Redundancy management of a LWR4+ for safe, collision free, robotic surgical applications, using environmental sensors

Danilo De Lorenzo, Simone Calò, Andrea Ciullo, Mirko Daniele Comparetti, Elena De Momi, Giancarlo Ferrigno
Department of Electronics, Information and Bioengineering
Politecnico di Milano
Milan, Italy
danilo.delorenzo@polimi.it

Mirko Kunze
Institute for Process Control and Robotics
Karlsruhe Institute of Technology
Karlsruhe, Germany

Abstract— **In surgical robotics, workspace sharing between robots and humans asks for safety-oriented control of the physical interaction. In this paper, a ROS – OROCOS system architecture for collision avoidance is presented for safe human-robot coexistence inside the operating room. A Microsoft Kinect sensor allows to track the user hand and to exploit the redundancy of the Kuka LWR4+ 7DoF manipulator moving the robot links far from a possible collision.**
Results show that the reaction time of the robot is about 80ms, when the user hand enters the collision risk region. A calibration between the Kinect and the robot workspace allows not to track the robot links with the Kinect sensor, reducing possible human-robot tracking occlusions.

Keywords—human-robot interaction; robotic surgery; redundancy; collision avoidance.

I. INTRODUCTION

In neurosurgery, several procedures can be performed with the help of robotic assistance. The robotic support can act directly on the surgeon tasks or in tool holding or for teaching [1][2]. In all these scenarios, the surgeon and the robot share the same workspace and safe human-robot interaction strategies must be considered. Collision detection and collision avoidance are needed inside an operating room where robots and clinicians work close together. In literature, collision detection and collision avoidance strategies have already been analyzed. Proprioceptive robot sensors are used for collision detection [3]. When the impact occurs, torque sensors detect the real torque on each joint; if the difference between this torque and the commanded torque is bigger than a threshold, the robot stops its movement.

In [4], the pose of the robot is computed in real time with a Kinect sensor. The pose of the operator's hand and the entire structure of the robot are extracted using a depth space approach. A repulsive vector is computed when there is a risk of collision between the robot and the operator. For real-time computation of this repulsive vector, Reflexxes Motion Libraries [5] are used. Both hand position and robot position are extracted using a depth camera. This approach could lead to risky situations in case the operator body overlaps the robot structure, compromising robot and skeleton detection.

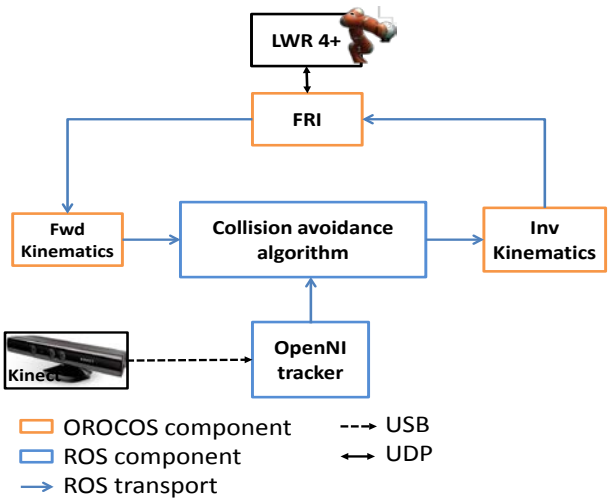

Fig. 1 ROS and OROCOS system software components and architecture.

The work here presented fits inside the Active EU project [6]. One of the purpose is to combine the ability of neurosurgeons with robotic instrument precision and repeatability and to exploit the 7th degree of freedom (redundancy) of a KUKA LWR4+ to improve robot maneuverability and avoid collision with operating room (OR) operators. If the robot is autonomously driven without any environmental information, it can accidentally hit the OR staff and/or the patient, leading to unsafe situations. Therefore, a high level controller to assure safe movements of the robot, avoiding any collision with the surgeon can be mandatory.

