Human-Robot Contactless Collaboration with Mixed Reality Interface*

Maram Khatib[a], Khaled Al Khudir[a,*] and Alessandro De Luca[a]

[a]Dipartimento di Ingegneria Informatica, Automatica e Gestionale, Sapienza Università di Roma, Via Ariosto 25, Roma, 00185, Italy

ARTICLE INFO

Keywords:
Control system
Robotics
Human-robot collaboration
Redundancy control
Collision avoidance
Mixed reality

ABSTRACT

A control system based on multiple sensors is proposed for the safe collaboration of a robot with a human. New constrained and contactless human-robot coordinated motion tasks are defined to control the robot end-effector so as to maintain a desired relative position to the human head while pointing at it. Simultaneously, the robot avoids any collision with the operator and with nearby static or dynamic obstacles, based on distance computations performed in the depth space of a RGB-D sensor. The various tasks are organized with priorities and executed under hard joint bounds using the Saturation in the Null Space (SNS) algorithm. A direct human-robot communication is integrated within a mixed reality interface using a stereo camera and an augmented reality system. The proposed system is significant for on-line, collaborative quality assessment phases in a manufacturing process. Various experimental validation scenarios using a 7-dof KUKA LWR4 robot are presented.

1. Introduction

The capability of handling tasks that involve interaction between humans and robots has become nowadays a highly desirable feature in both industrial and service environments [1], as well as one of the enabling technologies of Industry 4.0 [2, 3]. Robot co-workers should be able to share their workspace and collaborate safely with humans, combining and enhancing the skills of both parties [4]. A hierarchical control architecture to handle safe human-robot interaction can be organized in three functional layers: *safety*, *coexistence*, and *collaboration* [5]. Each layer addresses a desired robot behavior, preserving consistency with the lower layers in the architecture. The *safety* layer at the bottom is always active and deals with collision detection, most conveniently without resorting to extra sensors as in [6], specifying also how the robot should promptly react to undesired (and unavoidable) contacts. The intermediate layer is devoted to *coexistence*: it allows sharing a common workspace while the robot and the human perform independently their jobs. Collisions are prevented here, based on real-time information from external sensors monitoring the whole operation of the system [7]. Finally, physical [8] or contactless [9] human-robot *collaboration* is established in the top layer. In [10], these three control layers have been mapped into the four forms of interaction modes of the ISO 10218 standard [11, 12] (enhanced by the technical specification TS 15066 [13]). In this case, the *safety* layer is involved in all interaction modes, namely, the Safety-rated Monitored Stop (SMS), the Hand Guiding (HG), the Speed and Separation Monitoring (SSM), and the Power and Force Limiting (PFL) modes. Our *coexistence* layer handles specifically the SMS and SSM modes. Finally, the *collaboration* layer addresses tasks in the HG and the PFL modes.

Human-robot contactless collaboration can be achieved through direct communication using gestures [14] and/or voice commands [15]. Indirect communication during interactional context can also be considered by recognizing human intentions [16]. In [9], we proposed a passive communication for a contactless vision-based collaborative task, by imposing a coordinated motion between the robot and a human operator. For such a collaboration, localizing the human pose and detecting moving obstacles in the workspace should both be guaranteed.

For the 3D localization of human body parts, limbs, or head, different sensors can be employed, such as laser range finders [17] or vision/depth cameras for extracting the human pose [18]. Another modality is to attach a compact RGB-D (depth) sensor on the human body, and then localizing it with different techniques [19, 20]. In [9], we compared three different localization methods introduced in [21], [22], and [23]. All these techniques suffer from inefficiency during fast human motion, in highly dynamic environments, or when markers/features are not present. Moreover, they need a frequent and complex calibration phase. To overcome such problems, the tracking sensor of the Oculus Rift system (a HMD for Virtual Reality (VR) exploration) could be used, as we do in this paper. This sensor does not need markers or specific features, allows the human to look and move freely in the workspace, and provides a sufficiently accurate pose estimation both in static and dynamic environments, during fast human motion, and in bad lighting conditions. Furthermore, it can be used to introduce a mixed reality interface for end-user robot programming [24] or for helping the operator in the quality assessment of the product of an industrial process [25].

For the detection of obstacles in the robot workspace, several sensors and methods have been proposed. Laser and sonar sensors may monitor the workspace and detect obstacles that intersect a 2D scanning plane (usually, parallel to the floor and at the calf height or at the torso), allowing the robot to avoid at least parts of the human body [26, 27]. To cover the upper body (arms and chest), several inertial measurement units (IMUs) can be integrated [28]. Detection of

*This paper has been reviewed by FAIM 2020 fast-track process [FAIM paper 21212], and by the Editor-in-Chief of RCIM Journal, Prof. Lihui Wang.

Email: {khatib, alkhudir, deluca}@diag.uniroma1.it.
*Corresponding author.

the whole body (or, simultaneously, of several of its parts) can be achieved either by attaching passive or active markers to the body, or by extracting its shape from RGB/depth images as a 'point cloud' in the Cartesian space [29]. However, using the aforementioned methods, the robot would avoid only the human body and possibly neglect other dangerous obstacles in the workspace. In [30], a laser sensor was attached close to the robot end-effector to compute distances and danger zones from nearby obstacles. Unfortunately, repeating this arrangement for each robot link that may collide would be rather inefficient and too expensive. Alternatively, a visual workspace monitoring system can be used to determine a variable protective separation distance between the end-effector tool and a human operator [31].

In our work, we adopt the approach developed in [7] that uses one or more depth sensors (a single Kinect in our case) to monitor the workspace. A computationally efficient algorithm, which works directly in the so-called depth space of the sensor, evaluates in real time the distances between a number of control points on the robot and *any* other object (the whole human body and other static or dynamic obstacles) in the workspace. Based on this distance information, collisions can be avoided by using any preferred variant of the artificial potential fields method [32].

The goal of our research work is to define a framework for achieving a number of collaborative tasks that require coordinated robot-human motion, by integrating a suite of sensors in order to monitor the workspace, safely control the robot so as to avoid accidental collisions, and provide the user with awareness of the ongoing interaction task. The proposed framework is significant for human-robot collaborative phases of process quality assessment, e.g., within automotive manufacturing lines [33] or in surface finishing applications [25], where the robot should hold and present the processed work piece to the human operator in a specific position and orientation.

The main contributions of the paper can be summarized as follows.

- Definition and realization of a control scheme for contactless human-robot collaboration tasks and simultaneous safe coexistence, based on the multi-sensor system of Fig. 1.
- Integration of direct human-robot communication in a mixed reality interface that allows the operator to change online the collaboration mode, while providing the current status of the collaborative task.
- Specification of different coordinated motion tasks in which the robot end-effector follows a possibly time-varying desired pose (i.e., with position and pointing subtasks) relative to the head of a human operator in motion. Limitations in the motion coordination are identified and an algorithm is proposed to avoid the corresponding task singularities.
- Avoidance of any robot collision with other parts of the human operator body and with all the nearby obstacles, using an efficient distance evaluation method based on a Kinect depth sensor placed in the environment.
- Optimal execution of the above specified tasks in the presence of hard bounds on robot actuation, obtained by exploiting the available kinematic redundancy of the robot, organizing the multiple tasks by priority, and handling objectives and equality/inequality constraints in real time, based on the Saturation in the Null Space (SNS) algorithm [34]. The latter is implemented at the joint acceleration level, so as to guarantee also smoothness of the robot commands.

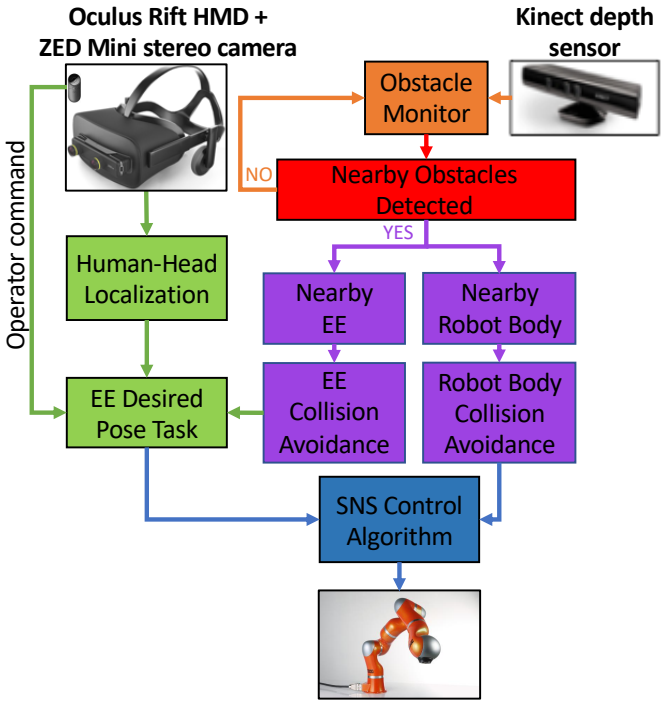


Figure 1: The proposed multi-sensor control scheme for safe human-robot contactless collaboration.

The paper is organized as follows. The human head localization and the mixed reality interface are introduced in Sec. 2. Section 3 presents the desired coordination tasks and the proposed task limit sphere. Section 4 presents the robot controller for motion coordination with simultaneous collision avoidance using the depth space approach. Experimental results with a KUKA LWR4 robot are reported in Sec. 5. A video of the experiments is also available in the supplementary material. Conclusions and future work are discussed in Sec. 6.

