Safe Human-Robot Collaboration via Collision Checking and Explicit Representation of Danger Zones

SUBMISSION DATE / POSTED DATE

01-03-2022 / 02-03-2022

CITATION

Lacevic, Bakir; Zanchettin, Andrea Maria; Rocco, Paolo (2022): Safe Human-Robot Collaboration via Collision Checking and Explicit Representation of Danger Zones. TechRxiv. Preprint.
https://doi.org/10.36227/techrxiv.19261643.v1

DOI

10.36227/techrxiv.19261643.v1

Safe Human-Robot Collaboration via Collision Checking and Explicit Representation of Danger Zones

Bakir Lacevic[1], Andrea Maria Zanchettin[2] and Paolo Rocco[2]

***Abstract*—This paper deals with safe human-robot collaboration in the context of speed and separation monitoring paradigm. The core of the approach is to continuously track the separation distance between the robot and the human. The robot speed is then adjusted according to the perceived distance so that it will be able to stop before eventually come into contact with the human. We present an approach that aims at maximizing the productivity of the robot, i.e., its speed, while keeping the prescribed safety requirements satisfied. The method is based on explicit representation of *danger zones* – regions around the robot, where safety requirements are violated. The motion is then generated such that the robot moves as fast as possible, while its danger zone still does not collide with human operators. The approach is validated within an experimental study.**

***Note to Practitioners*—This article was motivated by the problem of maximizing productivity of the robotic manipulator while ensuring the safety of human collaborator. The increase in productivity is achieved by a faster traversal of predefined paths without compromising the safety of the human, which is specifically defined by industrial standard. The approach requires limited knowledge on robot's dynamical properties. More precisely, we only need the braking time as a "lumped" representation of robot's inertia. The underlying optimization problem is conveniently resolved by introducing *danger zones* that allow for intuitive visualization and geometrical representation of the regions around the robot that must be avoided. On the other hand, the method assumes the representation of humans via typical geometric primitives, which can be obtained using off-the-shelf depth perception systems. The solution to the problem reduces to a repeated collision checking between danger zones and the human. Such an approach turns out to be suitable for real-time implementation due to availability of fast and efficient collision checking algorithms/libraries.**

***Index Terms*—Safe human-robot collaboration, Speed and separation monitoring, Safety standard, Trajectory scaling, Danger zones, Collision checking**

I. Introduction

In today's industrial contexts, removing physical barriers between human workers and robots has been a regular objective. Contemporary technologies that are in line with Industry 4.0 standards presume adaptable, reliable, and flexible production lines that require people and robots to work together closely. Accordingly, the area of human-robot collaboration (HRC) has grown dramatically from research lab demonstrations to real-world applications that are gaining traction even among small and medium-sized businesses. Emerging control techniques, coupled with contemporary hardware, geared at maintaining the most important feature of HRC – human safety – often make conventional safety standards too rigid and practically outdated [1], [2], [3], [4].

There are several techniques to investigating and ensuring the safety of humans in the presence of robots in the HRC literature. The initial efforts were focused on developing the first HRC-related safety criteria [5]. Later on, a succession of attempts were made to establish more detailed measures and apply them in a feedback control system to improve safety [6], [7], [8], [9], [10], [11].

In a series of publications, Haddadin et al. present an exhaustive investigation of physical human-robot interaction (HRI) including the themes of risk assessment, collision detection and reaction, and robot design [12], [13], [14]. The authors provide a thorough assessment of HRI safety, including several elements of the most important injury mechanisms.

In [15], an integrated framework for safe HRC is presented. It includes both social and physical components for finding an optimal velocity of robot. The authors argue that the inclusion of social component remarkably enhances the capabilities of the framework. The issue of safe collaboration in a wider context of cyber-physical approach is tackled in [16]. The system evaluates safety distance in real-time and uses closed-loop control to invoke actions necessary for preventing collisions. An interesting solution that involves human-in-the-loop concept for manipulation planning in the context of safe collaboration can be found in [17]. The developed system allocates the subtasks to robots and humans taking into account their respective advantages. The approach from [18] offers a learning-based hazard estimator that arbitrates between safety and efficiency. The training stage learns the control policy, while in the testing phase, guiding goal along a given path is selected as a trade-off between safety and task performance. In [19], the authors propose a Petri-nets-based scheduling method for collaborative assembly task. The approach devises optimal plans for assembly activities using online acquired knowledge. It is able to adapt to variations during a manufacturing process thus minimizing idle times for each agent.

In some recent techniques, the safety in collaborative setups is ensured by maintaining a speed-dependent separation gap between the human and the robot throughout operation [20],

[1]B. Lacevic is with University of Sarajevo, Faculty of Electrical Engineering, 71000 Sarajevo, Bosnia and Herzegovina, bakir.lacevic@etf.unsa.ba

[2]A.M. Zanchettin and P. Rocco are with Politecnico di Milano, Dipartimento di Elettronica, Informazione e Bioingegneria, 20133 Milano, Italy andreamaria.zanchettin@polimi.it, paolo.rocco@polimi.it

[21], [22]. This method, known as *speed and separation monitoring* (SSM), is compliant with more recent safety standard [23] and may even be used to make people collaborate with regular industrial robots. In a rather elaborate scheme proposed in [24], a multimodal perception system is used within a fuzzy-based inference system to assess the risk and dynamically compute the minimum protective distance.

Another technique known as *power and force limitation* (PFL) has been created for scenarios that enable physical interactions [30]. This method is based on establishing an upper bound of energy that may be transferred from the robot to human in the event of a collision [31]. Similarly to SSM, the PFL method is based on a safety standard. In addition, it involves the use of specially constructed manipulators [20], [30]. Recently, several attempts have been made to combine SSM and PFL techniques to further push the boundary of enhancing productivity while satisfying safety requirements according to the safety standard [32], [33]. In [27], the authors proposed an elaborate approach to compute minimum-time yet safe trajectories along specified paths in shared workspaces with humans. The safety is iteratively evaluated via a dedicated module that includes interaction/collision model whose role is to capture inertial properties along directions of interest. Furthermore, the replanning and recovery routines engage when the plan respectively violates safety requirements or when the higher speeds may be resumed. For an interested reader, [34] and [35] offer comprehensive views on human-robot-interaction safety mechanisms.

In [36], a method for determining dynamic separation distance and adjusting robot speed is provided. The validation is based on a reduced representation of human geometry that only accounts for the centroid of a single human part. In a series of papers, the authors devised a kinematic control technique that assures safety while maintaining a high degree of robot productivity [26], [25], [37]. The robot's velocity is calculated using an optimization-based real-time method in which safety is considered a hard constraint to meet [25]. Rather than examining merely a discrete set of locations, the technique considers the whole geometry of both the human and the robot. However, in order to make the optimization problem tractable, some sufficient conditions have been assumed, resulting in conservative safety requirements. An improvement of the above approach has been presented in [38], where the same problem has been solved with the exclusion of conservative assumptions. The validation of the method showed substantial speed-ups in robot's motion, without jeopardizing the safety requirements. The price was paid by the increase in computational complexity, which turned out to be a limiting factor with respect to real-time implementation involving more detailed geometric representation of robot and/or human.

In this study, we tackle the same problem as in [26], [25], [38] from the geometrical point of view. The main contribution of the paper is the introduction of danger zones: 3D volumes assigned to robot's links that represent regions of space where safety requirements are violated. The paper presents a simple approach to explicitly compute the danger zones. In particular, the possibility to represent the boundary of a danger zone as a polygonal mesh allows to use fast and efficient collision-checking routines for testing the possible violation of safety requirements, by simply checking whether a human intersects the danger zone. This approach reveals an underlying geometrical intuition of the method that is missing in the original analytical solution from [38]. More importantly, it relaxes the computational burden and therefore facilitates real-time implementation of the algorithm. A conceptually similar approach is proposed in [28], [29], where collisions between human and robot are investigated by online intersection tests for bounding volumes surrounding the robot's links and those evaluated around the human. The protective volumes used for bounding the links are simple capsules. Moreover, the approach requires the dynamic model of the robot.

It is worth pointing out that in this paper, we focus on scenarios where the knowledge about the robot's dynamics is limited. More precisely, the full dynamic model of the manipulator is considered unavailable. The only required parameter is the "braking time" T_b denoting the worst-case time necessary for the robot to transit from full-speed motion to a full stop, once the stop command has been issued. Table I gives a condensed comparison of features from the proposed approach and a selected set of related methods from the literature.

The preliminary results of this study can be found in [38]. This paper is complemented with the following extensions.

1) Analytical definition of danger zones is provided, along with a detailed procedure for their construction. Two options are outlined: the almost exact representation via triangular mesh, and an approximate convex representation. The complexity of the subsequent collision checking is discussed.
2) An improved method for adjusting the braking time T_b is proposed to further increase the performance in terms of shortening the production cycle.
3) The method is generalized to accommodate for more realistic geometric representation of robot's links.
4) The simulation study is broadened to include the comparison to two additional relevant methods.
5) Unlike in [38], the approach is experimentally validated.

The remainder of the paper is organized as follows. Section II provides the background of the proposed method, along with a brief description of the analytical approach from [38]. In Section III, we present the main contribution of the paper – the definition of danger zones and the manner in which they are used to generate robot motion. Section IV provides an extensive simulation study where the proposed approach is compared to the existing methods. In Section V, the results of the experimental validation are presented, while concluding remarks are drawn in Section VI.

II. BACKGROUND

This section provides the background on safety constraints used in the approach. Generally, it is based on SSM paradigm [23], and its derivation builds on work published in [26], [25] and [38]. At first, we consider the relationship between a single link and an obstacle. For simplicity, we assume the obstacle to be a point. Eventually, more general obstacle shapes will be considered: first a triangle in space, and finally an arbitrary

TABLE I: Summary of the comparison with existing methods

Paper	Dyn. model	Human representation	Trajectory generation	Perception	Danger zones
Byner *et al.* [20]	braking time	points of interest	imposing speed limits	laser scanner	no
Zanchettin *et al.* [25], [26]	braking time	triangular mesh	scaling	depth	no
Palleschi *et al.* [27]	reflected mass	points of interest	scaling w. replanning/recovery	depth	no
Costanzo *et al.* [24]	braking time	(thermal) point cloud	scaling	depth+thermal	no
Scalera *et al.* [28], [29]	full model	bounding volumes	scaling w. stopping trajectories	depth	yes (simplified)
This paper	braking time	triangular mesh	scaling	depth	yes

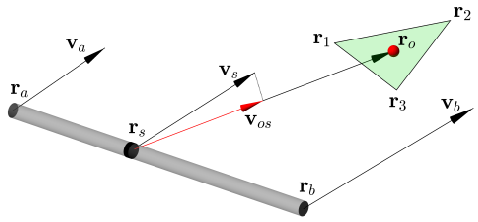

Fig. 1: A thin rigid beam representing one link and an obstacle (point or a triangle). Vectors denoted $\mathbf{r}$ stand for positions of respective points, where vectors denoted $\mathbf{v}$ stand for respective velocities.

polygonal mesh surface. Moreover, the robot's link is assumed to be a thin rigid beam (Fig. 1). Later on, we will extend this approach to include more general geometric shapes.