We introduce a preliminary system for collision avoidance, based on ROS and OROCOS platforms [7][8], which exploits the kinematic redundancy [9][10] of the Kuka LWR4+ and a

This work was partially funded by the ACTIVE FP7-ICT-2009-6-270460 European Project.

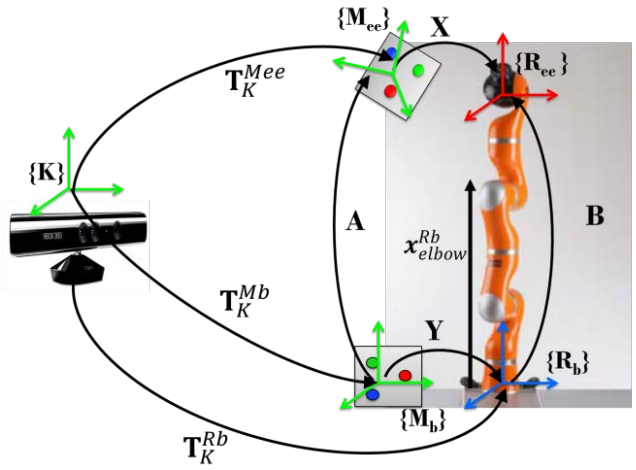

Fig. 2 Transformations involved in the calibration setup between the Kinect sensor and the LWR. **T, A, B, X, Y** are the transformations used during calibration procedure, {M} are the reference frames (RFs) of the markers, {R} are the RFs of the robot.

Microsoft Kinect sensor, calibrated inside the operational space of the robot.

II. Materials and Methods

The complete system is composed of a Microsoft Kinect sensor (Microsoft Corp., Redmond, WA, USA) and a KUKA Light-Weight-Robot 4+ (LWR) (Kuka Robotics, Augsburg, Germany). The system architecture is based on ROS and OROCOS platforms which communicate between them via ROS-transport (Fig. 1). OROCOS components are in charge of real-time communication with the LWR and of forward and inverse kinematic computation, while ROS modules are in charge of managing the information coming from the Kinect camera, using the OpenNI libraries [11] and the Point Cloud libraries [12]. We used the ROS `openni_tracker` component to gather the 3D Cartesian positions of the human body segments. In particular, the 3D position of an operator hand.

A. Kinect-robot calibration

As a first step, the calibration between the robot operational space and the Kinect camera space was performed. The goal of the calibration procedure was to compute the transformation ($\boldsymbol{T}_k^{Rb}$) from the Kinect reference frame (RF), {K}, to the robot base RF, {R_b} (Fig. 2). We glued paper colored (blue, green, red) 3 cm diameter circles on two rigid cardboards. Each cardboard was then attached onto the base and the end-effector (EE) of the robot. The three circles were used to build the two reference frames, {M_b} and {M_{ee}} and to compute the homogeneous transformations $\boldsymbol{T}_K^{Mb}$ and $\boldsymbol{T}_K^{Mee}$. With $\boldsymbol{T}_K^{Mb}$ and $\boldsymbol{T}_K^{Mee}$ known, we computed the **A** transformation between {M_b} and {M_{ee}} as:

$$\mathbf{A} = \mathbf{T}_{\mathrm{K}}^{\mathrm{Mb}\,-1} \cdot \mathbf{T}_{\mathrm{K}}^{\mathrm{Mee}} \qquad (1)$$

B is known via the forward kinematics while **X** and **Y** were determined using an hand-eye calibration procedure [13]. For the hand-eye procedure, the robot was moved in 14 different poses, exploring large part of the entire robot workspace. Once **Y** was known, it was possible to compute the calibration matrix between the robot base RF {R_B} and Kinect RF{K} ($\boldsymbol{T}_k^{Rb}$) as:

$$\mathbf{T}_{\mathrm{k}}^{\mathrm{Rb}} = \mathbf{T}_{\mathrm{K}}^{\mathrm{Mb}} \cdot \mathbf{Y} \qquad (2)$$

