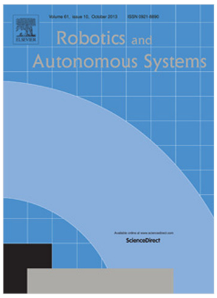
Journal Pre-proof



Safety barrier functions and multi-camera tracking for human-robot shared environment

Federica Ferraguti, Chiara Talignani Landi, Silvia Costi,
Marcello Bonfè, Saverio Farsoni, Cristian Secchi, Cesare Fantuzzi

PII: S0921-8890(19)30642-6
DOI: https://doi.org/10.1016/j.robot.2019.103388
Reference: ROBOT 103388

To appear in: *Robotics and Autonomous Systems*

Received date : 8 August 2019
Revised date : 21 October 2019
Accepted date : 25 November 2019

Please cite this article as: F. Ferraguti, C.T. Landi, S. Costi et al., Safety barrier functions and multi-camera tracking for human-robot shared environment, *Robotics and Autonomous Systems* (2019), doi: https://doi.org/10.1016/j.robot.2019.103388.

Safety Barrier Functions and Multi-Camera Tracking for Human-Robot Shared Environment

Federica Ferraguti[a,*], Chiara Talignani Landi[a], Silvia Costi[a], Marcello Bonfè[b], Saverio Farsoni[b], Cristian Secchi[a], Cesare Fantuzzi[a]

[a] *Department of Science and Methods for Engineering, University of Modena and Reggio Emilia, Italy*
[b] *Engineering Department, University of Ferrara, Italy*

Abstract

A new vision in human-robot collaboration has allowed to place robots nearby human operators, working close to each other in industrial environments. As a consequence, human safety has become a dominant issue, together with production efficiency. In this paper we propose an optimization-based control algorithm that allows robots to avoid obstacles (like human operators) while minimizing the difference between the nominal acceleration input and the commanded one. Control Barrier Functions are exploited to build safety barriers around each robot link, to guarantee collision-free trajectories along the whole robot body. Human accelerations and velocities are computed by means of a bank of Kalman filters. To solve obstruction problems, two RGB-D cameras are used and the measured skeleton data are processed and merged using the mentioned bank of Kalman filters. The algorithm is implemented on an Universal Robots UR5 in order to validate the proposed approach.

Keywords: human-robot interaction, collision avoidance, control barrier function

[*] Corresponding author
Email addresses: federica.ferraguti@unimore.it (Federica Ferraguti), chiara.talignanilandi@unimore.it (Chiara Talignani Landi), silvia.costi@unimore.it (Silvia Costi), marcello.bonfe@unife.it (Marcello Bonfè), saverio.farsoni@unife.it (Saverio Farsoni), cristian.secchi@unimore.it (Cristian Secchi), cesare.fantuzzi@unimore.it (Cesare Fantuzzi)

1. Introduction

The raise of collaborative robotics has allowed to create shared environments where robots and humans work closely. In this scenario, the high level skills of humans can be merged with robot reliability and repeatability, to obtain flexible and efficient production lines.

In this context, safety becomes of paramount importance, due to the tight cooperation between human and machines. Prolong endeavors have been made to develop an intrinsic safety for robots [1]. This means that, independently from any kind of failure, malfunctioning or incorrect use, the robot is safe to humans. Obviously this intrinsic safety can never be completely reached, since robotic applications have to take into account production performances. One important aspect about intrinsic safety is how to detect and behave when unexpected collisions along the robot body happen, during the working cycle. Bicchi et al [2] design a joint-actuation mechanism based on variable impedance that allows to keep high performances while guaranteeing a limited level of injury risk. The authors demonstrated that low stiffness is required at high speed, and vice versa. Haddadin et al [3] present methodologies and experimental tests to demonstrate how the robot is able to detect collisions and to distinguish between unexpected collisions and intended cooperation. They also showed that switching from a position control to zero-gravity torque control (i.e. compliant mode) gives the operator the feeling of a safe cooperation. If robot velocities are low, external proximity/contact-force sensors [4] or robot skin (e.g. Fraunhofer IFF Tactile Sensor Systems [5]) can safely detect human touch. Collision detection can also be implemented on industrial robots. Authors in [6] propose a method for limiting the forces applied by an industrial manipulator when collisions occur. They avoid the use of external sensors and they rely on motor current signals. By exploiting time-invariant dynamic models and neural networks, they are able to predict the nominal current during the motion and detect collisions.

Usually, when collisions happen, robots are stopped or switched in a compliant mode. However, it can be convenient to generate alternative paths for the robot, preserving task performance and reducing robot downtime or restarts. To prevent collisions from happening, exteroceptive sensors can be added to the robotic system to enhance control algorithms. For example, 3D or RGB-D cameras (e.g. Microsoft Kinect, Asus Xtion, Pilz SafetyEYE [7]), laser scanners or light projectors [8] detect intrusions of people inside the robot working area. The definition of virtual safety surfaces during robot operations (e.g. Fanuc Dual Check safety [9]) or during manual guidance [10] allows to define safe working regions that the robot cannot cross.

To prevent robots from stopping, exteroceptive sensors can be used in conjunction with control strategies to implement collision avoidance. For this purpose, solutions based on simple robot speed reduction or more complex motion adaptations (e.v. based on repulsive potential fields associated to obstacles) are often used. Optimization-based control algorithms, instead, would allow to avoid to slow-down the robot or to get stuck in local minima of the potential fields. Wang et al. [11] adopted an optimization-based algorithm and introduce the role of safety barriers certificates to ensure collision free movements in multirobot systems. The basic idea is to modify the nominal controller by means of solving a quadratic problem online, to satisfy the safety constraints. The control barrier functions are used to guarantee the forward invariance of the set of safe states: i.e. if a system starts in a collision-free configuration, it remains in collision free configurations. Control Barrier Functions around the end-effector are used in [12] to avoid collisions with the environment.

However, in shared workspaces, the human can collide with the robot along the whole robot body, not only against the end-effector. For this reason, including the whole kinematic structure of the robot inside the safety constraints is of utmost importance. In this paper we propose an optimization-based collision avoidance algorithm based on the Control Barrier Functions (CBFs) presented in [12]. We extend the construction of barrier functions on each robot link, guaranteeing collision free movements along the whole robot body. The resulting optimal control action, computed online, is the best one that approximates the nominal control action still preserving human safety. Preliminar results of the proposed approach were presented in [13]. In this paper, we also consider that to compute the optimization constraints, human joint velocities and accelerations are needed. As a significant extension of [13], we now describe how to estimate these values exploiting a bank of Kalman filters, that processes skeleton joint positions and orientations coming from a multi-camera system.

Therefore, the main contributions of the paper are the following:

- Two cameras are used to track the human body, to avoid obstruction problems. Indeed, a bank of Kalman filters is used to merge data from two cameras, to reduce measurements noise, to solve loss of data due to obstructions and to estimate velocities and accelerations of human body parts.
- The proposed algorithm allows to avoid collisions considering the whole human body. Indeed, we evaluate the point on the human body that is the closest to the robot body. Since this evaluation is performed at each instant of time, the closest point on the human body changes dynamically, according to the human movement, and the robot is moved consequently.
- We propose an extended experimental session, by showing the parameters tuning and performances of the bank of Kalman filters, both when occlusions happen and in static cases, to filter measurement noises. Moreover we show the overall performances of the two-cameras optimization-based algorithm when the robot is in proximity of a human operator.

2. Related Works

Different approaches were presented in the literature to deal with human safety and collision avoidance in a human-robot shared workspace. A common solution is represented by a velocity reduction when the robot approaches the human. Ragaglia *et al.* [14] propose a real-time solution to evaluate human occupancy and safely scale the robot velocity. A 3D camera and simple human kinematic model are used to predict the space the human will occupy within the robot stopping time. The robot velocity is reduced, in order to avoid task interruptions. Zanchettin *et al.* [15] present an optimization-based and real-time algorithm where safety is regarded as a hard constraint to be satisfied. The objective is to guarantee safety while keeping high performance levels, by means of a velocity reduction. The main idea is that the safety distance is always greater than robot velocity multiplied by its braking time. This strategy is applied in the joint space, to avoid time consuming calculations in the Cartesian space. Salmi *et al.* [16] propose a Dynamic Safety System that reduces the velocity of an industrial robot exploiting a primary non-safe device and a secondary certified device. These devices create different allowed robot working areas depending on human position.

A velocity reduction is not always the best choice, especially when the robot workspace is wide enough to allow a modification in the robot path. Polverini *et al.* [17] apply the concept of kinetostatic safety field on a redundant robot manipulator, avoiding both self-collisions and human-robot collisions. The safety field considers a rigid body as a source of danger and it depends on the relative position and velocity between the rigid body and the obstacle. Moreover, the safety field depends on the shape and the size of the source of danger. The generated joint velocity guarantees a collision-free movement. Levratti *et al.* [18] apply the concept of Danger Field presented in [19] on a mobile robot, in order to avoid human operators in a tire workshop. Ferraguti *et al.* [20] exploit the use of virtual fixtures to implement collision avoidance in a teleoperated environment, while suggesting a preferred path direction. Virtual fixtures are like assistive forces obtained by summing attractive and repulsive potential fields. Haddadin *et al.* [21] propose a variable attractor dynamics to obtain a reactive collision avoidance, dealing with external forces and able to serve as an interpolator with arbitrary desired velocity profile. Using potentials can effectively provide motion replanning and collision avoidance, but the system can be stuck in local minima and the replanning is somehow unpredictable. All these works use the distance between the robot and the obstacle as the key element to define safety.

Different approaches are based on the definition of a safey index. For example, in [22] the authors propose the potential impact force as a safety evaluation factor. The safety index is then computed exploiting the relative distance and velocity between the human and the robot, robot inertia and stiffness. Kulić *et al.* [23] design a safe planning and control strategy to evaluate danger in real-time along the entire robot arm. The danger index is formulated as a non-linear impedance controller and it is used to move the robot in a safe region when potential collisions are detected. The inputs to the safety module are the following robot arm configuration (i.e. position and velocity), the current human configuration and the estimated human's level of intent.

Finally, optimization-based algorithms can be used to obtain the best correction to the nominal path, with respect to safety constraints. Lin *et al.* [24] propose a velocity-based collision avoidance method in human guidance scenario. In [25] the authors consider a more general scenario, where the motion input is chosen with respect to the Safe Set, solving an optimization problem. The definition of a safety index allows to avoid dangerous regions that may lead to collisions. In this case, the controller is able to bring the system back inside the safe region, but only after the actual violation of the safety constraint.

In most of the mentioned works, the sensors detecting human presence and motion are commercial 3D or RGB-D cameras. Even though human motion tracking itself and the development of novel sensing technologies for this purpose are relevant research issues (see [26] and [27] for a survey), that would deserve further investigations, such commercial solutions are generally sufficient to support experimental validation, at least in a laboratory setup, of the safety related robot control algorithms, which is also the focus of this paper. On the other hand, the integration and fusion of data from multiple sensors (e.g. dual RGB-D cameras [28] or an RGB-D camera and an inertial measurement unit [29]) or the use of different estimation methods for position and orientation (see [30]) are useful solutions to improve the reliability of human motion tracking data and, therefore, of the validation experiments themselves. In this context, Kalman filters and Particle filters are common solutions to the sensor and data fusion problem. Other methods (e.g. Sliding Mode Observers [31]), potentially providing robust estimations, are rarely used for sensor/data fusion and, especially, are currently unexplored in human motion tracking.

