Enforcing Mobile Robot Safety Under Input Constraints

Francesco D'Orazio 1806836
Lorenzo Govoni 1796934
Riccardo Riglietti 1811527

March 2022

Contents

List of Figures

1 Introduction

Many cyber-physical systems are safety critical, namely, they require guarantees that safety constraints are not violated during operation. Safety is often modeled by defining a safe subset of the state space for a given system, within which the state trajectories must always evolve. Recently, methods such as Control Barrier Functions (CBFs), have become increasingly popular as a means of constructing and verifying such controllers. Quadratic Programs (QPs) enforcing CBF conditions have been used for synthesizing low-level controllers to ensure that system trajectories remain within the safe set [1].
In this project, we have addressed the issue of ensuring safety while having a good performance for a dynamical system subjected to input constraints solving the Adaptive Cruise Control (ACC) problem. Its formulation considers two vehicles, a leader and a (controlled) follower, where the second one would like to reach its maximum speed while remaining safe. The latter concept refers to the fact that the following car maintains always a certain distance from the leader avoiding collisions. In this framework the velocity profile of the vehicle in front is known, hence it can be included in the model formulation as a state if it's not constant.
Input limitations affect the solution in such a way that "naive" methods are no more able to always guarantee safety by letting the following vehicle enter in the unsafe region suggesting that a more conservative method is needed.
For designing a succeeding control law we have used an extension of the standard Control Barrier Functions theory building iteratively a suitable CBF applying the procedure reported in [2]. The result of this algorithm is an Input Constraint Control Barrier Function (ICCBF) that, by construction, is able to maintain safety under input constraints.
The standard method works defining safety as Hard Constraints, throught the CBF, and performance as Soft Constraint, managed with a Control Lyapunov Function (CLF). The results obtained with the latter methodology do not take into account the presence of limitations in the actuation system, since the procedure consists in clamping the control after its computation, and this reflects in the fact that the controlled vehicle starts to decelerate too late, exceeding from the boundary of the safe region.
Considering the same safety function $h(x)$ for both methods, we have discovered the superiority of ICCBF over CLF-CBF. Nevertheless [3] shows that, by changing on the fly the safety function, also CLF-CBF is able to maintain stability remaining on the boundary of the safe set. The authors solve an optimization problem in order to define the optimal CBF to be used in each situation and they also provide an explication of when each function must be used depending both on the instantaneous velocity of the vehicles and on the maximal fixed deceleration that they can apply by breaking as hard as possible.
The paper is structured as follows: section 2 presents some preliminaries on CBFs taking into account also the stability problem, CLF-CBF, and the presence of input constraints, ICCBF. Section 3 presents the definition of the problem of the Adaptive Cruise Control and the synthesis of the respective feedback controllers for ensuring safety with the corresponding simulation results.

2 Theory

Throughout this paper, we will suppose that we have a nonlinear input affine system:

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

with f and g locally Lipschitz, $x \in D \subset \mathbb{R}^n$ and $u \in \mathcal{U} \subset \mathbb{R}^m$ is the set of admissible inputs.

2.1 CBF

Safety can be framed in the context of enforcing invariance of a set, i.e. not leaving a safe set. In particular, we consider a set C defined as the superlevel set of a continuously differentiable function $h : D \subset \mathbb{R}^n \to \mathbb{R}$, yielding:

$$\begin{aligned} C &= \{x \in D \subset \mathbb{R}^n : h(x) \geq 0\}, \\ \partial C &= \{x \in D \subset \mathbb{R}^n : h(x) = 0\}, \\ \text{Int}(C) &= \{x \in D \subset \mathbb{R}^n : h(x) > 0\}, \end{aligned} \tag{2}$$

We refer to C as the safe set.
Let $u = k(x)$ be a feedback controller such that the resulting dynamical system

$$\dot{x} = f_{c\ell}(x) := f(x) + g(x)k(x) \tag{3}$$

is locally Lipschitz. To formally define safety, due to the locally Lipschitz assumption, for any initial condition $x_0 \in D$ there exists a maximum interval of existence $I(x_0) = [0, \tau_{max})$ such that $x(t)$ is the unique solution to *eq.(3)* on $I(x_0)$ [1].

