IEEE Robotics & Automation Magazine (RAM) paper, presented at ICRA 2023, London, UK. Cite as RAM paper.

Safety and Efficiency in Robotics: the Control Barrier Functions Approach

Federica Ferraguti, Chiara Talignani Landi, Andrew Singletary, Hsien-Chung Lin,
Aaron Ames, Cristian Secchi, Marcello Bonfè

***Abstract*—This paper aims at presenting an introductory overview of the theoretical framework of Control Barrier Functions (CBFs) and of its application to the design of safety-related controllers for robotic systems. The article starts by describing the basic concepts of CBFs and how they can be used to build optimization problems embedding CBF-based constraints, whose solution corresponds to the control input enforcing desired safety properties into the behavior of the controlled system. Simple examples, understandable for readers with a basic background in control theory and accompanied with source code for their simulation, are used to highlight the appealing features of the CBF-based approach. Then, more complex formulations applicable to robotic manipulators are introduced, recalling recent literature results allowing to implement a robust design methodology. Finally, it is shown that CBFs are suitable for the implementation of safe human-robot collaboration in a realistic industrial scenario, by means of the experimental validation in an industrial setup for collaborative robotics.**

I. Introduction

The concept of safety in robotics may have different interpretations, according to the application domain or even the kind of robots involved, like manipulators, mobile platforms, flying drones or walking humanoids. Safety-critical systems are those systems for which safety is of paramount importance and becomes one of the main design parameters. Examples of safety-critical applications in robotics are provided in [1] and include, e.g., collision avoidance in human-robot collaboration, stabilization of walking robots, adaptive cruise control in autonomous driving.

Whatever interpretation is considered, embedding safety objectives in the design of a controller for a robotic system requires a formal framework based on solid mathematical tools, so that the properties of the designed system can be theoretically verified before a potentially dangerous practical validation. A valuable example of such a formal framework is represented by the definition of Control Barrier Functions (CBFs), introduced in [2] as a means to constrain the behavior of a generic dynamic system and then exploited as a design tool for safe controllers in a multitude of applications. A recent survey on theoretical aspects of Control Barrier Functions and related practical case studies can be found in [1]. In this paper, we focus on the robotics field and we first present some examples at a basic level, but analysed with a detailed constructive approach, of safety constraints that can be enforced using CBFs. The basic examples presented in the next sections can also be fully simulated by interested readers, using source code available on Code Ocean[1]. Once highlighted the key features of the CBF-based approach, we recall the peculiar aspects to be considered before applying this approach on robotic manipulators for industrial environments, particularly those related to human-robot collaborations. The peculiarities of industrial manipulators are addressed by introducing more complex formulations of the CBFs, either based on robot kinematics, like those presented in [3], or on the partial knowledge of the Euler-Lagrange equations of robot dynamics [4] and their parameters (i.e. robot link masses, inertia tensors, etc.), as shown in [5]. The latter approach is also recalled here, in Section VI. However, in many industrial applications even a partial knowledge on the dynamic parameters of the manipulator may not be available. To demonstrate that in this case the CBF-based approach is still practical, we conclude the paper with the presentation of an experimental validation on a real industrial robot.

II. Background

Safety is centered around the idea of the constrained behavior of a system. In general, safety constraints may be inherently related to the features of the system (e.g. the rotation of the joints, in a robot, is limited and violating the limit may cause damages) or by the environment that may interact with the system (e.g. a robot could be prevented from reaching a given position because of an obstacle). For a dynamical system, safety constraints can be thought of as those delimiting a *safe* region of its state space, in which the state must remain all the time. Such a region, is what we define here as the *safe set*.

Consider a generic nonlinear control-affine system, whose dynamic model is expressed as:

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

with $x \in \mathcal{D} \subset \mathbb{R}^n$ being the state, $u \in U \subset \mathbb{R}^m$ being the input, for which the compact set U contains the feasible control values, and in which $f : \mathbb{R}^n \to \mathbb{R}^n$ and $g : \mathbb{R}^n \to \mathbb{R}^{n\times m}$ are assumed to be locally Lipschitz functions. We desire a given safe set $\mathcal{S} \subset \mathcal{D} \subset \mathbb{R}^n$ to be **forward invariant** for

Federica Ferraguti and Cristian Secchi are with the Department of Science and Methods for Engineering, University of Modena and Reggio Emilia, E-mail: {name.surname}@unimore.it

Chiara Talignani Landi and Hsien-Chung Lin are with FANUC America Corporation, E-mail: chiaratlandi@gmail.com, hsien-chung.lin@fanucamerica.com

Andrew Singletary and Aaron Ames are with the Department of Computing + Mathematical Sciences, California Institute of Technology, E-mail: {asinglet,ames}@caltech.edu

Marcello Bonfè is with the Department of Engineering, University of Ferrara, E-mail: marcello.bonfe@unife.it

[1] https://codeocean.com/capsule/8942623/tree

IEEE Robotics & Automation Magazine (RAM) paper, presented at ICRA 2023, London, UK. Cite as RAM paper.

the system, that is, $\forall x_0 \in \mathcal{S}, x(t) \in \mathcal{S}\ \forall t \geq 0$. In other words, if the system starts inside of its safe set, it will remain in the safe set for all time [1], [2]. Forward invariance of the safe set can be achieved by means of the definition of Control Barrier Function.

Definition 1 ([6])**.** Let $\mathcal{S}$ be the superlevel set of a continuously differentiable function $h : \mathcal{D} \to \mathbb{R}$:

$$\begin{aligned} \mathcal{S} &:= \{x \in \mathbb{R}^n \ : \ h(x) \geq 0\}, \\ \partial\mathcal{S} &= \{x \in \mathbb{R}^n \ : \ h(x) = 0\}, \\ \text{Int}(\mathcal{S}) &= \{x \in \mathbb{R}^n \ : \ h(x) > 0\}. \end{aligned}$$

The function h is a **Control Barrier Function (CBF)** if $\frac{\partial h}{\partial x}(x) \neq 0$ for all $x \in \partial\mathcal{S}$ and there exists an *extended class* $\mathcal{K}$ *function* α[2] such that for the system (1) and for all $x \in \mathcal{S}$:

$$\sup_{u \in U} \Big[\underbrace{\frac{\partial h}{\partial x} f(x) + \frac{\partial h}{\partial x} g(x) u}_{\dot{h}(x,u)} \Big] \geq -\alpha(h(x)). \tag{2}$$

The CBF h can then be used to regulate the input of the system to keep the system (1) safe. However, the synthesis of a *safety* controller satisfying the condition of (2) may require a complex analytic procedure. Moreover, the safety controller is generally not meant to operate by itself, but rather to support a primary, possibly unsafe, controller designed to achieve standard control objectives, like stability, regulation or tracking. A minimally invasive strategy to obtain this result is to setup an optimization-based control problem, whose objective is to minimize the difference between the *desired*, but potentially unsafe, control value u_{des} and the actuated one, so that the latter satisfies the CBF constraint. More precisely, the safe control value is obtained by solving a Quadratic Program (QP) defined as follows:

$$\begin{aligned} u^* = \underset{u \in U}{\text{argmin}} \ & \|u - u_{des}\|^2 \\ \text{s.t.} \ & \frac{\partial h}{\partial x} f(x) + \frac{\partial h}{\partial x} g(x) u \geq -\alpha(h(x)). \end{aligned} \tag{3}$$

The role of the $\alpha(.)$ function is to provide to the designer a way to modulate the action of the CBF, depending on whether a more conservative or a more aggressive behavior is desired. The most common choice of $\alpha(h)$ is simply to multiply $h(x)$ by a scalar value $\gamma > 0$, that means $\alpha(h(x)) = \gamma h(x)$. In this case, as the value of γ increases the CBF is less conservative (i.e. the nominal controller is less constrained). Ideally, as $\gamma \to \infty$, the CBF would activate only on the boundary of the set. On the other hand, allowing the system to get closer to the safety boundaries using high γ values may also lead to their violation in practical applications, because of inevitable discrepancies (e.g. modeling inaccuracies, unknown disturbances, etc.) between the formal model and the actual behavior of the physical system. Of course, different choices of the $\alpha(\cdot)$ shape are possible, like $h(x)^3$ or $\tan^{-1}(h(x))$ multiplied again by a tunable parameter as γ, for which similar considerations on the conservativism or "aggressiveness" of the CBF, depending on the slope of $\alpha(\cdot)$ near the safety boundaries, would apply.