Differently from previously mentioned works, in this paper we propose an optimization-based collision avoidance algorithm that keeps the system state inside a safe region, without violating the safety constraint. Moreover, the proposed algorithm allows to choose the optimal path by computing the minimum correction with respect to the nominal path. The optimization method also guarantees that the robot successfully reaches the target, without being stuck in local minima. To build the constraints in the optimization problem we adopted the formulation of Control Barrier Functions. The construction of a barrier function around each robot link allows to guarantee collision-free trajectories of the entire robot body. No velocity reduction is needed, hence the robot can fully exploit its workspace, increasing efficiency. With regard to human tracking, we exploit multiple RGB-D cameras and a combination of Linear Kalman Filters and Extended Kalman Filters (respectively for position and orientation data processing) based on second order differential kinematic models, as proposed in [32], to avoid visual occlusions from invalidating the performance obtained with the proposed control method during the experiments.

3. Problem Statement

As a consequence of the introduction of collaborative robots and human-robot interaction technologies, the safety standards have been updated in order to address the new co-working scenarios. International ISO 10218-1/2 [33, 34] safety standards have identified specific applications and criteria where collaborative operations can occur. More recently, the technical specification ISO/TS 15066 [35] has been introduced to specify safety requirements for collaborative industrial robot systems and the work environment, and supplements the requirements and guidance on collaborative industrial robot operations given in ISO 10218-1 and ISO 10218-2. The safety standards ISO 10218-1/2 and the tech-

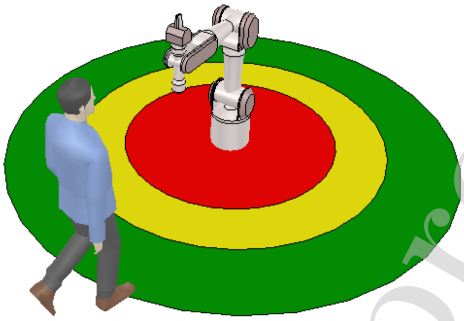


Figure 1: Safety zones for human-robot collaboration with Speed and Separation Monitoring technology.

nical specification ISO/TS 15066 describe four collaborative operating modes: Safety-rated Monitored Stop (SMS), Hand Guiding (HG), Speed and Separation Monitoring (SSM) and Power and Force Limiting (PFL). In particular, if a standard industrial robot is involved, the Speed and Separation Monitoring is typically used: in this modality, the speed and the trajectory of the robot are monitored and adjusted depending on the speed and position of the human operator into the shared workspace. The robot operates at full speed when the human moves in a *green zone*, at reduced speed in a *yellow zone* and it stops when the operator is in a *red zone* (Figure 1). The areas are inspected using scanners or vision systems to monitor the position of the human operator. However, the main disadvantage of this modality is that the robot is continuously stopped if the human access frequently the red zone, affecting the performance of the cooperative task.

In this paper we propose an initial step towards the preservation of the high performances typical of industrial robots, which are characterized by high speeds of execution. Indeed, we aim at avoiding stopping the robot or scaling its speed in the presence of the human operator in the shared workspace. The focus of the work is to create a safety aware human-robot shared environment where the robot executes its predefined task while the human operator is working nearby to collaborate at the same task or to perform a different operation. To prevent speed reduction or stopping the robot, we decided to exploit a collision avoidance algorithm to guarantee the human safety: in this way the robot can adapt its trajectory to avoid the obstacle (in this case the human operator) while accomplishing its task at full speed and without interruption. The collision avoidance algorithm has to continuously adjust the trajectory of the robot, considering at any moment the position of the human operator. It has to provide an effective motion re-planning while minimizing the distance between the original trajectory and the new one. Therefore an optimization-based algorithm is needed.

3.1. Proposed Control Architecture

In order to address the points raised in the Problem Statement, in this paper we propose the control architecture shown in Figure 2 and described in the following sections. In particular, the Human Tracking System combines the data coming from two cameras to continuously track the human body and to obtain its velocity. Then, this information is fed into the Safety Control: the robot nominal trajectory is adapted online, according to human movements and velocity, to avoid collisions between the human and the robot bodies. To minimize the difference between the nominal trajectory and the corrected one, an optimization algorithm is run online, based on robot accelerations. Finally, the computed safe position is given to the low level robot controller.

A detailed description of the control architecture components is given in Section 6.

4. Background on Control Barrier Functions

The constraints on robot control have to guarantee collision-free trajectories. A suitable mathematical tool to guarantee such a behavior is represented by the Control Barrier Functions (CBF) [12] which are briefly described in this section.

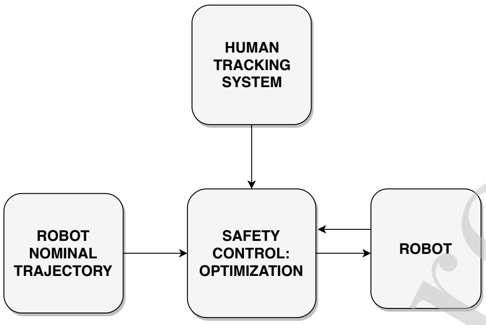


Figure 2: Overview of the proposed control architecture

Assuming we consider a control affine, nonlinear system of the form:

$$\dot{\chi} = f(\chi) + g(\chi)u \tag{1}$$

where $\chi \in \mathbb{R}^n$ is the system state and $u \in \mathcal{U} \subset \mathbb{R}^m$ is the control input, with $\mathcal{U}$ being the set of admissible control values. Functions f and g are locally Lipschitz continuous.

The control input should keep the system state inside a safe set to guarantee that the system remains in a safe region. We can achieve this goal exploiting CBFs. Basically, CBFs are non-negative functions that grow to infinity for states approaching to safety constraints, while they become smaller moving away from them.

A smooth function $h(\chi)$ is exploited to define a Control Barrier Function that constraints the system state to lay inside the safe region, i.e. for $h(\chi) > 0$, whereas $h(\chi) \leq 0$ indicates a violation of the constraint.

The safety region and its boundaries are defined by the set of admissible states $\mathscr{C}$:

$$\begin{aligned} \mathscr{C} &= \{\chi \in \mathbb{R}^n \mid h(\chi) > 0\}, \quad h : \mathbb{R}^n \to \mathbb{R} \\ \partial\mathscr{C} &= \{\chi \in \mathbb{R}^n \mid h(\chi) = 0\} \end{aligned} \tag{2}$$

The goal becomes to design a control law u that guarantees the forward invariance of the set $\mathscr{C}$ for all future times, i.e. if $\chi(0) \in \mathscr{C}$ then $\chi(t) \in \mathscr{C}$ for all $t \geq 0$. This behavior can be implemented using CBFs, since they relate the system control law with the constraint function $h(\chi)$.

More precisely, a function $B(\chi)$ qualifies as a CBF by fulfilling the following properties:

CBF-p_1 : A valid CBF is a non-negative function on $\mathscr{C}$

$$\inf_{\chi\in\mathscr{C}} B(\chi) \geq 0$$

CBF-p_2 : The barrier grows as the state χ approaches the constraint from inside the admissible set

$$\lim_{\chi\to\partial\mathscr{C}^+} B(\chi) = \infty$$

CBF-p_3 : The CBF grows with the growth rate (with $\gamma > 0$)

$$\dot{B}(\chi) \leq \frac{\gamma}{B(\chi)}$$

These properties make $B(\chi)$ behave like the inverse of a $\mathcal{K}$ function [1] [36]. 245 We can obtain the following formal definition relating the previous properties with the dynamics of a general system of the form (1).

Definition 4.1. *([12], Def. 4)*
Let a continuously differentiable $h : \mathbb{R} \to \mathbb{R}^n$ define $\mathscr{C}$ as in (2). A locally Lipschitz function $B(\chi) : \mathscr{C} \to \mathbb{R}$ is a Control Barrier Function, if its Lie 250 derivatives $\mathcal{L}_f B(\chi)$ and $\mathcal{L}_g B(\chi)$ are locally Lipschitz and if there exist class $\mathcal{K}$ functions α_1, α_2 and $\gamma > 0$ such that for all $\chi \in \mathscr{C}$

$$\frac{1}{\alpha_1(h(\chi))} \leq B(\chi) \leq \frac{1}{\alpha_2(h(\chi))} \tag{3}$$

$$\underset{u\in\mathcal{U}}{inf}\left[\mathcal{L}_f B(\chi) + \mathcal{L}_g B(\chi)u - \frac{\gamma}{B(\chi)}\right] \leq 0 \tag{4}$$

Since $B(\chi) \to \infty$ when $h(\chi) \to 0$ and $\dot{B}(\chi) \leq \frac{\gamma}{B(\chi)}$, equation (4) guarantees that CBF stops growing when the state approaches to the constraint.

[1] A continuous function $\alpha : [0, \inf) \to [0, \inf)$ is said to belong to class $\mathcal{K}$ if it is strictly increasing and $\alpha(0) = 0$ is verified.

Once we define an appropriate CBF, a relationship with the control value u is derived (i.e. provided that $\mathcal{L}_g B(\chi) \neq 0$ within the admissible set $\mathscr{C}$), yielding to the set of admissible control values $\mathcal{K}_B(\chi)$:

$$\mathcal{K}_B(\chi) = \{u \in \mathcal{U} \mid \mathcal{L}_f B(\chi) + \mathcal{L}_g B(\chi)u - \frac{\gamma}{B(\chi)} \leq 0\} \tag{5}$$

By applying a control input $u \in \mathcal{K}_B(\chi)$, we guarantee that the set $\mathscr{C}$ is forward invariant. It is worth noting that CBFs are defined inside the set of admissible states $\mathscr{C}$. This means that they are not able to resolve constraint violations. If the system goes outside the safety set,the controller is not able to bring the system back inside the safe region.

In the following, we will define the computation of the robot control input as an optimization problem. The objective of the problem is to minimize the norm of the difference between a nominal (i.e. potentially unsafe) control and the optimal one, that is constrained by CBFs (i.e. it must belong to a given $\mathcal{K}_B(\chi)$).

5. Trajectory optimization for collision avoidance

The control system has to solve online an optimization problem to generate a collision-free trajectory that is the closest to the nominal one. To this aim, the previously described CBFs will be exploited to build a set of virtual barriers. If all the robot links remain in a safe set, then the whole robot body is collision-free. Thus, it is required to compute all the possible collisions of each robot's link with each human's link.

To this aim, we need to define a model for each agent of the system (i.e. one for the human operator and one for the robot). These models allow to represent mathematically any entity of the system, define its behavior (both dynamic and kinematic), its spatial shape and the relations between entities. We chose to model the robot links and the human links using virtual capsules in order to enclose them inside the fittest and simplest shape, such that the distances can be easily computed. Given a couple of Cartesian points, a capsule [37] is a virtual object composed by two semi-spheres, centered in that points, and a cylinder, with longitudinal axis linking the two points.