Let a generic point on the link $\mathbf{r}_s = \mathbf{r}_a + s\left(\mathbf{r}_b - \mathbf{r}_a\right),\ s \in [0,1]$ be moving at the velocity $\mathbf{v}_s = \mathbf{v}_a + s\left(\mathbf{v}_b - \mathbf{v}_a\right),\ s \in [0,1]$, and a point $\mathbf{r}_o$ that belongs to an obstacle (e.g., a triangle with vertices $\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3$). The safety constraint can be expressed as follows [26]:

$$\|\mathbf{r}_{os}\| \geq T_b v_{os}, \tag{1}$$

where $\mathbf{r}_{os} = \mathbf{r}_o - \mathbf{r}_s$ and T_b is the braking time. For now, condition (1) holds for a specific pair of points $\mathbf{r}_s$ and $\mathbf{r}_o$. The optimal trajectory scaling procedure that is based on collision checking with danger zones, which will be explained later, ensures that this condition holds for arbitrary points $\mathbf{r}_s$ and $\mathbf{r}_o$, where $\mathbf{r}_s$ belongs to a link, and $\mathbf{r}_o$ belongs to a triangle. The quantity v_{os} is the magnitude of the vector $\mathbf{v}_{os}$, i.e., a projection of the vector $\mathbf{v}_s$ onto $\mathbf{r}_{os}$. The following holds:

$$v_{os} = \frac{\mathbf{v}_s^T \mathbf{r}_{os}}{\|\mathbf{r}_{os}\|}. \tag{2}$$

Using (2), condition (1) can be expressed as:

$$\|\mathbf{r}_{os}\|^2 - T_b \mathbf{r}_{os}^T \mathbf{v}_s \geq 0, \tag{3}$$

or alternatively in the following form:

$$\dot{\mathbf{q}}^T \mathbf{J}_s^T \mathbf{r}_{os} \leq T_b^{-1} \|\mathbf{r}_{os}\|^2, \tag{4}$$

where $\dot{\mathbf{q}}$ is the joint velocity vector and $\mathbf{J}_s$ the position Jacobian associated to point $\mathbf{r}_s$. For a point-obstacle at $\mathbf{r}_o$, it is necessary and sufficient to ensure that (3) holds $\forall s \in [0,1]$. The distance $\|\mathbf{r}_{os}\|$ is treated as a function of link parameter s and possibly other parameters that define the obstacle surface.

Establishing the value of the braking time T_b is pivotal in this analysis. Primarily, this value depends on the chosen mode of the stopping/braking operation that is specified by a relevant standard. In particular, the IEC Standard [39] distinguishes three different "categories" of stopping/braking operations:

- Stop Category 0: stopping by immediate removal of power to the machine actuators (i.e. an uncontrolled stop);
- Stop Category 1: a controlled stop with power available to the machine actuators to achieve the stop and then removal of power when the stop is achieved;
- Stop Category 2: a controlled stop with power remaining available to the machine actuators.

In robotic applications, Categories 0 and 1 may include the engagement of the electro-mechanical brakes at certain time instants. Some robot manufacturers declare braking times within the available product specifications (see e.g., [40]). Braking times vary with respect to robot's payload and the workspace zone in which it is operating. The most reasonable choice of the braking time T_b for the purposes of our approach is the largest value that corresponds to worst case scenario - robot moving at full speed while in the configuration with the highest inertia. Should this specification not be available, under the assumption that Stop Categories 1 or 2 apply, the braking time can be experimentally determined or estimated using the following formula:

$$T_b = \frac{J_{\max} \dot{q}_{1,\max}}{\tau_{1,\max}}. \tag{5}$$

Here, $J_{\max}$ stands for the robot's maximal moment of inertia, as perceived by the first joint, when the robot is fully outstretched, while $\dot{q}_{1,\max}$ and $\tau_{1,\max}$ denote maximal joint velocity and torque for the first axis. The quantity $J_{\max}$ may be estimated without the full knowledge of the robot's dynamics. Note that such obtained value of T_b can be augmented with the reaction time T_r, i.e., a time required by the system to detect and react to changes in the environment. As it will be discussed later, additional safety margins can be absorbed by the braking time T_b as well. For a more elaborated treatment of braking behavior in the context of robotic manipulators, the reader is referred to [41], [42], [43], [44],

Suppose $\dot{\mathbf{q}}_n(t)$ is a nominal joint velocity vector that corresponds to task execution in absence of human operators. In the context of consistent following of a geometrical path (that corresponds to the trajectory $\dot{\mathbf{q}}_n(t)$), we aim at maximizing the scaling coefficient δ for computing the joint velocity vector $\dot{\mathbf{q}}$ in the form $\dot{\mathbf{q}} = \delta \dot{\mathbf{q}}_n$. In other words, when humans/obstacles are present in the robot's workspace, the goal is to slow down the nominal trajectory $\dot{\mathbf{q}}_n(t)$ as little as possible, while at the same time satisfying the safety constraint (4).

To make the problem computationally more convenient, Zanchettin et al. [26] solved a more conservative version of the problem by forcing a sufficient condition:

$$\min_s \|\mathbf{r}_{os}\|^2 - T_b \mathbf{r}_{os}^T \mathbf{v}_s \geq 0, \tag{6}$$

or:

$$\beta s + \gamma + \min_s \|\mathbf{r}_{os}\|^2 \geq 0, \tag{7}$$

with $\beta = T_b \mathbf{r}_{ba}^T \mathbf{v}_a - T_b \mathbf{r}_{oa}^T \mathbf{v}_{ba}$, $\gamma = -T_b \mathbf{r}_{oa}^T \mathbf{v}_a$, $\mathbf{r}_{ba} = \mathbf{r}_b - \mathbf{r}_a$, $\mathbf{r}_{oa} = \mathbf{r}_o - \mathbf{r}_a$, and $\mathbf{v}_{ba} = \mathbf{v}_b - \mathbf{v}_a$.

The term $\min_s \|\mathbf{r}_{os}\|$ is a constant (distance from the point $\mathbf{r}_o$ to a link with endpoints $\mathbf{r}_a$ and $\mathbf{r}_b$). A quadratic condition (3) is therefore reduced to a linear one, which renders the problem significantly easier to solve. Taking into consideration n robot links and using the fact that β and γ are linear with respect to joint velocity vector $\dot{\mathbf{q}}$, the safety criterion can be expressed as [26]:

$$T_{bi} \mathbf{E}_i \dot{\mathbf{q}} \leq \mathbf{f}_i, \ \forall i = 1, 2, \ldots, n, \tag{8}$$

where T_{bi} is the maximum braking time up to joint i, $\mathbf{E}_i = [e_{i1} \ e_{i2}]^T$ with $e_{i1} = \mathbf{r}_{oa}^T \mathbf{J}_a$, $e_{i2} = \mathbf{r}_{oa}^T \mathbf{J}_b - \mathbf{r}_{ba}^T \mathbf{J}_a$, and $\mathbf{f}_i = \min_s \|\mathbf{r}_{os}\|^2 [1 \ 1]^T$.

The system of inequalities (8) represents constraints within a linear programming optimization problem that aims at maximizing the scaling coefficient δ for computing the joint velocity vector in the form $\dot{\mathbf{q}} = \delta \dot{\mathbf{q}}_n$ (see [26] for details). Such "linearized" version of the problem scales well with respect to more general representation of obstacles (e.g., triangles or convex polyhedra) [37], [25].

In [38], the problem is analytically resolved with the original quadratic constraints (3) or (4), without imposing sufficient conditions to make the optimization problem linear. The solution has been verified in a simulation study that showed significant performance improvements in terms of decreasing the production cycle without jeopardizing the safety constraints. For experimental validation however, it is required to ensure the real-time executability. While this is actually feasible for relatively simple geometric representation of human surface (i.e., a small number of points/triangles), our goal is to extend the scope of the method toward more complex scenarios by preventing safety constraint checks from being a computation bottleneck. In this regard, we propose a more intuitive, geometric solution to the same problem.

III. Proposed Approach - Geometric Solution

In this section, we describe an alternative approach to solving the same SSM problem. The main idea behind the approach is the following one. Given the state of the link (position and velocity), it is possible to conveniently represent the "danger zone", or the "region to avoid", i.e., the set of all points where the SSM criterion is violated. In particular, we are interested in computing the boundary of this region that is given by a closed surface.

Once this surface is computed, testing whether safety criterion is satisfied reduces to simple collision checks with environmental obstacles. To facilitate collision checks, tight approximations of the danger zones based on simple geometric primitives may be used. Otherwise, more general representation by triangular meshes is an option. In the latter case, a wide variety of libraries are available, e.g., PQP [45] or FCL [46], with extremely fast and efficient collision-checking routines. Assuming convex representation of danger zones and human body parts, specific algorithms such as Gilbert-Johnson-Keerthi (GJK) [47] can be used.

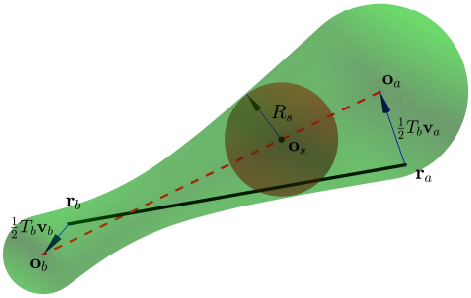

Fig. 2: The danger zone $\mathcal{DZ}(\mathbf{r}_a, \mathbf{r}_b, \mathbf{v}_a, \mathbf{v}_b)$ as a sphere-swept line segment

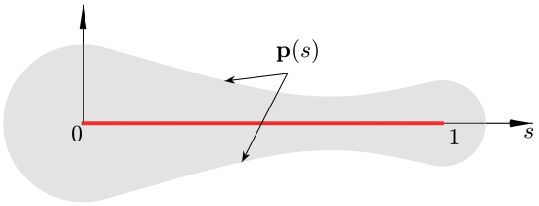

Fig. 3: A projection of the danger zone into a 2D local frame

A. Danger Zone Formulation

The computation of the danger zone boundary is based on one simple fact. Observe the point robot at a position $\mathbf{r}_s$ with velocity $\mathbf{v}_s$. All the points $\mathbf{r}$ that do not satisfy the condition (3), meet the following inequality:

$$\|\mathbf{r} - \mathbf{r}_s\|^2 - T_b (\mathbf{r} - \mathbf{r}_s)^T \mathbf{v}_s < 0. \tag{9}$$

This inequality can be rewritten as:

$$\left\|\mathbf{r} - \mathbf{r}_s - \tfrac{1}{2} T_b \mathbf{v}_s\right\|^2 < \tfrac{1}{4} T_b^2 \|\mathbf{v}_s\|^2. \tag{10}$$

The region corresponding to the above inequality is a ball $\mathcal{B}(\mathbf{o}_s, R_s)$ centered at $\mathbf{o}_s = \mathbf{r}_s + \frac{1}{2} T_b \mathbf{v}_s$, with a radius $R_s = \frac{1}{2} T_b \|\mathbf{v}_s\|$. For a link with endpoints $\mathbf{r}_a$ and $\mathbf{r}_b$, with respective velocities $\mathbf{v}_a$ and $\mathbf{v}_b$, the danger zone $\mathcal{DZ}$ becomes the union of balls $\mathcal{B}(\mathbf{o}_s, R_s)$, with s sweeping the interval $[0, 1]$:

$$\mathcal{DZ}(\mathbf{r}_a, \mathbf{r}_b, \mathbf{v}_a, \mathbf{v}_b) = \bigcup_{s \in [0,1]} \mathcal{B}(\mathbf{o}_s, R_s). \tag{11}$$

In other words, the danger zone $\mathcal{DZ}(\mathbf{r}_a, \mathbf{r}_b, \mathbf{v}_a, \mathbf{v}_b)$ is a sphere-swept segment with endpoints $\mathbf{o}_a = \mathbf{r}_a + \frac{1}{2} T_b \mathbf{v}_a$ and $\mathbf{o}_b = \mathbf{r}_b + \frac{1}{2} T_b \mathbf{v}_b$, where the radius of the sphere changes according to the following formula:

$$R_s = \frac{T_b}{2} \|\mathbf{v}_s\| = \frac{T_b}{2} \sqrt{\|\mathbf{v}_a\|^2 + 2 \mathbf{v}_a^T \mathbf{v}_{ba} s + \|\mathbf{v}_{ba}\|^2 s^2}. \tag{12}$$

Fig. 2 depicts an example of such region. Note that, unlike in [28], where the bounding volume for a link is a simple capsule, the danger zone $\mathcal{DZ}(\mathbf{r}_a, \mathbf{r}_b, \mathbf{v}_a, \mathbf{v}_b)$ assumes a shape that captures the velocity profile along the link.