2. Human-Robot Awareness

To perform a friendly contactless collaboration experience, both the robot and the human should be aware about each other current action and location. For this, we propose to use the Oculus Rift device together with its tracking sensor for human pose localization. On the other hand,

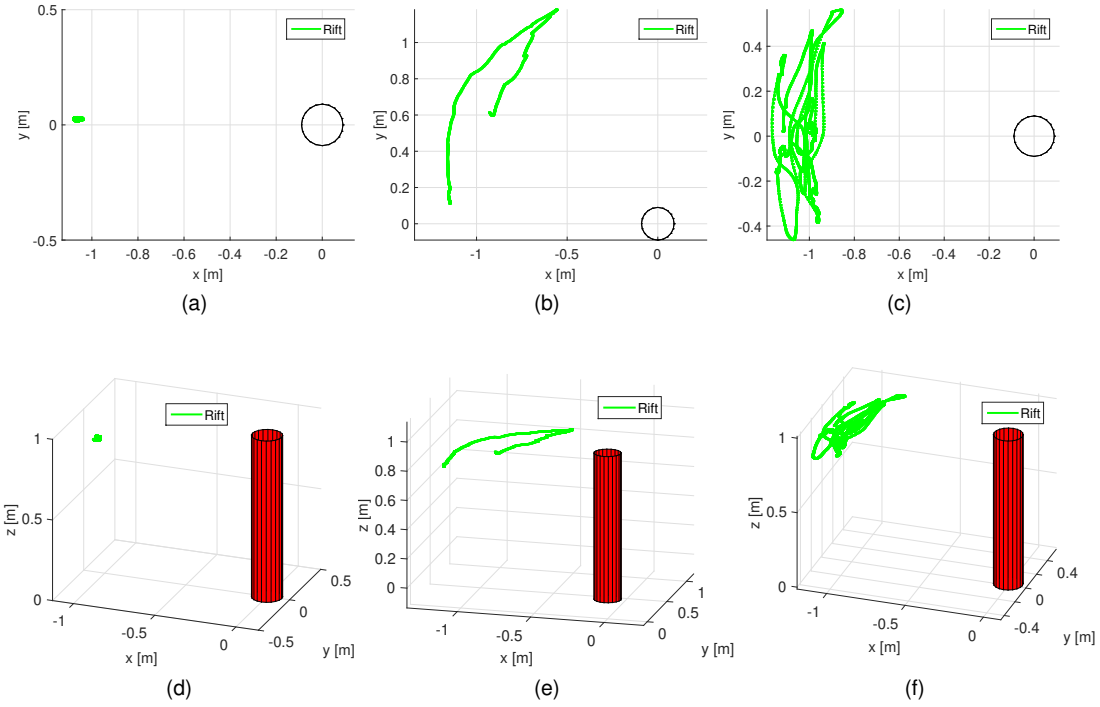


Figure 2: The estimated trajectory of a moving Oculus Rift during three human actions: (a-d) static; (b-e) slow motion; (c-f) fast and long motion. [top] The top view in 2D, and [bottom] for the 3D-view. The circle/cylinder denotes the robot position.

a Mixed Reality-Head Mounted Display (MR-HMD) interface is designed to enable the human to know what the robot is currently doing. The user will be able to connect with the robot directly by switching between different collaboration modes using the Oculus supplied controller. Furthermore, a depth sensor is used to compute the distances between the robot and close objects including the operator (more details in Sec. 4).

2.1. Human head localization

The Oculus system provides a Virtual/Augmented Reality experience by synchronizing the user view in the screen of the HMD with his head motion in the real world. This is done by estimating on line the six degrees of freedom of the device, including position and orientation represented by roll-pitch-yaw angles, and their first and second derivatives, through a sensor fusion process [35]. Data coming from the micro-electrical-mechanical sensors (MEMS) on the Rift, that include gyroscope, accelerometer, and magnetometer, and from the IR on the tracking sensor are combined. In our application the Oculus is used to provide the human head pose data to our control algorithm. For this, the tracking sensor should be located in a static place near to the human motion area, and a simple calibration procedure should be done each time the placement of the tracking sensor is changed. The tracking sensor is able to detect and localize the Rift in a distance range from 0.4 to 2.5 [m]. Multiple tracking sensors could be used to cover a larger area.

For our application, we checked the Oculus localization performance experimentally through different scenarios. The first case is shown in Fig. 2(a), where the Rift was mounted on a standing up human without moving for a duration of 60 [s] in a dynamic environment. In the second case, Fig. 2(b), the human was moving during the experiment. Finally, we tested the localization during fast and long duration motions, see Fig. 2(c). In all previous experiments, the Rift pose estimation was stable, continues and deterministic. This system has a simple setup, an easy initialization phase, and returns accurate HMD pose estimation relative to the desired world reference frame.

2.2. Mixed reality interface

To let the operator aware about the active robot task, and give him the possibility to command the robot directly and efficiently, we propose to add a mixed reality interface to the HMD. For this, a stereo camera is mounted to the Oculus Rift as shown in Fig. 3. Using the Unity cross-platform [36], the surrounding workspace of the operator can be rendered in the HMD screen and augmented with any useful information about the robot behavior and any desired optional commands.

For our proposed application, we designed a simple interface, as shown in Fig. 4, which consists of a static menu with four buttons represent the available collaboration modes. The user can switch between them using the Rift controller. The first mode is *follow*, where the robot should track a dynamic target position with respect to the human-head while pointing to it with a relaxed angle 5° or 90°. In the *circle* mode, the robot should achieve a variable circle that centered on a dynamic position w.r.t. the human head, and

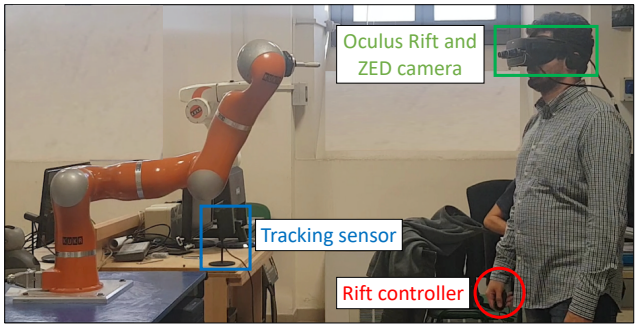


Figure 3: The hardware setup to achieve a mixed reality experience during human-robot collaboration.

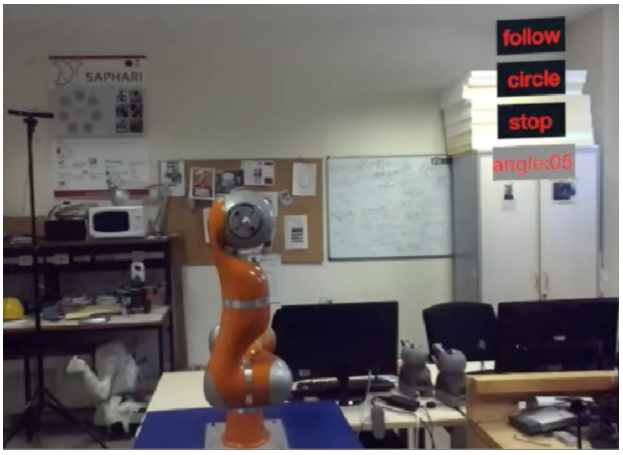


Figure 4: The mixed reality user interface on the Rift HMD screen lenses.

placed on a plane perpendicular to the line of sight of the human. The option *stop* will command the robot with the last computed target point reducing then the residual errors to zero, and finally remaining at rest. The last gray option is to choose between two pointing angles. After selecting the desired mode, the corresponding button is highlighted. More details about the desired robot tasks are given in the next section.

3. Coordinated Motion Tasks

For a robot with n joints, we can define a m-dimensional task to be executed. When $m < n$, the robot will be kinematically redundant for the given task. In this section, we define the coordination tasks of interest for the collaboration modes of our application. We propose also an algorithm for handling the tasks when these cannot be fully executed due to robot workspace limitations.

3.1. Positional task

Consider a desired task described in term of the robot end-effector position,

$$\boldsymbol{p}_{ee} = \boldsymbol{k}(\boldsymbol{q}) \quad \Rightarrow \quad \dot{\boldsymbol{p}}_{ee} = \boldsymbol{J}_{p_{ee}}(\boldsymbol{q})\dot{\boldsymbol{q}}, \tag{1}$$

where $\boldsymbol{q} \in \mathbb{R}^n$ is the robot configuration, $\boldsymbol{k}(.)$ is the direct kinematics, and $\boldsymbol{J}_{p_{ee}} = \partial \boldsymbol{k}/\partial \boldsymbol{q}$ is the $3 \times n$ Jacobian matrix

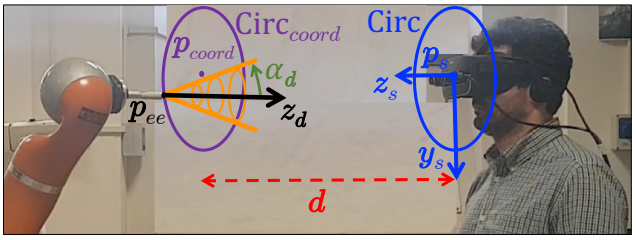


Figure 5: Frames and parameter definitions for the desired coordination tasks. Here, $z_e = z_d$ yielding $\alpha = 0$.

for this task. In this case, the positional error w.r.t a desired task $\boldsymbol{p}_{ee_d} \in \mathbb{R}^3$, can be defined as $\boldsymbol{e}_{p_{ee}} = \boldsymbol{p}_{ee_d} - \boldsymbol{k}(\boldsymbol{q})$. For the proposed contactless collaboration, three different positional tasks are defined as follows.