5.1. *Robot model*

Let us denote with $x \in \mathbb{R}^3$ the Cartesian position of the robot end-effector. We consider that:

- robot trajectories are planned in terms of desired acceleration $\ddot{x}_{des}$, to avoid non-smooth motions (i.e. discontinuous velocities).
- we define the system state as $\chi = [\chi_1 \quad \chi_2]^T = [x \quad \dot{x}]^T$. Therefore, the system dynamics is given by (1) whose input u is the desired end-effector acceleration $\ddot{x}_{des}$, and

$$f(\chi) = \begin{bmatrix} \chi_2 \\ 0_{3\times 1} \end{bmatrix} \qquad g(\chi) = \begin{bmatrix} 0_{3\times 3} \\ I_{3\times 3} \end{bmatrix} \tag{6}$$

To compute the distance between each robot link and the human operator we choose to decompose the robot body into subset of consecutive links. In this way we can explicitly take into account the kinematic couplings among the links of the robot. In particular, according to the serial chain of the manipulator, the first subset is obtained considering the first link connected to the robot base, the second one considering the first and the second link and so on.

Considering a serial manipulator with m degrees of freedom (DoF), defined by m links and m joints, the joint position vector $q = (q_1, ..., q_m)$ specifies uniquely the robot configuration. As shown in Figure 3, in the manipulator we can define m subsets of link, from the base to the end-effector, and each subset can be associated to a vector $\Theta_i = (q_1, ..., q_i) \in \mathbb{R}^i$ $(i = 1, ..., m)$ and to a Jacobian matrix $J_{p,i}(\Theta_i)$. If we denote with $x_{c,i} \in \mathbb{R}^3$ the midpoint on the last link of each subset, the corresponding Cartesian acceleration can be defined as

$$\ddot{x}_{c,i} = \dot{J}_{p,i}\dot{\Theta}_i + J_{p,i}\ddot{\Theta}_i, \quad i = 1...m \tag{7}$$

where $J_{p,i} \in \mathbb{R}^{3\times i}$ is the linear velocity part of each Jacobian submodel. It is worth noting that each submodel state $\chi_i = [x_{c,i} \quad \dot{x}_{c,i}]^T$ is subjected to a dynamics with the same structure of equation (6).

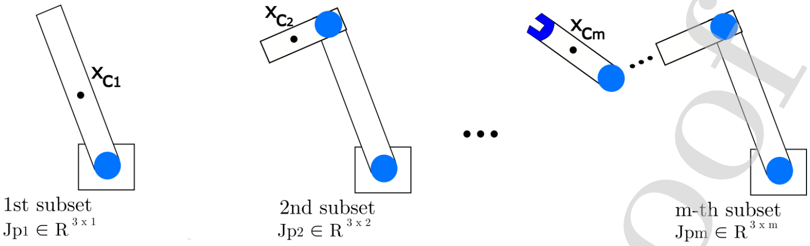

x_{C1}
x_{C2}
x_{Cm}
1st subset
$J_{p1} \in R^{3 \times 1}$
2nd subset
$J_{p2} \in R^{3 \times 2}$
m-th subset
$J_{pm} \in R^{3 \times m}$

Figure 3: Example of subset of consecutive links for a m-DoF robot manipulator

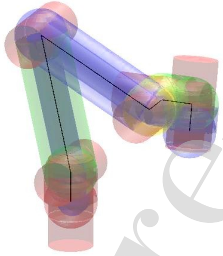


Figure 4: Capsules model for the manipulator

As previously introduced, the robot is modeled by using m capsules, as shown in Figure 4. Each capsule is composed by a cylindrical part that encloses the robot link, while the joints at the extremities are enclosed in two semi-spheres.

5.2. Human operator model

Regarding the model of the human operator, a first simple solution to be considered could be to enclose the whole body into a single element (eg.a sphere or a cylinder). However, this solution would be too conservative and simplistic for a correct human body representation. A less conservative and accurate human body representation consists in defining an obstacle around each body part.

We chose a 12 capsules model (as shown in Figure 5) to embody human parts, where each capsule is defined by the position of two human joints and a radius. The joints positions are obtained from the localization system, while the radius

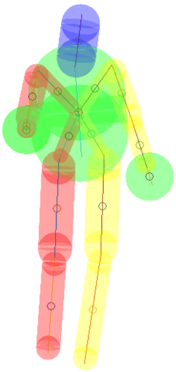


Figure 5: Capsules model for the human body

310 is defined a priori considering a generic human body. We chose 12 capsules since this is a trade-off between an accurate representation of the human body and the computational load for computing them online. The capsules incorporating hands and foots have a radius that allows to include even the fingers.

5.3. Obstacle distance

315 Once we have defined the human body and the robot body as a composition of capsules, we could evaluate the distance between them [38]. However, to perform this kind of operation online for each pair of robot capsules and human body capsules would take time, threatening human safety. To further simplify the computational load, we identify the closest human capsule for each robot link, i.e. the one at minimum distance from the link itself (Figure 6). 320

Thus, we first compute all the distances between human capsules and robot links middle-points:

$$d_{ij} = \|x_{c,i} - x_{h_j}\| - r_{c,i}, \quad \begin{aligned} i &= 1, ..., m \\ j &= 1, ..., 12 \end{aligned} \tag{8}$$

where $x_{c,i} \in R^3$ is the i-th link middle-point and $x_{h_j} \in R^3$ is the point on the

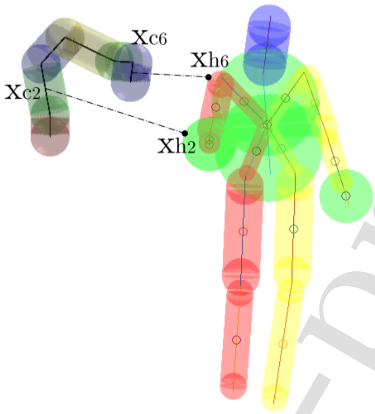


Figure 6: Example of human capsule closest to link 2 and link 6 of the robot

human capsule j which is closest to $x_{c,i}$. $r_{c,i}$ is the radius of the robot capsule. Since this evaluation is performed at each instant of time, the closest point on the human body changes dynamically, according to the human movement.

It is worth noting that the choice of the mid-points of robot links was a design choice. The theoretical findings of the paper hold for any other fixed point on the longitudinal axis of the considered robot link. Indeed, the distance is computed according to equation (8) where the radius of the robot capsule is constant along the capsule itself. As a consequence, the kinematic chain has to be built according to the chosen point along the robot link.

Then, in the optimization problem we only need to focus on a single capsule of the human body for each link of the robot. For ease of notation, in the following we will refer to the point x_{h_j}, corresponding to the closest human capsule, by directly using the term x_h.

5.4. Safety barrier function

As introduced in Sec.4, the constraints in the optimization problem, guaranteeing collision-free trajectories, are defined in terms of CBFs. We can consider the human operator as a moving obstacle with no control input and its skeleton is bounded inside capsules. Once we identified the closest capsule on the operator body, we can define a constraint function $h_i(\chi_i)$ on the state $\chi_i = [x_{c,i} \quad \dot{x}_{c,i}]^T$ for each subset of robot links, as:

$$h_i(\chi_i) = d_{ij} - D_s \geq 0, \quad i = 1, ..., m \tag{9}$$

where D_s is a desired distance to guarantee a safety margin for the operator.

Now we have to define an admissible CBF for the system. Consider the dynamic model of the robot in (1) and (6), where u_i is the control input, i.e. the desired acceleration for the i-th link middle-point. A candidate for the role of a CBF must satisfy the properties *CBF-p_1*, *CBF-p_2*, *CBF-p_3* (Sec. 4) and its time derivative must depend explicitly on the input. In other words, the relative degree between the CBF and the input must be one. This means that a function based only on the constraint $h_i(\chi_i)$, whose relative degree with respect to the input u_i is two, is not a valid candidate. However, the dependency of the control input u_i appears in the time derivative of the constraint function $\dot{h}_i(\chi_i)$. Therefore, an admissible CBF can be chosen as follow:

$$B_i(\chi_i) = -ln\left(\frac{h_i(\chi_i)}{1 + h_i(\chi_i)}\right) + a_E \frac{b_E \dot{h}_i(\chi_i)^2}{1 + b_E \dot{h}_i(\chi_i)^2} \tag{10}$$

The proposed CBF includes a logarithmic term to shape the barrier and a tunable term depending on $\dot{h}_i(\chi_i)$. By properly tuning the parameters a_E and b_E we can decide how far the system will stop from the constraint.

Given an acceleration input, the set of admissible control values (5) for each i-th robot link is defined as:

$$a_{Bi} u_i \leq \frac{\gamma}{B_i(\chi_i)} - b_{Bi} \quad i = 1, ..., m \tag{11}$$

where

$$
\begin{aligned}
a_{Bi} &= \frac{\partial B_i}{\partial \chi_i} g(\chi_i) \\
&= 2a_E b_E \frac{\dfrac{(x_{c,i} - x_h)^T(\dot{x}_{c,i} - \dot{x}_h)}{\|x_{c,i} - x_h\|}}{\left[1 + b_E \left(\dfrac{(x_{c,i} - x_h)^T(\dot{x}_{c,i} - \dot{x}_h)}{\|x_{c,i} - x_h\|}\right)^2\right]^2} \frac{(x_{c,i} - x_h)^T}{\|x_{c,i} - x_h\|}
\end{aligned}
\tag{12}
$$

and

$$
\begin{aligned}
& b_{Bi} = \frac{\partial B_i}{\partial \chi_i} f(\chi_i) = \\
& = -\frac{(x_{c,i} - x_h)^T(\dot{x}_{c,i} - \dot{x}_h)}{(\|x_{c,i} - x_h\|)(\|x_{c,i} - x_h\| - D_s)(1{+}\|x_{c,i} - x_h\| - D_s)} + \\
& + 2a_E b_E \frac{(x_{c,i} - x_h)^T(\dot{x}_{c,i} - \dot{x}_h)}{(\|x_{c,i} - x_h\|)\left[1 + b_E\left(\frac{(x_{c,i} - x_h)^T(\dot{x}_{c,i} - \dot{x}_h)}{\|x_{c,i} - x_h\|}\right)^2\right]^2} \cdot \\
& \left(-\frac{\ddot{x}_h(x_{c,i} - x_h)^T}{\|x_{c,i} - x_h\|} + \frac{\|\dot{x}_{c,i} - \dot{x}_h\|^2}{\|x_{c,i} - x_h\|} - \frac{\left[(x_{c,i} - x_h)^T(\dot{x}_{c,i} - \dot{x}_h)\right]^2}{\|x_{c,i} - x_h\|^3}\right)
\end{aligned}
\tag{13}
$$

In our work, differently from [12], we do not consider only the acceleration of the end-effector, but the acceleration of each link middle-point. Hence, we obtain a barrier function for each link, where the control input u_i is the desired Cartesian acceleration $\ddot{x}_{c,i}$ in (7). Consequently, we can rewrite each constraint for the i-th robot link (11) as:

$$
a_{Bi}(\dot{J}_{pi}\dot{\Theta}_i + J_{pi}\ddot{\Theta}_i) \leq \frac{\gamma}{B_i(\chi)} - b_{Bi} \quad i = 1, ..., m \tag{14}
$$

Since it's not possible to arbitrarily set the value of both joint velocities and accelerations, we choose the vector of joints accelerations $\ddot{q}$ as the optimal control input compatible with the Safety Barriers constraints. The current joint velocities are given as parameters to the control optimization.