Due to symmetry, the danger zone results from the rotation (about the line segment $\mathbf{o}_a \mathbf{o}_b$) of a corresponding plane figure that may be represented in a local frame (see Fig. 3). The

boundary $\mathbf{p}(s) \in \mathbb{R}^2$ of the given plane figure corresponds to the envelope of circle-swept segment and is given by:

$$\mathbf{p}(s) = \begin{bmatrix} sL - \frac{1}{L} R_s \frac{\partial R_s}{\partial s} \\ \pm R_s \sqrt{1 - \frac{1}{L^2}\left(\frac{\partial R_s}{\partial s}\right)^2} \end{bmatrix}, \tag{13}$$

where $L = \|\mathbf{o}_a - \mathbf{o}_b\|$. It is worth pointing out that the above parameterization of the envelope is valid if $\left|\frac{\partial R_s}{\partial s}\right| < L$, i.e., the radius of the circle should not change faster than the "traversal speed" of the segment. This can however, be easily proven by knowing the explicit formulas for R_s and L.

B. Danger Zone Construction

For the practical implementation of danger zones, a proper representation of the surfaces that correspond to their boundaries is necessary. A convenient approach is to compute a number of samples on the boundary $\partial\mathcal{DZ}$ of the danger zone $\mathcal{DZ}$. Such set of samples can then be used within any surface reconstruction algorithm (e.g., *ball-pivoting* [48]) to generate an explicit representation of $\partial\mathcal{DZ}$ via triangular mesh. This mesh can be further used for interference computation (collision detection, distance query, penetration estimation) with environment obstacles, which are represented in a same way or by means of other geometric primitives. Fig. 6 shows examples of danger zones represented by triangular meshes.

Clearly, there are many ways to generate the sample points on $\partial\mathcal{DZ}$. In the sequel, we describe two ways to do this. The first approach attempts at capturing the boundary $\partial\mathcal{DZ}$ almost exactly, while the second approach is based on the approximation of danger zone by its convex hull, which turns out not to be too conservative.

The idea behind the first approach is depicted in Fig. 4. The first step would be to sample the contour of the danger zone in the 2D local frame from Fig. 3. The frame is spanned by the vectors $\mathbf{n}$ and $\mathbf{s}$. For practical purposes, which will be justified later, the sampling of circular caps (each by N_1 points) and the mantle (by N_2 points) is performed separately. Once the contour is sampled, it is rotated with respect to the axis of the danger zone (i.e., the unit carrier vector $\mathbf{n}$) $N_3 - 1$ times with equal increments within the interval $[0, 2\pi]$. Thus, the sampling procedure is complete by which we end up with a total of $N_3\,(2N_1 + N_2)$ sample points.

The second approach is considerably simpler by taking into consideration only the spherical caps, without sampling the mantle. Convex hull of the spherical caps is actually the hull of danger zone itself. This stems from the fact that the mantle of danger zone is provably not strictly convex. Fig. 5 shows the convex approximation of the danger zone obtained using the second approach. Apart from slightly simpler construction, the main advantage of the second approach is the facilitation of subsequent collision checking, which in general performs better under the assumption of convex geometries. As it will be later shown via numerical investigation, the error due to described approximation is on average considerably small.

A unified pseudocode for both approaches is given in Algorithm 1. A Boolean input CONVEX determines whether first or second approach is used for constructing $\partial\mathcal{DZ}$. The first approach requires the surface to be reconstructed based on samples (e.g., [48]). This is performed in Line 24 of the Algorithm 1 by calling the function `ReconstructSurface`. This function returns TriangleList – a list of IDs of samples (vertices) that correspond to individual triangles (faces) in the mesh. It is worth pointing out that this function can be called only once, even offline. The resulting TriangleList can be reused for any subsequent mesh reconstruction, provided that the sampling scheme remains the same (i.e., keeping N_1, N_2, N_3 the same, as well as the ordering of the samples) for constructing the arbitrary $\partial\mathcal{DZ}$ (i.e., for arbitrary position and velocity profile of the link). In topological sense, the graph that corresponds to the mesh remains isomorphic regardless of the danger zone shape (see Fig. 6).

Algorithm 1 Construction of $\partial\mathcal{DZ}$

Inputs: $\mathbf{q}, \dot{\mathbf{q}} \in \mathbb{R}^6, N_1, N_2, N_3 \in \mathbb{N}$, Boolean CONVEX
Output: Boundary $\partial\mathcal{DZ}$ of the danger zone

1: $\mathbf{r}_a, \mathbf{r}_b \leftarrow \texttt{FK}(\mathbf{q}, a, b)$; ▷ *Forward kinematics*
2: $\mathbf{v}_a \leftarrow \mathbf{J}_a(\mathbf{q})\dot{\mathbf{q}}$;
3: $\mathbf{v}_b \leftarrow \mathbf{J}_b(\mathbf{q})\dot{\mathbf{q}}$;
4: $\mathbf{o}_a \leftarrow \mathbf{r}_a + \frac{1}{2}T_b\mathbf{v}_a$;
5: $\mathbf{o}_b \leftarrow \mathbf{r}_b + \frac{1}{2}T_b\mathbf{v}_b$;
6: $\mathbf{n} \leftarrow \mathbf{o}_b - \mathbf{o}_a$
7: $\mathbf{n} \leftarrow \frac{\mathbf{n}}{\|\mathbf{n}\|}$;
8: $\mathbf{s} \leftarrow \left(\mathbf{I}_{3\times3} - \mathbf{n}\mathbf{n}^T\right)[1\ 0\ 0]^T$;
9: **if** $\mathbf{s} = \mathbf{0}$ **then** ▷ *Make sure* $(\mathbf{s} \perp \mathbf{n}) \wedge (\mathbf{s} \neq \mathbf{0})$
10: $\quad\mathbf{s} \leftarrow \left(\mathbf{I}_{3\times3} - \mathbf{n}\mathbf{n}^T\right)[0\ 1\ 0]^T$;
11: $\mathbf{s} \leftarrow \frac{\mathbf{s}}{\|\mathbf{s}\|}$; ▷ $\mathbf{n}$ and $\mathbf{s}$ *form the orthonormal basis of the local frame*
12: $\mathrm{Cap\_a} \leftarrow \texttt{SampleHalfCircle}\left(\mathbf{o}_a, \frac{1}{2}T_b\mathbf{v}_a, -\mathbf{n}, \mathbf{s}, N_1\right)$;
13: $\mathrm{Cap\_b} \leftarrow \texttt{SampleHalfCircle}\left(\mathbf{o}_b, \frac{1}{2}T_b\mathbf{v}_b, \mathbf{n}, \mathbf{s}, N_1\right)$; ▷ *Uniform sampling of half-circles in the local frame with* N_1 *points*
14: **if** ¬CONVEX **then**
15: $\quad\mathrm{Mantle} \leftarrow \texttt{SampleMantle}\,(\mathbf{o}_a, \mathbf{o}_a, R_s(s), \mathbf{n}, \mathbf{s}, N_2)$; ▷ *Uniform sampling of the mantle in the local frame with* N_2 *points*
16: **for each** $k = 1 : (N_3 - 1)$ **do** ▷ *Rotating the samples from local frame* $N_3 - 1$ *times*
17: $\quad\theta \leftarrow \frac{k}{N_3} \cdot 2\pi$;
18: $\quad\mathrm{Cap\_a} \leftarrow \mathrm{Cap\_a} \cup \texttt{Rotate}\,(\mathrm{Cap\_a}, \mathbf{o}_a, \mathbf{n}, \theta)$;
19: $\quad\mathrm{Cap\_b} \leftarrow \mathrm{Cap\_b} \cup \texttt{Rotate}\,(\mathrm{Cap\_b}, \mathbf{o}_b, \mathbf{n}, \theta)$;
20: $\quad$**if** ¬CONVEX **then**
21: $\quad\quad\mathrm{Mantle} \leftarrow \mathrm{Mantle} \cup \texttt{Rotate}\,(\mathrm{Mantle}, \mathbf{o}_a, \mathbf{n}, \theta)$;
22: **if** ¬CONVEX **then**
23: $\quad\mathrm{SamplePoints} \leftarrow \mathrm{Cap\_a} \cup \mathrm{Mantle} \cup \mathrm{Cap\_b}$;
24: $\quad\mathrm{TriangleList} \leftarrow \texttt{ReconstructSurface}\,(\mathrm{SamplePoints})$;
25: $\quad\partial\mathcal{DZ} \leftarrow \texttt{TriangleMesh}\,(\mathrm{SamplePoints}, \mathrm{TriangleList})$;
26: **else**
27: $\quad\mathrm{SamplePoints} \leftarrow \mathrm{Cap\_a} \cup \mathrm{Cap\_b}$;
28: $\quad\mathcal{DZ} \leftarrow \texttt{ConvHull}\,(\mathrm{SamplePoints})$;
29: **return** $\partial\mathcal{DZ}$;

C. Computational cost – $\mathcal{DZ}$ construction and collision checking

The computational cost inherent to the notion of danger zones can be decomposed into two aspects. The first one is the cost of the danger zone construction itself. The second one is related to subsequent collision detection between danger zones and humans/obstacles. From the outline of Algorithm 1, it is clear that the construction phase requires only rather straightforward computations involving forward kinematics, Jacobians corresponding to link endpoints, generating a limited

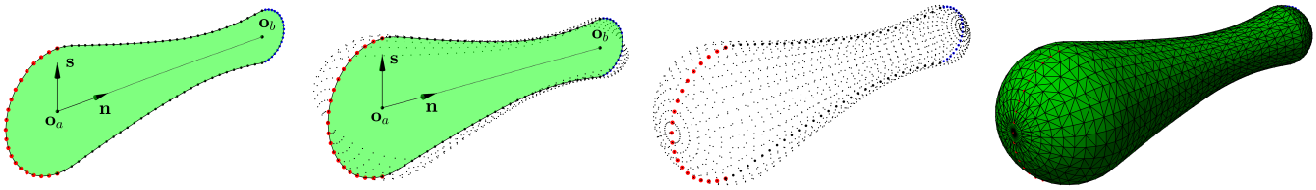

Fig. 4: Sampling scheme for representing the boundary $\partial\mathcal{DZ}$. Far left – sampling of the contour in the local frame. Left – rotating the sampled contour w.r.t. axis of the danger zone. Right – rotation is finished to complete the sampling procedure. Far right – triangular mesh is reconstructed based on samples.

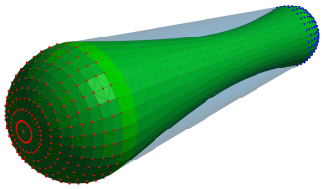

Fig. 5: Convex approximation of the danger zone

number of samples and performing pure rotations. In case of exact danger zone representation, one possibly expensive operation, i.e., surface reconstruction, can be pre-executed offline to provide the list of triangles that is invariant for a fixed sampling scheme and can be reused later. Therefore, the construction phase does not represent a potential bottleneck for real-time applications. On the other hand, the collision checking is expected to consume a substantial fraction of runtime. Here, one should separate two different modes of operation. A computationally more convenient one assumes the convex representation of danger zones. Moreover, it is expected that the human can be represented as a union of convex shapes (see e.g., [37]). In such setting, a natural approach to collision detection is via original GJK algorithm [47], or some of its enhanced incarnations [49], [50]. The convenience of GJK algorithm stems from its linear complexity $\mathcal{O}(N)$, where N is the total number of vertices of the two bodies that are being checked for collision [47], [49]. If the convex representation of danger zones is not allowed, one has to seek for an alternative algorithm that allows for non-convex models.