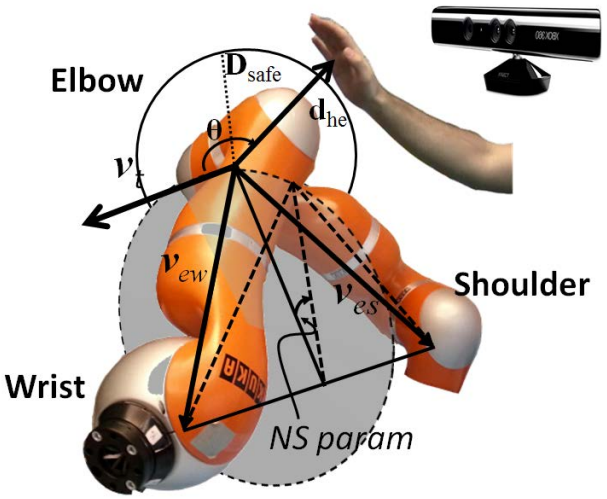

Fig. 3 Collision avoidance algorithm parameters. The null space parameter [10] (*NS param*) is the angle spanned by the triangle plane built as the virtual connection of the shoulder, the elbow and wrist of the robot. $\boldsymbol{v}_{es}$ is vector from elbow to the shoulder, $\boldsymbol{v}_{ew}$ is the vector from elbow to wrist, $\boldsymbol{v}_t$ is the tangent vector to the circumference virtually spanned by the robot elbow, while keeping fixed the wrist pose. $\boldsymbol{d}_{he}$ is the distance vector between the user hand and the elbow.

B. Collision avoidance algorithm

The collision avoidance strategy is aimed at exploiting the kinematic redundancy of the robot in order to keep the links of the robot far from the obstacle, while maintaining the same position of the EE.

As an exemplary scenario, we considered the robot with a fixed pose of its EE on the body of a patient (during a surgery) and the surgeon moving his hand close to the robot. The final goal was to update the value of the null space parameter (*NS param* in Fig. 3) [10] according to a high level control law.

The control law is based on the concept of "safe distance". We defined a spherical region of surveillance with a radius, D_{safe} = 35 cm around the elbow of the robot (Fig. 3). As soon as the hand of the operator enters in the virtual safety sphere, it is identified as an obstacle and the robot changes its configuration. The control law algorithm is detailed in Fig. 4. As a first step, the cartesian pose of the elbow, $\boldsymbol{x}_{elbow}^{Rb}$, is given in {R_b}, via the Fast Research Interface (FRI) [14]. The hand position, $\boldsymbol{x}_{hand}^{K}$, is acquired by the Kinect sensor with respect to {K} RF. Then, using the calibration matrix $\boldsymbol{T}_K^{Rb}$, the elbow position vector $\boldsymbol{x}_{elbow}^{Rb}$(Fig.1), is transformed into the {K} RF:

$$\mathrm{x}_{\mathrm{elbow}}^{\mathrm{K}} = \mathrm{T}_{\mathrm{K}}^{\mathrm{Rb}} \cdot \mathrm{x}_{\mathrm{elbow}}^{\mathrm{Rb}} \qquad (3)$$

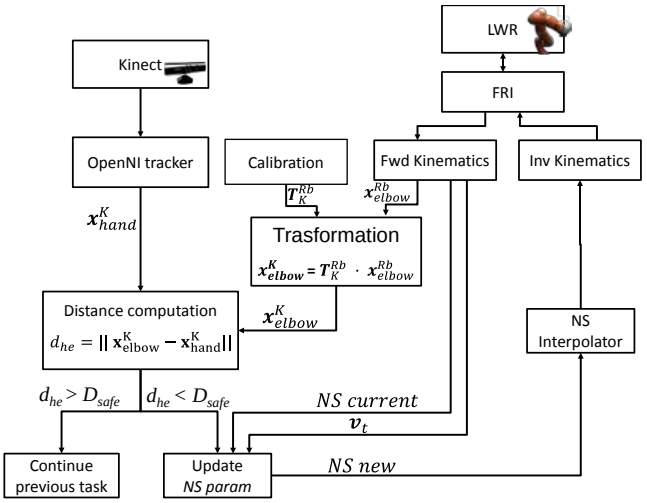

Fig. 4 Collision avoidance scheme.