3.1.1. Human head following

In the first positional task, the tip of the robot should follow a desired Cartesian point defined as

$$\boldsymbol{p}_{ee_d} = \boldsymbol{p}_{coord}(t) = \boldsymbol{p}_s(t) + {}^r\boldsymbol{R}_s(t)\boldsymbol{p}_{sc}, \tag{2}$$

which is attached to the moving Oculus Rift position $\boldsymbol{p}_s(t)$ and translated by $\boldsymbol{p}_{sc} = (x_{sc} \quad y_{sc} \quad z_{sc})^T$, where ${}^r\boldsymbol{R}_s(t)$ is the rotation matrix between the Rift frame and the world reference frame. In our case, $x_{sc} = y_{sc} = 0$ while $z_{sc} = d$, as shown in Fig. 5. The d value can be determined according to the necessary protective distance for SSM in ISO-TS 15066 technical specification. The desired task (2) is corresponding to the positional task of command *follow* in the mixed reality interface in Fig. 4.

3.1.2. Circular task

The second positional task is to track a circular path with variable center by the robot end-effector (EE). As shown in Fig. 5, the circle radius is $r = 0.2$ [m] and its center is at the point $\boldsymbol{p}_{coord}(t)$ which is attached to the moving Rift as in the previous task. In this case, the unit vector $\boldsymbol{z}_d$ (i.e., always parallel to $\boldsymbol{z}_s$) should be orthogonal on the desired circle as

$$\boldsymbol{p}_{ee_d} = \mathrm{Circ}_{coord}(s(t)) = \mathrm{Circ}(s(t)) + {}^r\boldsymbol{R}_s(t)\boldsymbol{p}_{sc}, \tag{3}$$

where

$$\mathrm{Circ}(s(t)) = \boldsymbol{p}_s(t) + r\,(\boldsymbol{u}\cos s(t) + \boldsymbol{n}\sin s(t)), \tag{4}$$

where $\boldsymbol{n}$ and $\boldsymbol{u}$ are any two orthonormal vectors to $\boldsymbol{z}_s$, $s(t)$ is the path parameter, and the translation $\boldsymbol{p}_{sc}$ value is as the previous case. The task (3) is corresponding to the positional task of command *circle* in the mixed reality interface in Fig. 4. Note that, in (2) and (3) the $\boldsymbol{p}_{ee_d}$ is always being updated according to the human-head motion localized by the Oculus Rift.

3.1.3. Stop task

The last positional task is corresponding to the command *stop*, where the robot should regulate to the last computed desired point $\boldsymbol{p}_{ee_d}$ from (2) or (3) and remains at rest.

Indeed, if an obstacle is getting close, the robot moves to avoid the collision and then resumes the task as soon as possible, as detailed in Sec. 4.

3.2. Pointing task

The previous positional tasks have dimension $m = 3$. If a classical 3D pointing task ($m = 2$) or a complete orientation task ($m = 3$) were added to the positional one, the task dimension would reach $m = 5$ or $m = 6$, respectively. For a standard industrial manipulator with $n = 6$ joints, this would imply that just one or no additional dof is left to the robot in order to achieve other tasks, i.e. collision avoidance. To milden this situation, in our framework we have proposed the use of a relaxed pointing task [9], which requires only one additional dof ($m = 3 + 1 = 4$). In this relaxed task the EE unit axis $\boldsymbol{z}_e(\boldsymbol{q})$ (the third column of the rotation matrix $\boldsymbol{R}_e(\boldsymbol{q})$ relative to the world frame) may point only approximately toward the human head and, in fact, should only belong to the surface of a pointing cone. This cone, which is again determined from the estimated head pose, has its apex located at $\boldsymbol{p}_{ee_d}(t)$ with an apex angle $\alpha_d > 0$ and unit axis $\boldsymbol{z}_d(t)$ which is ideally pointing at the human eyes as shown in Fig. 5. In this case,

$$\begin{aligned} \boldsymbol{z}_d(t) &= {}^r\boldsymbol{R}_{e_d}(t) \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^T, \\ {}^r\boldsymbol{R}_{e_d}(t) &= {}^r\boldsymbol{R}_s(t) \begin{pmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix}. \end{aligned} \tag{5}$$

That is, the current EE pointing can be expressed as

$$p_{rp}(\boldsymbol{q}) = \boldsymbol{z}_d^T \boldsymbol{z}_e(\boldsymbol{q}) = \cos\alpha, \tag{6}$$

where for a constant desired relative angle α_d,

$$p_{rp_d} = \cos\alpha_d, \tag{7}$$

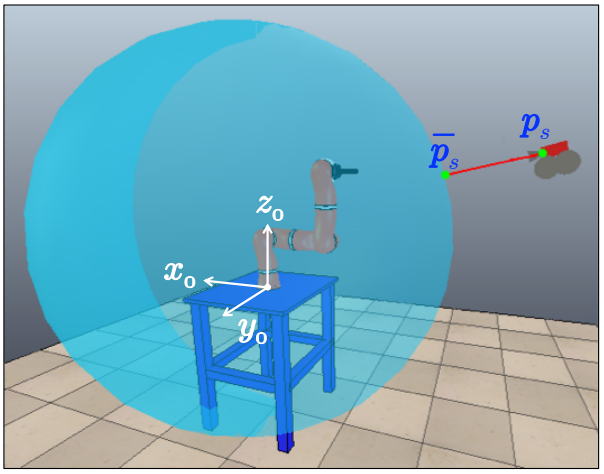
and in this case, the error is computed as

$$e_{rp} = p_{rp_d} - p_{rp} \in \mathbb{R}. \tag{8}$$

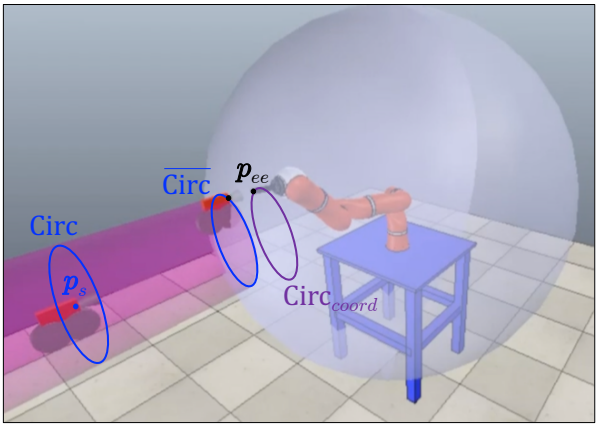
Through our mixed reality interface in Fig. 4, the user can determine the desired *angle* α_d to be 5° or 90°. Indeed, $\alpha_d = 5°$ is useful during a collaborative quality assessment procedure. On the other hand, $\alpha_d = 90°$ decreases the hazard of the EE critical tools.

3.3. Task limit sphere

During the human-robot coordinated motion, some limits may be violated when the operator moves outside the robot workspace, where the desired positional and/or pointing task cannot be fulfilled according to the current operator pose. For this situation, we propose a special treatment for the desired task to avoid any operational task singularity. As shown in Fig. 6, a Cartesian boundary is defined around the robot by a virtual sphere $\boldsymbol{S}$. In general, the sphere should be determined according to the robot allowable workspace. In our case, the center of the sphere is placed at the second joint (the robot shoulder), with a radius $r_s = 1$ [m] equal to the total length from the second joint to the tip of the EE auxiliary tool.



(a)



(b)

Figure 6: The task limit sphere represents a boundary for the coordination task. (a) When the sensor is outside the sphere in the position $\boldsymbol{p}_s$, the projected position $\overline{\boldsymbol{p}}_s$ on the surface of the sphere will be used as reference to compute the *follow* task in (2). (b) The desired circular task Circ_{coord} in (3), when all Circ points are out of the task limit sphere (see the accompanying video for a complete understanding).

The scheme in Fig. 7, together with the Algorithm 1, illustrates how we propose to deal with the coordination task limits. If the position of the human head is inside the sphere, the robot EE desired position for the *follow* task will be defined as in (2). If the human head goes beyond the bounding sphere, the desired task position will be accordingly relocated at the intersection point $\overline{\boldsymbol{p}}_s$ between the human line of sight $\boldsymbol{z}_s$ and the sphere $\boldsymbol{S}$. Otherwise, if there is no intersection, the desired task will not be updated and the robot will regulate for the last visible task. Furthermore, the task limits could be violated if the operator is looking to the outside of the robot workspace. Also in this case the desired task will not be updated as before. The motion control is resumed with the last estimate pose, as soon as the position

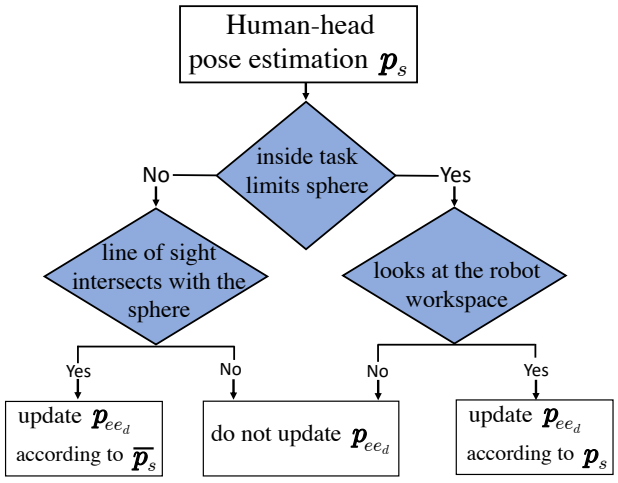


Figure 7: Different situations for the human-head pose, and how it gets modified according to the task limit sphere.

and pointing direction of the operator become again feasible for the task.