Most of contemporary algorithms for collision detection between general non-convex geometries are heavily based on *bounding volume hierarchies* (BVHs). BVH models aim at decomposing a body into a tree-like structure. Each node in the tree corresponds to a bounding volume that covers a certain subset of the original body. The root represents a volume that bounds the complete body, whereas the leaf nodes cover, or represent the basic primitives (e.g., triangles). Among others, the typical volumes used in BVHs are axis-aligned bounding boxes (AABBs) [51], oriented bounding boxes (OBBs) [52], spheres [53], sphere-swept volumes (SSVs) [45], and discrete orientation polytopes (k-DOPs) [54]. Whatever BVH model is picked for representing the bodies that need to be checked for collision, the procedure takes two phases. The first one is the construction phase, where BVHs are built, and the second one is the query phase where constructed BVHs are checked for collision. Due to tree-like structure of BVHs, the complexity

TABLE II: Runtimes (in μs) for collision/distance queries

	AABB	OBB	sphere-tree	k-dop	SSV	GJK (conv)
mean	181	37	36	134	197	28
std	88	29	14	46	75	11

of $\mathcal{O}(N \log N)$ can be achieved for construction phase [55], [52], where N is the number of triangles. The query phase is typically of similar expected complexity. However, in the worst case, it can be $\mathcal{O}(N^2)$ when each pair of primitives need to be checked for collision [53].

As pointed out in [56], the true cost of collision query is to a large extent scenario-specific. The input size, shape and relative proximity play a crucial role in the expected runtime. For this purpose, we have conducted a numerical study in order to select the most convenient approach to checking collision between danger zones and generic obstacles. A number (1000) of random scenarios is generated. Each scenario assumes a random danger zone (with random position/orientation) and a random convex obstacle in the relative vicinity of the danger zone. The proximity between objects is chosen such that the collision occurs in cca 50% of the time. The collision test is performed using the following different representations: convex approximation, AABB-tree, OBB-tree, sphere-tree, SSVs, and k-DOP tree. It is assumed that both objects are represented by the same type of BVH. For collision checking, FCL library [46] is used. In case of convex approximation of the danger zone, GJK algorithm available in FCL is used. A fixed sampling scheme for constructing the danger zone is assumed using 435 vertices. A randomly generated convex obstacle has cca 100 vertices. Table II shows average runtimes for collision checks when different types of BVHs are used. The results indicate that OBB- and sphere-tree-based representations imply the fastest collision detection on average. On the other hand, if the convex approximation of danger zone is used, combined with GJK-based distance computation, we obtain the fastest query.

D. Adaptive braking time

The analysis so far, along with the methods from [26], [25], [38] considered a constant braking time T_b. For obvious reasons, the specific value of T_b is estimated assuming worst case scenarios involving maximum robot speeds under configurations with the largest inertia. However, such an assumption may be too conservative in general, since it is not likely that the nominal trajectory of the robot, designed for typical tasks, is such that its state is near its domain's boundary all the time.

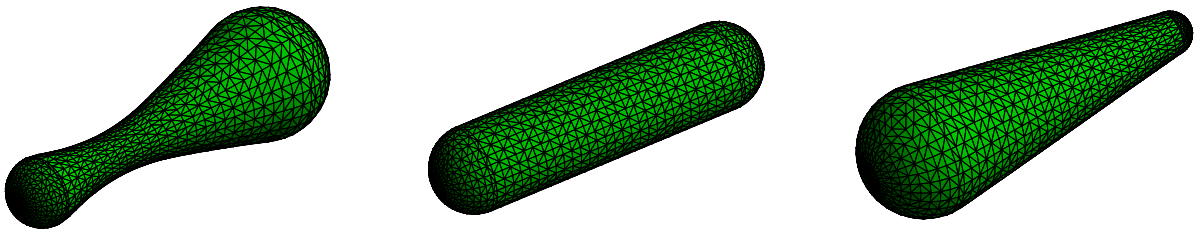

Fig. 6: Triangular mesh representation: danger zones for different velocity profiles of the link, obtained using the same sampling scheme. Graphs that correspond to the association of edges and vertices are isomorphic.

Consequently, the braking time T_b, now observed as a function of the robot's state, is in general smaller than the previously used worst case estimate. The problem of estimating the stopping time based on the current state has been addressed in the literature. For instance, in [28], [29], the stopping time is computed within an optimization procedure that accounts for the robot's current state, its dynamical model and the actuation torque bounds. A kinematic approach from [31] estimates the stopping time based on the current velocity of the robot and the specified joint acceleration and jerk bounds.

In this work, any reasonable reduction of the braking time (compared to the fixed, worst-case estimate) implies the reduction of danger-zone volumes, which further makes room for the increase in robot's speed and hence its productivity. In this regard, we inclined to the kinematic approach for assessing the braking time using the following intuitive formula:

$$T_b\left(\dot{\mathbf{q}}\right) = \max_i \frac{|\dot{q}_i|}{\ddot{q}_{i,\max}}, \tag{14}$$

where $\dot{q}_i$ and $\ddot{q}_{i,\max}$ are respectively actual joint velocity and joint acceleration bound, of i-th joint. Such formulation clearly requires the knowledge of joint acceleration bounds. Ideally, these are provided in the specifications by manufacturer. Otherwise, they need to be estimated, or determined experimentally. If a limited knowledge on robot's dynamics is available, the acceleration bounds can be induced from the torque bounds using the method from [57]. Alternatively, any informed, even conservative, estimate of acceleration limits may be beneficial in this regard.

E. Trajectory Scaling

The volume of danger zone $\mathcal{DZ}$ clearly depends on the robot state, i.e., on configuration $\mathbf{q}$ and the joint velocity $\dot{\mathbf{q}}$. The idea is to first compute the danger zones for all the links that correspond to nominal velocity $\dot{\mathbf{q}}_n$. Once the boundaries of danger zones have been computed, we check if any of them intersect with the human/obstacle. If there is no intersection, the robot continues to move according to the nominal trajectory. Otherwise, to prevent the collision between the human and danger zones, we have to tune the robot velocity $\dot{\mathbf{q}} = \delta\dot{\mathbf{q}}_n$, by computing a proper value of $\delta \in [0, 1]$. The optimal value of δ implies that the geometric shape which represents the human (possibly including additional safety margin) touches at least one danger zone, but does not penetrate any of them. We seek for such δ by using bisection approach, which is given via Algorithm 2. For the sake of simplicity, the danger zone $\mathcal{DZ}\left(\mathbf{q}, \dot{\mathbf{q}}\right)$ denoted in the pseudocode, stands for the union of danger zones with respect to all the robot's links.

Algorithm 2 δ-SEARCH (bisection)

$\delta_{\min} \leftarrow 0$;
$\delta_{\max} \leftarrow 1$;
$\delta_c \leftarrow 1$;
$\mathcal{DZ} \leftarrow$ `ConstructDZ` $(\mathbf{q}, \delta_c\dot{\mathbf{q}}_n, N_1, N_2, N_3, \mathrm{CONVEX})$
if `Collision` $(\mathrm{Obstacles}, \mathcal{DZ})$ **then**
  repeat
    $\delta_c \leftarrow \frac{1}{2}\left(\delta_{\min} + \delta_{\max}\right)$;
    $\mathcal{DZ} \leftarrow$ `ConstructDZ` $(\mathbf{q}, \delta_c\dot{\mathbf{q}}_n, N_1, N_2, N_3, \mathrm{CONVEX})$
    if `Collision` $(\mathrm{Obstacles}, \mathcal{DZ})$ **then**
      $\delta_{max} \leftarrow \delta_c$;
    else
      $\delta_{min} \leftarrow \delta_c$;
  until $\delta_{max} - \delta_{min} <$ Threshold
  $\delta \leftarrow \delta_c$;
return δ

Theoretically, it may happen that an abrupt decrease in the scaling factor δ may cause that the consequently desired deceleration cannot be achieved by joint actuators. More precisely, there may be a time instant when the robot will not slow down at the prescribed rate. However, we argue that this potential anomaly has no effect on safety requirements under the initially posed constraint that the robot has to maintain the ability to stop (in time T_b) before colliding with the obstacle.

In the worst case scenario, when the scaling factor δ suddenly changes from 1 to 0 (the robot has to promptly go from the nominal to a zero speed, i.e., a stop command is issued), the robot will clearly not be able to instantly decelerate. However, we still have the guarantee that the robot would stop within the next T_b seconds before a collision possibly happens. This is implied by the definition of stopping time, which is by default the result of the worst-case analysis where the motions of links with large inertias at full speed are considered. For an alternative setting, where the adaptive T_b is considered, we may apply the similar remarks. In any case, we emphasize that, within the experimental validation, we have not observed any violations of safety constraints related to the above-mentioned consideration.

F. Accommodating for Thick Robot Links

So far, the robot's links have been observed as thin line segments connecting adjacent joints. Clearly, such assumption is too simplistic for many HRC scenarios. To this end, we generalize the proposed approach to account for links with

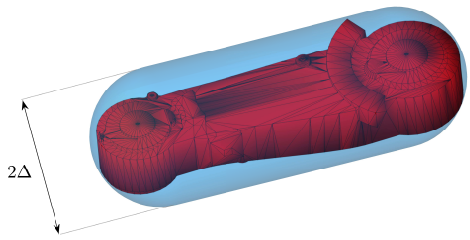

Fig. 7: CAD model of a generic link contained in a capsule

non-zero volumes. Accordingly, we emulate the approach from [37], by introducing the "clearance" parameter Δ, which acts as a buffer zone surrounding an idealized line-segment representation of the link. Thus, the link becomes a capsule, i.e., a Minkowski sum of a line segment and a sphere with the radius Δ. In other words, the capsule shape is used to approximate (as a superset bounding object) an arbitrary link geometry. Thus, a possibly complicated link geometry does not have to be explicitly represented via triangular mesh. Moreover, it is desirable that each capsule has the minimum possible radius, while still enclosing its corresponding link. The radii of such capsules can easily be acquired using typically available CAD models of the robot's links. Note that these capsules are never actually explicitly computed, nor have to be tessellated into triangles. Only their respective radii are necessary (see Fig. 7). The corresponding safety constraint is then given by:

$$v_{os} \leq T_b^{-1} \max\left\{0, \|\mathbf{r}_{os}\| - \Delta\right\}. \tag{15}$$

Despite the best of our efforts, the above constraint turned out to be difficult to use for the development of an explicit geometric representation of the danger zone. However, in a similar fashion as in [37], a sufficient condition can be imposed in the following form:

$$v_{os} \leq \beta T_b^{-1} \|\mathbf{r}_{os}\| \leq T_b^{-1} \max\left\{0, \|\mathbf{r}_{os}\| - \Delta\right\}, \tag{16}$$

where β is a suitably chosen constant. Picking a larger value for β implies that the above constraint is less conservative. It is easy to verify that the following value of β satisfies the rightmost inequality in (16):

$$\beta^* = \max\left\{0, 1 - \frac{\Delta}{d_{\min}}\right\}, \tag{17}$$

where

$$d_{\min} = \min_{s\in[0,1]} \|\mathbf{r}_{os}\|$$

is the minimum distance between the line segment representation of the link and the human/obstacle. Plugging $\beta = \beta^*$ in the leftmost inequality of (16) the safety constraint becomes:

$$v_{os} \leq \beta^* T_b^{-1} \|\mathbf{r}_{os}\| \equiv \left(T_b^*\right)^{-1} \|\mathbf{r}_{os}\|. \tag{18}$$

Clearly, if (18) is satisfied, then (15) holds as well. On the other hand, the constraint (18) preserves the structure of the original condition (1). Thus, exactly the same procedure for computing the geometrical representation of the danger zone can be conducted. The difference is that the clearance parameter is "absorbed" by the new, apparent braking time

$$T_b^* = \frac{T_b}{\beta^*}.$$

It is worth pointing out that the clearance parameter can simultaneously take into account both link thickness and the additional distance margin around the human/obstacle. Such margin can be interpreted as a threshold distance between the danger zone and a human, which must not be violated.