A. Issues in the formulation of CBFs and related solutions

Relative degree of $h(x)$: if the input of the system does not appear explicitly in the time derivative of the function h, the safety controller based on (2) cannot be designed. Formally, this means that a given function is a proper CBF *candidate* if its relative degree with respect to the input u is one[3]. The relative degree issue is particularly relevant for mechanical systems, just like robotic manipulators. For example, let us consider a one degree of freedom (DOF) translational mechanical system, whose displacement is denoted as $p \in \mathbb{R}$, controlled by an input force $u \in \mathbb{R}$. Assuming that the system is a pure mass M, its mathematical model is $M\ddot{p} = u$. A safety constraint defined only in terms of its displacement, for example requiring it to remain lower than a given constant p_M, would lead to the definition of the constraint function $h(p) = p_M - p$, whose time derivative is $\dot{h} = -\dot{p}$. Clearly, the latter is not directly influenced by the control input u, that would instead appear in the second order derivative (i.e $\ddot{h} = -u/M$). This confirms that the constraint function has a relative degree two and, therefore, that it is not a valid CBF candidate. A backstepping inspired approach to solve this issue was presented first in [7]. The corresponding formulation of CBF is then applied to enforce safety of human-robot collaborations in [8] and [3].

Actuation constraints: input/actuation constraints can be directly integrated into the QP formulation of CBF-based control [3], [7], [8].A more formal strategy to guarantee that such constraints do not compromise the existence of a solution to the QP problem is to define a *backup set*, that can be safely reached even in case of input saturation, a related *backup controller* and use the conditions derived from such definitions to enforce set invariance on an arbitrary safe set. Such a strategy has been applied to human avoidance for robotic manipulators in [9].

Sampled-data implementations or presence of delays and disturbances: even though the use of efficient QP solvers, so that the safe control value can be computed at high sampling frequencies, and proper tuning/shaping of the CBF or of the $\alpha(h)$ function in (2) is generally sufficient in practice, formal treatment of the discrete-time CBF formulation can be found in [10]. The issues related to the presence of time delays in the actuation of the control system or external disturbances adding to the controlled inputs, that may formally jeopardise the formal invariance of the safe set, are addressed in [11].

III. Control Barrier Functions as a Safety Filter

The solution of the QP defined by (3) fulfils the safety requirements expressed by a CBF and, at the same time, provides the smallest possible deviation from the nominal control value preserving the state of the system within its

[2]A continuous function $\alpha : (-b, a) \to (-\infty, \infty)$ is said to belong to extended class $\mathcal{K}$ for some $a > 0$, $b > 0$ if it is strictly increasing and $\alpha(0) = 0$.

[3]For a nonlinear control system, the relative degree of a function $h(x)$ is the number of times we have to differentiate h before the input u appears explicitly.

safe set. If the nominal control value does not violate the safety requirement, it would not be changed by the CBF-based optimization. In other words, the latter acts as a *safety filter*, that is constantly monitoring the behavior of the nominal controller and applying a corrective action only when necessary. Being the CBF defined on the measured state of the controlled system, such a safety filter is also a feedback controller, whose role in an overall control scheme can be depicted as in Fig. 1, assuming that the value u_{act} applied as the control input is equal to the solution of the QP at each control cycle u^*. Though the issues related to the convergence of the optimization algorithm to a feasible solution (within a reasonably small computation time) may be relevant, in the following we will assume that they can be addressed with a proper choice and setup of the numerical solver. For the introductory purpose of the paper, we will now focus on the formulation of the CBF for a very simple example, whose nominal controller is assumed to generate a constant value for u_{des}.

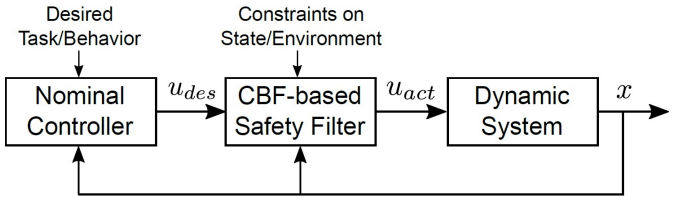

Fig. 1. The role of the CBF-based filter within the feedback control scheme for safe robotic systems.

A. A basic example: the double integrator case

Let us consider a further simplified version of the mechanical system described by $M\ddot{p} = u$. In particular, setting $M = 1$ we obtain the so-called *double integrator* system as $\ddot{p} = u$, in which $p \in \mathbb{R}$ is again the linear displacement and the input $u \in \mathbb{R}$ turns out to be the acceleration. Denoting the velocity as $\dot{p} = v$, we can assemble the state vector as $x = [p,\ v]^T$ and obtain a dynamic model in the form of (1), with the following substitutions:

$$f(x) = \begin{bmatrix} v \\ 0 \end{bmatrix} \quad g(x) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

A basic safety requirement on such a system could be to keep the values of each component of the state vector (i.e. position and velocity) within some given constant bounds:

$$\begin{aligned} p_m \leq p \leq p_M \\ v_m \leq v \leq v_M \end{aligned} \tag{4}$$

Such constraints clearly delimit a rectangular subset of the $(p\,v)$-plane. However, if we include also input constraints (i.e. the acceleration u is limited between u_m and u_M), then the feasible trajectories in the state space of the system should actually be confined within a safe set that can be depicted as in Fig. 2.

In particular, the curves delimiting the safe set in the upper right and lower left quadrants correspond to trajectories

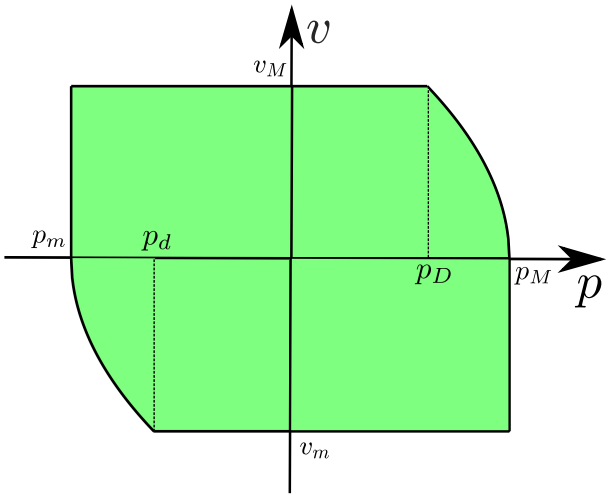

Fig. 2. Safe set for a double integrator with position, velocity and input (i.e. acceleration) bounds.

decelerating from the maximum and minimum velocities, respectively, until the system is stopped exactly at one of the extreme of the position interval. Denoting with p_D the position at which, if the velocity is equal to v_M (assumed to be positive), it is possible to stop in p_M applying the highest deceleration (i.e. u_m, which is assumed to be negative) for a given amount of time t_{dec}, the latter can be computed as follows:

$$v_M + u_m t_{dec} = 0 \quad \Rightarrow \quad t_{dec} = -\frac{v_M}{u_m}$$

Accordingly, the position p_D is:

$$p_M = p_D + v_M t_{dec} + \frac{1}{2} u_m t_{dec}^2 \quad \Rightarrow \quad p_D = p_M + \frac{v_M^2}{2u_m}$$

Similar considerations can be applied to compute the value of p_d, delimiting the boundary of the safe set in the lower left quadrant and corresponding to the position from which it is possible to stop in p_m, starting from the velocity v_m and applying the acceleration u_M (again, we assume that $v_m < 0$ and $u_M > 0$). The definition of a single smooth safety function of x, designed to be positive for states inside the green region of Fig. 2, to vanish at the boundaries and to be negative outside the interval as required by Definition 1 is not trivial. On one hand, if the QP defined by (3) has a single constraint, a closed-form solution can be derived from the Karush-Kuhn-Tucker (KKT) [12] conditions. On the other hand, a useful feature of the QP-based formulation of the controller is the possibility to embed quite easily multiple constraints, being bounds on the control value (i.e. $U = [u_m, u_M]$) among the most relevant ones. Even if the presence of multiple constraints prevents from computing a closed-form solution, the availability of several efficient numerical QP solvers makes the CBF-based control approach suitable for a large number of applications, taking into account a broad variety of safety conditions. Care must be taken, however, when multiple CBF constraints are implemented, to ensure that the resulting feasible region remains control invariant. In this example, such a

care is precisely the reason for computing the boundary curves depicted in Fig. 2.