5.5. Optimization problem

The quadratic optimization problem with linear constraints allows to compute the joints accelerations that will be used as control input. The objective of the optimization problem is to minimize the difference between the commanded control input $\ddot{q}$ and the desired end-effector Cartesian acceleration $\ddot{x}_{des}$. Meanwhile, it ensures safety by means of the safety barrier functions.

Assuming that the Jacobian matrices are incorporated in $\mathcal{R}$ and $\mathcal{S}$, as follows:

$$\mathcal{R}_i = a_{Bi}\dot{J}_{pi}, \quad \mathcal{R}_i \in \mathbb{R}^{1\times i} \tag{15}$$

$$\mathcal{S}_i = a_{Bi}J_{pi}, \quad \mathcal{S}_i \in \mathbb{R}^{1\times i} \tag{16}$$

the optimization problem is defined as follows:

$$\begin{aligned} \underset{\ddot{q}}{\text{minimize}} \quad & \|\ddot{x}_{des} - \dot{J}_{rob}\dot{q} - J_{rob}\ddot{q}\|^2 \\ \text{subject to} \quad & \mathcal{S}_1\ddot{\Theta}_1 \leq \frac{\gamma}{B_1(\chi)} - b_{B1} - \mathcal{R}_1\dot{\Theta}_1 \\ & \mathcal{S}_2\ddot{\Theta}_2 \leq \frac{\gamma}{B_2(\chi)} - b_{B2} - \mathcal{R}_2\dot{\Theta}_2 \\ & \quad \vdots \\ & \mathcal{S}_m\ddot{\Theta}_m \leq \frac{\gamma}{B_m(\chi)} - b_{Bm} - \mathcal{R}_m\dot{\Theta}_m \\ & \|\ddot{q}\|_\infty < \alpha \\ & \|\dot{q} + \ddot{q}\cdot\Delta t\|_\infty < \beta \end{aligned} \tag{17}$$

where J_{rob} is the Jacobian matrix of the full kinematic chain of the serial robot, from its base to the end-effector, and Δt is the sampling period. To obtain a feasible motion, joint acceleration and velocity bounds α and β are added to the constraints.

It is worth noting that the resulting controller is minimally invasive: it allows to precisely follow the nominal control input when the system is far from safety violation, while it modifies the behavior when a collision between a link and the obstacle is approaching.

6. Control Architecture Deployment

In this work we propose a multi-camera localization system to track human movements and obtain information about its velocity. Then, the proposed optimization-based collision avoidance algorithm replans the robot trajectory to minimize the difference between the nominal acceleration and the commanded one, to guarantee human safety.

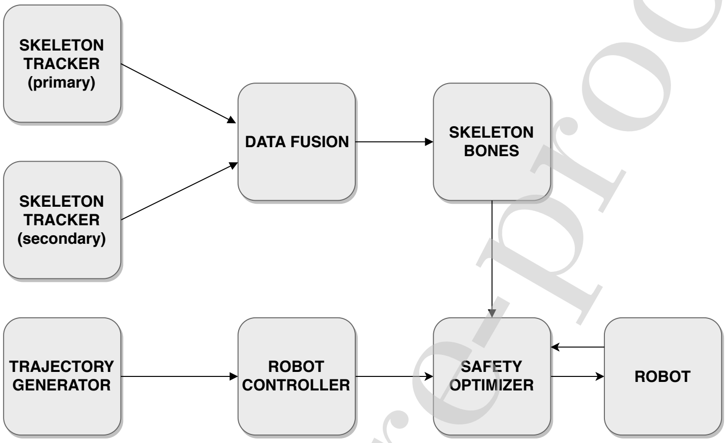


Figure 7: Control Architecture

The proposed control architecture is depicted in Fig. 7. To monitor the movement of the human operator in the shared workspace we exploit two RGB-D sensors, to obtain a reliable skeleton tracking. The data fusion is performed to merge the human skeleton data coming from the two sensors. Data fusion process exploits a bank of Kalman filters, to reduce data noise and to estimate the velocities and accelerations of the human operator joints. To avoid variability problems due to the variation of links length during skeleton tracking, the Skeleton Bones component has been introduced. It verifies if all the human body parts respect the reference values and it adjusts wrong measurements consequently. To guarantee human safety, we exploit the collision avoidance algorithm presented in Sec. 5.5, based on the current distance between the human body and the robot. The control strategy is minimally invasive, since it modifies the robot behavior (defined by the Trajectory Generator) only when the human safety is threatened.

Before going into details on each system component, we make a few assump-

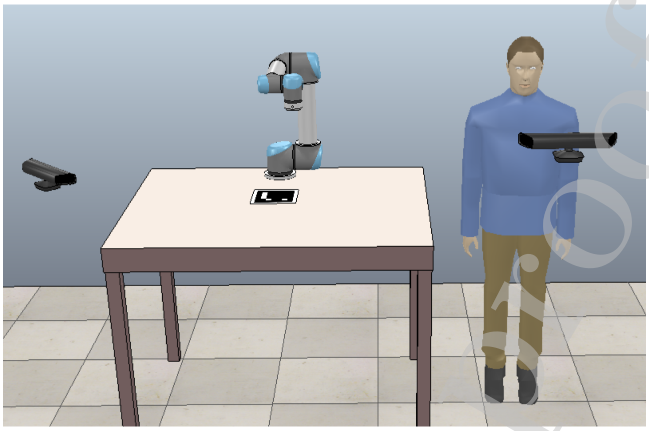


Figure 8: Environmental scene considered for the validation of the proposed system.

tions about the environmental scene we are considering for the validation of the proposed system, depicted in Figure 8:

- The robot is placed on a work table and the human operator can easily move around it. Since the robot is placed on a table, we can neglect the lower part of the human body during the evaluation of the safety distance for the optimization problem. However, this assumption was made according to the robotic setup we implemented for the validation experiments. It is worth noting that the algorithm works even for more general scenarios (e.g. the robot is placed on the floor) where the whole human body is considered for collision avoidance. Indeed the skeleton tracking algorithm allows to track the whole human body.

- The human operator can interact with the robot or work nearby it. Since in both cases the human is inside the robot workspace, the proposed method computes the safety distances and moves the robot in order to avoid collisions between the human and the robot.

- The shared workspace is monitored by two RGB-D sensors. They are placed at a different height with a different orientation. They both have to shot the whole scene (i.e. the entire shared workspace) to properly detect human and robot bodies. However, additional sensors can be used to obtain a more precise and reliable human tracking, at the expenses of an increasing computational load. On the opposite side, only one camera can be used, loosing the capability of safely managing occlusions. Finally, as mentioned in Section 1, different sensors can be used to track the human body, other than RGB-D cameras.

In the following we describe in details each component presented in Fig. 7.

6.1. Skeleton Tracker

The two Skeleton Tracker components allow to extract skeletal data from RGB-D sensors, as depicted in Figure 10: black dots represent human joints, while colored segments represent human links.

From the skeleton data, the information about the pose (i.e. position and orientation) of human joints is obtained, with respect to the camera reference frame. The use of two different cameras allows us to obtain more reliable data and alleviate the occlusion problem that arises when we relay on a single camera information. However, the use of two cameras implies to refer both image data to the same reference frame.

First, we define which camera is considered the main one (K_A in figure 9). Then, the raw data coming from the other camera (K_B) will be transformed according to the reference frame of the main one. The transformation matrix T_B^A, from the reference system B to A, is defined as:

$$T_B^A = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix}$$

To determine the rotational matrix R and the translation vector t inside the matrix T_B^A, we exploit joints data coming from the two cameras. In particular, when the human is standing still in front of the main and the secondary camera, joints data are acquired and exploited to minimize the following energy function:

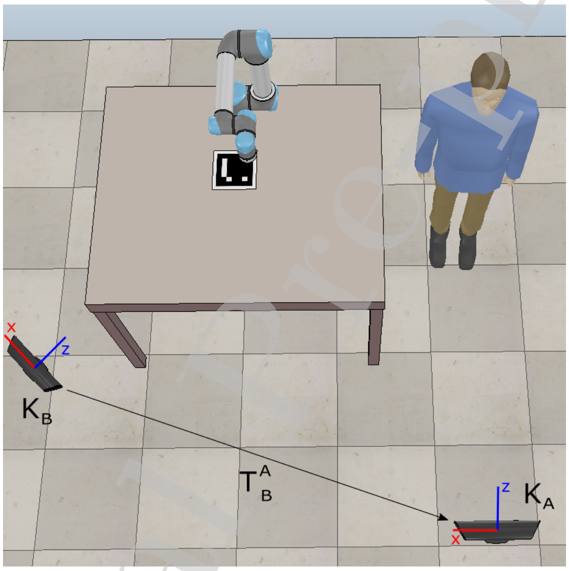


Figure 9: Reference frames K_A and K_B of the two cameras, and corresponding transformation matrix T_B^A

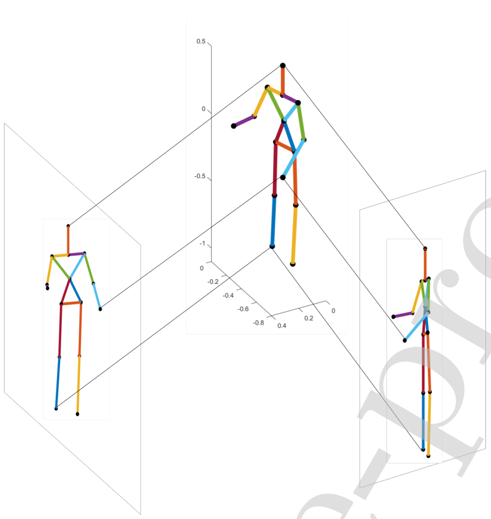


Figure 10: Example of data fusion: the data from the sensors are blended together to obtain more reliable data

$$J_R = \sum_i \| R(s_i - c_s) + t - (m_i - c_m) \|^2 \tag{18}$$

where c_m and c_s are the average centers of m_i and s_i, that are the set of joints positions from the main camera and the secondary one, respectively. By minimizing function J_R we obtain the transformation matrix to convert data from the secondary camera frame to the main one. Further details can be found in [39].

6.2. Data Fusion

Using multiple sensors allows us to obtain more reliable and consistent data, reducing the occlusion problem that could arise when using a single camera. The component of Data Fusion compares and blends together the data from both the cameras, as depicted in figure Figure 10.

Since the human is allowed to move freely inside the workspace of the robot, the data fusion process has to be performed online, to guarantee human safety.

