Towards an optimal avoidance strategy for collaborative robots

Andrea Maria Zanchettin[a,*], Paolo Rocco[a], Simone Chiappa[b], Roberto Rossi[b]

[a] *Politecnico di Milano, Dipartimento di Elettronica, Informazione e Bioingegneria, Piazza Leonardo Da Vinci 32, Milano {andreamaria.zanchettin, paolo.rocco}@polimi.it*

[b] *Smart Robots s.r.l., via San Martino 12, Milano, {roberto.rossi, simone.chiappa}@smartrobots.it*

Abstract

Collaborative robots represent a game changer in manufacturing for their ease of use, the reduced need of safeguarding hardware and, consequently, their extremely fast payback time. However, most of the collaborative robots available on the market are power and force limiting (PFL) devices. The main disadvantage of this type of collaborative operation is that the robot is forced to stop when a collision occurs, as the only way the robot is aware of the presence of the human is through its embedded torque or motor current monitoring algorithms. Albeit tolerable from a safety point of view, these collisions might dramatically reduce the performance of the robot in terms of productivity, ultimately jeopardising the economic attractiveness of a collaborative workstation. This paper introduces an avoidance strategy that suggests the robot alternative paths to be traversed, that are both collision free and optimal in terms of minimum traversal time. The control strategy makes use of a depth camera in order to enhance robot perception of the environment. Moreover, by properly exploiting information coming from these sensors, the control strategy itself is able to communicate to the robot the best decision to take with respect to the presence of one or more human operators. The method is experimentally validated on a Universal Robots UR5.

1. Introduction

Collaborative Robotics represents one of the most promising and quickly growing areas of research in robotics. In fact, while traditional industrial robots require to be segregated within cages to ensure that they do not come into contact with humans and cause bodily harm, collaborative robots, also referred to as cobots, are able to work directly in the proximity of human operators, sharing the same workspace and performing combined operations that require the precision of a robot and the problem-solving capabilities of the human. Human-Robot Interaction (HRI), see e.g. [2, 3] for a review, brings the classical and static production line to a whole new level of flexibility and efficiency. However, as human operators are working in close proximity of robots without barriers, the safety issue which was not present in the past, becomes dominant.

Safety standards for industrial environment as the ISO 10218 [4] and more recently the technical specification ISO TS 15066 [5], as well as many research studies have been elaborated in the past and recent years in order to address the safety issue during a human-robot collaboration. In [6] the first systematic methodology to study severity specifically for human-care robots is used in order to quantitatively assess risk during a collision. [7] defined the concept of velocity obstacle, that is the set of all velocities of a robot that will result in a collision: this can be used in order to evaluate possible human-robot collisions. A similar strategy has been also proposed in [8] which introduced the concept of inevitable collision states. The idea of the adoption of a danger index has been further improved in [9], where a danger field is defined in order to capture how dangerous the current posture and velocity of an articulated robot is with respect to the objects within the same environment. This quantity is then used to assess danger during a human-machine collaboration and fed back to a control scheme in

[*] Corresponding author.

The authors would like to thank Mr. Giacomo Amante, a former MSc student at Politecnico di Milano, who supported them in coding, debugging and performing the validation experiments. The original method described in this work is protected by an international patent application, PCT/IB2017/052932, [1].

order to make the motion of the robot safer. A related work also introduced the concept of safety field, [10], which has been proved to be computationally efficient even in the presence of obstacles with complicated shapes.

A remarkable contribution in risk assessment has been presented in [11], where a model-based injury index is defined as the dissipated kinetic energy in a potential inelastic impact. This index is integrated in a constraint-based pre-collision control strategy in order to minimise the energy in the direction of the impact. Regarding obstacle avoidance, a fundamental contribution is given in [12] and [13] where the concepts of artificial potential field and elastic bands are defined for manipulators and mobile robots, and used to guide them in real-time towards the goal while avoiding obstacles. The idea of generating repulsive forces from an obstacle to modify the path of the robot is also exploited and implemented in [14], where a complete collision avoidance framework, from perception of the environment to joint-level robot control is presented and implemented. A collision avoidance method based on a probabilistic representation of the space occupied by the worker has been presented in [15]. The method introduces a digital representation of the behaviour of the human which is then used mainly offline to plan almost surely collision free trajectories for the robot.

Other works, see e.g. [16], are relying on attractor dynamics able to shape the velocity profile of the manipulator to ensure a safe coexistence of humans and robots in the same workspace.

In the context of manufacturing environments, the requirement of safety rated devices and functionalities is usually considered as a constraint in implementing a collaborative workstation. Recently, [17] proposed the adoption of a safe network of unsafe devices to monitor the workspace around the robot and the position of the operator.

Vision based techniques are typically adopted to predict and modify the behaviour of a collaborative robot based on the perceived position of the human. For example [18] presents a framework from hand movement prediction in a collaborative scenario with safety guarantee. A similar approach has been developed in [20] based on time series prediction of human motion, [22], while an efficient method to evaluate the distance of the human from the robot has been presented in [21]. Other approaches, in turn, rely on other sensing technologies, such as artificial skins [19]. Today's collaborative robots typically rely on the so-called power and force limiting (PFL) clause, defined in [4] and better detailed in [5]. It follows that, since the major source of potential danger is due to the kinetic energy, the majority of collaborative robots are either slow or lightweight. Such a class of manipulators, however, is not endowed with perception capabilities, and safety is achieved by monitoring the potential severity of an impact. In case such an impact occurs, these robots are usually required to be reset from a protective stop state, thus causing a downtime of the production facility. In contrast, another way to achieve a safe coexistence of humans and robots is through the speed and separation monitoring (SSM) clause, see again [4] and [23, 24]. In a nutshell, the (collaborative) workspace is monitored through a safety-rated sensor, typically a camera or a laser scanner, and the velocity of the robot is modulated based on the separation distance between the operator and the robot itself.

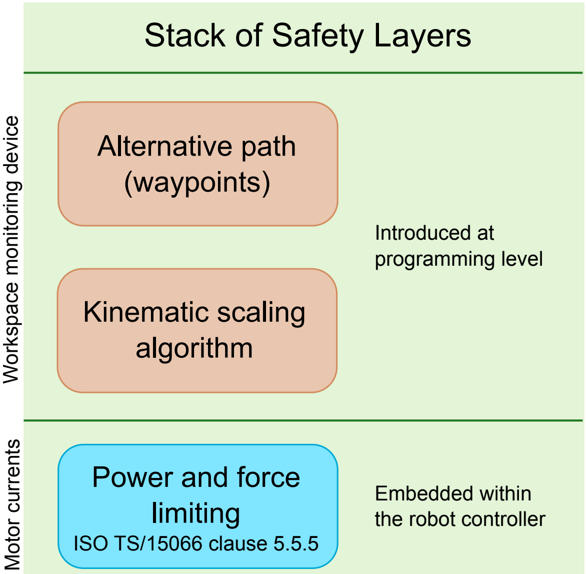

Stack of Safety Layers
Workspace monitoring device
Alternative path (waypoints)
Kinematic scaling algorithm
Introduced at programming level
Motor currents
Power and force limiting
ISO TS/15066 clause 5.5.5
Embedded within the robot controller

Figure 1: Hierarchy of safety functionalities proposed in this paper. The method is applicable to power and force limited robots, while a workspace monitoring device is adopted to compute alternative paths to avoid that the robot stops after a collision with the human operator.