With the latter considerations in mind, we now show how to exploit the extension of the QP with multiple CBF-based constraints to enforce a safe set composed by the intersection of multiple larger, but partially satisfying the overall safety requirement, sets. Indeed, the safe set of Fig. 2 can be formalized by combining the following functions:

$$\begin{aligned} h_1(x) = & \begin{cases} p_M - p + \frac{v^2}{2u_m} & \text{if } v > 0 \\ p_M - p & \text{otherwise} \end{cases} \\ h_2(x) = & \begin{cases} p - p_m & \text{if } v > 0 \\ p - p_m - \frac{v^2}{2u_M} & \text{otherwise} \end{cases} \\ h_3(x) = & \; v_M - v \\ h_4(x) = & \; v - v_m \end{aligned} \tag{5}$$

as CBF-based constraints. Thus, we can compute the safe control by solving the QP problem (3) with the constraint being applied to each function h_i, where $i \in \{1, 2, 3, 4\}$. It is worth noting that all the functions in (5) have relative degree one with respect to the input u. Moreover, despite their piecewise definition, the functions h_1, h_2 are continuously (though not infinitely) differentiable. On the other hand, the property of being infinitely differentiable is not required for a proper CBF implementation.

The plot in Fig. 3 shows the behavior of a double integrator, simulated with the code available on Code Ocean, whose desired input is a constant acceleration $u = -1$. Again, the desired input is filtered for safety by the optimization-based controller, whose CBF constraint is designed by choosing $\alpha(\cdot)$ as $\alpha(h(x)) = \gamma h(x)$. As can be seen, by increasing the value of γ the system is allowed to get closer to the boundaries of the safe set, delimited by $p_M = v_M$, $p_m = v_m = -1$, $u_M = 5$ and $u_m = -5$.

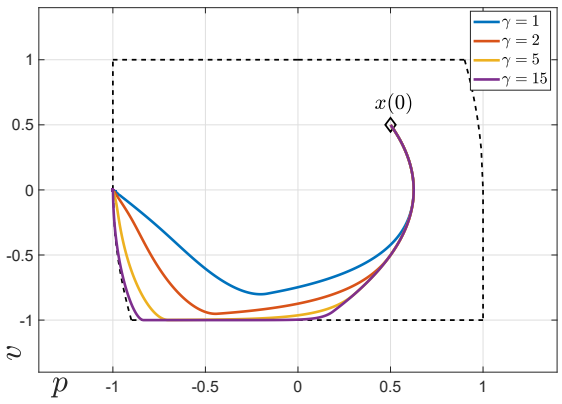

Fig. 3. Simulations with the companion code of the double integrator with position, velocity and input constraints.

IV. Safety applications in robotics

In this section we will address collision avoidance, since it is the most simple and representative example of robotic application where the CBF framework can be exploited to guarantee the safety constraints satisfaction. We will describe the example of an industrial manipulator, but the same approach can be exploited in the case of robot-robot collision in multi-robots systems. More complex formulation of the CBF framework may be required to address real-world human-robot collaboration, but this will be treated in Section VI. It is worth noting that the CBF-based approach to collision avoidance could be compared with the one based on Artificial Potential Fields (APFs), that is a staple in robotics. Such a comparison is outside the scope of this paper, but interested readers may refer to [13], in which it is shown that APFs are a special case of CBFs (given an APF it is possible to obtain a corresponding CBF, the converse is not true) and, especially, that CBFs outperforms APFs in terms of smoothness of the robot behavior and robustness against model uncertainty and noise.

A. Collision avoidance: basic example

Let us consider a system with two degrees of freedom (DOF), but a *single integrator* dynamics, namely $\dot{x} = u$, in which $x \in \mathbb{R}^2$ is the position and $u \in \mathbb{R}^2$ is the input, corresponding to a velocity. The model of (1) adapts to this case with:

$$f(x) = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad g(x) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

The formulation of a CBF-based safety controller for obstacle avoidance, assuming as a simple case that the obstacle is a static point x_{obs} in the state space and that the unsafe region is a circular area with radius R around x_{obs}, starts with the definition of the function:

$$h(x) = \|x - x_{obs}\|^2 - R^2 \tag{6}$$

whose value is strictly positive if the position x is outside of the obstacle, is zero if x lies on the circumference delimiting the unsafe region and is otherwise negative[4].

Again, such a function would be a CBF if we can guarantee that the velocity input u is such that:

$$\dot{h}(x, u) = \frac{\partial h}{\partial x}\dot{x} = \frac{\partial h}{\partial x}u = 2(x - x_{obs})u \geq -\alpha(h(x)) \tag{7}$$

For this example, we will also consider a nominal controller designed to force the system towards a target position x_{tgt} by means of a feedback law. Therefore, the desired input is computed as $u_{des} = K(x_{tgt} - x)$ and the safety filter is given by the solution of the QP:

$$\begin{aligned} u^* = \underset{u \in U}{\text{argmin}} \; & \|u - u_{des}\|^2 \\ \text{s.t.} \; & \frac{\partial h}{\partial x}u \geq -\alpha(h(x)) \end{aligned} \tag{8}$$

[4]An equivalent choice for the candidate CBF could be the geometric distance of x from the unsafe region, namely $h(x) = \|x - x_{obs}\| - R$. However, using the one proposed in the text the detailed formulation of the time derivative (here omitted) is a bit simpler, as can be seen consulting the companion code on CodeOcean.

Considering as usual $\alpha(h(x)) = \gamma h(x)$, we obtained with the companion code the simulations shown in Fig. 4. As can be seen, smaller values of γ force the system to remain at a higher distance from the obstacle, while using the larger value (among those used in the simulations) the system almost reaches the boundary of the unsafe region.

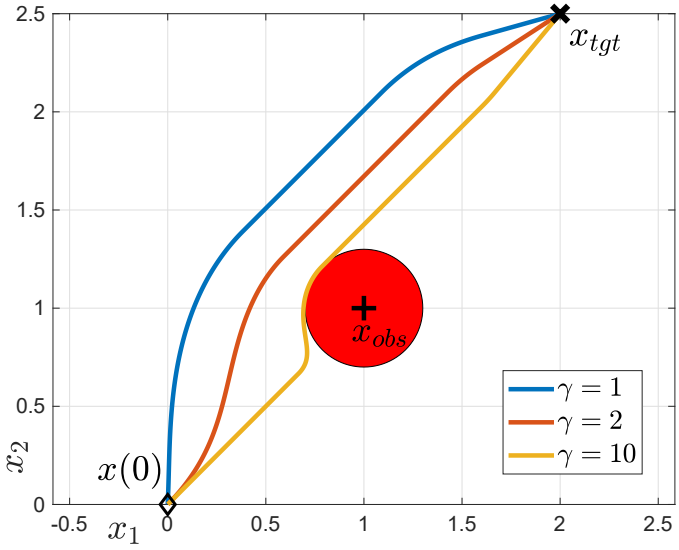

Fig. 4. Simulations of the 2D system avoiding an obstacle, depicted as the red disk centered in x_{obs}.

Plots comparing the desired (unsafe) and actuated (safe) control values can be visualized running the simulations code on Code Ocean.

The plot of Fig. 5 demonstrates that the system is always in a safe state, since the value of $h(x)$ remains strictly positive and, therefore, also the distance from the unsafe region.