Thus, the distance d_{he} between the hand and the robot elbow is computed. If d_{he} is less than D_{safe} the robot moves away changing its NS parameter, otherwise it keeps the current position.

The current joint positions are the inputs for the forward kinematics component which computes the current NS parameter (*NS current*) and a tangent vector ($\boldsymbol{v}_t$) (Fig. 3). The latter is the vector constructed as the cross product between the elbow-shoulder vector ($\boldsymbol{v}_{es}$) and the elbow-wrist vector ($\boldsymbol{v}_{ew}$) (Fig. 3):

$$\mathbf{v}_t = \mathbf{v}_{es} \times \mathbf{v}_{ew} \tag{4}$$

$\boldsymbol{v}_t$ is the tangent vector to the circle drawn by the movement of the elbow keeping the wrist and the shoulder fixed in position. $\boldsymbol{v}_t$ and the current *NS param* are published as ROS topics to the component in charge of updating the *NS param* value (*NS new*). The *NS param* value is updated starting from the *NS current*, with the following update rule:

$$NS\,new = NS\,current \mp k\left(\frac{D_{safe}}{d_{he}}\right) \tag{5}$$

with k being the proportional gain set to 0.05. The rule increases or descreases the *NS current* depending on the sign and the value of the cosine of the angle, θ, between $\boldsymbol{d}_{he}$ and $\mathbf{v}_t$ (Fig. 3).

We defined three regions for the possible *cos(θ)* values:

$\bar{c} < cos(\theta) < 1$ (Region 1),
$-\bar{c} < cos(\theta) < \bar{c}$ (Region 2),
$-1 < cos(\theta) < -\bar{c}$ (Region 3).

The complete *NS param* update rule is given by the following pseudo code:

```
If (d_he, < D_safe)
    If  (c̄ < cos(θ) < 1)
        NS new = NS current − k(D_safe / d_he);
    If  (−1 < cos(θ) < −c̄ )
        NS new = NS current + k(D_safe / d_he);
    Else
    Continue previous task
End
```

Region 2 is needed in order to avoid instability, due to the continuous change of the *NS param* from a positive value to a negative value due to the noise on $\boldsymbol{d}_{he}$.

Once the new *NS param* is known, the ROS component publish the new value (*NS new*) to the OROCOS component in charge of the interpolation (NS interpolator), according to the update rate of the FRI component (500 Hz). Then, the inverse kinematics component computes the new joint positions.

C. Experimental setup

a) Kinect hand tracking error

An operator freely moved both the hands in 3D space, handling a rigid cable of 50cm length at 1.2m distance from the Kinect sensors. Hand coordinates were acquired for 45s at 30Hz. The distance ($d_{H\text{-}H}$) between the two hands was computed, expecting $d_{H\text{-}H}$ to be equal to the length of the cable.

b) Kinect hand tracking error after calibration

An operator freely moved one hand in 3D space handling the LWR EE in order to overlap the hand position origin, $\boldsymbol{x}_{hand}^K$, with the {R_{ee}} RF origin; For this task, the robot was set to gravity compensation modality [14]. Hand positions were recorded for 120s with respect to {K} RF, while the EE position, $\boldsymbol{x}_{EE}^{R_b}$, were acquired via forward kinematics and then transformed in the {K} RF via the calibration matrix T_K^{Rb}. Then, the error distance between the hand position x_{hand}^K, and the EE origin x_{ee}^K was computed in {K} operational space:

c) Collision avoidance evaluation

The entire system was tested asking the operator to freely approach the robot elbow with his hand. The robot was approached 5 times from the left side and 5 from the right side, inside the safety zone. The trials were repeated at seven different update frequencies of the ROS component (10, 20, 30, 50, 100, 250, 500 Hz). The maximum allowed angular speed of the elbow was 1.4rad/s and the maximum Cartesian velocity was 0.2m/s, for safety reasons. We evaluated the time delay, T_d, as the time interval which starts as soon as the distance d_{he} is smaller than D_{safe} and ends when the robot starts to move (i.e. when *NS param* starts to change, Fig. 5).