According to the mostly updated safety recommendations, the implementation of either the PFL or the SSM clauses implies to halt the robot when a collision occurs, or when the distance is reduced under a certain amount. From a productivity point of view, however, in case of a continuous presence of the human operator within the collaborative workspace, frequent interruptions of the production cycle might be not tolerable. Frequent stops might also severely compromise the advantage in terms of fast payback time of the robotic investment that typically characterises collaborative applications. More in general, the role of digital human modelling systems for task planning as well as the safety related aspects are still open topics in both research and practical applications, [25].

One possible limitation of PFL robots is their intrinsic reduced speed. In order to mitigate the possible safety risks for the human operator, these robots are typically slower than traditional industrial manipulators. Moreover, they are controlled to stop whenever a collision occurs. On the other hand, they offer enhanced flexibility and reduced deployment effort, as they typically do not require safety fences. The benefits of collaborative robotics typically lie in reduced infrastructural investments and a minimum footprint. To further boost their return on investment (ROI), the downtime of these robots, especially due to collisions, should be minimised. This paper contributes in proposing a control strategy to prevent the robot from stopping along its programmed path. In particular, a method inspired by the SSM clause of [4] is introduced to insert waypoints along a pre-defined point to point motion that produce a collision free motion and to guarantee a minimum traversing time using a model-based prediction of the human occupancy. A relevant feature of the proposed algorithm is that it does not require a sophisticated research interface, as it only relies on typical programming primitives offered by any robotic programming language. The method is implemented within a Smart Robots[1] device and tested against a pick and place task, performed by a Universal Robots UR5 robot. The developed method (see the schematic representation in Fig. 1) allows to dramatically reduce the cycle time as compared to the one obtained by the safety countermeasures embedded in the robot controller. Moreover, in case of PFL robots, the workspace monitoring system is not required to be a safety-rated device.
The reminder of this paper is organised as follows. Section 2 summarises some background material that is used as a baseline for the development of the method. Section 3 details the developed strategy to evaluate the best alternative path that is proved to be collision free and entails some optimality in terms of minimum traversing time. The method has been then implemented and the outcome of the experiments are given in Section 4. Concluding remarks are finally given in Section 5.

2. Background material on safety constraints

This paper is based on the safety constraints originally introduced in [26], and more specifically on the concept of SSM: as distance between human and robot decreases, the velocity of the latter should be reduced accordingly. This fact can be analytically expressed imposing that at any time instant, the robot state of motion obeys the following constraints:

$$V\Delta T \leq \max\left(0, \mathrm{D} - K\Delta T\right) \tag{1}$$

where V represents the velocity of the robot in the direction of the human, ΔT is a reference time interval, e.g. the robot braking time, D the separating distance between the human and the robot, while K is the maximum velocity of the human, [27]. In [26] a complete characterisation of this kind of constraints has been discussed to consider all the points belonging to the robot. More specifically, with reference to Fig. 2, the following inequality

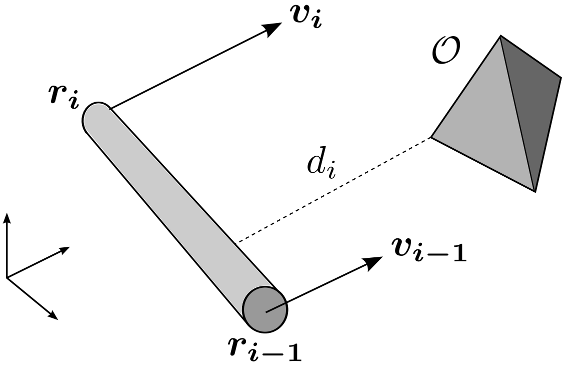


Figure 2: A link of the manipulator characterised by the position of its end-points $\boldsymbol{r_{i-1}}, \boldsymbol{r_i}$ and their velocities $\boldsymbol{v_{i-1}}, \boldsymbol{v_i}$ and a generic polytopic obstacle $\mathcal{O}$ The quantity d_i represents the minimum distance between the link and the obstacle.

$$\Delta T \boldsymbol{E}\left(\boldsymbol{p}\right) \dot{\boldsymbol{q}} \leq d_i^2 \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \forall \boldsymbol{p} \in \mathrm{vert}\left(\mathcal{O}\right) \tag{2}$$

[1] www.smartrobots.it/en/

can be proved to be equivalent to (1) for all the points belonging to the i-th link of the robot and all the points belonging to the human (regarded as an obstacle to be avoided). For the link i of a serial manipulator in a given configuration with velocity $\dot{\boldsymbol{q}}$, matrix $\boldsymbol{E}$ can be simply expressed by

$$\boldsymbol{E}(\boldsymbol{p}) = \begin{bmatrix} (\boldsymbol{p}-\boldsymbol{r_{i-1}})^T \boldsymbol{J_{i-1}} \\ (\boldsymbol{p}-\boldsymbol{r_{i-1}})^T \boldsymbol{J_i} - (\boldsymbol{r_i}-\boldsymbol{r_{i-1}})^T \boldsymbol{J_{i-1}} \end{bmatrix} \tag{3}$$

where $\boldsymbol{p}$ is a generic vertex of the polytopic obstacle $\mathcal{O}$, $\boldsymbol{r_i}, \boldsymbol{r_{i-1}}$ represent the position of the two end points, while $\boldsymbol{J_i}, \boldsymbol{J_{i-1}}$ are the corresponding position Jacobians (3-by-n matrices, where n is the number of degrees of freedom of the robot). Finally, in [28], a method to account for the motion of the human, i.e. the term $K\Delta T$ in (1), has been proposed based on the generation of the worst case swept volumes. This method allows to predict the occupancy of the human, see Fig. 3, during the time interval ΔT. The result is a set of polytopic volumes that are used in the calculation of the separation distance d_i in (2).

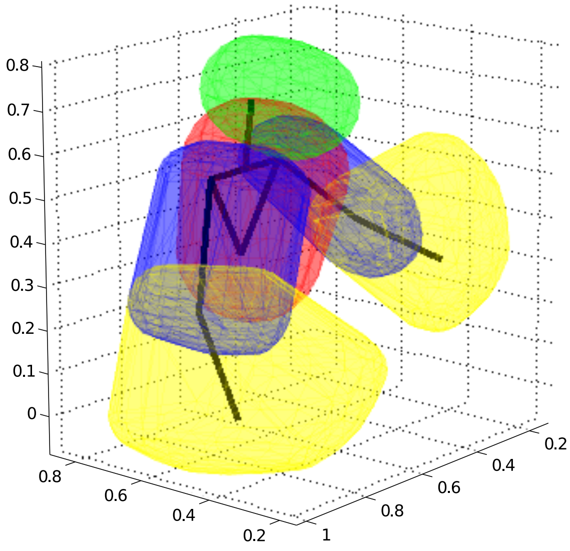


Figure 3: Prediction of the human occupancy during the time interval ΔT consisting in a set of polytopes, one per each body part (upper and lower arms, torso, head).

As explained in [26], the constraints in (2) can be used to modulate the robot velocity. In particular, if one introduces the scaling factor δ, the following linear programming (LP) problem can be solved

$$\max_{0\le\delta\le1} \delta \text{ subject to}$$

$$\Delta T \boldsymbol{E}(\boldsymbol{p})\,\dot{\boldsymbol{q}}\delta \le d_i^2 \begin{bmatrix}1\\1\end{bmatrix}, \forall \boldsymbol{p} \in \text{vert}(\mathcal{O})$$

in order to guarantee the maximum, yet safe velocity, along the pre-programmed path. As reported in [26], the previous LP problem can be actually solved without the adoption of a numerical solver. Each inequality, two per each pair link-vertex can be processed independently (and possibly in parallel) to compute the optimal value of δ.