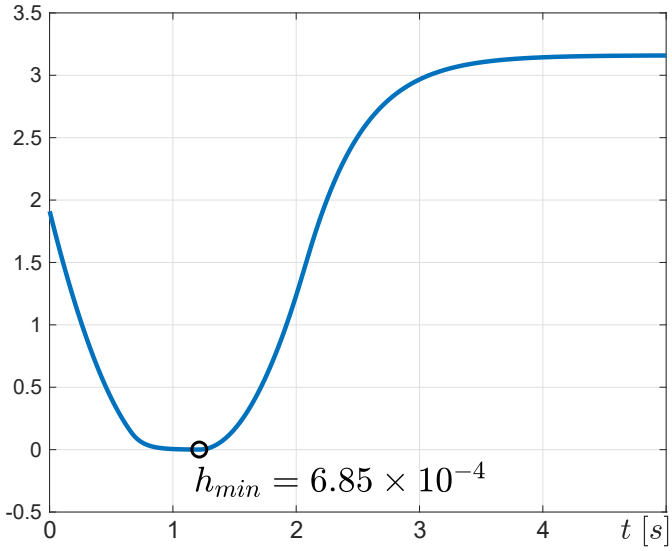

Fig. 5. Values of the CBF of (6) (simulation with $\gamma = 10$).

B. Collision avoidance for an industrial manipulator

The obstacle avoidance example can be easily extended to consider robotic systems with a more complex kinematic structure, like industrial manipulators. We will use the classical PUMA 560 as a case study. This choice is motivated by the fact that this manipulator, even though out of production since many years, is accurately described in many textbooks [14] and its kinematic and dynamic parameters are disclosed down to the deepest level of detail [15].

For this example, we will consider for obstacle avoidance only the center point of the robot wrist. Denoting the position in $\mathbb{R}^3$ of this point with p, its relationship with the angular positions of the robot joints is a nonlinear function, namely the forward kinematics function $p = F(q)$, where $q \in \mathbb{R}^6$ is the joint position vector (see [14], Section 3.7, for a detailed description of the PUMA 560 case). The latter will also be considered as the state of the system. Moreover, we will consider the vector of joint velocities as the input of the system, so that for the purpose of CBF-based control design the mathematical model is a straightforward extension of the 2-DOF single integrator, i.e. $\dot{q} = u$. The model of (1) fits again, considering a zero vector as the $f(\cdot)$ function and a 6×6 identity matrix as the $g(\cdot)$ function. Finally, we recall that $\dot{p} = J(q)\dot{q}$, where $J(q)$ is the configuration dependent Jacobian matrix of the manipulator, whose closed form expression can be obtained from differential kinematic analysis (see also [14], Exercise 5.5), noting that here we consider the center of the wrist, not the terminal flange of the manipulator, and we use only the translational part of the Jacobian matrix, i.e. the upper 3×6 part).

We can now formulate the safety objective related to the avoidance of a spherical obstacle, centered in p_{obs} and with radius R, as a CBF. Taking into account an additional parameter R_w, corresponding to the definition of a spherical safety volume around the robot wrist with that radius, we can define the following function:

$$h(q) = \|p - p_{obs}\|^2 - (R + R_w)^2 \tag{9}$$

whose derivative is related to the control input as follows:

$$\dot{h}(q, u) = 2\,(p - p_{obs})^T\, J(q)u \tag{10}$$

Denoting the Jacobian of the barrier with respect to the joint position vector as follows:

$$J_h(q) = \frac{\partial h}{\partial p}\frac{\partial p}{\partial q} = 2(p - p_{obs})^T J(q) \tag{11}$$

we can design the safety related controller by structuring the following QP, computing the joint velocity control input:

$$\begin{aligned} \dot{q}^* = \underset{\dot{q}\in\mathbb{R}^6}{\text{argmin}} \quad & \|\dot{q} - \dot{q}_{des}\|^2 \\ \text{s.t.} \quad & \dot{h}(q, \dot{q}) = J_h(q)\dot{q} \geq -\alpha(h(q)), \end{aligned} \tag{12}$$

where $\alpha(h(q)) = \gamma h(q)$ with $\gamma \in \mathbb{R}$, as we did in the previous examples. In this section and in the related simulations on CodeOcean, we assume that the velocity vector $\dot{q}^*$ is perfectly tracked by the robot joints.

The solution of the QP forces the center of the PUMA wrist to stay at a distance from p_{obs} higher than $R + R_w$, whatever is the desired joint velocity $\dot{q}_{des}$. This safety requirement is rather simple and may be acceptable for a large number of

applications (e.g. pick and place operations in an industrial scenario). However, the CBF-based safety controller can be easily extended to avoid collisions on any part of the manipulator, by considering multiple constraints (e.g. constraints on the distance between the middle point of each link and the obstacle) and, possibly, multiple obstacles.

The result of a simulation performed with the companion code, in which the desired input of the manipulator was set as a constant vector applying a velocity of 1 rad/s at the base joint and zero velocities for all the other joints[5], is shown in Fig. 6. The obstacle is represented as a red sphere, while the safety volume around the wrist is depicted as a green one. Both R and R_w were set equal to 0.1 m for the simulation. The 3D model of the PUMA 560 is drawn in the initial configuration, while the trajectory followed by the wrist when the safe control input is applied is shown as a blue line.

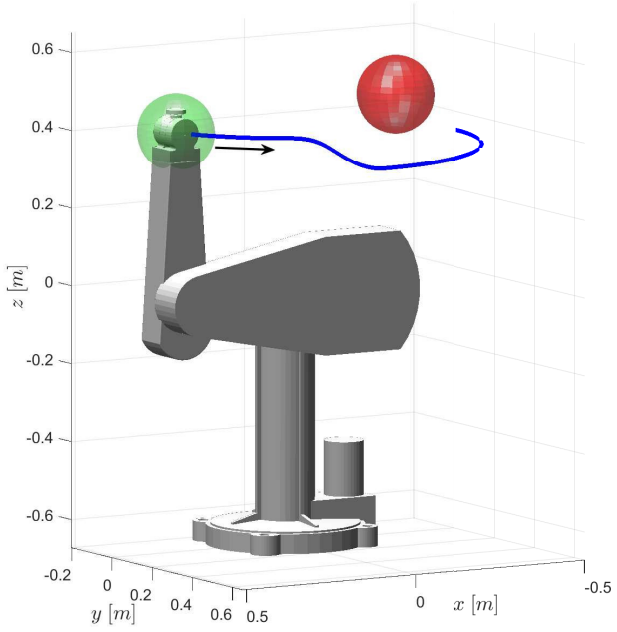

Fig. 6. Simulation of the 6-DOF manipulator avoding an obstacle: the green sphere is the safety volume around the wrist with radius R_w and the red sphere, centered in p_{obs} with radius R, is the obstacle.

The plot in Fig. 7 shows the velocity commands, limited to the first three joints, computed by the safety controller. Since the constraint is related to the wrist center, the safe velocities of the last three joints are always equal to their desired values, i.e. zero in this simulation, thus they are not shown. Running the simulations code on Code Ocean it is possible to visualize the values of the CBF of (9), which is always strictly positive, meaning that the green and the red spheres shown in Fig. 6 never intersect each other. Even though the formulation of the CBF is not, strictly speaking, equal to the geometric distance between the nearest two points on the surfaces of the spheres, it is also easy to relate the two quantities from $d = \sqrt{h(q) + (R + R_w)^2} - (R + R_w)$. Indeed, given the setting of R and R_w we can compute that the minimal distance reached during the simulation, corresponding to $h_{min} = 1.3 \times 10^{-3}$, is equivalent to 3.2 cm.

[5]The function $\alpha(\cdot)$ is chosen as in the previous examples, here considering only a value of $\gamma = 5$.

