ ranges from 0° to 360°.

3) Data Transmission: Once the motion capture sensor is acquiring raw data, which is processed as in [4] to obtain the desired joint angles as described in the previous section, they can be sent to the robot. For these data to be transmitted to the robot, a bridge must be implemented to make the system work in real time. The implementation of the bridge depends upon the specific characteristics of the robot and the sensor to be used.

B. Three dimensional teleoperation motion

1) Robot Motion Control: Contrary to the planar motion described in section II-A, a control based on inverse kinematics or differential inverse kinematics is more suitable for a three dimensional motion. In this case, the robot control will be defined with respect to Cartesian coordinates instead of joint angles. Since the motion is now three dimensional, a full tele-operation can be achieved.

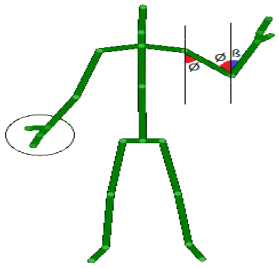

Fig. 2: Skeleton showing the important angles for the method. ϕ represents the angle relative to the torso and the forearm. $\phi + \beta$ represents the angle at the elbow, between the forearm and the arm.

Let the robot joint angles be represented by the joint configuration vector $\mathbf{q} = (q_1, q_2, \cdots, q_n)$, where n is the number of degrees of freedom of the robot. Let a Cartesian position be $\mathbf{x_i} = (x_i, y_i, z_i)$. Forward kinematics relates the joint configuration to an operational space configuration, such as the Cartesian space, as

$$\mathbf{x}_i = f_i(\mathbf{q}) \tag{4}$$

where f_i is the forward kinematics function. It can be seen in (4) that the Cartesian coordinates are a function that depends directly on the joint angles. This is the case of the joint control model described in section II-A. Since we are interested in the joint angles that will achieve a desired Cartesian position, (4) needs to be inverted as

$$\mathbf{q} = f_i^{-1}(\mathbf{x}_i), \tag{5}$$

which is referred to as inverse kinematics. Due to the non-linearities of (4), it is difficult to obtain the inverse map as in (5) in a general case. One usual approach to solve this problem is through differential inverse kinematics which acts as a linearization of the model around the working point. This relation is represented with a differential map which, in this case, corresponds to the Jacobian $J_i = \frac{\partial f_i}{\partial \mathbf{q}}$ so that

$$\dot{\mathbf{x}}_i = J_i \dot{\mathbf{q}}. \tag{6}$$

As (6) shows, the Jacobian relates the rate of change of the joint angles to the rate of change of the Cartesian coordinates. The system (6) is linear and needs to be solved for $\dot{\mathbf{q}}$. In the general case [2], this solution is given by

$$\dot{\mathbf{q}} = J_i^{\#} \dot{\mathbf{x}}_i + (I - J_i^{\#} J_i)\mathbf{z}, \tag{7}$$

where $J_i^{\#}$ denotes the Moore-Penrose pseudoinverse of J_i, and $\mathbf{z}$ is an arbitrary vector which is projected onto the nullspace of J_i using $(I - J_i^{\#} J_i)$. Note that if both $\mathbf{q}$ and $\mathbf{x}$ have the same size, leading to a square Jacobian, and if J_i is a nonsingular matrix, then the pseudoinverse becomes the usual inverse $J_i^{\#} = J_i^{-1}$. Using Euler integration, the joint velocities obtained with (7) can be transformed into joint positions as

$$\mathbf{q}_{k+1} = \mathbf{q}_k + \dot{\mathbf{q}}_k \Delta T \tag{8}$$

ISBN 978-1-4673-7835-2/16/$31.00©2016 IEEE

IEEE Catalog Number CFP16LAS-ART

where $\mathbf{q}_k$ is the joint configuration at time k. These joint values can then be directly sent to the robot.

2) Data acquisition: This part is similar to its counterpart in the two dimensional model. However, in this case the data process is only referred to the Cartesian coordinates obtained by the motion capture device instead of the joint angles. The constraint to be taken into account is that the positions relative to the human bones need to be normalized. The normalization depends on the specific robot that is used since different robots have different link lengths.

3) Data transmission: The data obtained with the motion capture device needs to be transmitted to the robot using a bridge which depends on the specific hardware. One option for this case is to use ROS (Robot Operating System) as a middleware for interconnecting the system.

III. Results and Discussion

For the validation of the proposed methodology, we used the *Kinect For Windows* as a motion capture device. The reason was its very low price compared to other motion capture systems, and the fact that it is open source. This sensor allows the recovery of the data through its SDK. The humanoid robot that we used for the experiments is NAO from Aldebaran Robotics, which resembles a human in its kinematical structure. This allows us to reproduce anthropomorphic movements.