The full procedure for computing the desired robot task is presented in Algorithm 1. First, the *incidence* is computed to check the intersection between the human line of sight $\boldsymbol{z}_s$ and the sphere $\boldsymbol{S}$. If an intersection exists, i.e., the variable *incidence* > 0, the two intersection points $(\overline{\boldsymbol{p}}_1, \overline{\boldsymbol{p}}_2)$ are computed. Indeed, these two points could be in front of the human (in the direction of the human line of sight) or behind him. For this, the corresponding directions $(\boldsymbol{l}_1, \boldsymbol{l}_2)$ are computed. If the point of intersection is in front of the human and the distance between him and the intersection point is more than r_s, the human is inside the sphere and looking toward the robot (line 10 of the algorithm). In this case, the current pose estimation will be used. If the operator is outside $\boldsymbol{S}$, the desired task will be updated according to the closest intersection point to the operator (lines 16 to 19 of the algorithm).

The same previous procedure is done in case of the desired circular task (3). When any point of the circle is outside the sphere, the desired task will be updated according to the point projection as in Fig. 6(b).

4. Collision Avoidance and Motion Control

In the proposed contactless collaboration, the human operator is supposed to work close to robot while the robot may live in an environment cluttered with obstacles. Therefore, both human safety and robot integrity should always be guaranteed. To determine closeness to obstacles, a single RGB-D camera is used together with the depth space approach of [7], evaluating the distances between a number of control points (including the EE) selected along the robot arm and any obstacle (including the human operator) in the workspace. In this work, as illustrated in Fig. 8, we consider $n_c = 9$ control points along the robot body, four of them are located between the third and fourth joints. While, the next four points are located between the fourth and sixth joints.

Algorithm 1 Coordinated task limit check

1: input: $\boldsymbol{p}_s$, $\boldsymbol{z}_s$, $\boldsymbol{c}_s$, r_s
2: $\boldsymbol{l} = \frac{\boldsymbol{z}_s}{\|\boldsymbol{z}_s\|}$, $a = \|\boldsymbol{l}\|^2$, $b = 2\boldsymbol{l}(\boldsymbol{p}_s - \boldsymbol{c}_s)$, $c = \|\boldsymbol{p}_s - \boldsymbol{c}_s\|^2 - r_s^2$
3: incidence $= b^2 - 4ac$
4: **if** incidence ≤ 0 **then**
5: use the last estimate pose
6: **else**
7: $d_1 = \frac{-2b+\sqrt{\text{incidence}}}{2a}$, $d_2 = \frac{-2b-\sqrt{\text{incidence}}}{2a}$
8: $\overline{\boldsymbol{p}}_1 = \boldsymbol{p}_s + d_1\boldsymbol{l}$, $\overline{\boldsymbol{p}}_2 = \boldsymbol{p}_s + d_2\boldsymbol{l}$
9: $\boldsymbol{l}_1 = \frac{\overline{\boldsymbol{p}}_1 - \boldsymbol{p}_s}{\|\overline{\boldsymbol{p}}_1 - \boldsymbol{p}_s\|}$, $\boldsymbol{l}_2 = \frac{\overline{\boldsymbol{p}}_2 - \boldsymbol{p}_s}{\|\overline{\boldsymbol{p}}_2 - \boldsymbol{p}_s\|}$
10: **if** $\|\boldsymbol{p}_s - \boldsymbol{c}_s\|^2 < r_s^2$ **then**
11: **if** $\{\boldsymbol{l}_1 == \boldsymbol{l}$ and $\|\boldsymbol{p}_1 - \boldsymbol{p}_s\|^2 > r_s^2\}$ or $\{\boldsymbol{l}_2 == \boldsymbol{l}$ and $\|\boldsymbol{p}_2 - \boldsymbol{p}_s\|^2 > r_s^2\}$ **then**
12: update the task with the current pose estimation
13: **else**
14: use the last estimate pose
15: **end if**
16: **else if** $\|\overline{\boldsymbol{p}}_1 - \boldsymbol{p}_s\|^2 > \|\overline{\boldsymbol{p}}_2 - \boldsymbol{p}_s\|^2$ and $\boldsymbol{l}_2 == \boldsymbol{l}$ **then**
17: update the task according to $\overline{\boldsymbol{p}}_2$
18: **else if** $\|\overline{\boldsymbol{p}}_2 - \boldsymbol{p}_s\|^2 > \|\overline{\boldsymbol{p}}_1 - \boldsymbol{p}_s\|^2$ and $\boldsymbol{l}_1 == \boldsymbol{l}$ **then**
19: update the task according to $\overline{\boldsymbol{p}}_1$
20: **else**
21: use the last estimate pose
22: **end if**
23: **end if**

The last control point is located on the robot EE tip. The advantage of the approach [7] is that point-to-object distances are evaluated directly in the depth space of the sensor, allowing large savings in computation times.

4.1. Distance computation in the depth space

Consider an obstacle point $\boldsymbol{o}$ and a generic control point $\boldsymbol{c}$, which are represented in the depth space respectively as ${}^D\boldsymbol{o} = (o_x\ o_y\ d_o)^T$ and ${}^D\boldsymbol{c} = (c_x\ c_y\ d_c)^T$. The first two coordinates represent the position of the projected point in the 2D image plane of the sensor, and the third coordinate represents the depth of this point as seen from the sensor. To compute the Cartesian distance $D(\boldsymbol{c}, \boldsymbol{o})$ between points $\boldsymbol{c}$ and $\boldsymbol{o}$, two different cases are considered.

- If $d_o > d_c$, then

$$\begin{aligned} D(\boldsymbol{c},\boldsymbol{o}) \backsimeq D_D({}^D\boldsymbol{c}, {}^D\boldsymbol{o}) &= \sqrt{a_x^2 + a_y^2 + a_z^2}, \\ a_x &= \frac{(o_x - \rho_x)d_o - (c_x - \rho_x)d_c}{l\ s_x}, \\ a_y &= \frac{(o_y - \rho_y)d_o - (c_y - \rho_y)d_c}{l\ s_y}, \\ a_z &= d_o - d_c, \end{aligned} \tag{9}$$

where D_D is the distance in the depth space, l is the focal length of the depth camera, (s_x, s_y) are the pixel

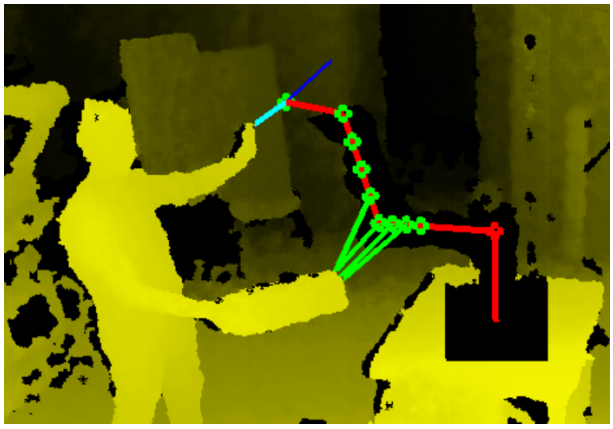


Figure 8: Snapshot of a depth image superposed with the computed distances between control points (green circles) on the robot arm and close objects located in the surveillance area. The picture shows also the computed distance between the end-effector and the human hand obstacle (cyan line), as well as its associated repulsive vector (blue line).

sizes and (ρ_x, ρ_y) are the pixel coordinates of the image plane center.

- If $d_o \leq d_c$, the depth of the obstacle is assumed conservatively to be equal to the depth of the control point ($d_o = d_c$), and the distance is then computed using (9).

To evaluate distances between $\boldsymbol{c}$ and all obstacle points sufficiently close to it, the distance evaluation is applied only to pixels in the depth image plane within a region of surveillance.

4.2. Reactive motion

Consider a generic unit vector between $\boldsymbol{c}$ and $\boldsymbol{o}$ defined in the depth space by

$$\boldsymbol{u}({}^D\boldsymbol{c}, {}^D\boldsymbol{o}) = \frac{(a_x\ a_y\ a_z)^T}{D_D({}^D\boldsymbol{c}, {}^D\boldsymbol{o})}. \tag{10}$$

In general, the *direction* of the desired reaction can be evaluated as the normalized *mean* vector $\hat{\boldsymbol{u}}_{mean}(\boldsymbol{c})$, or the *minimum* vector $\boldsymbol{u}_{min}(\boldsymbol{c})$ of the unit distance vectors between each control point $\boldsymbol{c}$ and all points of objects in the surveillance area, where

$$\begin{aligned} \boldsymbol{u}_{mean}(\boldsymbol{c}) &= \frac{1}{h}\sum_{i=1}^{h} \boldsymbol{u}({}^D\boldsymbol{c}, {}^D\boldsymbol{o}_i), \\ \hat{\boldsymbol{u}}_{mean}(\boldsymbol{c}) &= \frac{\boldsymbol{u}_{mean}(\boldsymbol{c})}{||\boldsymbol{u}_{mean}(\boldsymbol{c})||}, \end{aligned} \tag{11}$$

and

$$\boldsymbol{u}_{min}(\boldsymbol{c}) = \boldsymbol{u}({}^D\boldsymbol{c}, {}^D\boldsymbol{o}_{min}), \tag{12}$$

where h is the total number of all points of objects in the surveillance area of $\boldsymbol{c}$, and $\boldsymbol{o}_{min}$ is the nearest obstacle point to the $\boldsymbol{c}$.