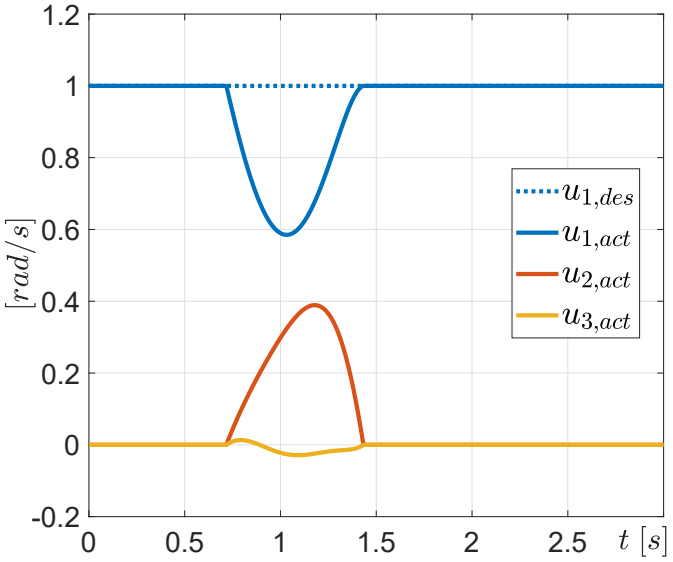

Fig. 7. Desired (unsafe) velocity of the first joint and actuated (safe) velocities of the first three joints of the manipulator considered in the example (the values of $u_{2,des}$ and $u_{3,des}$ are set to zero, thus they are not shown).

V. Peculiarities of Industrial Manipulators

Extending the CBF framework from the basic examples shown so far to a realistic industrial environment requires to address a number of additional issues, both theoretical and practical. First, it is important to remark that the safety objectives achieved through CBFs in the proposed examples rely on robot control hardware and software operating properly. From the industry point of view, instead, safety standards related to human-robot collaborations, like the ISO 10218-1 [6], require industrial robotic systems to be safe even in presence of hardware/software failures. However, fault tolerant CBF-based control is a highly advanced research topic, outside of the scope of this paper. On the other hand, the safety functions embedded in industrial robots, either certified as collaborative robots or not, are basically related to triggering an immediate stop, limiting the end-effector velocity or reducing the usable workspace within bounds fixed a priori and are executed on the basis of relatively low-level emergency detection mechanisms (e.g. push buttons, laser barriers, etc.). The safety concept expressed by the CBFs in the examples of this paper, instead, allows to explicitly address environment variations, assuming that an accurate perception system to measure them is available (e.g. 3D tracking of objects in the workspace).

That said, we will now highlight some peculiarities of industrial robots that have an important impact on the formulation of CBFs and then present two different approaches for the implementation of CBF-based safety control, one supported by simulations and the other related to experimental validation.

[6]http://www.iso.org

A first aspect that will be considered is the fact that robotic manipulators are governed by a second-order nonlinear dynamics, whose structure is well-known from the Lagrangian theory, leading to Euler-Lagrange equations [4]. The Euler-Lagrange model relates inputs at the level of joint torques to the velocities and accelerations of such joints, according to the inertial parameters of the mechanical structure (i.e. masses, centers of mass and inertia tensors for each link of the robot). Of course, such a model is much more complex than the first-order differential kinematics considered in Section IV-B. In the next section, it is shown that the safety controller described in that previous example, though conceptually valid and useful for an introduction to CBF, may not guarantee its theoretical behavior under all practical conditions. A more robust approach is then introduced in VI, summarizing the results recently presented in [5], in which a limited knowledge of the inertial parameters of the manipulator is exploited.

However, while Euler-Lagrange equations are known, an accurate estimation of their parameters, which is required for control design with an analytic approach like the CBF-based one, may be hard to achieve. This is especially true for control engineers working on the integration of an industrial manipulator in a given application, rather than on its manufacturing and assembly. On the other hand, the same industrial manipulators are generally equipped by their manufacturers with a factory calibrated controller, which is designed to accurately track the reference trajectories specified for either the pose of the end-effector or the position of the joints. Thanks to this embedded controller, it is possible to implement additional control loops, based on the feedback of some external devices (e.g. a vision system, a force/torque sensor) and designed according to purely kinematic models of the manipulator. In particular, a successful implementation of CBF-based safety control for robotic manipulators may exploit their second-order differential kinematics, so that accelerations are included in the model and, consequently, in the CBF formulation. Indeed, the embedded controller of an industrial manipulator is commonly quite efficient in compensating the nonlinearities of the robot dynamics, resulting in a behavior similar to the one of a (multi-dimensional) double integrator. These considerations motivated the approach presented in [3], in which safety is interpreted again as the absence of collisions between an industrial robot and a human body part, assuming that its velocity and acceleration are known from vision-based tracking. The same approach has been adopted for the development of an experimental validation setup at FANUC Advanced Research Laboratory, that contributed to this paper with the results discussed in Section VII.

VI. CBF-BASED OBSTACLE AVOIDANCE, BEYOND PURE KINEMATICS

In this section, we will address some of the potential concerns with utilizing the purely kinematic barrier functions, as done in Section IV-B on an ideal model of the PUMA 560 (i.e. a single integrator dynamics). Then, we will discuss possible alternatives for including the dynamics of the manipulator to provide more complete safety guarantees.

A. Limitations of purely kinematic barriers

On a real industrial manipulator, it is important to account for the tracking capabilities of its embedded controller. The QP of (12) is regulating the joint velocities of the manipulator, but on a physical system these velocities cannot be tracked perfectly. To demonstrate the consequences of this issue, we will now show the results obtained from simulations with a code including not only the kinematics of the manipulator, but also the full Lagrangian dynamics, computed with the Pinocchio solver[7], coupled with a simple proportional feedback controller for joint velocity tracking. Apart from these aspects, the CBF-based safety controller is the same of (12). Focusing on the values of the barrier function, we can analyze Fig. 8, showing three simulations executed with different tuning of the γ parameter. As can be seen from the plot, safety (i.e. CBF strictly positive all the time) is only guaranteed for certain values of γ. In particular, the lower values of γ result in safer trajectories, because the resulting joint velocity commands are more conservative and thus easier to track.

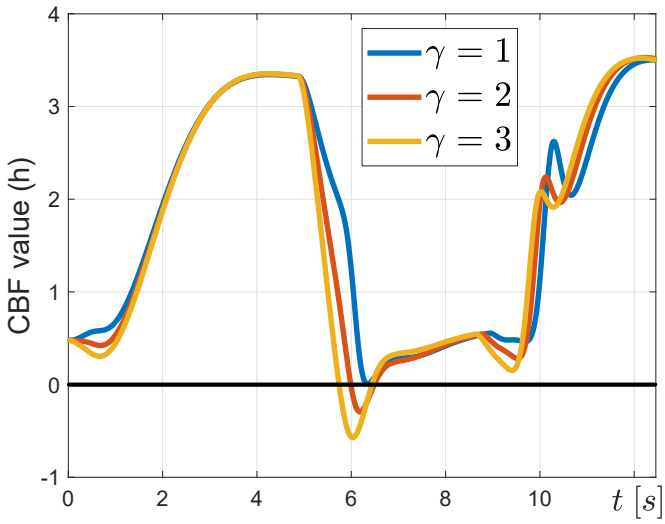

Fig. 8. Purely kinematic barrier function on the 6 DOF manipulator. Safety depends on choice of γ. The black line in the figure is the safe set boundary. The simulated robotic tasks and their execution are shown in the video http://youtube.com/watch?v=ilLRA7fpVBI.

B. Incorporating Dynamics for Safety Guarantees

It is possible to incorporate the dynamics of the manipulator to provide robustly safe guarantees for the true robotic system, as illustrated in both [9], [5]. The method showcased in [9] involves utilizing the knowledge of a *backup controller* that is able to keep the system safe to synthesize a CBF that guarantees safety in the presence of input bounds. However, this formulation is very notationally dense, and requires full knowledge of the underlying system dynamics, so it will not be showcased in this paper. Instead, we will outline the more recent approach introduced in [5].

