A VISION-BASED TELEOPERATION SYSTEM FOR ROBOTIC SYSTEMS

C. Nuzzi(1), S. Ghidini(1), R. Pagani(1), S. Pasinetti(1), G. Coffetti(1), G. Sansoni(1)
(1)Dip. di Ingegneria Meccanica e Industriale, Università degli Studi di Brescia
mail autore di riferimento: c.nuzzi@unibs.it

1. INTRODUCTION

Despite advances in robotic perception are increasing autonomous capabilities, the human intelligence is still considered a necessity in unstructured or unpredictable environments. Hence, also according to the Industry 4.0 paradigm, human and robots are encouraged to achieve mutual Human-Robot Interaction (HRI). HRI can be physical (pHRI) or not, depending on the assigned task. For example, when the robot is constrained in a dangerous environment or must handle hazardous materials, pHRI is not recommended. In these cases, robot teleoperation may be necessary. A teleoperation system concerns with the exploration and exploitation of spaces where the user presence is not allowed. Therefore, the operator needs to move the robot remotely [1]. Although plenty of human-machine interfaces for teleoperation have been developed considering a mechanical device, vision-based interfaces do not require physical contact with external devices. This grants a more natural and intuitive interaction, which is reflected on the task performance [2].

Our proposed system is a novel robot teleoperation system that exploits RGB cameras, which are easy to use and commonly available on the market at a reduced price. A ROS-based framework has been developed to supply hand tracking and hand-gesture recognition features, exploiting the OpenPose software [3] based on the Deep Learning framework Caffe. This, in combination with the ease of availability of an RGB camera, leads the framework to be strongly open-source oriented and highly replicable on all the ROS-based platform. It is worth noting that the system does not include the Z-axis control in this first version. This is due to the high precision and sensitivity required to robustly control the third axis, a precision that 3D vision systems are not able to provide unless very expensive devices are adopted. Our aim is to further develop the system to include the third axis control in a future release.

2. POSITIONING ERROR EVALUATION

A reliable teleoperation system is obtained if the robot correctly moves to the desired position with a low positioning error. In the proposed set-up, the positioning error is obtained as a sum of different errors, as shown in Fig. 1.

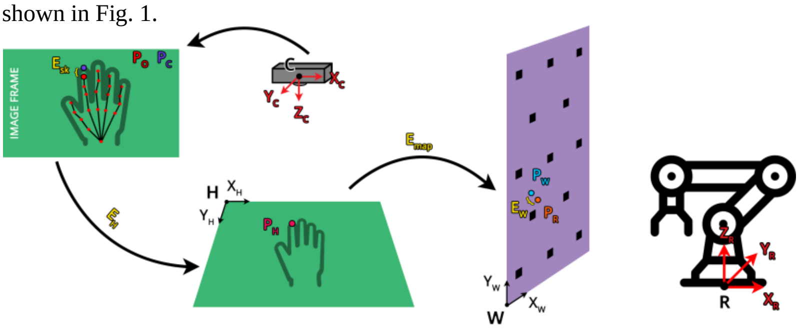

Fig. 1. Scheme of the calibration phases necessary to teleoperate the robot.

Starting from point $\mathbf{P_H}$, to where the user points its finger, the robot moves to point $\mathbf{P_R}$, which is calculated by the following formula:

$$P_R = \begin{bmatrix} \frac{1}{N}\sum_{n=1}^{N} x\, P_{On} \\ \frac{1}{N}\sum_{n=1}^{N} y\, P_{On} \end{bmatrix} + E_{sk} + E_H + E_{map} + E_W$$

We defined three experiments to determine the positioning errors $\mathbf{E_{sk}}$, $\mathbf{E_H}$ and $\mathbf{E_W}$. In our set-up, $\mathbf{E_{map}}$ was considered equal to none because reference system *H* and reference system *W* had the same dimensions but were oriented differently. Results are shown in Table 1.