I. Results

The Kinect hand tracking error showed accuracies which depends on hand velocities. The maximum median error was about 8 mm with interquartile range up to 8 cm.

Hand tracking error after calibration showed a median error of about 22 cm (16 cm and 30 cm, 25th and 75th percentile respectively).

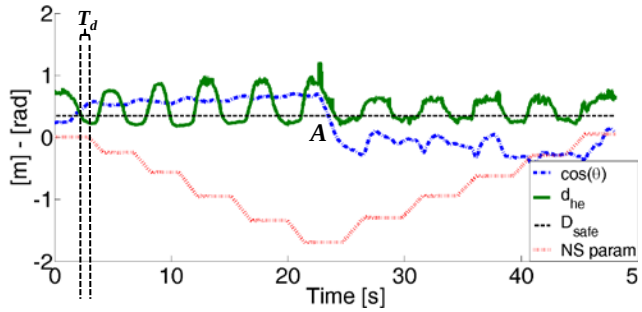

Fig. 5 Example of collision avoidance trial. The hand-elbow distance d_{he}, the safe threshold D_{safe} , the value of the *NS param* and the *cos(θ)* are shown . In A, there is a smooth change of the sign of the cos(θ) , when the user moved the hand on the other side of the robot.

Fig. 5 shows the results of a collision avoidance trial. As soon as the distance d_{he} goes below D_{safe} the robot start to move, changing the *NS param* (after T_d delay).
The time delay T_d (Fig. 6) quickly decreases with the ROS update rate, reaching a plateau below 0.1 ms at 100 Hz (80ms), without considerable improvement for higher frequencies.

II. Discussion and Conclusion

In this work, we showed a collision avoidance strategy able to change configuration of a Kuka LWR4+ robot, keeping fixed the pose of its EE. The calibration between the robot operational space and the Kinect sensor space allows not to track the robot skeleton with the Kinect sensor, avoid possible human – robot occlusions, since the robot link positions are always known from the robot kinematics.
Results showed that the time delay, T_d, between the detection of the possible collision and the actual motion of the robot links was below 0.1s for high update rates of the ROS modules. Given this time delay, it is possible to compute the minimum hand cartesian velocity to hit the robot as $\mathbf{v}_h = D_{safe}/T_d = 3.5$ m/s. We could not find references of speed of motions inside the OR, however, if this hand velocity can happen inside the OR, higher values of D_{safe} should be considered, since lower value of T_d. are not reachable in this architecture. Once the robot starts to move, the maximum speed of the user hand should be kept below the maximum allowed velocity of the elbow, se to 0.2 m/s. It should be noted that in the time delay, T_d, we neglected the delay due to the Kinect sensor computation of the hand position. This delay will be further evaluated tracking both the hand and the robot with an accurate external optical localization system. The collision avoidance method is not affected by the high Kinect tracking error (about 20cm), unless a specific safety distance is chosen to be considerably higher than the tracking accuracy.

The work is a preliminary study on the possibility to use such a system in the real OR, extending the algorithm to multi-body distance computation from all the joints of the robot. This will require increased computational costs that will be evaluated in further tests.

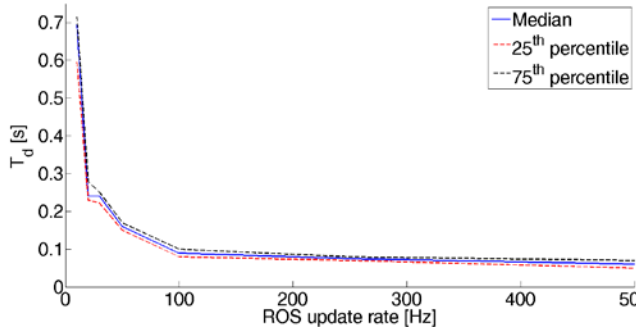

Fig. 6 Time delay T_d with respect to the ROS components update rate. Above 100Hz, the delay reaches a plateau of about 0.08s as median value.