[7] https://stack-of-tasks.github.io/pinocchio/

First, rather than simply considering the kinematics, we consider the Euler-Lagrange dynamic equation of the robot, given by:

$$D(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = u, \tag{13}$$

where $u \in \mathbb{R}^m$ is the vector of joint torques and $D(q)$ and $C(q,\dot{q})$ are respectively the inertia and Coriolis-centrifugal $m \times m$ matrices and $G(q) \in \mathbb{R}^m$ is the gravity torque vector. By incorporating the kinetic energy of the system into the barrier formulation, we can generate a so-called **energy-based safety constraint**, defined as :

$$h_D(q,\dot{q}) := -\frac{1}{2}\dot{q}^T D(q)\dot{q} + \gamma_e h(q) \geq 0. \tag{14}$$

in which $\gamma_e > 0$ is an additional scalar parameter to be properly tuned. Thanks to the kinetic energy term, we ensure that the CBF has a relative degree 1 with respect to the input, meaning that a purely positional constraint can be enforced. Due to the inclusion of the system dynamics through the kinetic energy term, we can now guarantee the safety of the true system described by (13). Since the kinetic energy term is positive semidefinite, we guarantee the fact that $h \geq h_D/\gamma_e$, meaning that safety is maintained ($h \geq 0$) whenever $h_D \geq 0$. Moreover, by taking γ_e large, we get a set that approaches the nominal safety set described by $h(q)$.

In many situations, it can be assumed that the embedded controller of the industrial robot implements a feedback law to compute the joint torques required to track the velocity commands, that we may denote as $\dot{q}_d$. The simplest choice of a feedback law is $u = -K_{\text{vel}}(\dot{q} - \dot{q}_d)$, where K_{vel} is an $m \times m$ feedback gain matrix. Assuming further that the underlying torque controller is known (or can be estimated), the CBF constraint can be implemented at the kinematic level:

$$\begin{aligned} \dot{q}_d^* = \ & \underset{\dot{q}_d \in \mathbb{R}^n}{\text{argmin}} \ \ \|\dot{q}_d - \dot{q}_{\text{des}}\|^2 \\ \text{s.t.} \ & \underbrace{\gamma_e J_h \dot{q} + \dot{q}^T K_{\text{vel}} \dot{q} - \dot{q}^T K_{\text{vel}} \dot{q}_d + G^T \dot{q}}_{\dot{h}_D(q,\dot{q},\dot{q}_d)} \geq -\gamma h_D, \end{aligned} \tag{15}$$

where the constraint embeds the knowledge of the K_{vel} feedback gain matrix coming from the feedback law.

By implementing the energy-based CBF constraint, safety can be guaranteed for all values of γ. To demonstrate this, we apply the energy-based CBF condition to a value of γ that previously resulted in a safety violation. As can be seen from Fig. 9, plotting the values of the CBFs, obstacle avoidance is guaranteed thanks to the CBF-based safety filter because $h(q)$ is always strictly positive. Moreover, the added computation time is minimal, with each QP solution taking roughly 19 μs on average (measured on a PC with Intel 9700K 3.7 GHz processor and 32GB of RAM). Indeed, QPs are computationally efficient and allow for implementation on the hardware in real-time. This is one of the main reasons why CBFs or QPs are highly sought after in industry.

The guaranteed safe condition is valid for any value of γ_e, but increasing this value improves the performance of the system, especially in terms of joint velocity tracking. While there is no theoretical upper bound on the value of γ_e such that safety is maintained, it cannot be so large that it results in numerical instability. Moreover, larger values of γ_e may lead to higher joint velocities, potentially unreachable because of the physical limitation of robot actuators. Such a limitation may be addressed as proposed in [3], [7], [8] or [9], as already discussed in Section II-A. The size of the kinetic energy term relative to the scaled safety set is irrelevant, as long as it remains positive.

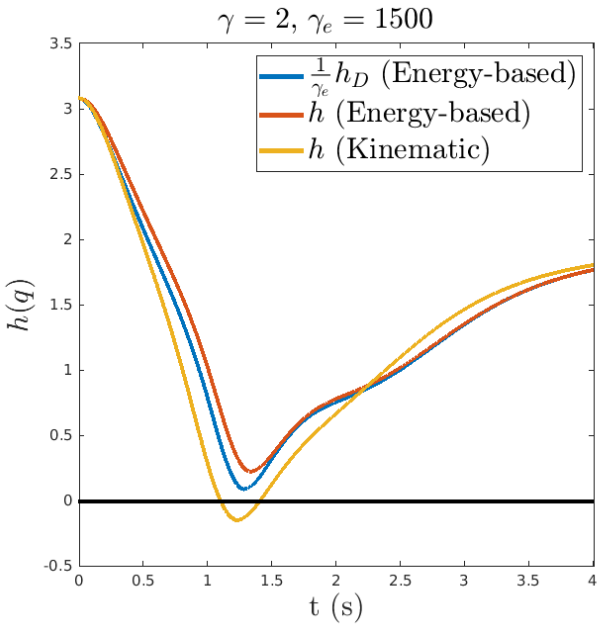

Fig. 9. Energy-based kinematic CBF on the 6 DOF manipulator: values of the CBFs during a simulation with $\gamma_e = 1500$. Obstacle avoidance is guaranteed since $h(q) > 0$ all the time. See http://youtube.com/watch?v=ilLRA7fpVBI for the video, and https://github.com/DrewSingletary/manipulator_asif_ros for the code.

VII. Experimental validation of CBF-based safety with kinematic control and delay compensation

This section will present the results of an experimental validation of the CBF-based control approach on a realistic industrial environment, setup by FANUC Advanced Research Laboratory.

A. Experimental System Setup

To show the effectiveness and feasibility of Control Barrier Function approach for obstacle avoidance, the experiments were performed on a FANUC CR-15*i*A collaborative robot[8]. Note that the FANUC CR and CRX series robots are certified to meet with the requirements of international standard ISO 10218-1, which reduce the human subjects' risk in the experiment. The experimental setup is shown in Fig. 10: while the robot moves in the workspace, its joint positions are retrieved from the robot controller and the positions of the control points are computed using direct kinematics. In the meantime, a VICON Motion Capture (MoCap) System[9] is used to track

[8]https://www.fanucamerica.com/products/robots/series/collaborative-robot/cr-15ia-cobot

[9]https://www.vicon.com/hardware/cameras/

the dynamic object at high frequency (100 Hz) by placing some reflective markers on the obstacle to be tracked.
It is worth mentioning that the perception part is not the main focus of this paper: in an industrial scenario, the MoCap system can be substituted by commercial 3D cameras (e.g. Microsoft Kinect or Intel Realsense) or safety sensors (e.g. laser scanner or Pilz Safety Eye). We chose a MoCap system because it provides an accurate obstacle position information at high-frequency. This allows to decouple the performances of the perception system from the control one, better highlighting the results of Control Barrier Functions.

The control software architecture is implemented on an external PC using OROCOS [10] and ROS [11] frameworks as shown in Fig. 11. The CBF-based safety optimization algorithm runs at the same frequency of robot low-level communication. The optimization problem is solved online using C++ code generated by CVXGEN [16]. Remote Motion Interface is a FANUC software option to online stream the position command to the robot controller from the external PC. High Speed Position Output is another FANUC software option that enables the external device to receive the robot joint position from robot controller in real-time[12]. Vicon_bridge [13] is a ROS node that can lively stream the position of an object tracked by the VICON system. Since the robot interface and the sensor interface only allow stream the position commands and measurements, it is necessary to perform integration and differentiation in order to get the appropriate input and output in between each module. To be more specific, the CBF-based optimization computes the safety filtered joints acceleration, which is then integrated to the position command q_{cmd}, whereas the measurements of the robot joints position q_{cur} and the tracked position of the obstacle p_{obs} are differentiated to obtain their respective velocity and acceleration. In this experiment, velocities and accelerations are estimated by using a Savitzky-Golay filter [17].

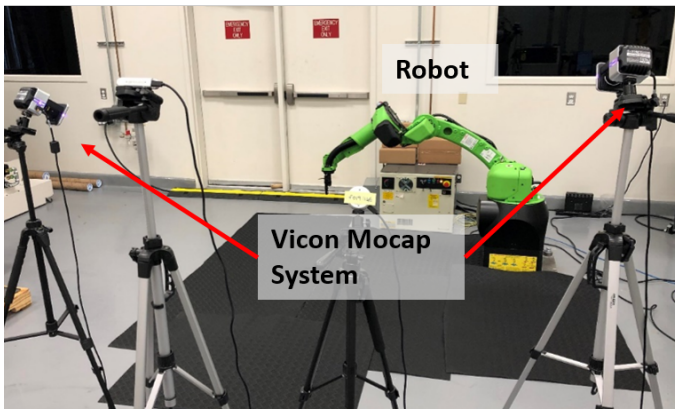

Fig. 10. Experimental setup including FANUC CR-15iA collaborative robot and Vicon Motion Capture system for obstacle tracking.