G. Human motion

The formulation of the proposed method does not explicitly include the motion of human operator. As such, it may lead to an impression that the approach is not capable of dealing with the moving obstacles. However, we underline that, by using some of the existing methods for incorporating the human velocity (perceived or predicted) for the generation of properly augmented human geometrical models, our method remains 100% applicable as formulated. In any case, we emphasize that, though it represents a crucial part in establishing safe human-robot collaboration, the consideration of human motion is not the focus of the paper.

In this work, we opted for improving the productivity by assuming less conservative safety requirements from the robot's side. The proposed algorithm is completely agnostic for what concerns how a human obstacle is represented and whether this representation accounts for the perceived/predicted motion of the human or not. The simplest approach is to compute the geometrical representation of the currently sensed human position, without considering their velocity. On the other hand, there are powerful methods to generate geometrical models of humans that account for the perceived or predicted motion (e.g., [37]). These models are usually augmented representations of those that are obtained simply by considering the current human's pose. In any case, the resulting model is typically represented via triangular mesh, geometric primitives, or point cloud(s), all being legitimate inputs to collision checking routines used by our approach.

In the process of experimental validation, we have assumed that the motion of human obstacle is limited by the maximal walking speed of $v_H = 1.6\frac{\mathrm{m}}{\mathrm{s}}$, as suggested in [23] when walking speed is not measured. This velocity bound translates into an additional safety margin within the clearance parameter in order to account for the human motion. This clearly may not be sufficient to cover pathological scenarios with rapid hand/arm motion where certain parts can exceed the assumed velocity bound. In this case, our method cannot guarantee the integrity of safety constraints, but with such assumptions, hardly can any. There is an obvious trade-off between the intention to increase productivity and preserve safety under a wide variety of working conditions. Theoretically, one might assume quite liberal bounds on the human velocity to account for potentially extremely fast movements. This would call for large clearance margins that would substantially slow the robot down.

IV. Simulation Study

This Section provides a simulation study, which aims at validating the proposed approach and comparing it to existing relevant methods. We compare the approach to the previous work of authors [26], [25] together with the recent works from other research groups [24], [28], [29]. The main objective of the used simulation framework is to enable fully reproducible settings for the fair comparison of considered methods. The study consists of three parts. The first part addresses path following in the environment filled with random obstacles. The second part examines the volumes of danger zones (regions to avoid, or dynamic safety zones) associated with relevant methods. The third part compares respective methods in a simulated human-robot collaboration scenario. The complete study is conducted in MATLAB and Robotics Toolbox [58].

A. Part I: static random obstacles

The first part of the study considers a wire model (a simplified representation via thin segments connecting local Denavit-Hartenberg frames) of Comau SmartSix 6 DOF industrial manipulator, which is exposed to 1000 different simulation runs with randomly generated circumstances. The following are the circumstances that are artificially created for each run:

1) There are N static point-obstacles in the environment, where N is a uniform random integer between 5 and 200. The obstacles are placed with respect to uniform random distribution within the axis-aligned box that defines the robot's workspace.
2) Initial and final configurations $\mathbf{q}_s$ and $\mathbf{q}_g$ are chosen at random, with the condition that $\rho(\mathbf{q}_s, \mathbf{q}_g) > \rho_0$, ρ being a C-space metric from [59], and ρ_0 a proper threshold.
3) The nominal path traversal time T_{sim} is picked at random (uniformly) between 1s and 5s.
4) The nominal trajectory assumes a quintic polynomial profile with zero boundary velocities and accelerations. The resulting path has to be collision-free.
5) The braking time is $T_b = 0.2823$s.

We are specifically interested in the following. Given a viable nominal trajectory in an obstacle-filled workspace, how fast the robot will traverse the path when its velocity is scaled according to different methods? In this part of the study, we compared the proposed method with the approach from [26], [25], and with the one from [24]. The method from [28], [29] is omitted from this part of the study. Reason for this is that the method does not continuously scale the robot's velocity during the path traversal. It actually executes the nominal trajectory if possible, while maintaining the dynamic safety zones that are computed based on estimated stopping time. If the safety constraint is violated (dynamic safety zone intersects the obstacle), a stopping command is issued, without adjusting the velocity in real-time to prevent robot from stopping if not necessary. However, the method is more relevant for the second part of study where the volumes of dynamic safety zones are compared to volumes of danger zones.

It is worth pointing out that the proposed method is completely equivalent to that from [38], in terms of the resulting scaling factor δ. The main difference is that the proposed method uses geometrical solution based on danger zones instead of the analytical one from [38]. Beside the original approach with constant braking time T_b, we examine the option where T_b is adjusted on-line by (14). For this purpose, we use a conservative estimate of the acceleration capabilities of the robot by setting $\ddot{\mathbf{q}}_{\max} = [6.5\ 8.5\ 10\ 15\ 20\ 20]\ \frac{\text{rad}}{\text{s}^2}$.

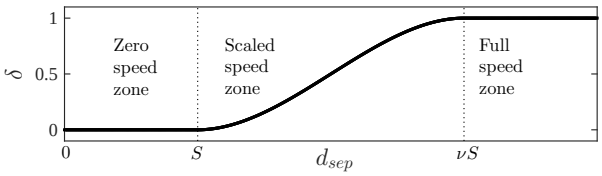

Fig. 8: Mapping from d_{sep} to δ from [24]. The middle part is a cubic polynomial that satisfies the continuity of the mapping and its first derivative.

In [24], the scaling factor δ is computed as a function of separation distance d_{sep} between the robot and the obstacle. The mapping $\delta = \delta(d_{sep}, S, \nu)$ is depicted in Fig. 8. The minimum protective separation distance S is computed in real-time using a modification of the one from [23]:

$$S = \alpha(t)\,[v_H T_R + v_H T_S + v_R T_R] + B + C + Z_S + Z_R, \quad (19)$$

where v_H is the maximum speed of the closest operator; v_R the maximum robot speed; T_R the time required by the machine to respond to the operator presence; T_S the response time of the machine that brings the robot to a safe, controlled stop; C the intrusion distance safety margin, i.e., an additional distance, based on the expected intrusion toward the critical zone prior to the actuation of the protective equipment; Z_R the robot position uncertainty; Z_S the operator position uncertainty, and B is the Euclidean distance traveled by the robot while braking. The function $\alpha(t)$ is the output of a fuzzy inference system that performs a suitable risk assessment.

To simplify the implementation of this scaling method (and hence make it faster), we have assumed that all the terms on the right-hand-side of (19) were zero, except for the variable B – the Euclidean distance traveled by the robot while braking. To relax the circumstances for the method even more, we conservatively estimated the distance B for each link as $B_i = \frac{1}{2} v_{i,\max} T_b$, where $v_{i,\max}$ is the maximum velocity of the i-th link (which happens to be the velocity of one of its endpoints). Moreover, a separation distance $d_{sep,i}$ is determined for each link. Then, a tentative scaling factor is computed for each link as $\delta_i = \delta(d_{sep,i}, B_i, \nu)$, using the mapping from Fig. 8, where the value $\nu = 2$ is assumed (as in the validation part of [24]). Finally, the velocity scaling factor δ is chosen as $\delta = \min_i \delta_i$.

We denote the path traversal times $T_p(k)$, $k = 1, 2, \ldots, 1000$ for the proposed method based on fixed braking time T_b. For the modified approach with adaptive T_b, the traversal times are $T_{pa}(k)$. Similarly, the times that correspond to methods from Zanchettin et al. [26], [25] and Costanzo et al. [24] are denoted respectively by $T_Z(k)$ and $T_C(k)$. Fig. 9 shows distributions of logged traversal times in the form of violin/box plots. Fig. 10 depicts distributions of ratios of traversal times collected from 1000 simulation runs. Universally, the proposed approach implies faster path following

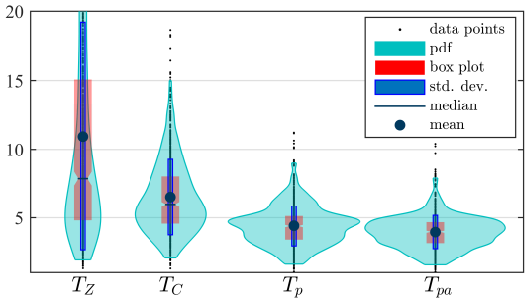

Fig. 9: Distributions of path traversal times for different methods captured by violin/box plots. Subscript Z stands for the method from [26], [25], C for the one from [24], while pa and p stand for the proposed methods (with and without braking time adaptation, respectively).

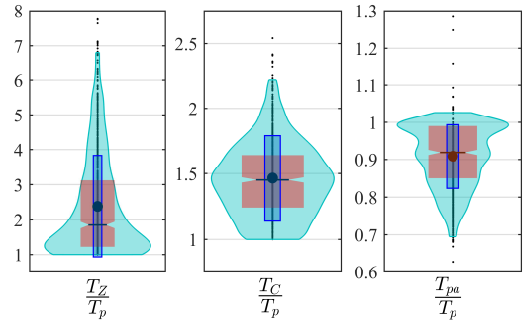

Fig. 10: Ratios of path traversal times for different methods captured by violin/box plots. Plots are shown separately due to scale differences. Subscript Z stands for the method from [26], [25], C for the one from [24], while pa and p stand for the proposed methods (with and without braking time adaptation, respectively).

when compared to the competing algorithms. On average, the proposed method is more than twice as fast than the one from [26], [25], and ca. 45% faster than the one from [24]. Note that, by construction, the method is still consistent with safety requirements. Moreover, the modified approach with adaptive braking time T_b introduced the additional speedup of ca. 10% on average.

B. Part II: Danger zone volumes

In this part of the study, the proposed method is compared to the one from Zanchettin et al. [26], [25] and that from Scalera et al. [28], [29]. The objective is to compare the volumes of the danger zones proposed in this work to volumes of their respective analogues: *regions to avoid* from [26], [25], and *dynamic safety zones* from [28], [29]. For a simpler nomenclature, we will refer to all of these regions as danger zones. The method from [24] is not included in this study since it does not explicitly address the volumes of space where the safety constraint is violated[1]. We denote the danger zones of the respective methods by $\mathcal{DZ}_p$, $\mathcal{DZ}_{pa}$, $\mathcal{DZ}_Z$ and $\mathcal{DZ}_S$. Indices are inherited from the notation used in the first part of the study, with the exception of $\mathcal{DZ}_S$ denoting the danger zone from Scalera et al. [28], [29]. Fig. 11 depicts typical shapes of danger zones associated to a specific robot's state. Note that the method from [28], [29] requires full dynamical model of the robot. For this purpose, we used a dynamic model of Comau SmartSix robot, which is unnecessary for the approach proposed in this paper. Moreover, to simplify the implementation (and hence make it less conservative), reaction time, obstacle velocity, and potential uncertainties are neglected. Thus, the resulting danger zone reflects only the motion of robot's links (see [28], [29] for details).

Let μ_p, μ_{pa}, μ_Z and μ_S be the volumes of respective danger zones $\mathcal{DZ}_p$, $\mathcal{DZ}_{pa}$, $\mathcal{DZ}_Z$ and $\mathcal{DZ}_S$. Here, we refer to the volume of the union of danger zones associated to all the links. Due to their inherent rotational symmetry, the volumes μ_p, μ_{pa} and μ_S can be obtained analytically. For $\mathcal{DZ}_Z$, the volume μ_Z is estimated using quasi-Monte Carlo numerical integration [60] that uses a low-discrepancy Hammersley sequence (with 10^5 points in the robot's workspace) [61]. The danger zone volume is estimated as the ratio of number of points that belong to a danger zone versus total number of points. To compare the relevant methods, a 1000 feasible random robot states $(\mathbf{q}, \dot{\mathbf{q}})$ have been generated, for which the volumes of corresponding danger zones are computed/estimated. Fig. 12 shows distributions of obtained volumes in the form of violin/box plots. Fig. 13 depicts distributions of ratios of traversal times collected from 1000 simulation runs.