On the other hand, the *magnitude* of the desired reaction can be defined to take into account the *nearest* object to $\boldsymbol{c}$ (at a distance $D_{min}(\boldsymbol{c})$) as [7]

$$u(\boldsymbol{c}) = \frac{u_{max}}{1 + e^{(D_{min}(\boldsymbol{c})(2/\rho)-1)\gamma}}, \tag{13}$$

where

$$D_{min}(\boldsymbol{c}) = \min_{i=1}^{h} D(\boldsymbol{c}, \boldsymbol{o}_i), \tag{14}$$

u_{max} is the maximum admissible magnitude, ρ is the danger threshold distance in the surveillance area, and the factor $\gamma > 0$ shapes the exponential decay rate. In practice, for a large value of γ, the magnitude u of the repulsive vector equals u_{max} when $D_{min}(\boldsymbol{c}) \approx 0$, and approximately vanishes when the distance reaches the ρ value, where beyonds ρ the $u(\boldsymbol{c})$ is not defined. According to (11), (12) and (13) the general repulsive vector associated to a control point can be defined in the world reference frame as

$$\begin{aligned} {}^r\boldsymbol{u}(\boldsymbol{c}) &= {}^r\boldsymbol{R}_D\ {}^D\boldsymbol{u}(\boldsymbol{c}) \\ &= {}^r\boldsymbol{R}_D\, u(\boldsymbol{c})\, \boldsymbol{u}_{mean/min}(\boldsymbol{c}), \end{aligned} \tag{15}$$

where ${}^r\boldsymbol{R}_D$ is the rotation matrix between the depth camera frame and the world reference frame.

4.3. Robot end-effector safety assessment

In our case, the repulsive vector associated to the robot EE control point, i.e. $\boldsymbol{c} = \boldsymbol{p}_{ee}$, is defined in the world reference frame by

$$\begin{aligned} {}^r\boldsymbol{u}(\boldsymbol{p}_{ee}) &= {}^r\boldsymbol{R}_D\ {}^D\boldsymbol{u}(\boldsymbol{p}_{ee}) \\ &= {}^r\boldsymbol{R}_D\, u(\boldsymbol{p}_{ee})\, \hat{\boldsymbol{u}}_{mean}(\boldsymbol{p}_{ee}). \end{aligned} \tag{16}$$

Using (16), the magnitude of the desired reaction considers the nearest object to the EE, whereas the direction takes into account all objects in the danger area. This hybrid reaction scheme allows the robot to escape from possible oscillating behaviors resulting from the topology of multiple close obstacles in the surveillance area [7]. For EE collision avoidance, the repulsive vector in (16) is considered as a repulsive velocity that directly modifies the EE original desired velocity $\dot{\boldsymbol{p}}_{ee_d}$ for the coordinated positional task (defined from Sec.3.1) as [7]

$$\dot{\boldsymbol{p}}_r = \dot{\boldsymbol{p}}_{ee_d} + {}^r\boldsymbol{u}(\boldsymbol{p}_{ee}), \tag{17}$$

where

$$\dot{\boldsymbol{p}}_{ee_d} = v\,\frac{\boldsymbol{p}_{ee_d} - \boldsymbol{p}_{ee}}{\|\boldsymbol{p}_{ee_d} - \boldsymbol{p}_{ee}\|}. \tag{18}$$

The Cartesian speed namely the velocity magnitude v of $\boldsymbol{p}_{ee_d}$ is evaluated at discrete instants $t_k = kT$, being $T > 0$ the sampling time, as

$$\begin{aligned} v_k &= \min\{v_{\max}, v(t_k)\}, \\ v(t_k) &= k_p\|\boldsymbol{p}_{ee_d,k} - \boldsymbol{p}_{ee,k}\| - k_d v_{k-1}, \end{aligned} \tag{19}$$

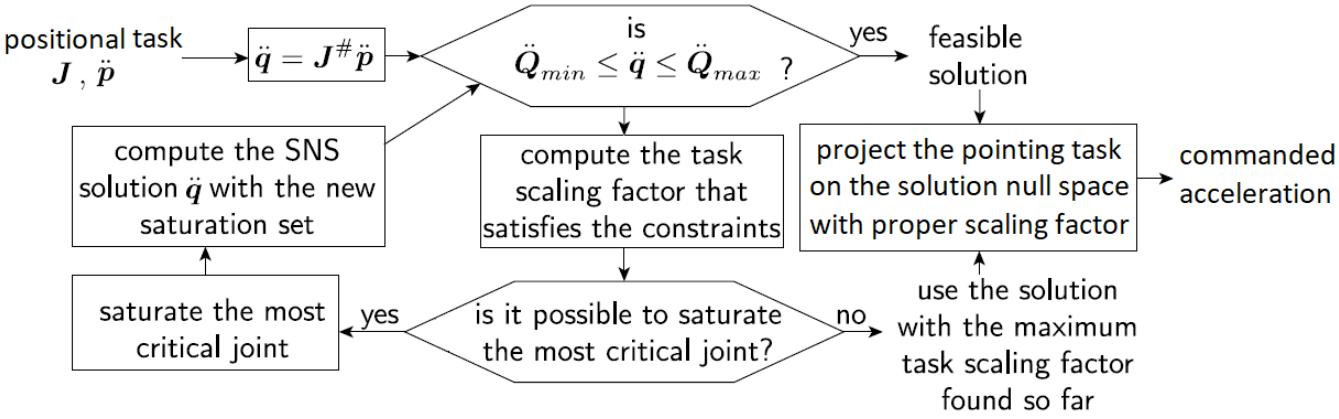


Figure 9: The block diagram of the SNS algorithm.

where $v_{\max}$ is the maximum velocity magnitude, $k_p > 0$, $k_d > 0$, and v_{k-1} is the previous sample (for smoothing purposes). The $v_{\max}$ value can be determined according to the distance d (see Sec. 3.1.1) between the human and the EE, according to the SSM mode in ISO-TS 15066. Furthermore, several risk zones can be considered, with different danger threshold distances ρ and different maximum repulsive magnitude u_{max}. To avoid discontinuities at the joint velocity, we choose to work at the acceleration level. From (17), the commanded task acceleration $\ddot{\boldsymbol{p}}_r$ is obtained as

$$\ddot{\boldsymbol{p}}_r = k_v(\dot{\boldsymbol{p}}_r - \dot{\boldsymbol{p}}_{ee}), \tag{20}$$

where $k_v > 0$.

4.4. Robot body safety assessment

For the robot body collision avoidance, the repulsive vector (15) associated to each control point can be transformed approximately to the robot joint space. Then, all corresponding joint velocities can be accumulated algebraically to be used as the robot desired joint task, or projected in the null space of the robot main Cartesian task [30]. In this case, if there are multiple obstacles moving oppositely to the same control point, the repulsive vectors will cancel/reduce the effect of each other and the collision could be unavoidable.

Instead, in this work the repulsive vectors associated to control points along the robot body are treated as Cartesian constraints with artificial forces that are translated into hard joint velocity and acceleration constraints and used later in the SNS algorithm [34]. In this case, a 'risk of collision' can be defined through the function (similar to (13))

$$f(D_{min}(\boldsymbol{c})) = \frac{1}{1 + e^{(D_{min}(\boldsymbol{c})(2/\rho)-1)\gamma}}. \tag{21}$$

Accordingly, a Cartesian constraint force can be defined as $\boldsymbol{u}_{min}(\boldsymbol{c})f(D_{min}(\boldsymbol{c}))$ and converted to the joint space by

$$\boldsymbol{h}(\boldsymbol{q}) = \boldsymbol{J}_c^T(\boldsymbol{q})\big[\boldsymbol{u}_{min}(\boldsymbol{c})f(D_{min}(\boldsymbol{c}))\big], \tag{22}$$

where $\boldsymbol{J}_c$ is the analytic Jacobian of the direct kinematics for the position of the control point $\boldsymbol{c}$. Each component of the n-dimensional vector $\boldsymbol{h}$ represents the 'degree of influence' of the Cartesian constraint on the homologous joint. Next, the admissible velocity limits of each joint will be reshaped using the risk of collision function (21) according to the rule

$$\begin{aligned} \text{if } h_i > 0, \quad & \dot{q}_{max,i} = \dot{Q}_{max,i}\,(1 - f(D_{min}(\boldsymbol{c})) \\ \text{else}, \quad & \dot{q}_{min,i} = -\dot{Q}_{max,i}\,(1 - f(D_{min}(\boldsymbol{c})), \end{aligned} \tag{23}$$

where $\pm\dot{Q}_{max,i}$ are the (symmetric) original bounds on the ith joint velocity, i.e., $|\dot{q}_i| \leq \dot{Q}_{max,i}$, for $i = 1, \ldots, n$. As a result, the modified velocity limits will be converted as bounds on the actual acceleration commands, namely

$$\ddot{Q}_{min,i} = \frac{\dot{q}_{min,i} + \dot{q}_i}{T} \leq \ddot{q}_i \leq \frac{\dot{q}_{max,i} + \dot{q}_i}{T} = \ddot{Q}_{max,i}, \tag{24}$$

for $i = 1, \ldots, n$. Multiple Cartesian constraints, arising from different obstacles, can be taken into account by considering, for each joint i, the maximum degree of influence of all these virtual constraints. Applying (23) and (24), the robot will immediately stop when the collision cannot be avoided. Indeed, this property is consistent with the SSM mode in ISO-TS 15066.

4.5. SNS algorithm for the first priority task

At this stage, we can apply the simplest version of the SNS algorithm at the acceleration level [34], considering the joint acceleration limits (24), which already embed collision avoidance for the robot body, and exploiting robot redundancy to realize the other desired tasks as much as possible (see Fig. 9). In our framework, the first priority task for the robot is to follow with its EE a specific position trajectory $\boldsymbol{p}_{ee_d}(t)$ that is coordinated with the motion of the head of the human operator, as defined in section 3.1. This will be the case, unless the acceleration command $\ddot{\boldsymbol{p}}_r$ in (20) includes the velocity modification resulting from the EE collision avoidance scheme in (16) and (17).