$\mathbf{E_{sk}}$**:** is computed as the difference between T_p, which correspond to the real *px* coordinates of the index finger in the acquired frame, and A_p, which correspond to the index finger coordinates estimated by OpenPose. It is worth noting that, since our procedure computes an average over $N = 7$ keypoint positions to reduce the skeleton estimation noise, we obtain both T_p and A_p as the average position over *N* consecutive frames (Fig. 2a). We considered *14* different locations of workspace *H*, corresponding to *98* couples of image frames and index joint estimations (*14 * 7*).

$\mathbf{E_H}$**:** is computed as the difference between theoretical positions (T_p), which correspond to the real calibration master checkerboard dimensions in *mm*, and the actual positions (A_p), which correspond to the checkerboard grid points calculated by OpenCV in *px* and converted in *mm* using the calibration matrixes. We used a total of *54* positions taken from the calibration master (Fig. 2b).

$\mathbf{E_W}$**:** this error has two components: $\mathbf{E_{WA}}$ which is due to the calibration procedure between reference system *W* and reference system *R*, and $\mathbf{E_{WB}}$, which is due to the robot internal errors such as motor vibrations and encoder resolution. $\mathbf{E_{WA}}$ is computed similarly to $\mathbf{E_H}$, considering the difference between T_p and A_p where T_p are the coordinates of the markers in reference system *W* and A_p are the positions of the markers calculated using the transformation matrix obtained between *W* and *R*. In this case, we used workspace *W* markers, for a total of *13* points. $\mathbf{E_{WB}}$ is computed as the difference between T_p, which correspond to the coordinates of the marker centroid of reference system *W*, and A_p, which correspond to the laser dot centroid calculated by a LabVIEW software using a second vision system mounted behind the glass panel of reference system *W*. In this case, we moved the robot from its proprietary software to *7* positions *3* times each (Fig. 2c).

TABLE 1. *Results of the experiments. It is shown that there is high variability but not bias.*

	E_{sk} [px]		E_H [mm]		E_{WA} [mm]		E_{WB} [mm]	
	X-axis	**Y-axis**	**X-axis**	**Y-axis**	**X-axis**	**Y-axis**	**X-axis**	**Y-axis**
Average	2.1	-2.6	-8.3	-10.3	2.0	0.5	0.7	1.7
Std. Dev.	3.6	3.5	6.4	7.4	2.1	0.3	0.1	0.1

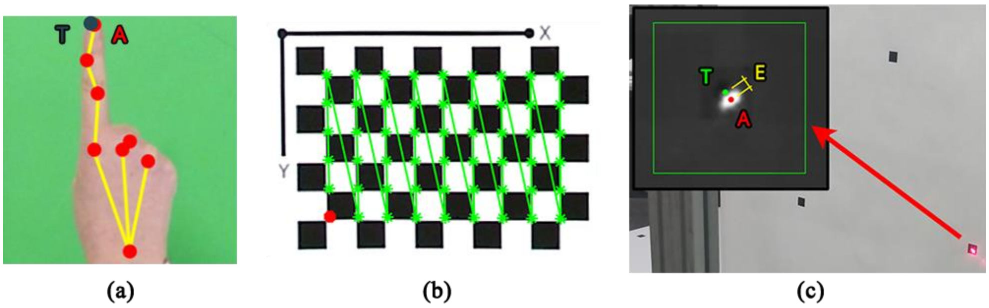

Fig. 2. Examples of the experiments carried out to determine the positioning errors. (a) E_{sk}, (b) E_H, (c) E_{WB}.

BIBLIOGRAPHY

[1] J. Vertut e P. C. Coeffet, Robot Technology: Teleoperation and Robotics Evolution and Development, vol. 3A, Prentice Hall, 1986.

[2] R. G. Boboc, H. Moga e D. Talabă, «A Review of Current Applications in Teleoperation of Mobile Robots,» 2013.

[3] S. Tomas, J. Hanbyul, M. Iain e S. Yaser, «Hand Keypoint Detection in Single Images using Multiview Bootstrapping,» in *CVPR*, 2017.