We use a bank of the quaternion-based Kalman filters described in [40] (i.e. for each tracked joint), to remove the noise from skeleton tracker data and to blend the information received from the two sensors. In addition, the Kalman filtering module allows to estimate the human joints velocities and accelerations (both linear and angular), starting from the human joint poses. We denote with p_{Ji} the pose associated to the i-th joint, that includes the position x_{Ji} of joint itself and the orientation of the human link starting from that joint, expressed as a quaternion q_{Ji}. The bank of Kalman filters provides an estimate of the following vectors:

$$\xi_{Ji} = \begin{bmatrix} x_{Ji} \\ \dot{x}_{Ji} \\ \ddot{x}_{Ji} \end{bmatrix} ; \quad \vartheta_{Ji} = \begin{bmatrix} q_{Ji} \\ \omega_{Ji} \\ \alpha_{Ji} \end{bmatrix} \tag{19}$$

in which $\dot{x}_{Ji}$, $\ddot{x}_{Ji}$, ω_{Ji} and α_{Ji} denote respectively the linear velocity and acceleration of the joint and the angular velocity and acceleration of the link. The vector ξ_{Ji} has 9 scalar components and is estimated by a Linear Kalman Filter (LKF), while ϑ_{Ji} has 10 components and is estimated by an Extended Kalman Filter (EKF), based on the kinematic model of quaternion differentiation (see [40] for further details).

It is worth noting that data fusion is implemented by updating the same quaternion-based kinematic model every time that new skeleton data are received from any of the two cameras. The data are assumed to be captured asynchronously, similarly to the case discussed in [32]. The LKF/EKF update mechanism is executed using a single process noise covariance matrix and two different measurement noise covariance matrices. The latter are differently tuned to take into account the discrepancies in the two cameras, mostly due to their different placement inside the operating scenario.

In our work we take into account both data missing from the cameras and camera occlusions. In particular, we refer to ”data missing” when a camera is disconnected or is not able to track the human body (i.e. skeleton tracking matrices containing human joints poses are empty). Instead, we refer to

”occlusions” when the camera keeps track of the human, but a few joints are hidden and the corresponding poses are not available. When data are missing only from one camera, or occlusions occur, the Kalman Filter relies on the information coming from the other camera. An example on how we manage this situation is shown in Section 7.1.2.

If data are missing from both cameras for consecutive time steps, a low level time-out stops the robot to preserve human safety.

6.3. Skeleton Bones

One problem that can arise when using a skeleton tracker is the variability of link lengths, because the length of body segments may change during the tracking.

To overcome this problem we introduce the Skeleton Bones component. When a human operator enters for the first time in the scene and stops in front of a camera, a measurement on his/her body is performed. Then the reference values for each body link are computed and stored in this component for that specific operator. During the online tracking of the human body, each time the Skeleton Bones component receives joints data, it computes the current links lengths. Then, exploiting the stored values, it adjusts the position of the joints in order to respect the reference lengths. To avoid a continuous adjustment, we set a threshold ϵ: when the difference between the current and the reference values is grater than ϵ, the correction is made.

When we consider a human link, we call J_a the first joint we meet if we try to reach the link starting from the torso, while J_{a+1} the following one. If a correction has to be made, we decided to always move J_{a+1}.

To perform collision avoidance between the robot and the human operator, the poses, velocities and acceleration of the human joints, coming from data fusion, have to be referred to the robot frame. Therefore we have to define a transformation matrix from the main camera reference frame to the robot base frame.

To determine the transformation matrix, we use a marker, placed on the

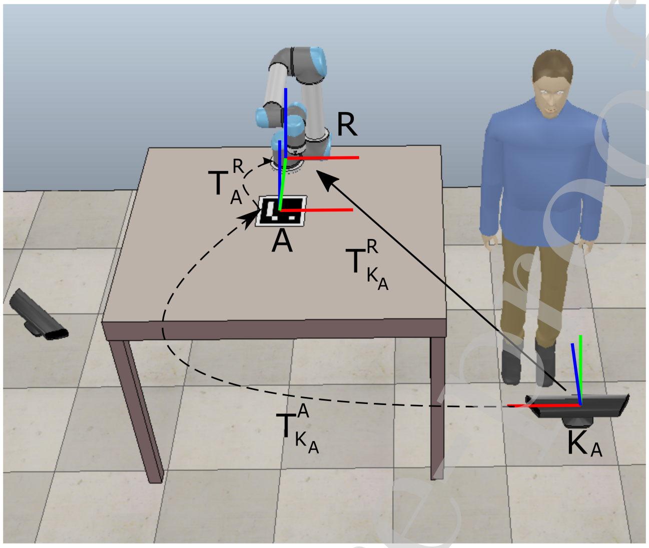


Figure 11: Transformation matrices between the main camera and the marker ($T^A_{K_A}$), the marker and the robot base (T^R_A) and the main camera and the robot base ($T^R_{K_A}$).

table near the robot base. Using this marker it is possible to learn the camera pose with respect to the marker, obtaining the transformation matrix $T^A_{K_A}$ in figure 11. We need a further transformation from the marker to the robot base (T^R_A), to obtain the final transformation matrix from the main camera to the robot base frame:

$$T^R_{K_A} = T^R_A T^A_{K_A} \tag{20}$$

6.4. Trajectory Generator and Robot Controller

To test the proposed system, we monitor the human movements when interacting with a robot that is following a predefined path, with a fixed end-effector orientation. We planned robot trajectory offline, computing the end-effector positions and velocities in the Cartesian space. Then a PD controller we obtain the desired acceleration as:

$$\ddot{x}_{des} = K_P(x_{ref} - x) + K_D(\dot{x}_{ref} - \dot{x}) \tag{21}$$

where $K_P \in \mathbb{R}^{3\times3}$ and $K_D \in \mathbb{R}^{3\times3}$ are positive definite diagonal matrices and x_{ref} and $\dot{x}_{ref}$ are the reference pose and velocity of the end-effector, while x and $\dot{x}$ are the current position and velocity of the robot. This acceleration is fed to the Safety Optimizer component as the desired acceleration for the optimization problem.

6.5. Safety Optimizer

Once the human body has been tracked correctly and its pose, velocity and acceleration have been computed, the Safety Optimizer component exploits this information to avoid collisions when the human operator interferes with the robot trajectory. According to the algorithm presented in Sec. 5.5, this component determines the minimum deviation from the nominal acceleration if a collision is approaching.

At the beginning of Sec. 6 we made the assumption that the robot is placed on a work table. Hence, we can reduce computational load of the Safety Optimizer component by reducing the number of human-robot capsules we consider for the collision avoidance. Specifically, we consider only the upper part of the human body and we neglect the first robot link because robot base can not move to avoid the obstacle.

Since collision avoidance relies on the solution of an optimization problem, cases of not convergence can occur. This means that the system can not find a feasible solution and human safety is threatened. To preserve safety in a conservative way, we command the robot a scaled version of the last commanded velocity (in our tests we reduced the velocity by 20%), until robot stops if no feasible solutions are found repeatedly.

In the optimization problem we also need the values of $\dot{x}_h$ and $\ddot{x}_h$ (12)(13). As previously described, the bank of Kalman filters for data fusion estimates the linear velocities and accelerations of each joint and the angular velocities and accelerations of each link. Given this knowledge and considering that x_h belongs to a rigid body represented by the human link, we can calculate $\dot{x}_h$ and $\ddot{x}_h$ by means of well-known kinematic equations. Such equations propagate the estimated velocities and accelerations of the link up to x_h, as follows:

$$\begin{aligned} \dot{x}_h &= \dot{x}_{J_a} + \omega_{J_a} \times (x_h - x_{J_a}) \\ \ddot{x}_h &= \ddot{x}_{J_a} + \omega_{J_a} \times (\omega_{J_a} \times (x_h - x_{J_a})) + \alpha_{J_a} \times (x_h - x_{J_a}) \end{aligned} \tag{22}$$

where: x_{J_a} is the first joint of the considered link (i.e. the first joint we meet before reaching x_h starting from the torso joint); $\dot{x}_{J_a}$, $\ddot{x}_{J_a}$, ω_{J_a} and α_{J_a} are the related velocities and accelerations as estimated by the LKF/EKF processing module previously described.

7. Experimental Validation

Different experiments were performed to show the effectiveness of the proposed algorithm. The first experiment was used to validate the bank of Kalman filters as estimators of human acceleration and velocity. Then, the second experiment showed how the system manages the occlusion of a camera to guarantee reliable data. Finally, the optimization-based algorithm was tested in a human-robot shared environment, where the robot had to follow a nominal trajectory while the human was working nearby. The results show that the robot deviates from the nominal trajectory to avoid the collision with the human operator, preserving his/her safety during the execution of a task.

The proposed approach, based on CBFs for a serial manipulator, has been implemented and tested on a Universal Robots UR5, a 6-DoF industrial manipulator designed for collaborative applications. For the first experiment, regarding the validation of the Kalman filter banks, we used a 6-DoF Puma 260 robot. As RGB-D cameras, we used both Intel Realsense D415 and Asus Xtion, to test the robustness of the Kalman filter on different hardware. The software components were developed using ROS and OROCOS frameworks. The optimization problem was solved online exploiting C++ code generated by CVXGEN software.

7.1. Human body tracking

The first phase of the experimental validation was focused on human body tracking. First, we tested the effectiveness of Kalman filters banks for velocity

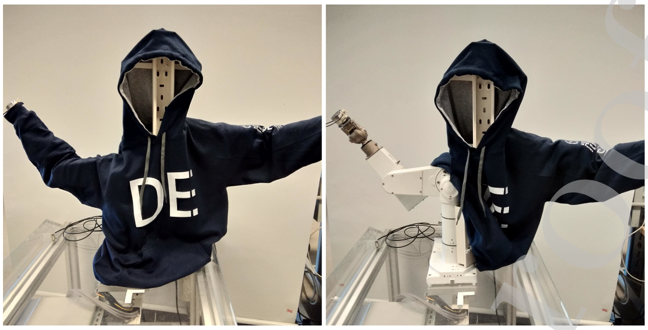


Figure 12: Experimental setup for testing the skeleton tracking software

and acceleration estimation. To cope with camera occlusion issues, we then evaluated data fusion from multiple RGB-D cameras.

7.1.1. Estimation of Velocities and Accelerations

The practical execution of Kalman filters requires the tuning of covariance matrices associated to the process noise and the measurement noise. Such matrices are generally denoted respectively as **Q** and **R**.

The **R** matrix is related to the measurement technology and it should be evaluated when the observed process is in a stationary condition. The **Q** should be instead tuned to achieve a good compromise between smoothness and desired dynamics of the estimation output. For this purpose, it would be important to compare the estimated values of the Kalman filter outputs with their true values or, at least, with values estimated with an accurate technique.

For human body tracking, we developed a solution to obtain such reference values from a robotic emulation of a human. In particular, we inserted a Puma 260 6-DOF manipulator into the sleeve of a hooded sweatshirt and realized a fixed structure to sustain the other sleeve and the hood of the sweatshirt. In the robot initial configuration, the whole assembly mimics the upper body of a human in a T-shape posture, as shown in Figure 12. When the robot is moving, instead, the structure mimics the motion of the right arm of the human.