Definition 1 *The set C is forward invariant if for every $x_0 \in C$, $x(t) \in C$ for $x(0) = x_0$ and all $t \in I(x_0)$. The system (3) is safe with respect to the set C if it is forward invariant.*

Definition 2 *The function $h(x)$ is a Control Barrier Function (CBF) if there exists an extended class-$\mathcal{K}_\infty$ function α such that for the control system (1):*

$$\sup_{u \in U} \left[L_f h(x) + L_g h(x) u \right] \geq -\alpha(h(x)) \;\; \forall x \in D \tag{4}$$

Now we can consider the set consisting of all control values that render C safe:

$$K_{cbf}(x) = \{u \in \mathcal{U} : L_f h(x) + L_g h(x) u + \alpha(h(x)) \geq 0\} \tag{5}$$

In order to synthesize the controller we want to modify an existing one in a minimal way so as to guarantee safety. Suppose we are given a feedback controller $u = k(x)$ for the control system *eq.(1)* and we wish that it makes the evolution be confined inside the safe set. We start by noticing that the conditions on safety given in *eq.(5)* are affine in u. Thus, we can consider the following Quadratic Program (QP) based controller that finds the minimum perturbation w.r.t. u:

$$\begin{aligned} u^*(x) = \underset{u \in \mathbb{R}^m}{\text{argmin}} \;\; & \frac{1}{2} ||u - k(x)||^2 \\ s.t. \;\; & L_f h(x) + L_g h(x) u \geq -\alpha(h(x)) \end{aligned} \tag{6}$$

where here we assumed that $U = \mathbb{R}^m$. Thus, when there are no input constraints, since we have a single inequality constraint, the *eq.(6)* has a closed-form solution given by the min-norm controller.

2.2 CLF-CBF

The dual objective to safety is the one of stabilizing the system. Suppose we want to stabilize the nonlinear control system *eq.(1)* to a point $x^* = 0$, i.e. driving $x(t) \to 0$. In a nonlinear context, this can be achieved by finding a feedback control law that drives a positive definite function, $V : D \subset \mathbb{R}^n \to \mathbb{R} \geq 0$, to zero. That is, if

$$\exists \; u = k(x) \;\; \text{s.t.} \;\; \dot{V}(x, k(x)) \leq -\gamma(V(x)) \tag{7}$$

where

$$\dot{V}(x, k(x)) = L_f V(x) + L_g V(x) k(x),$$

then the system is stabilizable to $V(x^*) = 0$, i.e. $x^* = 0$. Concretely, $V(x)$ is a Control Lyapunov Function (CLF) if it is positive definite and satisfies:

$$\inf_{u \in \mathcal{U}} \; [L_f V(x) + L_g V(x) u] \leq -\gamma(V(x)) \tag{8}$$

where γ is a class-$\mathcal{K}$ function.
The QP based formulation of safety-critical controllers suggests a means in which to unify safety and stability. We consider the following QP problem:

$$\begin{aligned} u^*(x) = \underset{(u,\delta) \in \mathbb{R}^{m+1}}{\operatorname{argmin}} \quad & \frac{1}{2} u^T H(x) u + p\delta^2 \\ s.t. \quad & L_f V(x) + L_g V(x) u \leq -\gamma(V(x)) + \delta \\ & L_f h(x) + L_g h(x) u \geq -\alpha(h(x)) \end{aligned} \tag{9}$$

where here $H(x)$ is any positive definite matrix (pointwise in x), and δ is a relaxation variable that ensures solvability of the QP as penalized by $p > 0$, i.e. to ensure the QP has a solution one must relax the condition on stability to guarantee safety.