In the SNS algorithm, the joint acceleration satisfying the first task is computed through some iterations (at the current sampling instant) based on Jacobian pseudoinversion as

$$\ddot{\boldsymbol{q}}_1 = \ddot{\boldsymbol{q}}_{N,1} + (\boldsymbol{J}_1\boldsymbol{W}_1)^{\#}(s_1\ddot{\boldsymbol{p}}_r - \dot{\boldsymbol{J}}_1\dot{\boldsymbol{q}} - \boldsymbol{J}_1\ddot{\boldsymbol{q}}_{N,1}), \quad (25)$$

with the first iteration being initialized with $\boldsymbol{W}_1 = \boldsymbol{I}$, $\ddot{\boldsymbol{q}}_{N,1} = 0$, and $s_1 = 1$. If the joint acceleration in (25) exceeds any of the limits (24) related to the robot body collision avoidance, it will be modified by bringing back to its saturated value the most violating command and projecting it into the null space of the task Jacobian of the enabled (non-saturated) joints (i.e., by suitably modifying the values of $\boldsymbol{W}_1$, and $\ddot{\boldsymbol{q}}_{N,1}$). This is repeated until all acceleration limits are satisfied or when the rank of $\boldsymbol{J}_1\boldsymbol{W}_1 < m$. In the latter case, a proper scaling factor $s_1 \in (0, 1)$ is necessarily used to reduce the original $\ddot{\boldsymbol{p}}_r$ and obtain feasibility. In practice, joint motions that are in contrast with the Cartesian constraints will be scaled down. When a constraint is too close, all joint motions that are not compatible with it will be denied.

4.6. SNS algorithm for the second priority task

The relaxed pointing task defined in section 3.2 can be realized by minimizing the following cost function [9]:

$$H(\boldsymbol{q}) = \frac{1}{2}e_{rp}^2 = \frac{1}{2}\left(\cos\alpha_d - \boldsymbol{z}_d^T\boldsymbol{z}_e(\boldsymbol{q})\right)^2. \quad (26)$$

This will be considered as our second (lower) priority task, thus preserving the higher priority positional task and still without violating the constraints on the joint acceleration commands associated to the robot body collision avoidance requirement. Therefore, the negative gradient of the cost function (26) with a step $k_g > 0$ is projected in the auxiliary null-space projector $\boldsymbol{P}$ given by

$$\boldsymbol{P} = (\boldsymbol{I} - ((\boldsymbol{I} - \boldsymbol{W}_2)(\boldsymbol{I} - \boldsymbol{J}_1^{\#}\boldsymbol{J}_1))^{\#})(\boldsymbol{I} - \boldsymbol{J}_1^{\#}\boldsymbol{J}_1), \quad (27)$$

where initially $\boldsymbol{W}_2 = \boldsymbol{I}$. For each saturated $\ddot{\boldsymbol{q}}_{1i}$, we shall set $\boldsymbol{W}_{2ii} = 0$. Iterations proceed then as for the first task. The final commanded joint acceleration will take the form

$$\ddot{\boldsymbol{q}}_{com} = \ddot{\boldsymbol{q}}_1 + s_2\boldsymbol{P}(-\boldsymbol{D}\dot{\boldsymbol{q}} - k_g\nabla H(\boldsymbol{q})), \quad (28)$$

where $s_2 \in (0, 1)$ is a proper scaling factor introduced only if feasibility with respect to the hard inequality constraints cannot be recovered. The addition of a (diagonal) damping matrix $\boldsymbol{D} > 0$ in the null space is strictly recommended when working with acceleration commands, in order to eliminate any uncontrolled self-motion velocity. The complete multi-task SNS algorithm at the acceleration level can be found in [34].

5. Experimental Evaluation

5.1. Setup

In the experimental setup we have considered a KUKA LWR4 manipulator with $n = 7$ revolute joints. The robot should execute one of the desired coordination tasks defined in Sec. 3 and selected by the operator using the Oculus controller, while avoiding collision with the human and with any static or dynamic obstacle in the environment. The proposed control system in Fig. 1 is implemented using C++ through the ROS 2 middleware. The control framework is implemented according to the data flow diagram in Fig. 10. The KUKA robot is commanded using the position control mode through the Fast Research Interface (FRI) library [37], with a control cycle of $T = 5$ [ms].

For collision avoidance, the workspace is monitored by a Microsoft Kinect depth sensor that captures 640×480 depth images at a frequency of 30 Hz. The camera is fixed at a horizontal distance of 1.5 [m] and at a height of 1.2 [m] w.r.t. the robot base frame. A simple camera calibration process is done in order to compute the transformation between the camera and the world frame. This process is mandatory only once for each camera pose change.

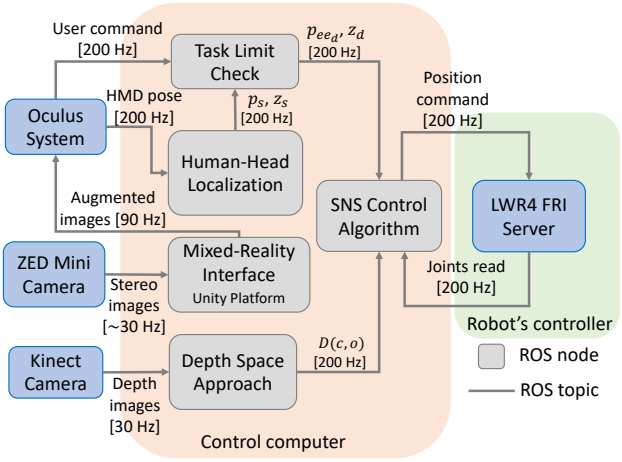


Figure 10: Data flow diagram for the proposed multi-sensor control system.

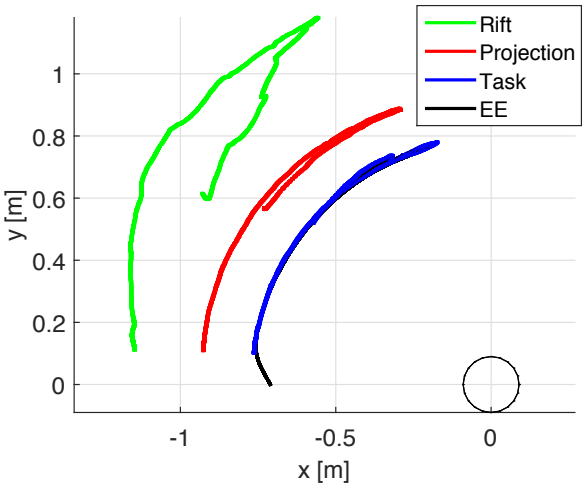


Figure 11: The coordination task during the first experiment. Green traces: Estimated position of the human head. Red traces: Head pose projection on the task limit sphere. Blue traces: Desired positional task. Black traces: EE position. In practice, blue and black traces are superposed. The circle denotes the robot base location.

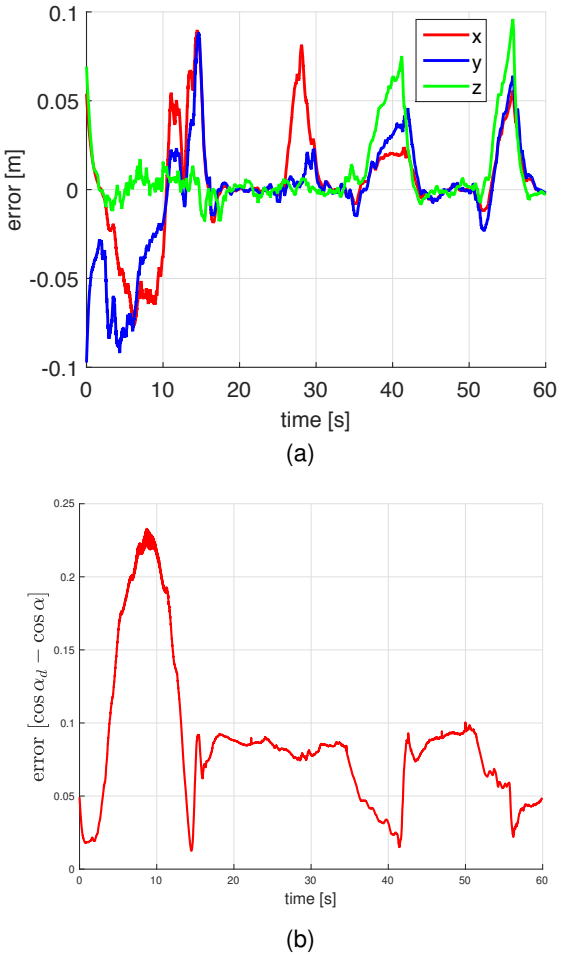


Figure 12: First experiment: The operator is moving and uses his hand as a dynamic obstacle. (a) EE position errors. (b) EE orientation error.

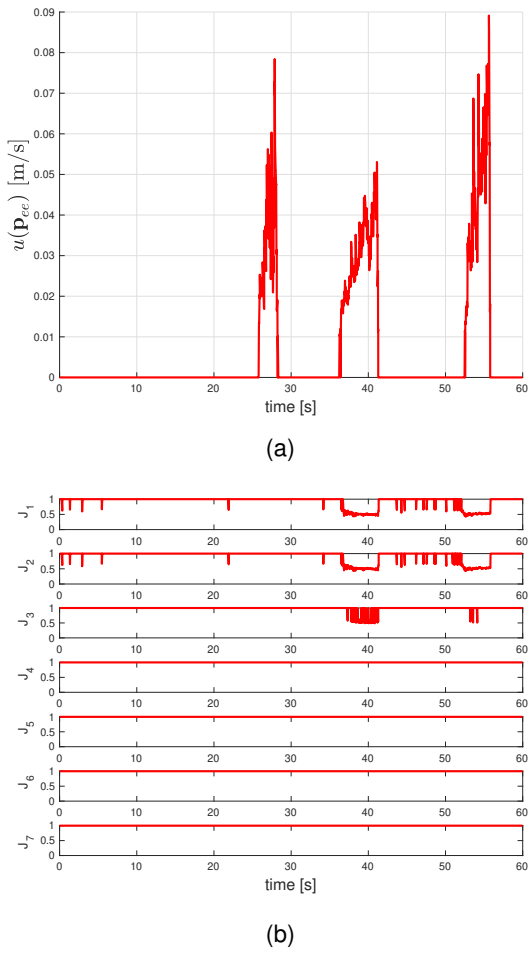


Figure 13: First experiment: (a) EE reaction magnitude. (b) Joint limit scaling factors.