First, an experimental analysis of the noise affecting the skeleton tracker was executed. The measurements used for the evaluation were obtained by detecting the upper body skeleton of the robotic emulator with the Puma 260 at rest. It is worth noting that this stationary condition cannot be achieved when tracking a real human, due to little movements of the human body even when it is still. The covariance matrix was calculated from measurement data. The covariance matrix may vary according to the camera model and to the placement of the camera itself. In our case, with Intel Realsense D415 camera placed at a distance of 3 meters from the robot, we obtained the following setting for each one of the LKFs:

$$\mathbf{R} = \mathbf{diag}(3.2 \times 10^{-6}, 8.4 \times 10^{-7}, 1.91 \times 0^{-7})$$

and for the quaternion-based EKFs:

$$\mathbf{R} = \mathbf{diag}(1.238 \times 10^{-5}, 4.756 \times 10^{-5}, 2.957 \times 0^{-5}, 4.99 \times 10^{-5})$$

To evaluate the process noise covariance matrix, we followed the rule of thumb stating that in a Kalman filter used for differentiation higher values should be associated to the matrix entries related to higher order derivatives. In other words, the process noise on accelerations is assumed to be much higher than that affecting the velocities. After some trials, we obtained the following values for the LKFs:

$$\mathbf{Q} = \mathbf{diag}(10^{-5}, 10^{-5}, 10^{-5}, 10^{-2}, 10^{-2}, 10^{-2}, 10^{-1}, 10^{-1}, 10^{-1})$$

and for the quaternion-based EKFs:

$$\mathbf{Q} = \mathbf{diag}(10^{-5}, 10^{-5}, 10^{-5}, 10^{-5}, 10^{-4}, 10^{-4}, 10^{-4}, 10^{-2}, 10^{-2}, 10^{-2})$$

The accuracy of the outputs of the Kalman filters has been evaluated by comparing them with the reference values from the forward kinematics of the Puma 260. In particular, we focused on the elbow joint orientation and on the wrist joint position, assuming that they match respectively with the orientation of the third link and with the center of the spherical wrist of the Puma robot. Velocities and accelerations of the robot links were calculated by processing kinematics data with a Savitzky-Golay differentiation filter. Since the filter is non-causal, we assume that its output is the best estimate of first and second order derivatives that we can get from the robot joints position measurements.

Figures 13(a), 13(b), 14(a) and 14(b) show the comparison between reference values (blue lines) and estimated values (red lines) of, respectively, the linear velocities and accelerations of the wrist and the angular velocities and accelerations of the elbow. The plots refer to a sequence of motions in which the base joint (i.e. the human left shoulder), the second and the third joint of the Puma 260 moved of 60 degrees from the initial configuration shown in Figure 12, either simultaneously or one at a time. As depicted from figures, the outputs of the Kalman filter are similar to the reference values during most of the motions, with a limited delay. We found that the highest discrepancies between references and estimates were especially related to motions involving the third joint (i.e. the human elbow). This effect is due to the elbow position tracked from the RGB-D camera, that is not rigidly matching the Puma robot joint one. Indeed the elbow position is changing its placement on the sweatshirt sleeve according to the angle between the second and third link of the robot.

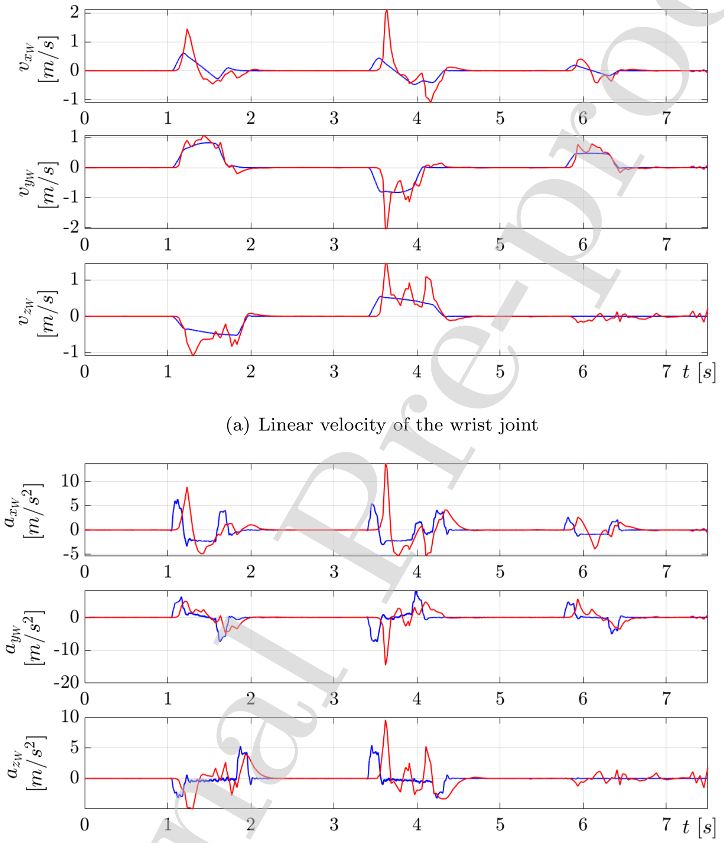
7.1.2. Camera occlusion

In the data fusion component we handle the problem of one of the two sensors occlusion. In the following plots we show the recorded values of the joint *torso* because it is usually the most stable during the tracking.

The plots in Figure 15 show the raw data coming from the principal camera (blue dashed line), the secondary camera (green dotted line) and the output of the filter (red solid line). We highlighted the time period where we detected an occlusion of one of the sensors.

In particular, we can observe two occlusion episodes:

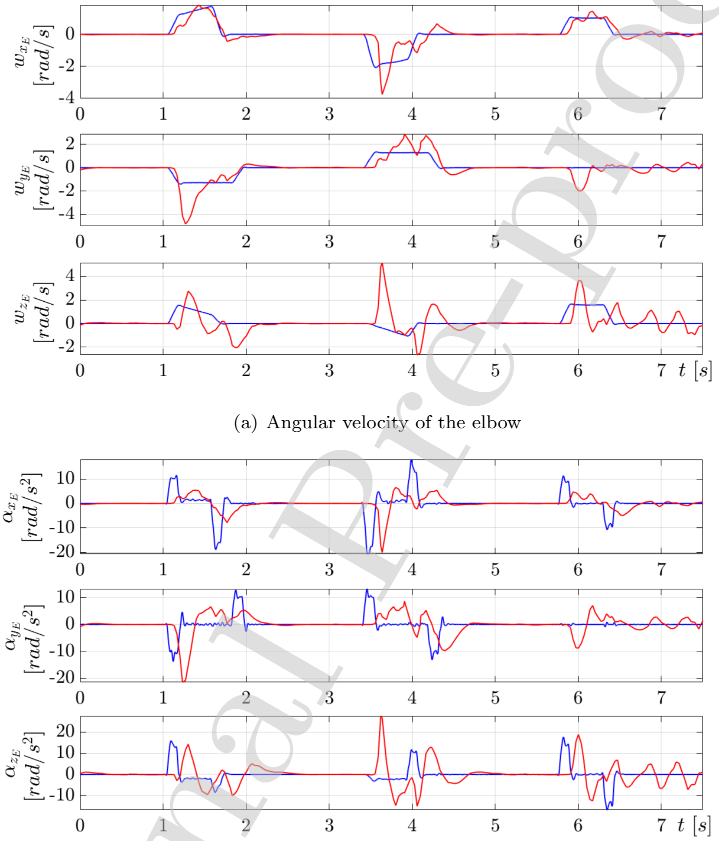
- in the first episode the occluded camera was the second one. During



(a) Linear velocity of the wrist joint

(b) Linear acceleration of the wrist joint

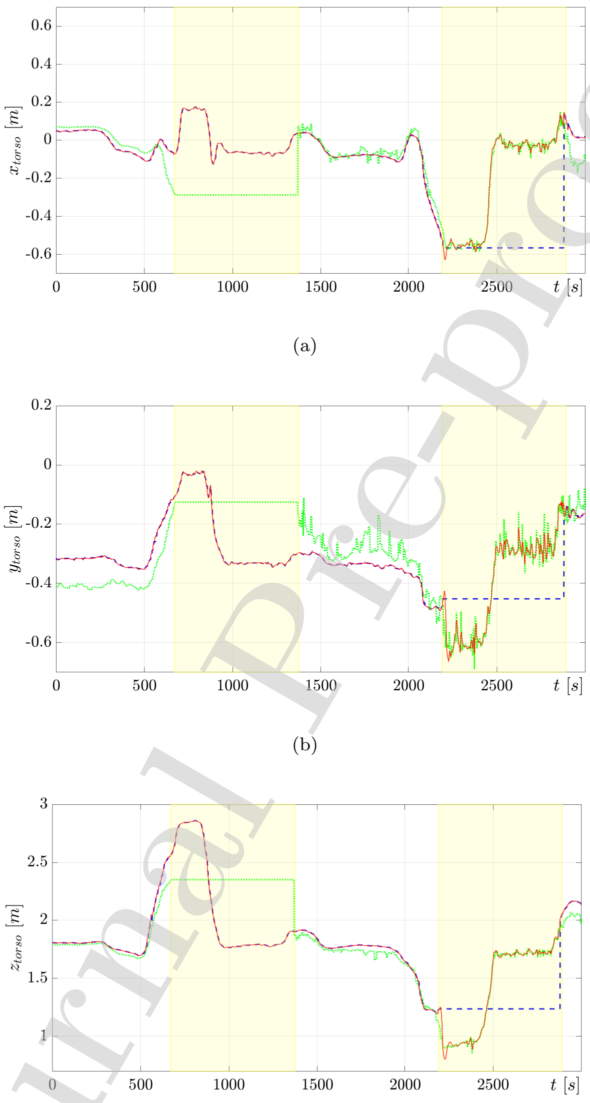
Figure 13: Comparison between the estimated data from the Kalman filter (red line) and calculated data from Puma 260 kinematics (blue line).



(a) Angular velocity of the elbow

(b) Angular acceleration of the elbow

Figure 14: Comparison between the estimated data from the Kalman filter (red line) and calculated data from Puma 260 kinematics (blue line).



(a)

(b)

(c)

Figure 15: Comparison between the cameras raw data and the output data from the data fusion component. Joint torso is chosen as example. Data coming from the principal camera are depicted with a blue dashed line, the secondary camera ones with a green dotted line and the output of the filter with a red solid line. We highlighted the time period where we detect an occlusion of one of the sensors.

the occlusion, the sensor kept sending the same value for a short time, as the joint *torso* was completely still. In the meantime, the principal camera kept recording the joint *torso* movements. As a result, even if the secondary camera could not send reliable data, the data fusion component computed a reliable output by exploiting only the principal camera data;

- in the second episode the occluded camera was the principal one. In this case, the Kalman filter exploited the data from the secondary sensor even if they were more noisy with respect to the principal camera. We can also note that the system reduced the noise from the secondary camera by removing the outliers.