3. Online generation of waypoints

The main contribution of this paper concerns the generation of alternative paths to prevent the robot from stopping or reducing its speed in presence of one or more obstacles along its path. In fact, the speed scaling algorithm developed in [26] only allows the robot to modulate (decrease or increase) its velocity along the path to minimise the risk for the operator. In order to increase the efficiency of the collaborative robot, to minimise the cycle time even when the human operator is present in the workspace of the robot, and to be compliant with a standard control architecture (i.e. without the need of sophisticated research interfaces), a new algorithm based on alternative paths has to be developed.

Differently from the kinematic scaling algorithm, however, the possibility to modify the pre-programmed path entails additional constraints to be included. Along the modified path, the robot has to satisfy the upper

$$\Delta T \dot{\boldsymbol{q}} \le \boldsymbol{q^{sup}} - \boldsymbol{q}$$

and lower joint limits

$$-\Delta T \dot{\boldsymbol{q}} \le \boldsymbol{q} - \boldsymbol{q^{inf}}$$

In many collaborative applications, the robot has to stay within the collaborative area, which is typically specified in terms of virtual walls. In order to enforce this kind of constraints, one can specify the wall in terms of its normal vector $\boldsymbol{v}$ and include, for each link of the robot, the following constraint

$$-\Delta T \boldsymbol{v}^T \boldsymbol{J_i} \dot{\boldsymbol{q}} \le d_i$$

where $\boldsymbol{J_i}$ is the positional Jacobian of the end-point of the link and d_i represents the distance between the end-point of the link $\boldsymbol{r_i}$ and the wall, see also Fig. 4.

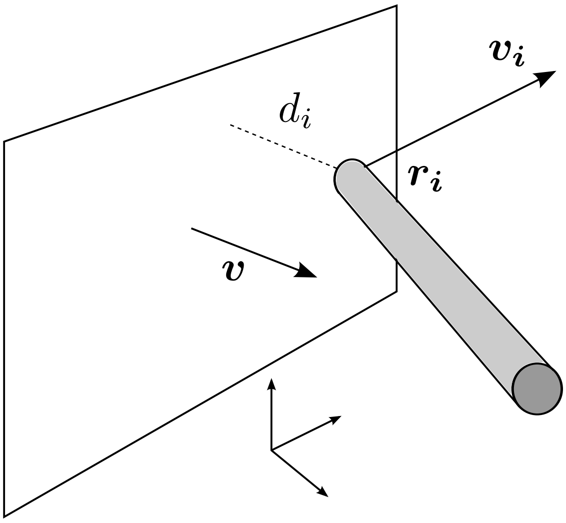


Figure 4: Virtual wall delimiting the collaborative workspace and quantities adopted to define the corresponding constraints.

Another important aspect to consider is the limited workspace of the robot. More in particular, a constraint can be defined in order to ensure that at any time the robot will be not completely stretched. To this end, the following constraint can be adopted:

$$2\Delta T\left(\boldsymbol{r_n}-\boldsymbol{r_0}\right)^T\boldsymbol{J_n}\dot{\boldsymbol{q}}\leq\max\left(0,r^2-\left\|\boldsymbol{r_n}-\boldsymbol{r_0}\right\|^2\right)$$

where $\boldsymbol{r_n}$ and $\boldsymbol{J_n}$ represent the position of the end point and the corresponding Jacobian, respectively, $\boldsymbol{r_0}$ is the position of the base of the robot, while r is the maximum reach of the robot, see Fig. 5.

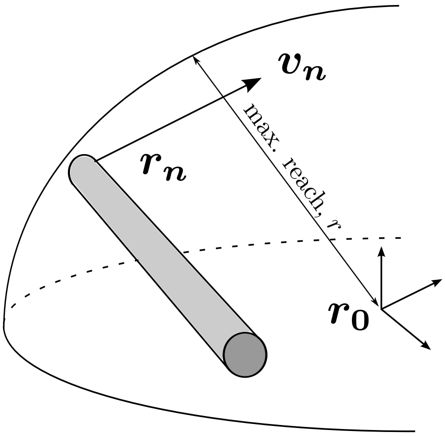


Figure 5: Virtual hemisphere delimiting the robot workspace and quantities adopted to define the corresponding constraints.

All the aforementioned constraints, together with the safety ones described in the previous Section, turn out to be linear with respect to the joint velocity vector $\dot{\boldsymbol{q}}$ and can be collected and written in a more compact way as follows:

$$\Delta T\boldsymbol{A}\dot{\boldsymbol{q}}\leq\boldsymbol{b} \tag{4}$$

where $\dot{\boldsymbol{q}}$ represents the velocity the manipulator can maintain for ΔT seconds, provided that it will be able to stop immediately at the end of the same time interval. In order to account for the braking time T_b of the manipulator, which is typically specified, in terms of each axis, as Category 1 Safety Stop [29], we can decompose ΔT in $\Delta T = T_b + \tilde{\Delta T}$, where $\tilde{\Delta T}$ now represents the time interval during which the manipulator can travel at speed $\dot{\boldsymbol{q}}$, while being still able to halt, with a safety monitored stop, without colliding with an obstacle or violating the constraints.

3.1. Geometric interpretation of matrix $\boldsymbol{A}$

The geometric interpretation of (4), and in particular of matrix $\boldsymbol{A}$, can be used to provide relevant information about possible directions to avoid because of the presence of an obstacle or, in general, because of the boundary of a certain constraint. In particular, the structure of matrix $\boldsymbol{A}$ can be exploited resorting to its singular value decomposition (SVD). SVD is expressed through matrices $\boldsymbol{N}$ and $\boldsymbol{\Lambda}$, such that $\boldsymbol{A}^T\boldsymbol{A}=\boldsymbol{N}\boldsymbol{\Lambda}\boldsymbol{N}^T$ where $\boldsymbol{N}$ contains a set of orthogonal vectors with unitary norm, so that every column of $\boldsymbol{N}$ represents a direction in joint space. The matrix $\boldsymbol{N}^*=\begin{bmatrix}-\boldsymbol{N} & \boldsymbol{N}\end{bmatrix}$ contains, within its columns, the directions returned by the eigenvalue decomposition of $\boldsymbol{A}^T\boldsymbol{A}$ accounted with both positive and negative sign. Columns of $\boldsymbol{N}^*$ thus represent specific directions in joint speed space and will be consequently called *Main Directions*[2]. Figure 6 shows the relationships between the introduced quantities for a two-DOF planar robot. The quantity

$$\dot{\psi}_i^{max}=\max_{\dot{\psi}_i}\dot{\psi}_i \text{ s.t. } \dot{\psi}_i\Delta T\boldsymbol{A}\boldsymbol{n_i}\leq\boldsymbol{b},\boldsymbol{n_i}\in\operatorname{col}\left(\boldsymbol{N^*}\right) \tag{5}$$

[2] Consider a unit velocity $\dot{\boldsymbol{q}}$, $\|\dot{\boldsymbol{q}}\|=1$. The direction in the joint space which maximises the product $\boldsymbol{A}\dot{\boldsymbol{q}}$ corresponds to the eigenvector of $\boldsymbol{A}^T\boldsymbol{A}$ associated to the maximum eigenvalue. In fact, it is well known that $\|\boldsymbol{A}\|=\sup_{\|\dot{\boldsymbol{q}}\|=1}\|\boldsymbol{A}\dot{\boldsymbol{q}}\|=\sqrt{\rho\left(\boldsymbol{A}^T\boldsymbol{A}\right)}=\sqrt{\max_i\lambda_i}$, where the spectral radius $\rho\left(\cdot\right)$ corresponds to the largest eigenvalue λ_i of $\boldsymbol{A}^T\boldsymbol{A}$.