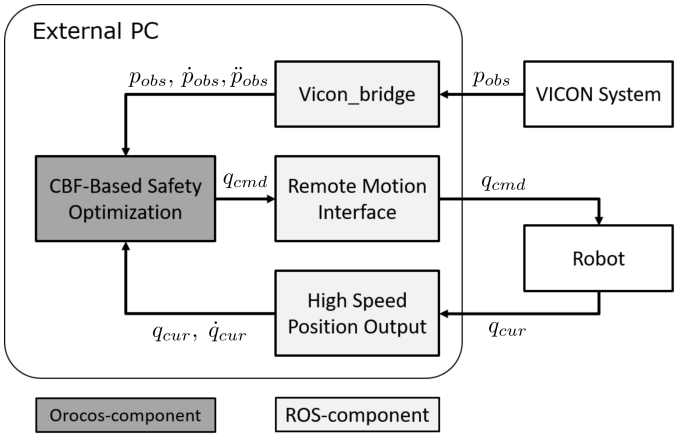

Fig. 11. The control software architecture that includes the CBF-based safety optimization, the communication modules to the robot controller (Remote Motion Interface and High Speed Position Output) and to the Mocap system (Vicon_bridge).

B. Experimental Validation

In order to verify whether the Control Barrier Function can be applied to an industrial manipulator even when a direct interface with the torque control level is not feasible, the robot system was modeled by the second-order differential kinematic model relating the Cartesian acceleration of a given control point p on the robot (e.g. the end-effector or the middle point of a specific link) and joints velocity $\dot{q}$ and acceleration $\ddot{q}$:

$$\ddot{p} = J(q)\ddot{q} + \dot{J}(q, \dot{q})\dot{q} \tag{16}$$

where $J(q)$ and $\dot{J}(q, \dot{q})$ are the Jacobian matrix of the robot's control point and its derivative. Moreover, we considered a fixed and known delay τ between current and commanded joints position, namely $q_{cur}(t) = q_{cmd}(t - \tau)$.

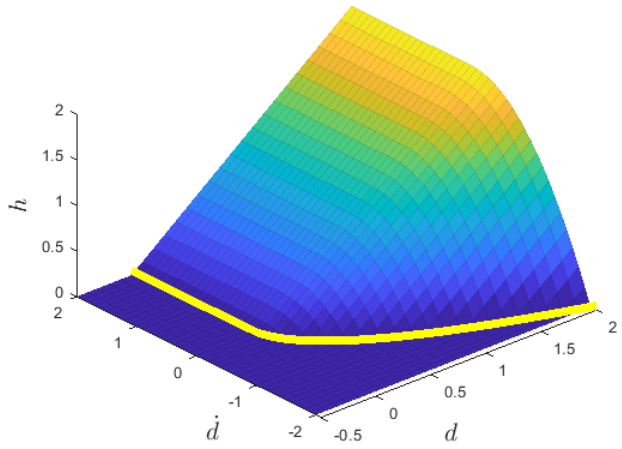

Fig. 12. The safe set of the experiment validation. The yellow curve is the boundary of the safe set ($h(d, \dot{d}) = 0$), where it is a straight line at $d = 0$ when $\dot{d} > 0$ and it becomes a parabolic curve ($d - \dot{d}^2/2a_{max}$) when $\dot{d} \leq 0$

[10] https://orocos.org/
[11] https://www.ros.org/
[12] The detail of Remote Motion Interface and High Speed Position Output can be found in the FANUC software manual. For more detail, please consult the FANUC Customer Resource Center (CRC) at https://www.fanucamerica.com/support/robot/crc-registration.
[13] https://github.com/ethz-asl/vicon_bridge

Indeed, for an industrial manipulator coupled to its embedded controller we can assume a quite efficient compensation of nonlinear dynamic effects, providing accurate tracking of commanded motions, but such an architecture would also show a fixed delay due to: communication between an external PC and the robot controller; low-level internal processing time. To mitigate this problem a solution including a simple predictor can be implemented [18], with the assumption that the low-level controller can perfectly track the commanded input (in our case, a joint position command).

For this experiment we used a Control Barrier Function $h(d, \dot{d})$ that depends on the distance d between the obstacle point and a control point on the robot and its time derivative $\dot{d}$. We adopted the concept from [19] to formulate the safe set in the experiment. The control barrier function is formulated as:

$$h(d, \dot{d}) = \begin{cases} d, & \dot{d} > 0, \\ d - \frac{\dot{d}^2}{2a_{max}}, & \text{otherwise}, \end{cases} \tag{17}$$

where a_{max} is assumed to be the robot maximum acceleration/deceleration. The safe set enforces the distance to be strictly positive, but does not enforce a limit on the derivative $\dot{d}$ if it is positive, since the condition $\dot{d} > 0$ means that robot and obstacle are moving away from each other. If, instead, $\dot{d}$ is negative, the obstacle is approaching to the robot, the additional term $\dot{d}^2/(2a_{max})$ is the safe deceleration distance that ensures the system staying within the safe set. Figure 12 shows the safe set and its boundary.

Given the CBF $h(d, \dot{d})$ and the modeling assumptions previously described, the QP implementing the safety filter can be formulated as described in [3]. The safe optimization is formulated as:

$$\begin{aligned} \ddot{q}^* = \arg\min_{\ddot{q}} \ & \|u_{des} - J\ddot{q} - \dot{J}\dot{q}\|^2 \\ s.t. \ & \dot{h}(d, \dot{d}) \geq -\gamma h(d, \dot{d}) \\ & \|\dot{q}\| \leq \dot{q}_{max} \\ & \|\ddot{q}\| \leq \ddot{q}_{max} \end{aligned} \tag{18}$$

where u_{des} is the desired control acceleration, $\dot{q}_{max}$ and $\ddot{q}_{max}$ are the magnitude of the joint velocity and acceleration limit. It is worth noting that the full formulation of the QP used for the experiment is similar to the one in [3]. Hence, we omit some details for brevity, include a CBF-based constraint for each link of the robot that may collide with the obstacle, considering of course different distances and Jacobian matrices in each constraint.

In the experiment, the human operator grabs a box and moves the box around the collaborative robot when the robot is tracking along a straight line back and forth. The box top right corner is tracked by the MoCap system.The execution of the experiment can be seen in the accompanying video.

To further analyze the behavior of the control barrier function, a set of experiments is conducted by using the same recorded VICON data, but different Control Barrier Function parameters. In this way, we can compare the behavior change while tuning the parameters. Here, we will focus on the value of a scalar γ multiplying the value of the barrier function in the constraint relating h and $\dot{h}$, as was considered in all of the simulated examples presented before. Figure 13 shows the trend of the Control Barrier Function according to different values of γ. As depicted, a lower γ is more conservative, meaning that a greater distance from the obstacle is kept, while a higher γ pushes the system on the boundary ($h = 0$) faster. Finally, Fig. 14 represents the robot end-effector trajectory when performing collision avoidance with respect to the object movement (light-blue dotted line). By using a predictor the value of the commanded position (colored solid line) and the actual position tracked by the low-level control (colored dashed line) are practically coincident. By tuning the γ parameter, the resulting obstacle avoiding behavior can be more or less conservative.