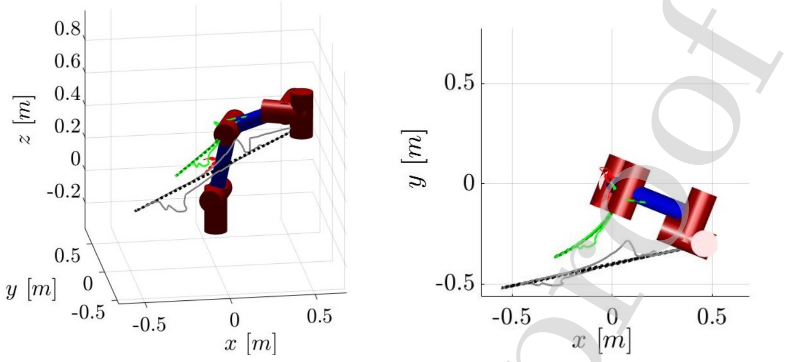
We can note that the secondary camera data are more unstable and noisy than the one from the main camera. Moreover, there is a small offset in the data received from the two sensors. This offset is due to the data reference frame transformation from the secondary camera to the principal one. To overcome this problem, the Kalman filter had to closely follow the data from the principal camera. Hence, a different priority was given to the sensors: once new data arrive to one of the two sensors, the joint data are updated with different covariance matrices (i.e. principal camera data are considered more reliable than the secondary ones).

7.2. Human-Robot sharing the same workspace

To test the effectiveness of the proposed algorithm, we performed an experiment involving a human operator and a robot moving in the same workspace, as shown in the attached video. During the experiment, the robot followed a desired trajectory, while guaranteeing human safety. The desired robot trajectory, in terms of accelerations and velocities, was planned using an offline simulator. The movement of the human operator was monitored by two RGB-D sensors and a skeleton tracker software. When the human operator approached the robot during its motion, the system computed the minimum deviation from the nominal trajectory to guarantee obstacle avoidance and human safety. The parameters setting that was used during the experiment is described in Table 1.

Table 1: Parameters setting for the experiment

Data Fusion	
Update loop time period:	0.04 s
Number of connected cameras:	2
Linear velocity threshold:	2 m/s
Angular velocity threshold:	10 rad/s
Linear acceleration threshold:	10 m/s^2
Angular acceleration threshold:	30 rad/s^2
Skeleton Bones	
Update loop time period:	0.04 s
Reader and Cartesian Controller:	
Update loop time period:	0.04 s
Matrices of PD controller:	$\mathbf{K_P} = diag\,(20, 20, 20, 10, 10, 10)$ $\mathbf{K_D} = diag\,(20, 20, 20, 10, 10, 10)$
Safety Optimizer:	
Update loop time period:	0.04 s
α:	1.4 rad/s
β:	8 rad/s
γ:	30
Δt	0.04 s
D_s	0.10 m
a_E	2
b_E	1
UR5 Bridge	
Update loop time period:	0.008 s



(a) 3D View

(b) Top view

Figure 16: 3D plot of the nominal trajectory (dotted lines) and of the executed one (solid lines) of the second (red), third (green) and fifth (black) link during obstacle avoidance.

7.2.1. Robot movements

The robot movements during the experiment are shown in Figure 16. In this figure we represent the initial configuration of the robot and the trajectories of the second, third and fifth links. For each link we plot the nominal trajectory (dotted lines) and the executed one (solid lines). In this experiment, the human approached the robot by moving the right arm. The closest human capsule was detected as the human hand, while the closest robot capsule was around the fifth link. As we can see from the figure, the fifth link trajectory had the major correction, while other links moved accordingly.

7.2.2. Trajectory of the end-effector

In Figure 18 we compare the nominal trajectory of the end-effector with the executed one in presence of an obstacle. The plots in Figure 17 show the distance between each link and the human operator, in particular with the related $x_{h,i}$.

By relating the plot in Figure 17 with the plots of the robot movements (18(a), 18(b), 18(c)), we can see how the robot tries to deviate from the nominal trajectory when at least one distance link-obstacle is inside the gray band. D_r indicates the minimum distance we want to preserve to guarantee the human

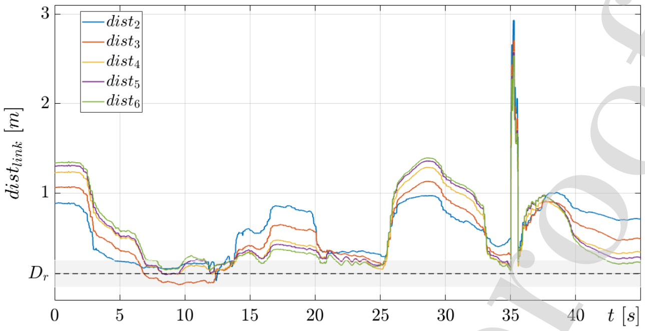


Figure 17: Distance between links and obstacle ($D_r = 0.10$ m).

safety. The more the distance gets closer to this value, the more the robot deviation is significant.

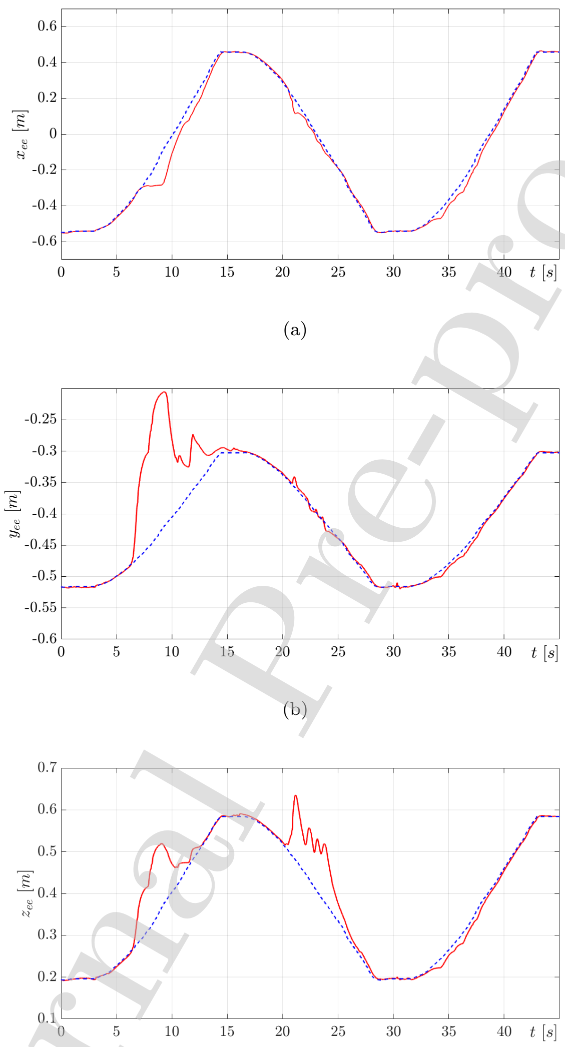
In the last plot (18(c)) we can notice that the computed deviation is non-smooth. This behavior is probably due to the use of a PD controller and the evaluation of the instantaneous position of the two agents in the optimization problem, without considering the evolution of robot and human movements.

7.2.3. Non-convergence of the optimization algorithm

In Figure 20 and Figure 19 the time periods where the optimization did not converge are highlighted. When the optimization does not converge, the commanded velocity is scaled from the previous value by a 20%. Only after a sequence of non-convergence episodes, the resulting velocity is almost null and the robot stops, preserving human safety. The robot will resume its movements only after the operator moves away from the robot.

8. Conclusions

In this work we propose an optimization-based algorithm for collision avoidance to guarantee human safety in a shared human-robot environment. The algorithm minimizes the difference between the desired acceleration input and the commanded one, such that the robot keeps following the desired trajectory



(a)

(b)

(c)

Figure 18: Comparison between the nominal trajectory (blue line) of the end-effector and the executed one (red line), in relation to the human-robot distance.

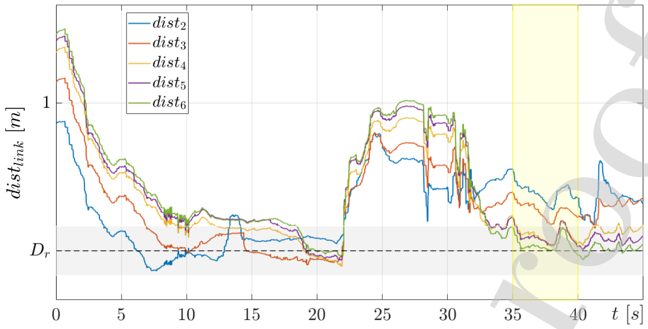
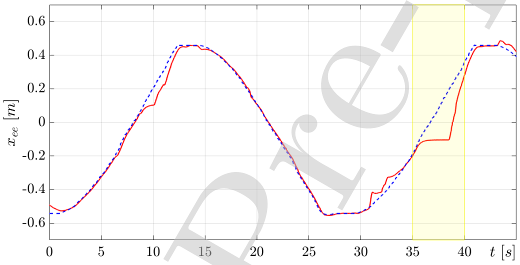
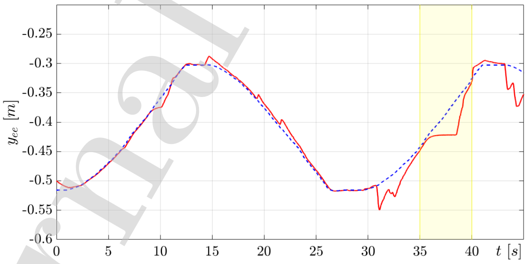


Figure 19: Distance between links and obstacle ($D_r = 0.10$ m).



(a)



(b)

Figure 20: Comparison between the nominal trajectory (blue line) of the end-effector and the executed one (red line), when the system cannot converge. We omitted axis z plot since it is similar to axis y one.

while guaranteeing human avoidance. To guarantee a collision-free trajectory along the whole robot body, we built a Control Barrier Function along each robot link and we computed the distance between each human-robot link. To avoid obstruction problems, we implemented a two cameras system with data fusion, to merge information coming from the two devices. We exploited a bank of Kalman filters to solve the obstruction problem and to compute human velocities and accelerations. In the experimental session we demonstrated the reliability of the Kalman filter banks and the effectiveness of the proposed collision avoidance algorithm in a scenario where human and robots share the same workspace.

The main issue of the proposed method is related to the limitations of the vision camera system. For example it can happen that a human joint is incorporated in the robot body and it is seen as a whole with the robot. In this case, raw data coming from the sensors are not reliable and the optimization problem may not converge. Further works aim at investigating the use of more reliable RGB-D cameras and skeleton tracking solutions suitable for industrial applications.

Future works aim also at implementing prediction of human movements, to avoid robot reactive behaviors and to plan a smooth robot trajectory during the deviation from the nominal path. Human intention can be also exploited to predict when the human itself is trying to avoid the robot. Moreover, we aim at combining the proposed method with the Power and Force Limitation modality, to obtain a safe behavior according to ISO 15066. This would allow to safely manage collisions, for example when the optimization algorithm does not converge. Furthermore, in order to certify the safety of the robotic system, we have to exploit certified depth cameras. However, the main limitation of current certified sensors is the low update frequency, that is not compatible with human movements that occur suddenly. Future works aim at finding the best trade-off between safe certified vision systems and environments that change abruptly. Finally, we would like to implement the proposed method in a real industrial environment for a collaborative robotics application.

References

[1] L. Villani, J. De Schutter, Safety for physical human-robot interaction, in: B. Siciliano, O. Khatib (Eds.), Springer Handbook of Robotics, Springer Berlin Heidelberg, 2008, Ch. 57.