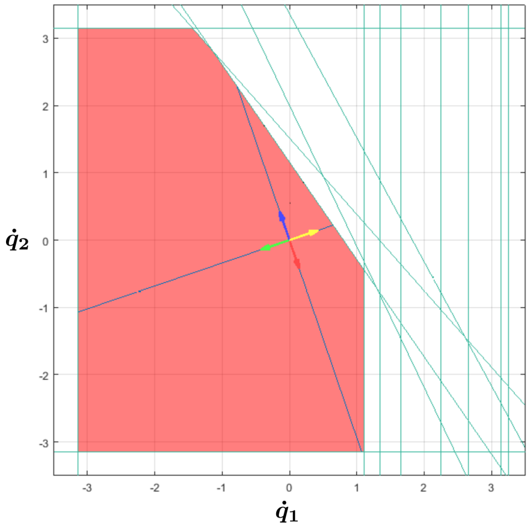


Figure 6: Constraints for a planar robot. The coloured area represents the admissible velocities in the joint space $\dot{\boldsymbol{q}}$ which are consistent with the constraints in (4). The four arrows represent the Main Directions.

represents the maximum speed that the robot can afford in the corresponding Main Direction $\boldsymbol{n_i}$ due to the presence of constraints. The manipulator can maintain the joint velocity $\dot{\boldsymbol{q}} = \dot{\psi}_i^{max}\boldsymbol{n_i}$ for ΔT seconds without violating any of the constraints in (4). Therefore the quantity $\Delta \boldsymbol{q_i^{max}} = \dot{\psi}_i^{max}\boldsymbol{n_i}\tilde{\Delta T}$, which now accounts for a generically non null braking time T_b, represents the maximum joint displacement along the Main Direction $\boldsymbol{n_i}$ which guarantees a collision free motion.

Using the direct kinematic mapping $\boldsymbol{x} = \boldsymbol{f}(\boldsymbol{q})$ from the joint space to the operational space, one can express the previously computed joint displacement in terms of linear displacement of the end-effector, i.e.

$$\boldsymbol{\Delta x_i^{max}} = \boldsymbol{J}\Delta \boldsymbol{q_i^{max}} = \boldsymbol{J}\dot{\psi}_i^{max}\boldsymbol{n_i}\tilde{\Delta T}$$

where $\boldsymbol{J}$ represents the linear Jacobian of the manipulator. In other words, the quantity

$$\boldsymbol{x_{i,\beta}} = \boldsymbol{f}(\boldsymbol{q}) + \beta_i \boldsymbol{\Delta x_i^{max}}$$

represents a set of candidate waypoints which are reachable from the current configuration $\boldsymbol{q}$ in $\tilde{\Delta T}$ seconds, without violating the constraints in (4), while $\beta_i \in [0, 1]$ represents a scaling factor.

3.2. Selection of scaling factor β and of the optimal waypoint

As the waypoints calculated in the previous step do not account for the position of the goal of the robot during a certain motion, it is possible that the alternative path including one of the generated waypoints introduces a significant overshoot with respect to the target position. In order to limit this effect, the scaling factor β_i can be used to avoid this effect, by simply setting

$$\beta_i = \min\left(1, \frac{\|\boldsymbol{x}^{trg} - \boldsymbol{f}(\boldsymbol{q})\|}{\boldsymbol{\Delta x_i^{max}}}\right)$$

This way the distance of the robot from the target once the waypoint is reached will not be higher than the actual distance $\|\boldsymbol{x}^{trg} - \boldsymbol{f}(\boldsymbol{q})\|$.

For the same reason, waypoints that are opposite with respect to the direction of the target are neglected. In terms of scaling factor, we force β_i to be zero when $(\boldsymbol{x}^{trg} - \boldsymbol{f}(\boldsymbol{q}))^T \boldsymbol{\Delta x_i^{max}} < 0$. The selection of the scaling factor β_i is then performed as follows

$$\beta_i = \begin{cases} 0, \text{if } \left(\boldsymbol{x}^{trg} - \boldsymbol{f}(\boldsymbol{q})\right)^T \boldsymbol{\Delta x_i^{max}} < 0 \\ \min\left(1, \dfrac{\|\boldsymbol{x}^{trg} - \boldsymbol{f}(\boldsymbol{q})\|}{\boldsymbol{\Delta x_i^{max}}}\right), \text{elsewhere} \end{cases}$$

This situation is depicted in Fig. 7.

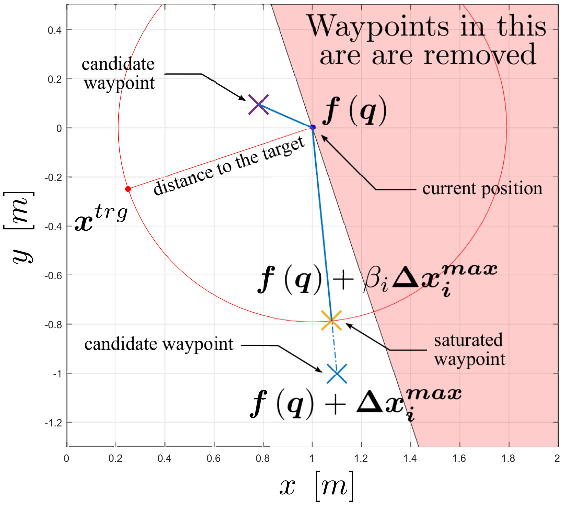


Figure 7: Pruning and saturation of candidate waypoints depending on the target position $\boldsymbol{x}^{trg}$ to be reached by the robot.

The next operation to be performed consists in the selection of the optimal waypoint among those generated, and possibly modified to remove the overshoots. At current stage, the candidate waypoints are those with $\beta_i > 0$, i.e. those not discarded because opposite to the current direction of motion, and the algorithm has to select the best one among this set. To this end, the waypoint that minimises the overall traversing time T_i^{trg} is selected as the optimal waypoint, i.e. the one that minimises

$$T_i^{trg} = \frac{\left\|\boldsymbol{x_{i,\beta}} - \boldsymbol{f}\left(\boldsymbol{q}\right)\right\|}{\dot{\psi}_i^{max}\left\|\boldsymbol{J}\boldsymbol{n_i}\right\|} + \frac{\left\|\boldsymbol{x}^{trg} - \boldsymbol{x_{i,\beta}}\right\|}{\dot{\psi}_i^{max}\left\|\boldsymbol{J}\boldsymbol{n_i}\right\|}$$

where the first term represents the time to reach the waypoint $\boldsymbol{x_{i,\beta}}$, while the second term is the time to reach the target $\boldsymbol{x}^{trg}$ from the waypoint. It is important to the remind that all the original waypoints, i.e. those obtained prior to the possible saturation, are expected to be reached in $\tilde{\Delta T}$ seconds, therefore the two quantities in the calculation of T_i^{trg} can be further arranged as follows:

$$\begin{aligned} T_i^{trg} &= \frac{\beta_i\left\|\boldsymbol{\Delta x_i^{max}}\right\|}{\dot{\psi}_i^{max}\left\|\boldsymbol{J}\boldsymbol{n_i}\right\|} + \frac{\left\|\boldsymbol{x}^{trg} - \boldsymbol{x_{i,\beta}}\right\|}{\dot{\psi}_i^{max}\left\|\boldsymbol{J}\boldsymbol{n_i}\right\|} = \\ &= \tilde{\Delta T}\left[\beta_i + \frac{\left\|\boldsymbol{x}^{trg} - \boldsymbol{x_{i,\beta}}\right\|}{\left\|\boldsymbol{\Delta x_i^{max}}\right\|}\right] \end{aligned} \tag{6}$$