The Kinect for Windows v1 sensor detects depth based on structured light. Combining both *depth by stereo* and *depth by focus* techniques, the Kinect sensor provides a three dimensional reconstruction of a 3D space. The algorithm used by Microsoft to recognize a person relies on the comparison of the current obtained data with an internal database using a tree-type algorithm [8]. If the operation is successful, the person is properly recognized and represented as a skeleton. The function to achieve this recovery in the Kinect SDK is called *Skeletal Tracking*. This provides the position (in Cartesian coordinates) of the main joints in a human being. Contrary to its predecessor, the Kinect v2 is based on the principle of time-of-flight.

A. Results of the two-dimensional telepresence

This method uses the Kinect sensor v1 with its corresponding SDK V1.8, which can only be installed on Windows. The NAO robot is programmed using the interface provided by Aldebaran Robotics called NAOqi. This work uses C++ to program the robot motion.

The joint angular limits of the robot limbs do not correspond to the values used in section II-A2. For example, the joint angle corresponding to the shoulder neither begins at 0° nor ends at 180° degrees, but ranges from -18° to 76° (LShoulderRoll) as Fig. 3 depicts. This figure also shows that the values for RelbowRoll range from 0° to 88.5° instead of the assumed ones. To solve this problem, a linear map relating these values was used. If the robot angular values exceed the joint limits, the robot stops moving due to its internal safety procedure. Another factor that leads to automatically stopping the motion is the nominal torque which appears when "sharp" movements

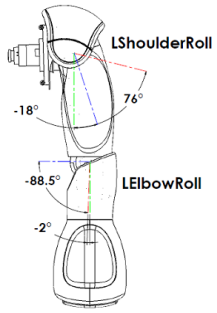

Fig. 3: NAO's arm design. Image of Aldebaran [10].

are sent to NAO. For instance, if a sudden movement is sent to the robot, the program will end the motion execution as a safety measure. Because of this issue, the data from the Kinect are restricted to the robot joint limit values with the purpose of avoiding physical damage and undesired interruptions. This is similar to clamping these values as in [9]. The angles are multiplied by a factor obtained by experimental tests to achieve better result. Likewise, tests were made to adjust the speed to reproduce the motion avoiding interruptions due to torques or angular limits.

B. Results of the three-dimensional telepresence

To generate motion for the robot in three dimensions we used a control system based on inverse kinematics. Since we are interested in reproducing the motion shown by a human demonstrator, it is important to ensure that the reference frames of both the motion sensor and the robot are consistent. In the general case, these reference frames have a different orientation as clearly presented in [4], where a Kinect v2 and a 7-DOF KUKA robotic arm were used. To solve this problem, it is necessary to use a rotation matrix that can be obtained from a prior calibration process. In our case, contrary to [4], we are assuming that the chest remains fixed throughout the motion and then, it is straightforward to obtain the rotation matrix relating both frames from simple inspection.

The software for robot control has been mainly developed for Linux since most robots use a Linux-based real-time embedded computer. We used ROS (Robot Operating System) in Linux as a middleware to get the acquired motion from the sensor, and to send the generated joint command to the robot. Since the Kinect SDK needed to extract the human motion data has been developed only for Windows, we needed to send the data from a machine running Windows to a machine running Linux. This was achieved through a logical (non-physical) serial connection, using the *rosserial* package, which acted as a bridge between Windows and ROS in Linux. This connection scheme is shown in Fig. 4.

The human motion is then used as input to the control system where inverse differential kinematics is used to generate the desired joint angles $q_{desired}$ using (8). The NAOqi Bridge package acts as an interface between ROS and the NAOqi which control the robot. It takes the desired joint angles that have been published to a ROS topic and sends them to NAOqi.

ISBN 978-1-4673-7835-2/16/$31.00©2016 IEEE

IEEE Catalog Number CFP16LAS-ART

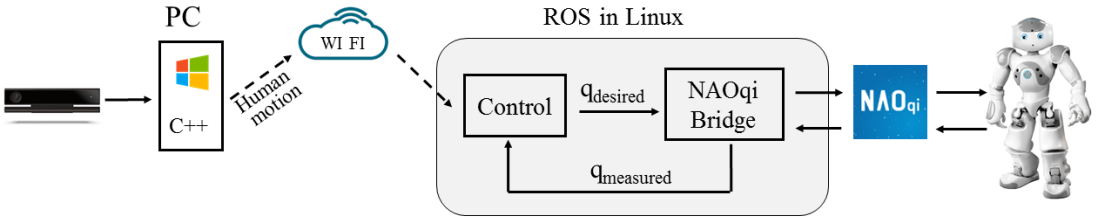

Fig. 4: Scheme showing the parts of the system used for the 3D motion, from the human motion acquisition (with the Kinect) to the humanoid robot execution.