For human head localization, a single tracking sensor of the Oculus system is located in a static place and directed toward the robot workspace. Another simple calibration is done to compute the transformation between the Rift and the world frame. For the mixed reality experience, a ZED-Mini camera is attached to the Oculus Rift as shown in Fig. 3. The system runs on core i9-9.9k CPU @3.10 GHz, with 32 GB RAM and NVIDIA GeForce RTX 2070 GPU.

In the following case studies, we consider one fixed danger zone for each control point, with $\rho = 0.3$ [m], $\gamma = 5$, and $u_{max} = 1.5$ [m/s]. However, further safety zones with variable threshold distances [31] can be integrated easily. The applied motion control parameters are $v_{\max} = 0.4$ [m/s], $k_p = 0.5$, $k_d = 0.05$, $k_v = 200$, and $k_g = 10$.

5.2. Results

The results of two typical experiments are presented in the following. A video of the second experiment is available in the supplementary material.

5.2.1. First experiment

In the first experiment, the *follow* command is activated with a desired pointing angle $\alpha_d = 5°$ during the whole time. The human operator moves initially and the robot follows his head for the first 15 [s] approximately. Every time a dynamic obstacle (in this case, the operator hand) or a static obstacle (the table supporting the robot base) is getting closer, the EE will try to achieve the task while primarily avoiding collision. If this is impossible, the robot will move away from the obstacle, increasing thus the coordination error. When the operator moves the hand back away from the robot, the EE resumes in full the coordinated task. In Fig. 11, since the operator head position is always out of the predefined sphere of task limits, the corresponding projection is used to define the reference values for the coordination task. The difference between the actual EE position and the desired one is due instead to obstacle avoidance.

In Fig. 12, the large initial values of the tracking error are due to the relatively fast motion of the human operator. These errors can be reduced by increasing the motion control gains. However, a trade-off between low Cartesian errors and high EE velocity, should be taken into account. The

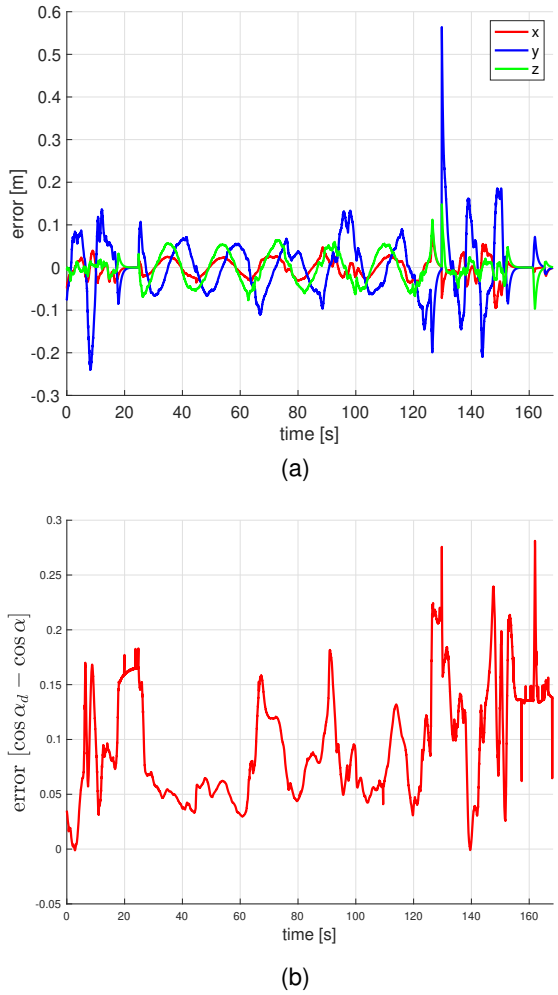


Figure 14: Second experiment: The operator is changing frequently the desired coordinated task. (a) EE position errors. (b) EE orientation error.

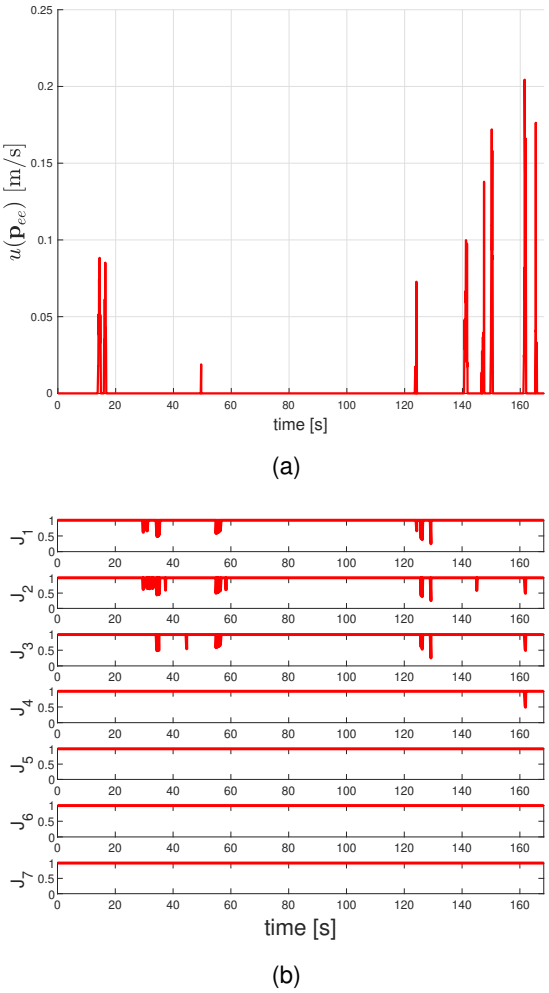


Figure 15: Second experiment: (a) EE reaction magnitude. (b) Joint limit scaling factors.

successive increases in the position error are due instead to obstacle avoidance, as indicated also by the three peaks in the EE reaction magnitude in Fig. 13(a). The scaling factor $(1 - f(D_{min}(\boldsymbol{c}))$ on the velocity limits (23) is shown in Fig. 13(b). Only the limits corresponding to joints that are more influenced by the presence of the obstacle (joints 1 to 3) are reduced. Finally, between $t = 43$ [s] to $t = 51$ [s], an obstacle is getting nearer to the first and second joints resulting in a modification of the actual joint limits, and thus in a robot reconfiguration in its joint space. However, using the SNS algorithm, the robot is able to exploit the available redundancy to follow accurately the desired EE position while avoiding the obstacle. Since the pointing task has a lower priority than the positional task, its corresponding error is relatively higher along the experiment.

5.2.2. Second experiment

In this experiment the operator is interacting in front of the robot for about three minutes, and changing frequently the desired coordinated task by choosing it from the supported virtual augmented list using the Oculus controller in his hand, see Fig. 3. At start, the *follow* command is activated with a desired pointing angle $\alpha_d = 5°$, as in the first experiment. The robot EE moves to comply with the desired coordinated task, minimizing the position and orientation errors. During the experiment, the operator moves his hand toward the robot, acting as a dynamic obstacle.

The positional error in Fig. 14(a) increases every time the robot is not able to achieve the task because of the need of avoiding obstacles. The robot reaction magnitude (13) in Fig. 15(a) indicates in fact how close is the nearest obstacle to the robot EE. If there is no conflict with the higher priority task, the EE keeps the desired orientation (Fig. 14(b)). At about $t = 17$ [s], the *stop* command is activated for 6 [s]. Thus, the robot is controlled to regulate its EE at the last desired task position. At $t = 24$ [s], the operator activates the *circle* command and keeps standing in front of the robot for 35 [s]. Afterward, he moves in the workspace with the desired circle task changing accordingly. Later on, the *stop* and *follow* commands are activated again respectively (see the accompanying video for a complete understanding).

Figures 15(a-b) show clearly how the proposed system is able to avoid obstacles efficiently, without resorting to fast EE motion or large reduction of the joint motion feasible range.

It must be noted that multiple obstacles may affect at the same time the motion of the robot, while acting on different parts of the structure. The robot system may also get stuck in the limit. The proposed control scheme handles these situations as well, with the SNS method smoothly stopping the joint motion.

6. Conclusions

We have proposed a multi-sensor control system that allows realizing contactless coordinated motion tasks between a human and the end-effector of a robot that tracks the human head, while simultaneously avoiding collisions with any static or dynamic obstacle in the workspace. The system is supported with a mixed reality interface to provide the operator with different information about the current robot task and let him/her communicate directly with the robot to change the desired task during the collaboration. This can be seen as another building block of a hierarchical control architecture devoted to safe human-robot interaction. In fact, our system integrates two enabling technologies that characterize digitalization of production (Industry 4.0), namely augmented reality and collaborative human-robot tasks. This solution may be useful at different stages of a manufacturing line, e.g., when the operator should check online a desired level of quality in a service or product.