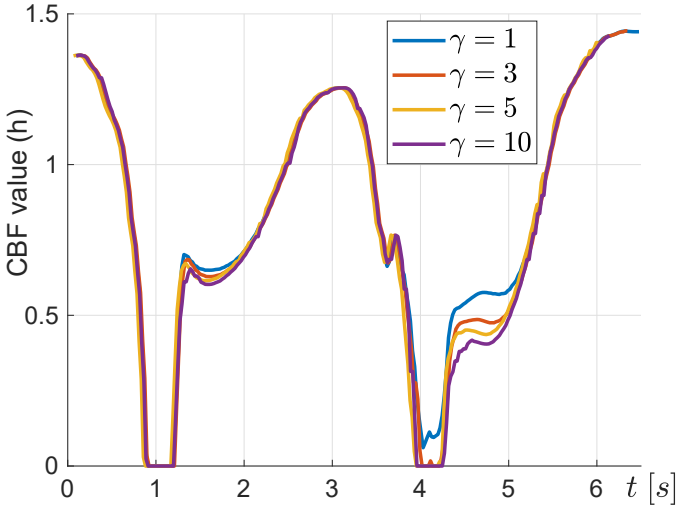

Fig. 13. Evolution of Control Barrier Function over time, according to different values of γ.

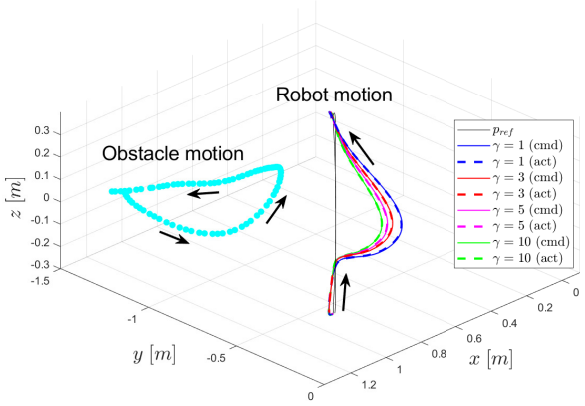

Fig. 14. Robot end-effector trajectory according to different values of γ: colored solid line represents commanded position and colored dashed line represents the actual position tracked by the low-level control. The black solid line represents robot nominal trajectory. The light-blue dotted trajectory represents the obstacle movement.

VIII. Discussion and Conclusions

This paper has presented the use of Control Barrier Functions as a tool to design safety-related constraints for robotic systems and how to exploit such constraints in an optimization-based approach to safety control. First, the basic concepts of CBFs together with simple and generic examples of safety-related control design were introduced. Then we focused on both theoretical and practical issues involved by the application of a CBF-based safety control approach to robotic manipulators, especially for the avoidance of collisions that may endanger humans during their collaboration with a robot in an industrial environment. For this purpose, we recalled that manipulators are governed by a complex and nonlinear dynamics, but that can be analytically described. The knowledge of robot dynamics can be exploited in the formulation of barrier functions providing a formal guarantee of the desired safety property, namely collision-free motion in presence of humans. On the other hand, we also remarked that a precise knowledge of parameters in the dynamics of an industrial robot may not be fully disclosed to its end-users or system integrators, possibly in charge of implementing application-specific and safety-related software. Therefore, we included an experimental validation of a CBF-based safety controller designed on the basis of the kinematic model of the manipulator, which is easier to calibrate, and relying on communication features of the industrial robot controller that are commonly available to all of its users.

In this final section, we would like to emphasize that collision-avoidance is an important feature for safe robot control in general and safe human-robot collaboration in particular. However, different definitions of safety in the latter context may be applicable. For example, an increasing number of research studies and practical applications are addressing the safety requirements defined by the ISO Technical Specification 15066[14]. In the ISO/TS 15066, specific definitions are given to the operating modes in which contacts between humans and robots are admitted, namey *Hand Guidance* (HG) and *Power and Force Limitation* (PFL). In the first mode it is the human that voluntarily touches the robot to drag it for teaching purposes, while in the second accidental impacts during the execution of a robotic task are considered. If such events happen, the robot must limit the force and pressure applied to the involved human body part, according to values specified by the ISO/TS 15066 itself on the basis of studies on pain onset levels. The requirements on force and pressure can be transformed into constraints on the relative velocity between human and robot in case of impact, according to the guidelines given in the annex of the ISO document. The analysis of such safety specifications may then lead to a CBF-based formulation, following the suggestions given in this tutorial. A first work in this direction is described by [20], adopting the same approach used for the experimental validation described in this paper. Further research on the application of Control Barrier Functions as a tool to guarantee safety in either human-robot interaction or other domains of robotics and on the practical realization of CBF-based efficient safety controllers is therefore easily predictable and actually suggested by the authors.

[14] http://www.iso.org

IX. Acknowledgments

The authors would like to thank FANUC America Corporation, Tetsuaki Kato, the General Manager of FANUC Advanced Research Laboratory, and Jason Tsai, the Advisor of FANUC Advanced Research Laboratory.

References

[1] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications," in *2019 18th European Control Conference (ECC)*, 2019, pp. 3420–3431.

[2] P. Wieland and F. Allgöwer, "Constructive safety using Control Barrier Functions," *IFAC Proceedings Volumes*, vol. 40, no. 12, pp. 462 – 467, 2007, 7th IFAC Symposium on Nonlinear Control Systems.

[3] F. Ferraguti, C. Talignani Landi, S. Costi, M. Bonfè, S. Farsoni, C. Secchi, and C. Fantuzzi, "Safety barrier functions and multi-camera tracking for human–robot shared environment," *Robotics and Autonomous Systems*, vol. 124, 2020.

[4] R. Featherstone and K. A. Publishers, *Robot Dynamics Algorithm*. USA: Kluwer Academic Publishers, 1987.

[5] A. Singletary, S. Kolathaya, and A. D. Ames, "Safety-critical kinematic control of robotic systems," *IEEE Control Systems Letters*, pp. 1–1, 2021.

[6] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs for safety critical systems," *IEEE Transactions on Automatic Control*, vol. 62, no. 8, pp. 3861–3876, 2017.

[7] S. Hsu, X. Xu, and A. D. Ames, "Control barrier function based quadratic programs with application to bipedal robotic walking," in *2015 American Control Conference (ACC)*, 2015, pp. 4542–4548.

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

[9] A. Singletary, P. Nilsson, T. Gurriet, and A. D. Ames, "Online active safety for robotic manipulators," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 173–178.

[10] A. Agrawal and K. Sreenath, "Discrete control barrier functions for safety-critical control of discrete systems with application to bipedal robot navigation," in *Robotics: Science and Systems*, 2017.

[11] A. Singletary, Y. Chen, and A. D. Ames, "Control barrier functions for sampled-data systems with input delays," in *2020 59th IEEE Conference on Decision and Control (CDC)*, 2020, pp. 804–809.

[12] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge University Press, 2004.

[13] A. Singletary, K. Klingebiel, J. Bourne, N. A. Browning, P. Tokumaru, and A. D. Ames, "Comparative analysis of control barrier functions and artificial potential fields for obstacle avoidance," *CoRR*, vol. abs/2010.09819, 2020. [Online]. Available: https://arxiv.org/abs/2010.09819

[14] J. J. Craig, *Introduction to Robotics: Mechanics and Control; Solutions Manual for Introduction to Robotics: Mechanics and Control*, 3rd ed. Pearson, 2005.

[15] P. Corke and B. Armstrong-Helouvry, "A search for consensus among model parameters reported for the puma 560 robot," in *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, 1994, pp. 1608–1613 vol.2.

[16] J. Mattingley and S. Boyd, "CVXGEN: A code generator for embedded convex optimization," *Optimization and Engineering*, vol. 13, no. 1, pp. 1–27, 2012.

[17] J. Chen, P. Jönsson, M. Tamura, Z. Gu, B. Matsushita, and L. Eklundh, "A simple method for reconstructing a high-quality ndvi time-series data set based on the savitzky–golay filter," *Remote sensing of Environment*, vol. 91, no. 3-4, pp. 332–344, 2004.

[18] I. Karafyllis and M. Krstic, *Predictor Feedback for Delay Systems: Implementations and Approximation*. Birkhäuser, 2017.

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

[20] F. Ferraguti, M. Bertuletti, C. T. Landi, M. Bonfè, C. Fantuzzi, and C. Secchi, "A control barrier function approach for maximizing performance while fulfilling to ISO/TS 15066 regulations," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5921–5928, 2020.