Finally, the optimal waypoint $\boldsymbol{x_{i^{opt},\beta}}$ can be selected as

$$i^{opt} = \arg\min_i \left[\beta_i + \frac{\left\|\boldsymbol{x}^{trg} - \boldsymbol{x_{i,\beta}}\right\|}{\left\|\boldsymbol{\Delta x_i^{max}}\right\|}\right]$$

The cost function adopted accounts for both the distance of the robot to the target (numerator) and the distance to the obstacle (denominator), as the waypoint allows the robot to travel as much as possible in the corresponding direction without hitting the obstacles.
Finally, when the considered waypoint $\boldsymbol{x_{i,\beta}}$ is too far from the desired target position $\boldsymbol{x}^{trg}$, it is convenient to consider an additional waypoint $\boldsymbol{x_{i,\beta}}'$ to be reached before the target. To this end, the following additional waypoint is considered

$$\boldsymbol{x_{i,\beta}}' = \boldsymbol{x_{i,\beta}} + \alpha_i\left(\boldsymbol{x}^{trg} - \boldsymbol{f}\left(\boldsymbol{q}\right)\right)$$

only when $\alpha_i > 0$, where

$$\alpha_i = 1 - 2\frac{\beta_i\left\|\boldsymbol{\Delta x_i^{max}}\right\|}{\left\|\boldsymbol{x}^{trg} - \boldsymbol{f}\left(\boldsymbol{q}\right)\right\|}$$

In this case, the cost function in (6) is modified as follows to account for the additional waypoint:

$$T_i^{trg} = \tilde{\Delta T}\left[\beta_i + \frac{\left\|\boldsymbol{x}^{trg} - \boldsymbol{x_{i,\beta}}'\right\| + \left\|\boldsymbol{x_{i,\beta}}' - \boldsymbol{x_{i,\beta}}\right\|}{\left\|\boldsymbol{\Delta x_i^{max}}\right\|}\right]$$

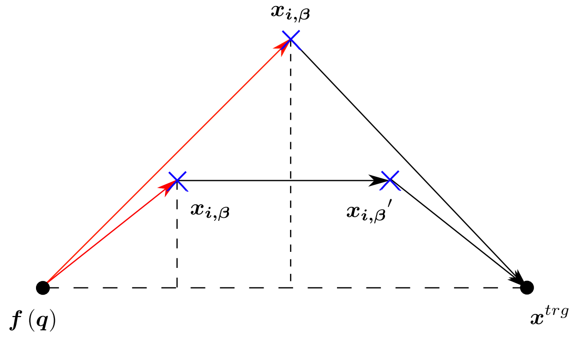

$\boldsymbol{x_{i,\beta}}$
$\boldsymbol{x_{i,\beta}}$
$\boldsymbol{x_{i,\beta}}'$
$\boldsymbol{f}\left(\boldsymbol{q}\right)$
$\boldsymbol{x}^{trg}$

Figure 8: Selection of waypoint $\boldsymbol{x_{i,\beta}}$ and additional waypoint $\boldsymbol{x_{i,\beta}}'$ in case of significant distance to the target position $\boldsymbol{x}^{trg}$.

and correspondingly, the optimisation procedure can be rewritten as

$$i^{opt} = \arg\min_i \left[\beta_i + \frac{\left\|\boldsymbol{x}^{trg} - \boldsymbol{x_{i,\beta}}'\right\| + \left\|\boldsymbol{x_{i,\beta}}' - \boldsymbol{x_{i,\beta}}\right\|}{\left\|\boldsymbol{\Delta x_i^{max}}\right\|}\right]$$

The adoption of the additional waypoint will result in a trapezoidal path from the current position to the goal, through the two waypoints $\boldsymbol{x_{i,\beta}}$ and $\boldsymbol{x_{i,\beta}}'$. The corresponding situation is shown in Fig. 8.

3.3. Overall algorithm

Algorithm 1 reports a pseudo-code of a possible implementation of the overall strategy for generating and selecting the optimal waypoint and the corresponding dodging path. The algorithm is fed with the current position of the robot in the joint space $\boldsymbol{q}$, its target position in the Cartesian space $\boldsymbol{x}^{trg}$, the matrices characterising the constraints $\boldsymbol{A}$ and $\boldsymbol{b}$, and the time horizons ΔT and $\tilde{\Delta T}$ (or alternative the time horizon ΔT and the braking time T_b). The output of the algorithm is in terms of the optimal waypoint $\boldsymbol{x_{i^{opt},\beta}}$, the corresponding linear velocity $\left\|\boldsymbol{J}\dot{\psi}_{i^{opt}}^{max}\boldsymbol{n_i}\right\|$ the robot should adopt to reach the waypoint, as well as the additional waypoint $\boldsymbol{x_{i^{opt},\beta}}'$, in case of a trapezoidal dodging maneouver.

An overall picture of the developed method that is implemented in the Smart Robots device is reported in Fig. 10. The sensor, embedded in the device, extracts the silhouette of the operator in terms of skeletal points which are then processed to return the human swept volumes (see Fig. 3 and [28]). The swept volumes are then regarded as obstacles and the corresponding matrix $\boldsymbol{E}$ in (3) is