To this end, the proposed control system deals with four main subproblems. The first one is providing a bidirectional awareness between the robot and the human. For this, we propose to use the Oculus Rift HMD with an attached stereo camera. In this case, the collaborator pose can be localized continuously and accurately to be used for defining the robot desired tasks. At the same time, a mixed reality interface is built to provide the operator with the current robot state and give him the ability to control the robot directly.

The second problem is defining suitable Cartesian tasks for the desired contactless collaboration. For this, we propose different possible coordination tasks which involve three positional variables and only one angular component. The tasks are defined in order to pursue the operator head motion while pointing to it. This can be done either in a regulation mode or by tracking a specified circular path. If the desired task is out of the robot workspace, a sphere of task limit is presented for task adjustment. Furthermore, the proposed relaxed pointing task decreases the overall task dimension which improves the robot dexterity and manipulability to perform the collaboration while avoiding any obstacle.

The last two problems consist of achieving the desired coordination tasks while keeping far from any collision. In this work, we resort to the depth space approach [7] to compute online the distances between the robot and any object in the monitoring area. This information is used to define proper collision avoidance tasks for the robot body and its end-effector. Finally, the SNS algorithm for strict prioritized task control is used at the acceleration level [34]. In this case, the control scheme gives the highest priority to collision avoidance of the whole robot body, whereas the second priority is still preventing end-effector collisions. The desired positional task is in the third rank of the stack of tasks, while the relaxed pointing task has the least priority. Indeed, safe collaborative tasks could be defined and combined differently, and other priority orders could be assigned as well. This is a subject of further study. Note that, the SNS algorithm could have been applied, perhaps in a simpler way, also at the level of velocity commands. However, acceleration commands allow to avoid the joint velocity discontinuities that arise due to the switching of saturated joints.

Various enhancements could be done to boost the proposed control system. First, the mixed reality environment can be supported by various useful augmented objects, e.g. the desired end-effector Cartesian path. Furthermore, it is possible to let the operator design the desired task as a preprocess before starting the collaboration. To improve the collision avoidance performance, a second fixed depth camera can be used to avoid gray zones or sensor occlusion. Also, redundancy in monitoring capabilities can be integrated, e.g., by adding laser scanners to compensate for any unexpected behavior of the depth sensors. The extra degrees of freedom of the robot may be even better exploited by using also inequality constraints to shape the desired operation of the task [38]. Finally, since a stereo camera is used for building the mixed reality interface, it is possible to investigate how to exploit it as well for robot collision avoidance.

Supplementary material

The video associated with this article can be found by attachment.

References

[1] A. De Santis, B. Siciliano, A. De Luca, and A. Bicchi. An atlas of physical human-robot interaction. *Mechanism and Machine Theory*, 43(3):253–270, 2008.

[2] M. Hägele, K. Nilsson, J.N. Pires, and R. Bischoff. Industrial robotics. In B. Siciliano and O. Khatib, editors, *Handbook of Robotics (2nd Ed.)*, pages 1385–1421. Springer, 2016.

[3] Y. Lu. Industry 4.0: A survey on technologies, applications and open research issues. *J. of Industrial Information Integration*, 6:1–10, 2017.

[4] V. Villani, F. Pini, F. Leali, and C. Secchi. Survey on human-robot collaboration in industrial settings: Safety, intuitive interfaces and applications. *Mechatronics*, 55:248–266, 2018.

[5] A. De Luca and F. Flacco. Integrated control for pHRI: Collision avoidance, detection, reaction and collaboration. In *Proc. 4th IEEE Int. Conf. on Biomedical Robotics and Biomechatronics*, pages 288–295, 2012.

[6] S. Haddadin, A. De Luca, and A. Albu-Schäffer. Robot collisions: A survey on detection, isolation, and identification. *IEEE Trans. on Robotics*, 33(2):1292–1312, 2017.

[7] F. Flacco, T. Kröger, A. De Luca, and O. Khatib. A depth space approach for evaluating distance to objects – with application to human-robot collision avoidance. *J. of Intelligent & Robotic Systems*, 80, Suppl. 1:7–22, 2015.

[8] A. Cherubini, R. Passama, A. Crosnier, A. Lasnier, and P. Fraisse.

Collaborative manufacturing with physical human–robot interaction. *Robot. Comput. Integr. Manuf.*, 40:1–13, 2016.

[9] M. Khatib, K. Al Khudir, and A. De Luca. Visual coordination task for human-robot collaboration. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 3762–3768, 2017.

[10] E. Magrini, F. Ferraguti, A.J. Ronga, F. Pini, A. De Luca, and F. Leali. Human-robot coexistence and interaction in open industrial cells. *Robot. Comput. Integr. Manuf.*, 61:101846, 2020.

[11] ISO 10218-1:2011, Robots and Robotic Devices – Safety Requirements for Industrial Robots. Part 1: Robots, since July 1, 2011. URL http://www.iso.org.

[12] ISO 10218-2:2011, Robots and Robotic Devices – Safety Requirements for Industrial Robots. Part 2: Robot Systems and Integration, since July 1, 2011. URL http://www.iso.org.

[13] ISO TS 15066:2016, Robots and Robotic Devices – Collaborative Robots, since February 15, 2016. URL http://www.iso.org.

[14] O. Mazhar, B. Navarro, S. Ramdani, R. Passama, and A. Cherubini. A real-time human-robot interaction framework with robust background invariant hand gesture detection. *Robot. Comput. Integr. Manuf.*, 60:34–48, 2019.

[15] A. Rogowski. Industrially oriented voice control system. *Robot. Comput. Integr. Manuf.*, 28(3):303–315, 2012.

[16] C. Nehaniv, K. Dautenhahn, J. Kubacki, M. Haegele, C. Parlitz, and R. Alami. A methodological approach relating the classification of gesture to identification of human intent in the context of human-robot interaction. In *Proc. IEEE Int. Work. on Robot and Human Interactive Communication*, pages 371–377, 2005.

[17] M. Svenstrup, S. Tranberg, H.J. Andersen, and T. Bak. Pose estimation and adaptive robot behaviour for human-robot interaction. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 3571–3576, 2009.

[18] L. Villani, A. De Santis, V. Lippiello, and B. Siciliano. Human-aware interaction control of robot manipulators based on force and vision. In *Proc. 7th Work. on Robot Motion Control*, pages 209–225. Lecture Notes in Control and Information Sciences, vol. 396, Springer, 2009.

[19] P.J. Besl and N.D. McKay. A method for registration of 3-D shapes. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 14(12): 239–256, 1992.

[20] A.J. Davison. Real-time simultaneous localisation and mapping with a single camera. In *Proc. 9th IEEE Int. Conf. on Computer Vision*, pages 1403–1410, 2003.

[21] H. Kato and M. Billinghurst. Marker tracking and HMD calibration for a video-based augmented reality conferencing system. In *Proc. 2nd IEEE/ACM Int. Work. on Augmented Reality*, pages 85–94, 1999.

[22] G. Klein and D. Murray. Parallel tracking and mapping for small AR workspaces. In *Proc. 6th IEEE/ACM Int. Symp. on Mixed and Augmented Reality*, pages 225–234, 2007.

[23] J. Serafin and G. Grisetti. NICP: Dense normal based point cloud registration. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 742–749, 2015.

[24] S.Y. Gadre, E. Rosen, G. Chien, E. Phillips, S. Tellex, and G. Konidaris. End-user robot programming using mixed reality. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 2707–2713, 2019.

[25] F. Ferraguti, F. Pini, T. Gale, F. Messmer, C. Storchi, F. Leali, and C. Fantuzzi. Augmented reality based approach for on-line quality assessment of polished surfaces. *Robot. Comput. Integr. Manuf.*, 59: 158–167, 2019.

[26] I. Ulrich and J. Borenstein. VFH+: Reliable obstacle avoidance for fast mobile robots. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 1572–1577, 1998.

[27] O. Brock and O. Khatib. High-speed navigation using the global dynamic window approach. In *Proc. IEEE Int. Conf. on Robotics and Automation*, pages 341–346, 1999.

[28] M. Safeea and P. Neto. Minimum distance calculation using laser scanner and imus for safe human-robot interaction. *Robot. Comput. Integr. Manuf.*, 58:33–42, 2019.

[29] M. P. Polverini, A. M. Zanchettin, and P. Rocco. Real-time collision avoidance in human-robot interaction based on kinetostatic safety field. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 4136–4141, 2014.

[30] B. Lacevic, P. Rocco, and A. M. Zanchettin. Safety assessment and control of robotic manipulators using danger field. *IEEE Trans. on Robotics*, 29(5):1257–1270, 2013.

[31] J. Saenz, C. Vogel, F. Penzlin, and N. Elkmann. Safeguarding collaborative mobile manipulators - evaluation of the VALERI workspace monitoring system. *Procedia Manufacturing*, 11:47–54, 2017.

[32] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *Int. J. of Robotics Research*, 5(1):90–99, 1986.

[33] G. Michalos, S. Makris, N. Papakostas, D. Mourtzis, and G. Chryssolouris. Automotive assembly technologies review: Challenges and outlook for a flexible and adaptive approach. *CIRP J. Manuf. Sci. and Technol.*, 2(2):81–91, 2010.

[34] F. Flacco, A. De Luca, and O. Khatib. Control of redundant robots under hard joint constraints: Saturation in the null space. *IEEE Trans. on Robotics*, 31(3):637–654, 2015.

[35] Oculus. URL https://developer.oculus.com.

[36] Unity. URL https://unity.com.

[37] T. Kröger. Fast research interface library. URL https://cs.stanford.edu/people/tkr/fri/html.

[38] M. Khatib, A. Al Khudir, and A. De Luca. Task Priority Matrix at the acceleration level: Collision avoidance under relaxed constraints. *IEEE Robotics and Automation Lett.*, 5(3):4970–4977, 2020.