[2] A. Bicchi, G. Tonietti, Dealing with the safety-performance tradeoff in robot arms design and control, IEEE Robotics and Automation Magazine 11 (2).

[3] S. Haddadin, A. Albu-Schaffer, A. De Luca, 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.

[4] A. Cirillo, P. Cirillo, G. De Maria, C. Natale, S. Pirozzi, A proximity/contact-force sensor for human safety in industrial robot environment, in: Proceedings of IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM), 2013, pp. 1272–1277.

[5] Tactile Sensor Systems, `https://www.iff.fraunhofer.de/en/business-units/robotic-systems/technologies/tactile-sensor-systems.html`, accessed: 2019-03-13.

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

[7] Piltz safety eye, `https://www.pilz.com/en-INT/eshop/00106002207042/SafetyEYE-Safe-camera-system`, accessed: 2019-03-12.

[8] C. Vogel, M. Poggendorf, C. Walter, N. Elkmann, Towards safe physical human-robot collaboration: A projection-based safety system, in: Pro-

ceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2011, pp. 3355–3360.

[9] Dual Check Safety (DCS) Explained, `http://motioncontrolsrobotics.com/dual-check-safety-dcs-explained/`, accessed: 2019-03-13.

[10] L. Bascetta, G. Ferretti, G. Magnani, P. Rocco, Walk-through programming for robotic manipulators based on admittance control, Robotica 31 (7) (2013) 1143–1153.

[11] L. Wang, A. D. Ames, M. Egerstedt, Safety barrier certificates for collisions-free multirobot systems, IEEE Transactions on Robotics 33 (3) (2017) 661–674.

[12] M. Rauscher, M. Kimmel, S. Hirche, Constrained robot control using control barrier functions, in: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2016, pp. 279–285.

[13] C. T. Landi, F. Ferraguti, S. Costi, M. Bonfè, C. Secchi, Safety barrier functions for human-robot interaction with industrial manipulators, in: Proceedings of European Control Conference (ECC), 2019.

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

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

[16] T. Salmi, I. Marstio, T. Malm, J. Montonen, Advanced safety solutions for human-robot-cooperation, in: Proceedings of ISR 2016: 47st International Symposium on Robotics, 2016, pp. 1–6.

[17] M. P. Polverini, A. M. Zanchettin, P. Rocco, Real-time collision avoidance in human-robot interaction based on kinetostatic safety field, in: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2014, pp. 4136–4141.

[18] A. Levratti, G. Riggio, C. Fantuzzi, A. De Vuono, C. Secchi, Tirebot: A collaborative robot for the tire workshop, Robotics and Computer-Integrated Manufacturing 57 (2019) 129–137.

[19] 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.

[20] F. Ferraguti, N. Preda, M. Bonfe, C. Secchi, Bilateral teleoperation of a dual arms surgical robot with passive virtual fixtures generation, in: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2015, pp. 4223–4228.

[21] 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: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2010, pp. 3109–3116.

[22] K. Ikuta, H. Ishii, M. Nokata, Safety evaluation method of design and control for human-care robots, The International Journal of Robotics Research 22 (5) (2003) 281–297.

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

[24] H.-C. Lin, Y. Fan, T. Tang, M. Tomizuka, Human guidance programming on a 6-dof robot with collision avoidance, in: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2016, pp. 2676–2681.

[25] H.-C. Lin, C. Liu, Y. Fan, M. Tomizuka, Real-time collision avoidance algorithm on industrial manipulators, in: Proceedings of IEEE Conference on Control Technology and Applications (CCTA), 2017, pp. 1294–1299.

[26] H. Zhou, H. Hu, Human motion tracking for rehabilitation—a survey, Biomedical Signal Processing and Control 3 (1) (2008) 1 – 18.

[27] A. Filippeschi, N. Schmitz, M. Miezal, G. Bleser, E. Ruffaldi, D. Stricker, Survey of motion tracking methods based on inertial sensors: A focus on upper limb human motion, Sensors 17 (6).

[28] L. Zhang, J. Sturm, D. Cremers, D. Lee, Real-time human motion tracking using multiple depth cameras, in: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2012, pp. 2389–2395. `doi:10.1109/IROS.2012.6385968`.

[29] G. Du, P. Zhang, D. Li, Human–manipulator interface based on multisensory process via kalman filters, IEEE Transactions on Industrial Electronics 61 (10) (2014) 5411–5418.

[30] G. Du, P. Zhang, X. Liu, Markerless human–manipulator interface using leap motion with interval kalman filter and improved particle filter, IEEE Transactions on Industrial Informatics 12 (2) (2016) 694–704.

[31] I. Boiko, M. Chehadeh, Sliding mode differentiator/observer for quadcopter velocity estimation through sensor fusion, International Journal of Control 91 (9) (2018) 2113–2120.

[32] S. Farsoni, C. T. Landi, F. Ferraguti, C. Secchi, M. Bonfè, Real-time identification of robot payload using a multirate quaternion-based kalman filter and recursive total least-squares, in: 2018 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2018, pp. 2103–2109.

[33] ISO 10218-1-2011, Robots and robotic devices – Safety requirements for industrial robots. Part 1: Robots (since July 1, 2011).
URL `http://www.iso.org`

[34] ISO 10218-2-2011, Robots and robotic devices – Safety requirements for industrial robots. Part 2: Robot systems and integration (since July 1, 2011).
URL http://www.iso.org

[35] ISO TS 15066:2016, Robots and robotic devices – Collaborative robots (since February 15, 2016).
URL http://www.iso.org

[36] H. K. Khalil, Nonlinear systems; 3rd ed., Prentice-Hall, Upper Saddle River, NJ, 2002, the book can be consulted by contacting: PH-AID: Wallet, Lionel.
URL https://cds.cern.ch/record/1173048

[37] H. Lin, C. Liu, Y. Fan, M. Tomizuka, Real-time collision avoidance algorithm on industrial manipulators, in: 2017 IEEE Conference on Control Technology and Applications (CCTA), 2017, pp. 1294–1299. doi:10.1109/CCTA.2017.8062637.

[38] M. Safeea, N. Mendes, P. Neto, Minimum distance calculation for safe human robot interaction, Procedia Manufacturing 11 (2017) 99 – 106, 27th International Conference on Flexible Automation and Intelligent Manufacturing, FAIM2017, 27-30 June 2017, Modena, Italy. doi:https://doi.org/10.1016/j.promfg.2017.07.157.
URL http://www.sciencedirect.com/science/article/pii/S2351978917303633

[39] K.-Y. Yeung, T. H. Kwok, C. Wang, Improved skeleton tracking by duplex kinects: A practical approach for real-time applications, Journal of Computing and Information Science in Engineering 13 (2013) 041007. doi:10.1115/1.4025404.

[40] S. Farsoni, C. T. Landi, F. Ferraguti, C. Secchi, M. Bonfè, Compensation of load dynamics for admittance controlled interactive industrial robots using

a quaternion-based kalman filter, IEEE Robotics and Automation Letters
855 2 (2) (2017) 672–679. doi:10.1109/LRA.2017.2651393.

Federica Ferraguti received the M.Sc. degree in Industrial and Management Engineering and the Ph.D. in Industrial Innovation Engineering from the University of Modena and Reggio Emilia, Italy, in 2011 and 2015. She has been visiting student at the Rehabilitation Engineering Lab at ETH Zurich, Switzerland in 2013. She is currently an Assistant Professor at the University of Modena and Reggio Emilia. In 2017 she received the “Fabrizio Flacco Young Author Best Paper Award” from the IEEE Robotics and Automation Society – Italian Chapter. Her research deals with surgical robotics, telerobotics, control of robotic systems and human-robot physical interaction, collaborative robotics.

Chiara Talignani Landi received the M.Sc. degree in Mechatronics Engineering in 2016 from the University of Modena and Reggio Emilia, Italy. She is currently a PhD student in Industrial Innovation Engineering at the University of Modena and Reggio Emilia. She visited the Mechanical Sysytems Control laboratory in Berkeley (CA, USA) in 2018. Her researh focuses on collaborative robotics and Human-Robot Physical Interaction.

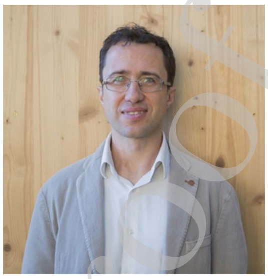
Silvia Costi received the Master degree in Computer Engineering
at University of Modena and Reggio Emilia, Italy, in 2019.
She is currently a Research Fellow at University of Modena and Reggio Emilia.
Her research activities include robotics and human-robot collaboration.

Marcello Bonfè received the M.Sc. degree in Electronic
Engineering in 1998, from the University of Ferrara, and the Ph.D. in Information
Engineering in 2003, from the University of Modena and Reggio Emilia. He is currently
Assistant Professor of Automatic Control at the University
of Ferrara, Italy. His main research interests are:
motion planning and control for robotics, modelling
and control of mechatronic systems, fault detection
and fault tolerant control.

SAVERIO FARSONI received the M.Sc. degree in Informatics and Automation Engineering and the Ph.D. degree in Engineering Sciences from the University of Ferrara, in 2012 and 2016, respectively. He has published about 20 refereed journal and conference papers. His research activities include robotics, fault diagnosis, fault tolerant control and system identification.

Cristian Secchi graduated in Computer Science Engineering at the University of Bologna in 2000 and he received his Ph.D. in Information Engineering in 2004 from the University of Modena and Reggio Emilia, where he is currently Associate Professor. His Ph.D. thesis has been selected as one of the three finalists of the 5th Georges Giralt Award for the best Ph.D. thesis on robotics in Europe. He participated to the CROW project, selected as one of the finalists for the 2010 EUROP/EURON Technology Transfer Award for the best Technology transfer project in Europe. He has been an Associate Editor for the IEEE Robotics and Automation Magazine, for the IEEE Transactions on Robotics and IEEE Robotics and Automation Letters. His research deals with human-robot physical interaction, telerobotics, mobile robotics and surgical robotics.

Cesare Fantuzzi received the M.S. degree in Electrical Engineering and Ph.D. degree in System Engineering from the University of Bologna (Italy) in 1990 and 1995, respectively. Prof. Fantuzzi is a Full Professor of Automatic Control at University of Modena and Reggio Emilia. He has held a variety of positions as project leader in several applied research programs developed in cooperation with small, large and multinational companies in the field of packaging machinery builder and manufacturing sector. His research interests are principally in the theory and application of discrete-event control for automatic machinery and in the modeling and control of Mechatronics system and robots.



UNIMORE
UNIVERSITÀ DEGLI STUDI DI
MODENA E REGGIO EMILIA

Dipartimento di Scienze
e Metodi dell'Ingegneria

Federica Ferraguti
Assistant Professor
Department of Science and Methods of Engineering
University of Modena and Reggio Emilia
via G. Amendola 2, 42122
Reggio Emilia , Italy
e-mail: federica.ferraguti@unimore.it

Reggio Emilia, August 8, 2019

Subject: Submission of a research paper to the Robotics and Autonomous Systems (RAS) journal

Dear Editor-in-Chief,

We declare that we have no conflict of interest in the authorship or publication of this contribution.

Sincerely

Federica Ferraguti.

Federica Ferraguti