Algorithm 1

```
Input: $\boldsymbol{q}, \boldsymbol{x}^{trg}, \boldsymbol{A}, \boldsymbol{b}, \Delta T, \tilde{\Delta T}$
Output: $\boldsymbol{x}_{\boldsymbol{i^{opt}},\boldsymbol{\beta}}, \boldsymbol{x}_{\boldsymbol{i^{opt}},\boldsymbol{\beta}}{}', \left\| \boldsymbol{J}\dot{\psi}_{i^{opt}}^{max} \boldsymbol{n}_{\boldsymbol{i^{opt}}} \right\|$
compute $\boldsymbol{N}, \boldsymbol{\Lambda}$ such that $\boldsymbol{A}^T\boldsymbol{A} = \boldsymbol{N}\boldsymbol{\Lambda}\boldsymbol{N}^T$;
$\boldsymbol{N}^* \leftarrow \begin{bmatrix} \boldsymbol{N} & -\boldsymbol{N} \end{bmatrix}$;
for all $\boldsymbol{n_i} \in \mathrm{col}\,(\boldsymbol{N}^*)$ do
    $\dot{\psi}_i^{max} \leftarrow +\infty$;
    for all $\boldsymbol{a_j}^T \in \mathrm{row}\,(\boldsymbol{A})$ do
        if $\boldsymbol{a_j}^T\boldsymbol{n_i} > 0$ then
            $\dot{\psi}_i^{max} \leftarrow \min\left(\dot{\psi}_i^{max}, \dfrac{b_j}{\Delta T \boldsymbol{a_j}^T \boldsymbol{n_i}}\right)$;
    $\boldsymbol{\Delta x_i^{max}} \leftarrow \boldsymbol{J}\dot{\psi}_i^{max}\tilde{\Delta T}$;                 ▷ candidate waypoint
    if $\left(\boldsymbol{x}^{trg} - \boldsymbol{f}\,(\boldsymbol{q})\right)^T \boldsymbol{\Delta x_i^{max}} > 0$ then
        $\beta_i \leftarrow \min\left(1, \dfrac{\left\|\boldsymbol{x}^{trg} - \boldsymbol{f}\,(\boldsymbol{q})\right\|}{\boldsymbol{\Delta x_i^{max}}}\right)$;               ▷ valid waypoint
    else
        $\beta_i \leftarrow 0$;                                  ▷ invalid waypoint: removed
    if $\beta_i = 0$ then
        $\boldsymbol{x_{i,\beta}} \leftarrow \emptyset$;
        $\boldsymbol{x_{i,\beta}}' \leftarrow \emptyset$;
        $T_i^{trg} \leftarrow +\infty$;
    else
        $\boldsymbol{x_{i,\beta}} \leftarrow \boldsymbol{f}\,(\boldsymbol{q}) + \beta_i \boldsymbol{\Delta x_i^{max}}$;
        $\alpha_i \leftarrow 1 - 2\dfrac{\beta_i \left\|\boldsymbol{\Delta x_i^{max}}\right\|}{\left\|\boldsymbol{x}^{trg} - \boldsymbol{f}\,(\boldsymbol{q})\right\|}$;
        if $\alpha_i > 0$ then
            $\boldsymbol{x_{i,\beta}}' \leftarrow \emptyset$;                          ▷ triangular path
            $T_i^{trg} \leftarrow \tilde{\Delta T}\left[\beta_i + \dfrac{\left\|\boldsymbol{x}^{trg} - \boldsymbol{x_{i,\beta}}\right\|}{\left\|\boldsymbol{\Delta x_i^{max}}\right\|}\right]$;
        else
            $\boldsymbol{x_{i,\beta}}' \leftarrow \boldsymbol{x_{i,\beta}} + \alpha_i\left(\boldsymbol{x}^{trg} - \boldsymbol{f}\,(\boldsymbol{q})\right)$;          ▷ trapezoidal path
            $T_i^{trg} \leftarrow \tilde{\Delta T}\left[\beta_i + \dfrac{\left\|\boldsymbol{x}^{trg} - \boldsymbol{x_{i,\beta}}'\right\| + \left\|\boldsymbol{x_{i,\beta}}' - \boldsymbol{x_{i,\beta}}\right\|}{\left\|\boldsymbol{\Delta x_i^{max}}\right\|}\right]$;
$i^{opt} \leftarrow \arg\min_i \left(T_i^{trg}\right)$;                     ▷ selecting optimal waypoint
return $\boldsymbol{x}_{\boldsymbol{i^{opt}},\boldsymbol{\beta}}, \boldsymbol{x}_{\boldsymbol{i^{opt}},\boldsymbol{\beta}}{}', \left\| \boldsymbol{J}\dot{\psi}_{i^{opt}}^{max} \boldsymbol{n}_{\boldsymbol{i^{opt}}} \right\|$;
```

computed. Finally, the scaling factor δ is evaluated, while Algorithm 1 is executed to compute the candidate waypoint.

4. Implementation and experiments

The algorithm detailed in the previous Section has been coded within a dedicated CPU within the Smart Robots device, see Fig. 12(a). The robot and the device communicate with a standard Ethernet connection and all the calculations are synchronous with the communication cycle, which runs approximately every 30 *ms* as shown in Fig. 9. In particular the robot sends to the device the current position, in terms of joint angles $\boldsymbol{q}$ and the corresponding motion target $\boldsymbol{x}^{trg}$. Based on the full silhouette of the human operator and on the prediction of his/her occupancy, see Fig. 3, the Smart Robots device computes first the constraint matrix in (3), and then evaluates

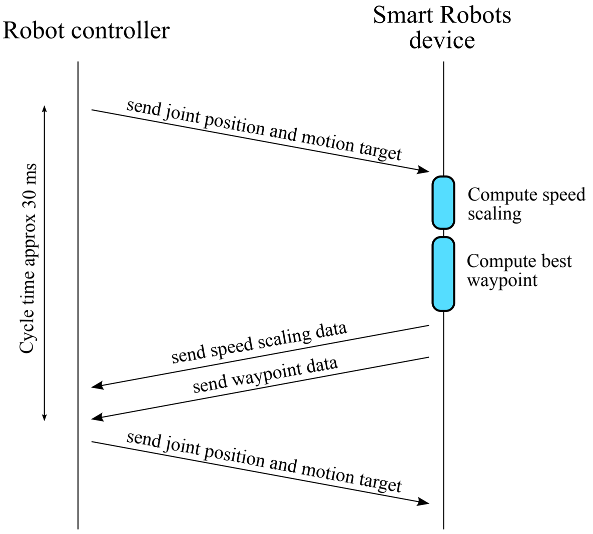


Figure 9: UML sequence diagram describing the flow of information exchanged between the robot controller and the Smart Robots device, the cycle is repeated every 30 ms.

the scaling factor δ, as well as the candidate waypoint(s) $\boldsymbol{x_{i^{opt}},_{\beta}}, \boldsymbol{x_{i^{opt}},_{\beta}}'$ and the corresponding velocity $\left\| \boldsymbol{J}\dot{\psi}_{i^{opt}}^{max}\boldsymbol{n_{i^{opt}}} \right\|$. The corresponding UML sequence diagram is shown in Fig. 9.

The controller of the robot, then evaluates the need for an evasive manoeuvre through an algorithm coded in its native programming language. The behaviour implemented within the robot controller is sketched in Fig. 11, while the corresponding pseudo-code is reported in the following:

```
def SmartMoveL(x_trg, v_max, a_max)
  EnableSpeedScaling()
  while delta > delta_threshold
    MoveL(x_trg, v_max, a_max)
  else
    AbortCurrentMove()
    DisableSpeedScaling()
    MoveL(waypoint_1, v_waypoint, a_max)
    if trapezoidal_path
      SmartMoveL(waypoint_2, v_max, a_max)
    end
    SmartMoveL(x_trg, v_max, a_max)
  end
end
```

The task that has been used during the experiments consists in a simple pick and place between two points SOURCE and TARGET. The robot has to move a bottle from a SOURCE position A to a TARGET position B, see Fig. 12. The maximum velocity of the robot has been set to 250 mm/s and the nominal cycle time is of 7.2 s.

Two type of experiments have been run for the proposed benchmark application. During the first set of experiments, the robot operates without the Smart Robots device. The maximum tolerable force, which is typically decided during the risk assessment, has been set to its default value of 150 N. During each cycle an operator place his/her hand along the path of the robot, causing a Category 0 stop triggered by the collision detection algorithm embedded in the robot controller. The operator is then responsible for acknowledging the safety exception on the teach pendant, has to move back the robot on the original path, and to manually resume the operation. A collision is then responsible for an increased cycle time that highly depends on availability of the teaching device at hand, as well as on how much the operator is familiar with the system.

During the second set of experiments, in turn, the Smart Robots device has been activated, allowing the robot to avoid the contact with the human by adopting a dodging manoeuvre. The same benchmark pick and place task has been adopted. In this case, when a dodging manoeuvre is successfully performed, the robot can continue its task without triggering a Category 0 stop. When this is not possible, the speed modulation along the path allows the robot to reduce its speed (and consistently its kinetic energy) to possibly avoid, or at least mitigate, the effect of collisions. During the several runs of the experiment, no collision has been registered, and the evasive manoeuvre has been successfully completed in all the runs. Figure 13 reports an example of the avoidance strategy, while Fig. 14 shows some snapshots related to the evasive manoeuvre performed by the robot during one of the experiments.