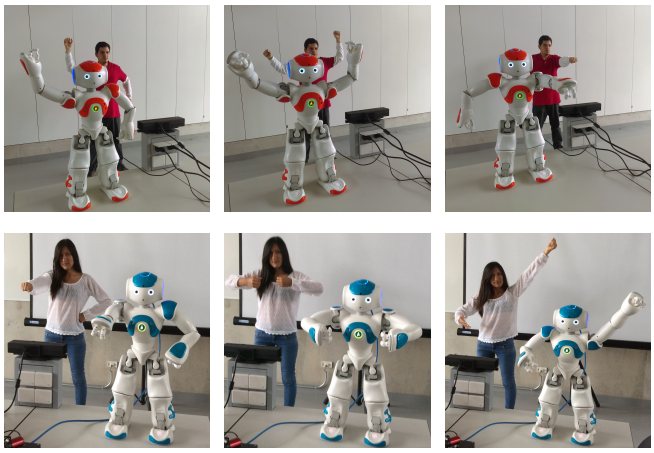

Fig. 5: Example of the results of the three-dimensional telepresence system under three different positions.

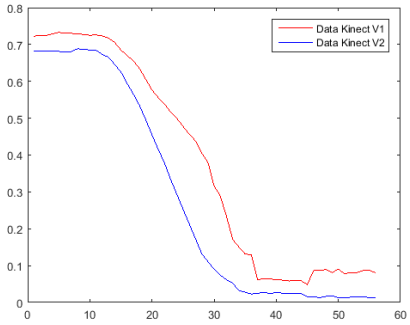

Fig. 6: Comparison of Kinect v1 (red) and v2 (blue) in terms of smoothing in the motion acquisition. $x-$axis is time and $y-$axis is the position in x.

Similarly, it takes the joint measured angles $q_{measured}$ coming from the NAOqi and feedbacks them publishing in a ROS topic that the controller can read. The NAOqi is the ultimate interface between the robot and the host computer, which is normally used to control the NAO robot. The results of three dimensional telepresence are shown in Fig. 5. The software we developed for this work is freely available[1].

In our experience, the Kinect v1 presented problems when some joints start overlapping with other joints. To analyze these problems, we chose a movement parallel to the ground where the arm starts fully extended and moves towards the torso. Fig. 6 shows the resulting path: we can see that the v2 maintains a smoother track than the v1. Therefore, we used the Kinect v1 only for the two dimensional approach, but we used the Kinect v2 for the three dimensional motion.

IV. Conclusions

A method for the human teleoperation of a humanoid robot for motion in two dimensions and three dimensions was presented in this paper. The proposed approach was validated using the NAO robot and the Kinect sensor. There exist different applications for telepresence. In particular, this work wants to use telepresence as a method for long-distance teaching with active participation of the users. Moreover, this work could be used as an interactive tool for teaching children with autism in order to improve their social abilities. This is a first step to achieve full telepresence. Future work will involve audio and video transmission since the humanoid platform (NAO) possesses these sensors.

[1]https://github.com/utecrobotics/nao_kinect_teleop

References

[1] W. Suleiman, E. Yoshida, F. Kanehiro, J.-P. Laumond, and A. Monin, "On human motion imitation by humanoid robot," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2008, pp. 2697–2704.

[2] A. Cela, J. J. Yebes, R. Arroyo, L. M. Bergasa, R. Barea, and E. López, "Complete low-cost implementation of a teleoperated control system for a humanoid robot," *Sensors*, vol. 13, no. 2, pp. 1385–1401, 2013.

[3] S. Filiatrault and A.-M. Cretu, "Human arm motion imitation by a humanoid robot," in *IEEE International Symposium on Robotic and Sensors Environments (ROSE)*, 2014, pp. 31–36.

[4] Y. Yang, H. Yan, M. Dehghan, and M. H. Ang, "Real-time human-robot interaction in complex environment using kinect v2 image recognition," in *IEEE International Conference on Cybernetics and Intelligent Systems (CIS)*, 2015, pp. 112–117.

[5] H. Hisahara, S. Hane, H. Takemura, Y. Chin, T. Ogitsu, and H. Mizoguchi, "3d point cloud–based virtual environment for safe testing of robot control programs," *International Conference on Intelligent Systems, Modelling and Simulation (ISMS)*, pp. 24–27, 2015.

[6] H. Fürntratt and H. Neuschmied, "Evaluating pointing accuracy on kinect v2 sensor," in *International Conference on Multimedia and Human-Computer Interaction (MHCI)*, 2014, pp. 124.1–124.5.

[7] S. Shamsuddin, L. I. Ismail, H. Yussof, N. I. Zahari, S. Bahari, H. Hashim, and A. Jaffar, "Humanoid robot nao: Review of control and motion exploration," in *IEEE International Conference on Control System, Computing and Engineering (ICCSCE)*, 2011, pp. 511–516.

[8] J. Shotton, T. Sharp, A. Kipman, A. Fitzgibbon, M. Finocchio, A. Blake, M. Cook, and R. Moore, "Real-time human pose recognition in parts from single depth images," *Communications of the ACM*, vol. 56, no. 1, pp. 116–124, 2013.

[9] D. Raunhardt and R. Boulic, "Progressive clamping," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2007, pp. 4414–4419.

[10] A. Robotics, "Nao software 1.14. 5 documentation," *URL http://www. aldebaran-robotics. com*, 2014.