Danger zones corresponding to proposed method have substantially smaller volume when compared to other approaches. On average, μ_p is ca. 3 times smaller than μ_Z, and an order of magnitude smaller than μ_S. Furthermore, if the method with adaptive braking time T_b is used, the danger zone volume is additionally reduced by more than 20% on average.

C. Part III: Human-robot scenario

In this part of the study, we validate the proposed method in a simulation that closely matches the HRC scenario. For this purpose, a human-shaped moving obstacle with a surface specified by triangles (ca. 350) is simulated. The human motion is simulated with the help of the Carnegie Mellon Motion Capture Database [62] and the MATLAB Motion Capture Toolbox [63]. The original motion from [62] considers human skeleton model, which we generalize by associating volume to the human figure and expressing it with triangular meshes. The setup is depicted in Fig. 14. The end-effector follows a predefined path, while the human conducts a series of manual tasks within the manipulator's reach. The simulation is performed for two different parameterizations of the same geometric path and for the same trajectory scaling methods used in the first part of the simulation study. For simplicity, and due to the fact that it does not introduce exceptional performance improvement, we omit the method based on adaptive braking time T_b from this part of the study. Resulting time profiles for the scaling factor δ are shown in Fig. 15. The signal δ_p denotes the scaling factor resulting from the method proposed in this paper. Similarly, signals δ_Z and δ_C correspond respectively to methods from Zanchettin et al. [26], [25] and Costanzo et al. [24]. Relative to existing approaches, the new method clearly allows for substantially faster path following.

[1]There might be a possibility to associate a suitable volume-based interpretation of this method, however, we believe that addressing this question would not add much value to this study.

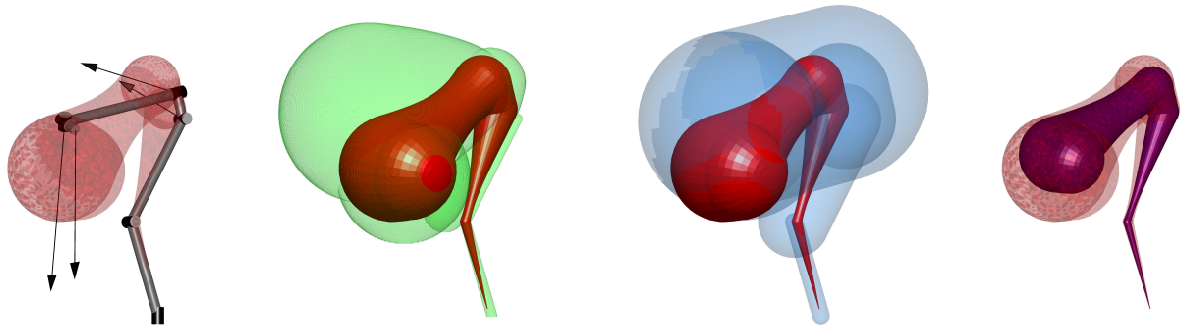

Fig. 11: Danger zones w.r.t. different methods around a 6 DOF manipulator. Far left – $\mathcal{DZ}_p$ (transparent red), the linear velocities of link endpoints indicated by arrows. Left – $\mathcal{DZ}_p$ (solid red) and $\mathcal{DZ}_Z$ (transparent green, [26], [25]). Right – $\mathcal{DZ}_p$ (solid red) and $\mathcal{DZ}_S$ (transparent blue, [28], [29]). Far right – $\mathcal{DZ}_p$ (transparent red) and $\mathcal{DZ}_{pa}$ (solid purple).

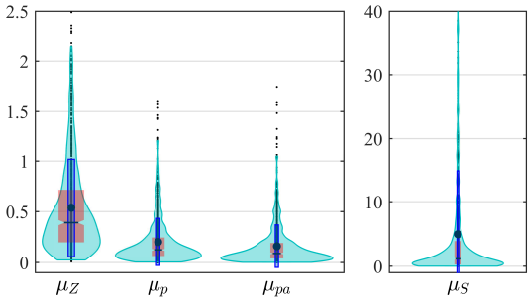

Fig. 12: Distributions of danger zone volumes for different methods captured by violin/box plots. Separated plots are due to scale differences. Subscript Z stands for the method from [26], [25], S for the one from [28], [29], while pa and p stand for the proposed methods (with and without braking time adaptation, respectively).

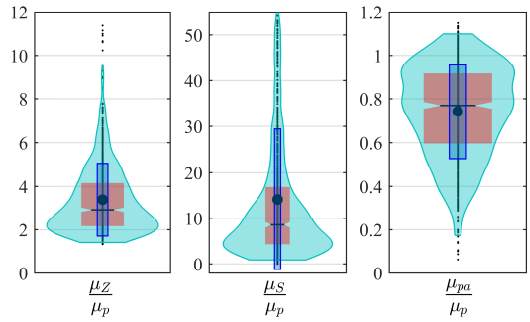

Fig. 13: Ratios of danger zone volumes for different methods captured by violin/box plots. Separated plots are due to scale differences. Subscript Z stands for the method from [26], [25], S for the one from [28], [29], while pa and p stand for the proposed methods (with and without braking time adaptation, respectively).

This is fully consistent with the results from the first part od the simulation study. Furthermore, we may note that, as the nominal trajectory becomes slower, the relative difference between traversal times decreases.

Finally, a curious comparison is presented in Fig 16. It juxtaposes the history of scaling factor δ_p from the proposed method to "would-be" values of scaling factors δ_Z and δ_C, which are computed in the background, but not actually used to scale the trajectory. For most of the time, computed values of δ_Z and δ_C are smaller than δ_p. This reveals the advantage of proposed method from another point of view.

V. Experiments

A. Experimental Setup and Implementation Details

The proposed approach has been tested using a Comau SmartSix robot. The robot is controlled by an open version of the industrial Comau C4G controller, that allows for a fast external interface with sampling time of $\Delta t = 2$ms. The algorithm is implemented in C language on an external real-time Linux PC that is connected to the C4Gopen controller via real-time Ethernet. Each 2ms, the PC sends to the controller the control commands. This information is actually the increment of joint variables that the robot has to cover in 2ms to reach the desired position. In particular, the scaling of joint velocity is implemented using the notion of scaled time τ. The evolution of the scaled time, and corresponding desired joint configuration are governed by the following equations:

$$\tau_{k+1} = \tau_k + \delta\Delta t, \quad \mathbf{q}_{k+1} = \mathbf{q}_n\left(\tau_{k+1}\right),$$

where $\mathbf{q}_n(t)$ denotes the prescribed nominal trajectory. Analogous approach to scaling can also be found in [26], [24], [12].

The robot can send back to the PC different types of information, among them: joint variables and homogeneous transformation matrix of the end-effector frame.

In order to detect the obstacle, i.e., the human operator, a range camera (Microsoft Kinect V2) has been selected. The camera is connected to a Mini PC Intel NUC that communicates with the Linux PC via Ethernet. Thus, the information from the camera is sent to the Linux PC and used by the algorithm. The sampling frequency of the perception system is 30Hz. Fig. 17 shows the overall system architecture.

The original information retrieved from Kinect has the format of a point-cloud. There are several possible ways to induce a useful volumetric representation of the human from point-cloud-based inputs. One option is to first extract the skeleton model and later use geometric primitives (e.g., spheres, cylinders, capsules, or other shapes) to compute a segmented representation of human body (head, torso, upper arms, forearms, etc.). Each relevant body part can be then represented by a triangular mesh, or a separate point-cloud. Any of these structures can be later used for collision detection with danger zones. Another option is to process the point cloud without segmentation and obtain the unique triangular mesh. The simplest option is to represent the human as a single blob

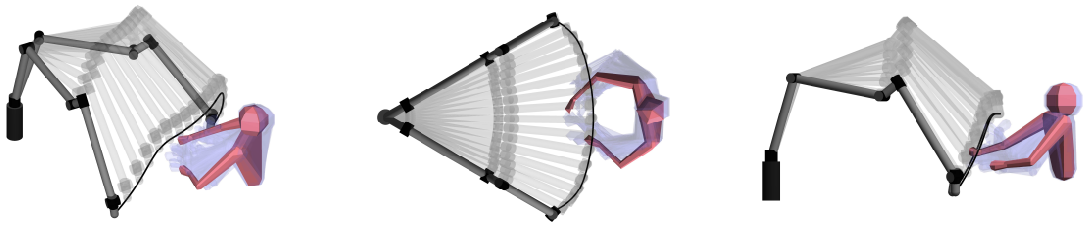

Fig. 14: Distinct views of collaboration scenario involving a human and a 6 DOF manipulator. Motion of human/robot is depicted by transparent volumes, while the path of the end-effector is shown by solid black line.

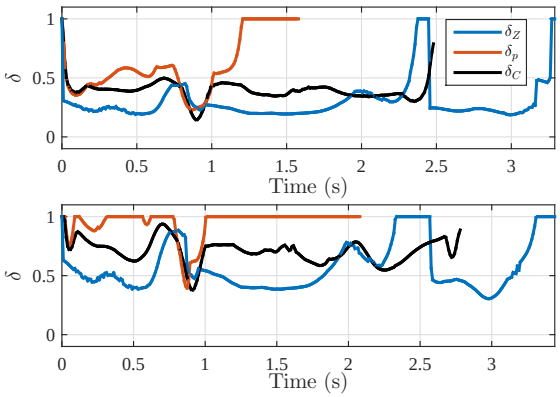

Fig. 15: Scaling factor δ time profiles for two different nominal trajectories along the same geometric path. The first nominal trajectory lasts 1s (top), the second one lasts 2s (bottom). Subscript Z stands for the method from [26], [25], C for the one from [24], while p stands for the proposed methods.

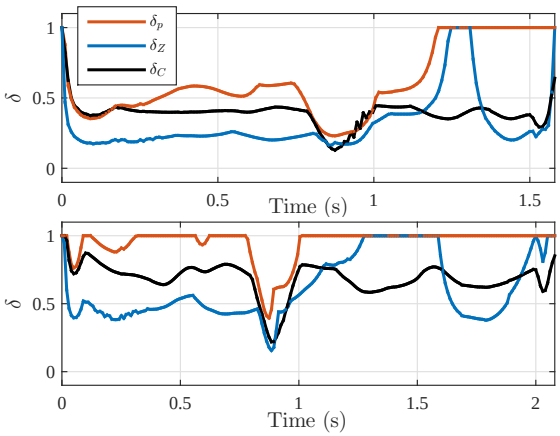

Fig. 16: History of the scaling factor δ_p compared to "would-be" profiles of factors δ_Z and δ_C. The first nominal trajectory lasts 1s (top), the second one lasts 2s (bottom). Subscript Z stands for the method from [26], [25], C for the one from [24], while p stands for the proposed methods.

(e.g., a capsule or a cylinder). In principle, our method does not assume anything specific in terms of geometric representation of humans/obstacles. In the description of approach, we tacitly assume the most general representation, i.e., a triangular mesh that could capture a wide variety of surfaces.

Nevertheless, in order to facilitate the implementation, the human operator is represented by a single capsule. The Microsoft Kinect detects the position of the operator's shoulders. The axis of the capsule passes through the midpoint of the segment connecting two shoulders. The radius of the capsule is computed as the distance between one of the shoulders and the midpoint. The capsule is further augmented by the additional margin – a worst-case distance that the obstacle can cover in the next time interval assuming the usually adopted maximal walking speed $v_H = 1.6\frac{\mathrm{m}}{\mathrm{s}}$ [23], [20].