In order to evaluate the benefits of the additional safety-related layers introduced in this paper (see Fig. 1), the cycle time has been used as a Key Performance Indicator (KPI). In fact, while the robot can be still considered as safe in the considered application, the adoption of control strategies to avoid stopping its motion are more interesting from a productivity point of view. The differences, in terms of cycle times, between the two strategies are reported in Fig. 15. During the first set of experiments, the average cycle time has been of 13.11 s, with a stan-

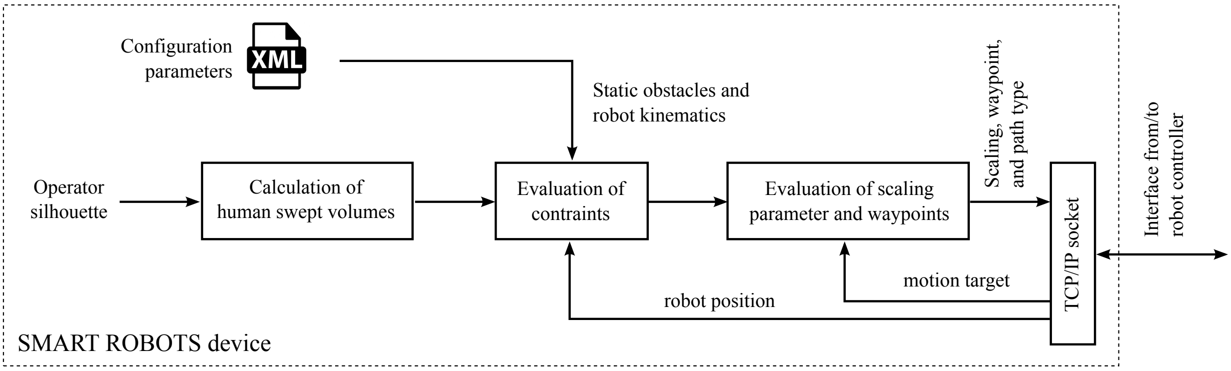


Figure 10: Block diagram of the developed algorithm and its interface from and to the robot.

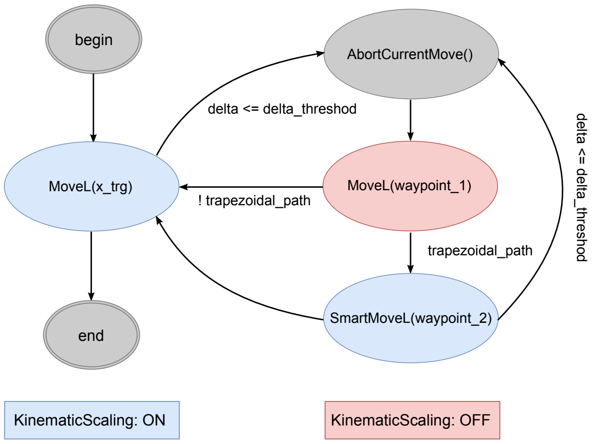


Figure 11: State machine representation of the SmartMoveL instruction. Notice that this newly defined instruction can be called recursively during a point to point motion.

dard deviation of 0.90 s (on average 82% more than the nominal case). In turn, during the second set of experiments, which include the avoidance strategy developed in this work, the cycle time was of 9.47 s with a standard deviation of 1.29 s, thus only 32% more than in the nominal execution. It is then clear that the avoidance strategy developed and described in this work outperforms, in terms of a lower cycle time, the standard safety countermeasures implemented within the robot controller (Wilcoxon single tailed test, $p < 0.0001$).

5. Conclusions

State of the art workspace monitoring devices can be used to enhance the robot perception of its surrounding environment. The gathered information can be used to suitably control the motion of the robot with respect to the presence of one or more human operators. Within modern collaborative robots, safety issues are typically handled with the PFL clause which severely limits the velocity of the payload capability of such robots. In this scenario, it is paramount to maintain a constant level of productivity, thus avoiding the robot to stop because of (harmless) collisions with the operator. In this work, we have introduced a method which evaluates alternative execution paths by inserting waypoints along the motion, so to avoid unintended contacts with the operator and to keep the velocity of the manipulator as much as possible close to its maximum value. The approach, which can be ported on all robot controllers with just an Ethernet connection, has been verified within an experimental benchmark application, showing the capability of the developed method to significantly reduce the cycle time eliminating the idle time of the robot and of the human operator.

References

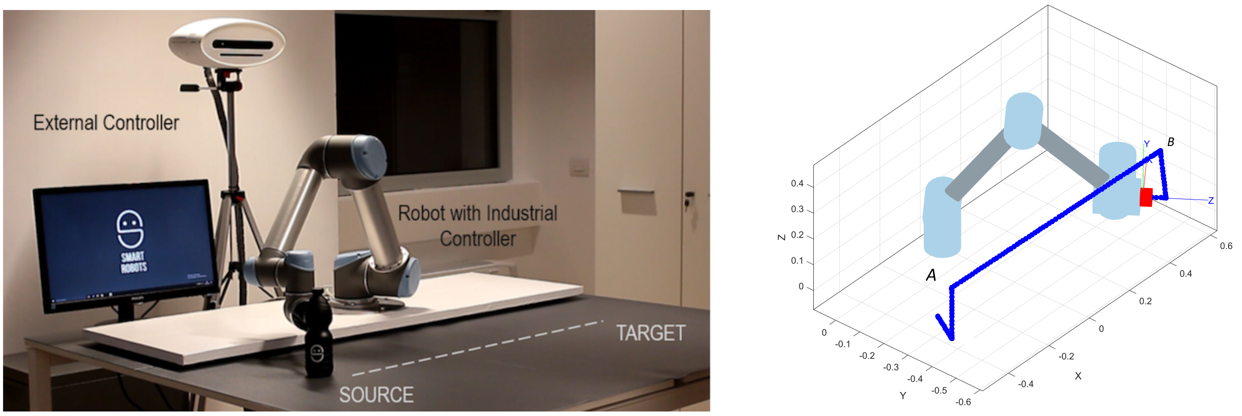
[1] A. M. Zanchettin, P. Rocco, Method and device for controlling the motion of one or more collaborative robots, patent, PCT/IB2017/052932 (2017).

[2] A. Ajoudani, A. M. Zanchettin, S. Ivaldi, A. Albu-Schäffer, K. Kosuge, O. Khatib, Progress and prospects of the human–robot collaboration, Autonomous Robots (2017) 1–19.

[3] V. Villani, F. Pini, F. Leali, C. Secchi, Survey on human–robot collaboration in industrial settings: Safety, intuitive interfaces and applications, Mechatronics (2018, online).

[4] ISO TC184/SC2, ISO 10218 - Robots for industrial environments – Safety requirements – Part 1: Robot (2011).

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



(a) Layout of the experiments for the pick and place task

(b) 3D representation of the layout and of the nominal pick and place path

Figure 12: Layout of the experimental setup comprising a Universal Robots UR5 robot and a Smart Robots device.