References

[1] T. R. K. Varma, P. R. Eldridge, A. Forster, S. Fox, N. Fletcher, M. Steiger, P. Littlechild, P. Byrne, A. Sinnott, K. Tyler, S. Flintham, "Use of the NeuroMate Stereotactic Robot in a Frameless Mode for Movement Disorder Surgery," Stereotact Funct Neurosurg, vol. 80, pp. 132-135, 2003.

[2] G. R. Sutherland, I. Latour, A.D Greer, "Integrating an image-guided robot with intraoperative MRI: a review of the design and construction of neuroArm," IEEE Eng Med Biol Mag., vol. 27(3), pp.59-65, 2008.

[3] A. De Luca, A. Albu-Schaffer, S. Haddadin, G. Hirzinger, "Collision Detection and Safe Reaction with the DLR-III Lightweight Manipulator Arm," IEEE/RSJ International Conference on Intelligent Robots and Systems, 2006, pp.1623-1630, Oct. 2006, doi: 10.1109/IROS.2006.

[4] F. Flacco, T. Kroger, A. De Luca and O.Khatib, "A depth space approach to human-robot collision avoidance," IEEE International Conference on Robotics and Automation (ICRA), pp.338-345, 14-18 May 2012, doi: 10.1109/ICRA.2012.6225245

[5] T. Kröger, "Opening the door to new sensor-based robot applications — The Reflexxes Motion Libraries", in Proc. of the IEEE International Conference on Robotics and Automation, Shanghai, China, May 2011.

[6] G. Ferrigno, G. Baroni, F. Casolo, E. De Momi, G. Gini, M. Matteucci and A. Pedrocchi, "Medical Robotics and Computer-Aided Therapy", IEEE PULSE, vol. 2(3), pp. 55-61, 2011.

[7] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Y. Ng, "ROS: an open-source Robot Operating System," in Proc. Open-Source Software workshop of the International Conference on Robotics and Automation (ICRA), 2009.

[8] H. Bruyninckx, P.Soetens and B. Koninckx. The Real-Time Motion Control Core of the Orocos Project. IEEE International Conference on Robotics and Automation, p. 2766—2771, 2003

[9] A. De Luca and L. Ferrajoli, "Exploiting robot redundancy in collision detection and reaction," IEEE/RSJ International Conference on Intelligent Robots and Systems. IROS 2008, pp.3299-,3305, 22-26 Sept. 2008, doi: 10.1109/IROS.2008.4651204

[10] M. Shimizu, H. Kakuya, Y. Woo-Keun, K. Kitagaki and K. Kosuge, "Analytical Inverse Kinematic Computation for 7-DOF Redundant Manipulators With Joint Limits and Its Application to Redundancy Resolution," IEEE Transactions on Robotics, vol. 24(5), pp.1131-1142, Oct. 2008, doi: 10.1109/TRO.2008.2003266.

[11] OpenNi Framework, Documentation Homepage "http://www.openni.org (accessed: Nov. 2012)", Internet, 2012.

[12] R. B. Rusu, and S. Cousins, "3D is here: Point Cloud Library (PCL)," 2011 IEEE International Conference on Robotics and Automation (ICRA), pp. 1-4, 9-13 May 2011, doi: 10.1109/ICRA.2011.5980567

[13] M. D. Comparetti, A. Vaccarella, I. Dyagilev, M. Shoham, G. Ferrigno, and E. De Momi, "Accurate multi-robot targeting for keyhole neurosurgery based on external sensor monitoring," Proc Inst Mech Eng H., vol. 226(5), pp. 347-59, May 2012.

[14] G. Schreiber, A. Stemmer, and R. Bischoff, "The fast research interface for the kuka lightweight robot", IEEE International Conference on Robotics and Automation (ICRA 2010) Workshop on Innovative Robot Control Architectures, 2010.