TABLE III: Relative errors due to convex approximation of danger zones

Link	1	2	3	4	5	6
σ [%]	0	3.9	0.9	6.5	0	0.7

For collision checking between the human and danger zones, GJK algorithm [47] is used, implemented via *openGJK* library [49]. GJK algorithm assumes convex geometries of the shapes tested for intersection. Therefore, we used a convex approximation of danger zones (see Fig. 5). Each danger zone is sampled by ca. 200 points, resulting in ca. 400 triangles.

A question arises how large is the approximation error caused by this assumption. To this end, we have performed a numerical study where we compared the volumes μ_p of the original danger zones to volumes μ_{conv} obtained via convex representation. We subjected the robot to 10^5 random states and computed the respective danger zone volumes for each link separately. The mean relative approximation errors $\sigma = \frac{\mu_{conv}-\mu_p}{\mu_p} \cdot 100\%$ are reported in Table III. The results indicate that the error implied by approximating the danger zone with its convex hull is nearly marginal. Clearly, if the original danger zones shapes (which are non-convex in general) are used for validation, algorithms other than GJK have to be used. In particular, based on results from Table II, it is advised to use BVH-based representation via OBBs [52], or spheres [53], both available e.g., in [46].

Within experimental study, the robot executes a palletizing operation following the "bridge" path according to typical trapezoidal velocity profile. The duration of the nominal trajectory is 5s, with an acceleration and deceleration phase of 1s each, while the cruise speed is set to $658\mathrm{mms}^{-1}$. When the human is not present in the scene, the robot fulfills its task, i.e., follows the path at the nominal programmed speed. As soon as the operator enters the workspace, he/she is tracked by the depth camera and the speed of the robot is adjusted accordingly. The experimental setup is reported in Fig. 18.

B. Results

The experimental study covers three distinct scenarios:

- **experiment** 1: the human operator is far away from the robot, which can follow the path at the programmed, nominal speed $\dot{\mathbf{q}} = \dot{\mathbf{q}}_n$, i.e., $\delta = 1$;

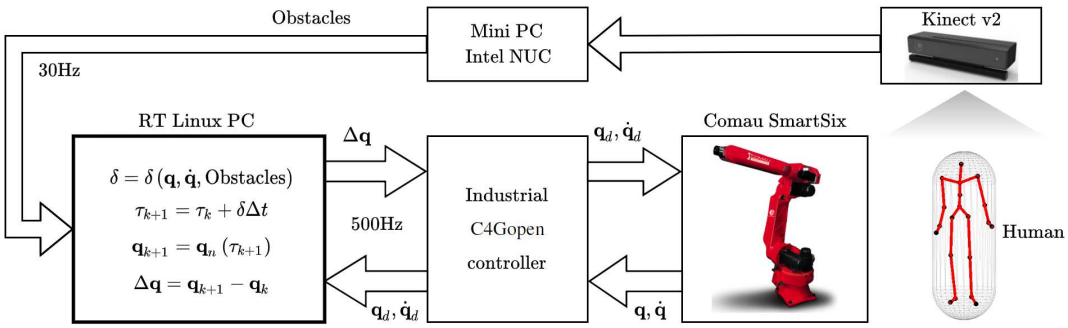

Fig. 17: Overall system architecture

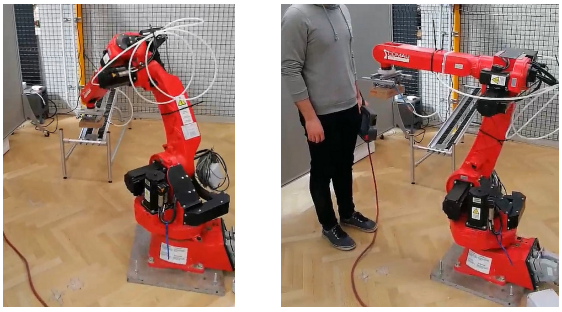

Fig. 18: Experimental setup with Comau Smart6 robot

- **experiment** 2: the human operator is closer to the robot, which has to scale down the velocity in order to satisfy the safety constraint. However, the robot is not forced to stop;
- **experiment** 3: the human operator gets excessively close to the robot, which forces it to stop and maintain zero speed until the human moves far enough.

In particular, we are interested in the profiles of the following quantities: the scaling factor δ, the the consequent speed of the path traversal, and the distance $d(\mathcal{DZ}, \text{human})$ between the danger zone $\mathcal{DZ}$ and the human.

From Fig. 19, reporting the results of the first experiment, one can see that the minimum distance $d(\mathcal{DZ}, \text{human})$ computed while the manipulator moves at nominal speed, is always greater than the threshold (which is set to zero). Therefore, there is no need to scale down the velocity. Hence, the scaling factor δ is always equal to 1 and the robot follows nominal velocity trajectory.

Fig. 20 shows the relevant signals related to the second experiment. Note that, at a certain time instant ($t \approx 2$s), the distance $d(\mathcal{DZ}, \text{human})$ reaches the threshold, meaning that the human model starts colliding with the danger zone $\mathcal{DZ}$. As soon as this occurs, the scaling factor δ assumes values less than 1. In other words, the robot cannot follow the path with the nominal programmed speed without violating the safety constraints. However, the scaling factor δ is optimal in the sense that the resulting speed is maximum possible, while still keeping the safety constraints satisfied, i.e., maintaining the contact between the human and the danger zone without the penetration. While $\delta < 1$, the distance $d(\mathcal{DZ}, \text{human})$ takes zero value (until $t \approx 3.2$s). Once the circumstances allow, the value of δ returns to 1 and the robot resumes to follow the path with the nominal speed.

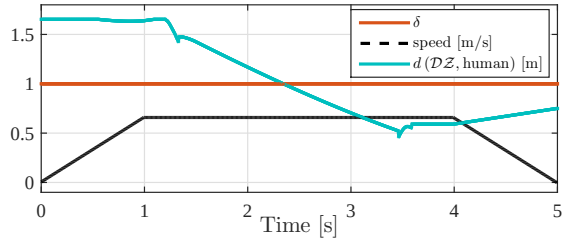

Fig. 19: Velocity, minimum distances and scaling factor δ comparison during experiment 1

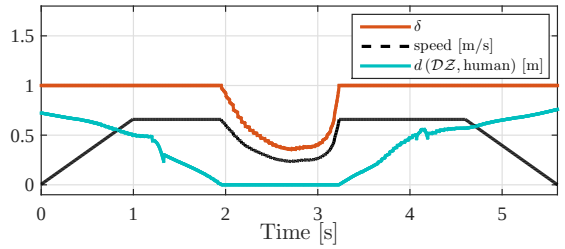

Fig. 20: Velocity, minimum distances and scaling factor δ comparison during experiment 2

The third experiment (Fig. 21) is conceptually similar to the second one – as soon as the distance $d(\mathcal{DZ}, \text{human})$ reaches the critical value, the scaling factor δ becomes smaller than 1 and the nominal velocity profile cannot be sustained. However, due to extreme human-robot proximity, there is no non-zero velocity that can satisfy the safety constraint. Therefore, both the scaling factor δ and the velocity approach zero. When the human-robot distance starts increasing, due to human operator moving away, the scaling factor δ increases as well, until it returns to 1. Finally, the robot may resume the nominal velocity profile.

VI. Conclusion

This study presented an approach to ensure safe human-robot collaboration using a geometric method that explicitly represents the danger zones around the robot. Danger zones are regions of space where safety constraints are violated. The constraints have been formulated in accordance to speed and separation monitoring paradigm. A simple way to use danger zones for scaling the trajectory is presented. The overall approach is validated within an experimental study involving

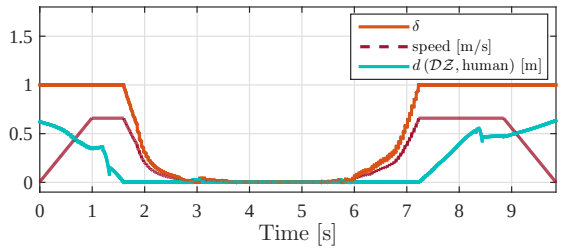

Fig. 21: Velocity, minimum distances and scaling factor δ comparison during experiment 3

an industrial manipulator and a depth-camera-based perception system.

Future work will explore the possibilities to further enhance the robot's productivity without jeopardizing safety constraints by considering more detailed knowledge on robot's inertial parameters.

ACKNOWLEDGMENT

Authors would like to thank Luca Carraro for his help with conducting the experiments. We also thank Dinko Osmankovic for his valuable advice on collision detection.

REFERENCES

[1] "RIA/ANSI R15.06 - 1999 American national standard for industrial robots and robot systems - safety requirements, New York: American National Standards Institute," 1999.

[2] "ISO 2006. Robots for industrial environments, Safety requirements part 1: Robot. standard ISO10218, International Organization for Standardization." 2006.

[3] J. Fryman and B. Matthias, "Safety of industrial robots: From conventional to collaborative applications," in *ROBOTIK 2012; 7th German Conference on Robotics*, 2012, pp. 1–5.

[4] A. Ajoudani, A. M. Zanchettin, S. Ivaldi, A. Albu-Schäffer, K. Kosuge, and O. Khatib, "Progress and prospects of the human–robot collaboration," *Autonomous Robots*, vol. 42, no. 5, pp. 957–975, 2018.

[5] K. Ikuta, M. Nokata, and H. Ishii, "General danger evaluation method for control strategy of human-care robot," *Journal of the Robotics Society of Japan*, vol. 19, no. 1, pp. 81–90, 2001.

[6] D. Kulic and E. A. Croft, "Real-time safety for human-robot interaction," *Robotics and Autonomous Systems*, vol. 54, no. 1, pp. 1–12, 2006.

[7] B. Lacevic and P. Rocco, "Kinetostatic danger field-a novel safety assessment for human-robot interaction," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2010, pp. 2169–2174.

[8] B. Lacevic, P. Rocco, and A. M. Zanchettin, "Safety assessment and control of robotic manipulators using danger field," *IEEE Transactions on Robotics*, vol. 29, no. 5, pp. 1257–1270, 2013.

[9] M. P. Polverini, A. M. Zanchettin, and P. Rocco, "Real-time collision avoidance in human-robot interaction based on kinetostatic safety field," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2014, pp. 4136–4141.

[10] ——, "A computationally efficient safety assessment for collaborative robotics applications," *Robotics and Computer-Integrated Manufacturing*, vol. 46, pp. 25–37, 2017.

[11] A. M. Zanchettin, B. Lacevic, and P. Rocco, "Passivity-based control of robotic manipulators for safe cooperation with humans," *International Journal of Control*, vol. 88, no. 2, pp. 429–439, 2015.

[12] S. Haddadin, A. Albu-Schaffer, A. De Luca, and G. Hirzinger, "Collision detection and reaction: A contribution to safe physical human-robot interaction," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2008, pp. 3356–3363.

[13] S. Haddadin, A. Albu-Schäffer, and G. Hirzinger, "Requirements for safe robots: Measurements, analysis and new insights," *The International Journal of Robotics Research*, vol. 28, no. 11-12, pp. 1507–1527, 2009.

[14] A. Albu-Schaffer, O. Eiberger, M. Grebenstein, S. Haddadin, C. Ott, T. Wimbock, S. Wolf, and G. Hirzinger, "Soft robotics," *IEEE Robotics & Automation Magazine*, vol. 15, no. 3, pp. 20–30, 2008.

[15] B. Sadrfaridpour and Y. Wang, "Collaborative assembly in hybrid manufacturing cells: an integrated framework for human–robot interaction," *IEEE Transactions on Automation Science and Engineering*, vol. 15, no. 3, pp. 1178–1192, 2017.

[16] N. Nikolakis, V. Maratos, and S. Makris, "A cyber physical system (cps) approach for safe human-robot collaboration in a shared workplace," *Robotics and Computer-Integrated Manufacturing*, vol. 56, pp. 233–243, 2019.