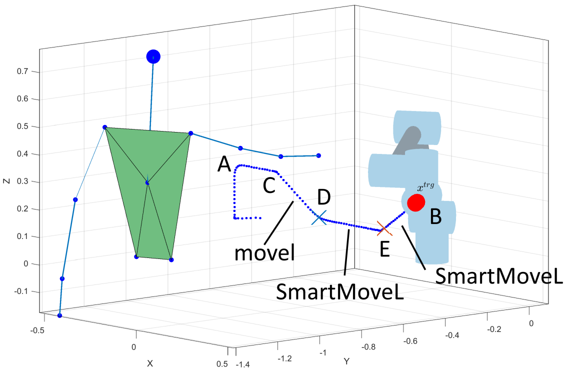


Figure 13: Example of avoidance strategy. A standard SmartMoveL is executed between point A and point B. During the path, and in particular in position C, the Smart Robots device triggers an evasive manoeuvre towards point D which is executed by the robot with a standard MoveL, then since the distance to the target point B is still high, the trapezoidal profile is used and a SmartMoveL is performed towards E and then towards the final destination B.

[6] D. Kulić, E. A. Croft, Real-time safety for human–robot interaction, Robotics and Autonomous Systems 54 (1) (2006) 1–12.

[7] P. Fiorini, Z. Shiller, Motion planning in dynamic environments using velocity obstacles, The International Journal of Robotics Research 17 (7) (1998) 760–772.

[8] T. Fraichard, H. Asama, Inevitable collision statesa step towards safer robots?, Advanced Robotics 18 (10) (2004) 1001–1024.

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

[10] M. P. Polverini, A. M. Zanchettin, P. Rocco, A computationally efficient safety assessment for collaborative robotics applications, Robotics and Computer-Integrated Manufacturing 46 (2017) 25–37.

[11] R. Rossi, M. P. Polverini, A. M. Zanchettin, P. Rocco, A pre-collision control strategy for human-robot interaction based on dissipated energy in potential inelastic impacts, in: Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on, IEEE, 2015, pp. 26–31.

[12] O. Khatib, Real-time obstacle avoidance for manipulators and mobile robots, in: Autonomous robot vehicles, Springer, 1986, pp. 396–404.

[13] O. Brock, O. Khatib, Elastic strips: A framework for integrated planning and execution, in: Experimental Robotics VI, Springer, 2000, pp. 329–338.

[14] F. Flacco, T. Kröger, A. De Luca, O. Khatib, A depth space approach to human-robot collision avoidance, in: Robotics and Automation (ICRA), 2012 IEEE International Conference on, IEEE, 2012, pp. 338–345.

[15] S. Pellegrinelli, F. L. Moro, N. Pedrocchi, L. M. Tosatti, T. Tolio, A probabilistic approach to workspace sharing for human–robot cooperation in assembly tasks, CIRP Annals-Manufacturing Technology 65 (1) (2016) 57–60.

[16] S. Haddadin, H. Urbanek, S. Parusel, D. Burschka, J. Roßmann, A. Albu-Schäffer, G. Hirzinger, Real-time reactive motion generation based on variable attractor dynamics and shaped velocities, in: Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on, IEEE, 2010, pp. 3109–3116.

[17] N. Pedrocchi, F. Vicentini, M. Matteo, L. M. Tosatti, Safe human-robot cooperation in an industrial environment, International Journal of Advanced Robotic Systems 10 (1) (2013) 27.

[18] Y. Wang, X. Ye, Y. Yang, W. Zhang, Collision-free trajectory planning in human-robot interaction through hand movement prediction from vision, in: Humanoid Robotics (Humanoids), 2017 IEEE-RAS 17th International Conference on, IEEE, 2017, pp. 305–310.

[19] D. Hughes, J. Lammie, N. Correll, A robotic skin for collision avoidance and affective touch recognition,

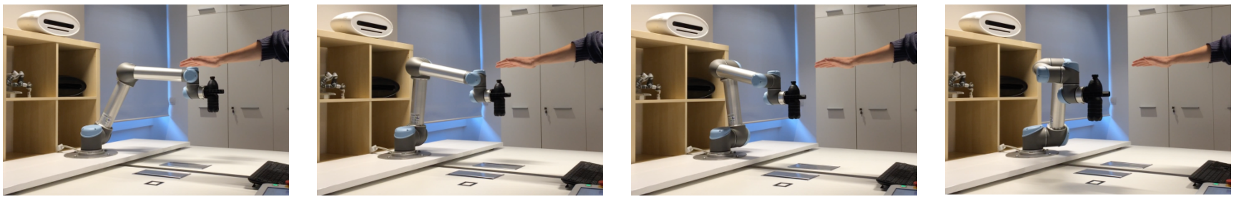


Figure 14: Snapshots of one of the experiments showing the evasive movement of the robot to avoid the hand of the operator.

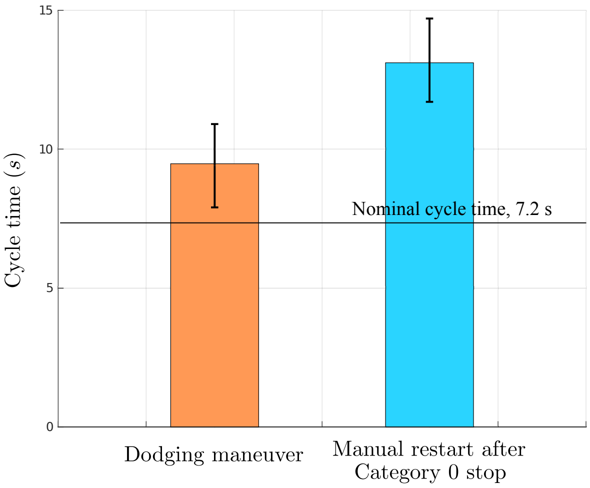


Figure 15: Registered cycle times during the two sets of experiments (the nominal cycle time is of 7.2 seconds).

IEEE Robotics and Automation Letters 3 (3) (2018) 1386–1393.

[20] Y. Wang, Y. Sheng, J. Wang, W. Zhang, Optimal collision-free robot trajectory generation based on time series prediction of human motion, IEEE Robotics and Automation Letters 3 (1) (2018) 226–233.

[21] D. Han, H. Nie, J. Chen, M. Chen, Dynamic obstacle avoidance for manipulators using distance calculation and discrete detection, Robotics and Computer-Integrated Manufacturing 49 (2018) 98–104.

[22] J.-H. Chen, K.-T. Song, Collision-free motion planning for human-robot collaborative safety under cartesian constraint, in: 2018 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2018, pp. 1–7.

[23] J. A. Marvel, Performance metrics of speed and separation monitoring in shared workspaces, IEEE Transactions on Automation Science and Engineering 10 (2) (2013) 405–414.

[24] J. A. Marvel, R. Norcross, Implementing speed and separation monitoring in collaborative robot workcells, Robotics and computer-integrated manufacturing 44 (2017) 144–155.

[25] P. Tsarouchi, S. Makris, G. Chryssolouris, Human–robot interaction review and challenges on task planning and programming, International Journal of Computer Integrated Manufacturing 29 (8) (2016) 916–931.

[26] A. M. Zanchettin, N. M. Ceriani, P. Rocco, H. Ding, B. Matthias, Safety in human-robot collaborative manufacturing environments: Metrics and control, IEEE Transactions on Automation Science and Engineering 13 (2) (2016) 882–893.

[27] ANSI/RIA R15 06 1999 ”Safety requirements for industrial robots and robot systems”.

[28] M. Ragaglia, A. M. Zanchettin, P. Rocco, Safety-aware trajectory scaling for human-robot collaboration with prediction of human occupancy, in: Advanced Robotics (ICAR), 2015 International Conference on, IEEE, 2015, pp. 85–90.

[29] IEC, EN 60204 - ”Safety of machinery. Electrical equipment of machines. General requirements” (2006).