[17] M. Raessa, J. C. Y. Chen, W. Wan, and K. Harada, "Human-in-the-loop robotic manipulation planning for collaborative assembly," *IEEE Transactions on Automation Science and Engineering*, vol. 17, no. 4, pp. 1800–1813, 2020.

[18] X. Zhao, T. Fan, Y. Li, Y. Zheng, and J. Pan, "An efficient and responsive robot motion controller for safe human-robot collaboration," *IEEE Robotics and Automation Letters*, 2021.

[19] A. Casalino, A. M. Zanchettin, L. Piroddi, and P. Rocco, "Optimal scheduling of human-robot collaborative assembly operations with time Petri nets," *IEEE Transactions on Automation Science and Engineering*, 2019.

[20] C. Byner, B. Matthias, and H. Ding, "Dynamic speed and separation monitoring for collaborative robot applications–concepts and performance," *Robotics and Computer-Integrated Manufacturing*, vol. 58, pp. 239–252, 2019.

[21] J. A. Marvel and R. Norcross, "Implementing speed and separation monitoring in collaborative robot workcells," *Robotics and computer-integrated manufacturing*, vol. 44, pp. 144–155, 2017.

[22] A. Campomaggiore, M. Costanzo, G. Lettera, and C. Natale, "A fuzzy inference approach to control robot speed in human-robot shared workspaces," in *16th International Conference on Informatics in Control, Automation and Robotics, ICINCO 2019*, vol. 2. SciTePress, 2019, pp. 78–87.

[23] ISO TC184/SC2, *ISO/TS 15066 Robots and robotic devices – Safety requirements for industrial robots – Collaborative operation*, 2013.

[24] M. Costanzo, G. De Maria, G. Lettera, and C. Natale, "A multimodal approach to human safety in collaborative robotic workcells," *IEEE Transactions on Automation Science and Engineering*, 2021.

[25] A. M. Zanchettin, N. M. Ceriani, P. Rocco, H. Ding, and B. Matthias, "Safety in human-robot collaborative manufacturing environments: Metrics and control," *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 2, pp. 882–893, 2015.

[26] A. M. Zanchettin and P. Rocco, "Path-consistent safety in mixed human-robot collaborative manufacturing environments," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 1131–1136.

[27] A. Palleschi, M. Hamad, S. Abdolshah, M. Garabini, S. Haddadin, and L. Pallottino, "Fast and safe trajectory planning: Solving the cobot performance/safety trade-off in human-robot shared environments," *IEEE Robotics and Automation Letters*, 2021.

[28] L. Scalera, A. Giusti, R. Vidoni, V. Di Cosmo, D. Matt, and M. Riedl, "Application of dynamically scaled safety zones based on the ISO/TS 15066: 2016 for collaborative robotics," *Int. J. Mech. Control*, vol. 21, pp. 41–49, 2020.

[29] L. Scalera, R. Vidoni, and A. Giusti, "Optimal scaling of dynamic safety zones for collaborative robotics," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 3822–3828.

[30] P. Aivaliotis, S. Aivaliotis, C. Gkournelos, K. Kokkalis, G. Michalos, and S. Makris, "Power and force limiting on industrial robots for human-robot collaboration," *Robotics and Computer-Integrated Manufacturing*, vol. 59, pp. 346–360, 2019.

[31] L. Joseph, J. K. Pickard, V. Padois, and D. Daney, "Online velocity constraint adaptation for safe and efficient human-robot workspace sharing," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 11 045–11 051.

[32] P. Svarny, M. Tesar, J. K. Behrens, and M. Hoffmann, "Safe physical HRI: Toward a unified treatment of speed and separation monitoring together with power and force limiting," *arXiv preprint arXiv:1908.03046*, 2019.

[33] N. Lucci, B. Lacevic, A. M. Zanchettin, and P. Rocco, "Combining speed and separation monitoring with power and force limiting for safe collaborative robotics applications," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6121–6128, 2020.

[34] A. Zacharaki, I. Kostavelis, A. Gasteratos, and I. Dokas, "Safety bounds in human robot interaction: A survey," *Safety Science*, vol. 127, p. 104667, 2020.

[35] Z. Bi, C. Luo, Z. Miao, B. Zhang, W. Zhang, and L. Wang, "Safety assurance mechanisms of collaborative robotic systems in manufacturing,"

Robotics and Computer-Integrated Manufacturing, vol. 67, p. 102022, 2021.
[36] P. Glogowski, K. Lemmerz, A. Hypki, and B. Kuhlenkötter, "Extended calculation of the dynamic separation distance for robot speed adaption in the human-robot interaction," in *2019 19th International Conference on Advanced Robotics (ICAR)*. IEEE, 2019, pp. 205–212.
[37] M. Ragaglia, A. M. Zanchettin, and P. Rocco, "Trajectory generation algorithm for safe human-robot collaboration based on multiple depth sensor measurements," *Mechatronics*, vol. 55, pp. 267–281, 2018.
[38] B. Lacevic, A. M. Zanchettin, and P. Rocco, "Towards the exact solution for speed and separation monitoring for improved human-robot collaboration," in *2020 29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*. IEEE, pp. 1190–1195.
[39] IEC, EN 60204, *IEC, EN 60204—"Safety of machinery, Electrical Equipment of machines, General requirements*, 2016.
[40] ABB Robotics, *Robot stopping distances according to iso 10218-1, ID: 3HAC048645-001*, 2017.
[41] J. Montonen, I. Marstio, and T. Malm, "Dynamic safety system for safe collaboration with industrial robots," in *Advanced Human-Robot Collaboration in Manufacturing*. Springer, 2021, pp. 141–152.
[42] B. Lindqvist, "Multi-axis industrial robot braking distance measurements: For risk assessments with virtual safety zones on industrial robots," Master's thesis, University West, Department of Engineering Science, Trollhättan, SWEDEN, 2017.
[43] A. Labusch, T. Bellmann, K. Sharma, and J. Bals, "Worst case braking trajectories for robotic motion simulators," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 3297–3302.
[44] T. Dietz and A. Verl, "Simulation of the stopping behavior of industrial robots using a complementarity-based approach," in *2011 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*. IEEE, 2011, pp. 428–433.
[45] E. Larsen, S. Gottschalk, M. C. Lin, and D. Manocha, "Fast proximity queries with swept sphere volumes," Technical Report TR99-018, Department of Computer Science, UNC Chappel Hill, Tech. Rep., 1999.
[46] J. Pan, S. Chitta, and D. Manocha, "FCL: A general purpose library for collision and proximity queries," in *2012 IEEE International Conference on Robotics and Automation*. IEEE, 2012, pp. 3859–3866.
[47] E. G. Gilbert, D. W. Johnson, and S. S. Keerthi, "A fast procedure for computing the distance between complex objects in three-dimensional space," *IEEE Journal on Robotics and Automation*, vol. 4, no. 2, pp. 193–203, 1988.
[48] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, and G. Taubin, "The ball-pivoting algorithm for surface reconstruction," *IEEE transactions on visualization and computer graphics*, vol. 5, no. 4, pp. 349–359, 1999.
[49] M. Montanari and N. Petrinic, "OpenGJK for C, C# and Matlab: Reliable solutions to distance queries between convex bodies in three-dimensional space," *SoftwareX*, vol. 7, pp. 352–355, 2018.
[50] S. Cameron, "Enhancing GJK: Computing minimum and penetration distances between convex polyhedra," in *Proceedings of international conference on robotics and automation*, vol. 4. IEEE, 1997, pp. 3112–3117.
[51] G. v. d. Bergen, "Efficient collision detection of complex deformable models using AABB trees," *Journal of graphics tools*, vol. 2, no. 4, pp. 1–13, 1997.
[52] S. Gottschalk, M. C. Lin, and D. Manocha, "OBBtree: A hierarchical structure for rapid interference detection," in *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. ACM, 1996, pp. 171–180.
[53] S. Quinlan, "Real-time modification of collision-free paths," Ph.D. dissertation, Stanford University, 1995.
[54] J. T. Klosowski, M. Held, J. S. Mitchell, H. Sowizral, and K. Zikan, "Efficient collision detection using bounding volume hierarchies of k-DOPs," *IEEE Transactions on Visualization & Computer Graphics*, no. 1, pp. 21–36, 1998.
[55] C. Ericson, *Real-time collision detection*. CRC Press, 2004.
[56] S. A. Gottschalk, *Collision queries using oriented bounding boxes*. The University of North Carolina at Chapel Hill, 2000.
[57] A. Del Prete, "Joint position and velocity bounds in discrete-time acceleration/torque control of robot manipulators," *IEEE Robotics and Automation Letters*, vol. 3, no. 1, pp. 281–288, 2017.
[58] P. Corke, *Robotics and Control: Fundamental Algorithms in MATLAB®*. Springer Nature, 2021, vol. 141.
[59] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.
[60] H. Niederreiter, *Random number generation and quasi-Monte Carlo methods*. Siam, 1992, vol. 63.
[61] J. M. Hammersley, "Monte Carlo methods for solving multivariable problems," *Annals of the New York Academy of Sciences*, vol. 86, no. 3, pp. 844–874, 1960.
[62] CMU, "CMU graphics lab motion capture database," 2003, http://mocap.cs.cmu.edu.
[63] N. D. Lawrence, "MOCAP toolbox for MATLAB," 2009, http://github.com/lawrennd/mocap.

Bakir Lacevic received the Dipl.-Ing. and Magister degrees in automatic control from the University of Sarajevo, Sarajevo, Bosnia and Herzegovina, in 2003 and 2007, respectively, and the Ph.D. degree in information technology from Politecnico di Milano, Milano, Italy, in 2011. Currently, he is an Associate Professor with the Faculty of Electrical Engineering, University of Sarajevo, where he teaches courses in robotics and modeling/identification of dynamic systems. Dr. Lacevic has coauthored ca. 60 papers in robotics, automatic control, optimization and computational intelligence. He has been a member of the IEEE Robotics and Automation Society since 2010. His research interests include robotic motion planning, human-robot interaction, optimization and machine learning.

Andrea Maria Zanchettin received the M.Sc. degree in computer science engineering and the Ph.D. degree in information technology from the Politecnico di Milano, Milan, Italy, in 2008 and 2012, respectively. During Spring 2010, he spent a research stay at the Department of Automatic Control (Reglerteknik), Lund University. From January 2012 to February 2014, he was a Temporary Research Assistant and, from March 2014 to September 2016, a fixed-term Assistant Professor with the Dipartimento di Elettronica, Informazione e Bioingegneria (DEIB), Politecnico di Milano, where he is currently an Associate Professor. He has coauthored around 90 papers in automatic control and intelligent human–robot interaction. Dr. Zanchettin has been a member of the IEEE Robotics and Automation Society since 2009. In September 2014, he was a recipient of the Young Author Best Paper Award sponsored by the Italian Chapter of the IEEE Robotics and Automation Society (I-RAS). In 2019, he has been elected as the Chair of I-RAS.

Paolo Rocco is currently a Full Professor in automatic control and robotics with the Politecnico di Milano, Milan, Italy, where he has served as the Chair of the B.Sc. and M.Sc. Programs on Automation and Control Engineering. He is also a co-founder of Smart Robots, a spin-off company of the Politecnico di Milano. He has authored about 200 papers in the areas of robotics, motion control, and mechatronics. His research interests concern industrial robotics, with a particular focus on safe and productive human–robot interaction. Prof. Rocco is currently in the Board of Directors of euRobotics, the association of all stakeholders in robotics in Europe, private part in the Public Private Partnership (PPP) SPARC. He has served in various positions in the editorial boards of journals and conferences, including being a Senior Editor for the IEEE ROBOTICS AND AUTOMATION LETTERS. At present he serves as an Associate Editor for the IFAC journal *Mechatronics*. He has been in charge of several research projects with industrial partners and public bodies. He serves as an expert for the European Commission and for the Italian Ministry of University and Research for the evaluation of proposals and ongoing projects.