2.3 ICCBF

Suppose the safe set C associated with $h(x)$ cannot be rendered forward invariant by any controller $u \in \mathcal{U}$. Then we define a function $b_1 : X \to \mathbb{R}$ and a set C_1 as follows [2]

$$\begin{aligned} b_1(x) = & \inf_{u \in \mathcal{U}} \; [\dot{h}(x, u) + \alpha_0(h(x))] \\ C_1 = & \{x \in X : b_1(x) \geq 0\} \end{aligned} \tag{10}$$

where the shorthand $\dot{h}(x, u) = L_f h(x) + L_g h(x) u$ is used and α_0 is some user specified extended class-$\mathcal{K}$ function. Now suppose there exists a Lipschitz continuous controller $u \in \mathcal{U}$ such that C_1 is forward invariant, then the set $C \cap C_1$ is also forward invariant. In defining $b_1(x)$, we allowed the user to specify the extended class-$\mathcal{K}$ function α_0. However not all class-$\mathcal{K}$ functions will admit a controller that renders C_1 forward invariant. Either a different class-$\mathcal{K}$ function could be used, or we can repeat the same steps. Consider the dynamical system expressed in *eq.(1)* with bounded control inputs $u \in \mathcal{U}$ and a safe set C defined by a function $h : X \to \mathbb{R}$, as per *eq.(2)*. Assume $h(x)$ is not a ZCBF, i.e. does not render C forward invariant. We define the following sequence of functions:

$$\begin{aligned} b_0(x) &= h(x) \\ b_1(x) &= \inf_{u \in \mathcal{U}} [L_f b_0(x) + L_g b_0(x) u + \alpha_0(b_0(x))] \\ b_2(x) &= \inf_{u \in \mathcal{U}} [L_f b_1(x) + L_g b_1(x) u + \alpha_1(b_1(x))] \\ &\vdots \\ b_{i+1}(x) &= \inf_{u \in \mathcal{U}} [L_f b_i(x) + L_g b_i(x) u + \alpha_i(b_i(x))] \\ &\vdots \\ b_N(x) &= \inf_{u \in \mathcal{U}} [L_f b_{N-1}(x) + L_g b_{N-1}(x) + \alpha_{N-1}(b_{N-1}(x))] \end{aligned} \tag{11}$$

where each α_i is some user-specified class-$\mathcal{K}$ function, and N is a positive integer. We assume the functions f, g, h are sufficiently smooth such that $b_N(x)$ and its derivative are defined. Each

function $b_i : X \to \mathbb{R}$ is a scalar function that only depends on the state. The time derivative $\dot{b}_i(x,u) = \frac{\partial b}{\partial x}\dot{x} = L_f b_i(x) + L_g b_i(x)u$ is still affine in u.

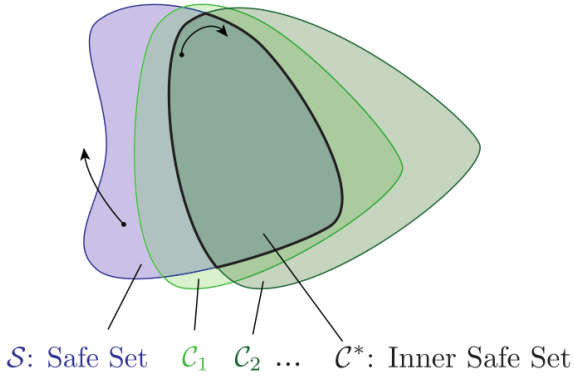


Fig. 1: *Visual representation of ICCBF method*

Next, we define a family of sets, *Fig.(1)*,

$$\begin{aligned} C_0 &= \{x \in X : b_0(x) \geq 0\}, \\ C_1 &= \{x \in X : b_1(x) \geq 0\}, \\ &\vdots \\ C_i &= \{x \in X : b_i(x) \geq 0\}, \\ &\vdots \\ C_N &= \{x \in X : b_N(x) \geq 0\}, \end{aligned} \tag{12}$$

and the intersection of these sets will be our safe region

$$C^* = C_0 \cap C_1 \cap \ldots \cap C_N \tag{13}$$

Definition 3 *For the above construction if there exists a class-$\mathcal{K}$ function α_N such that*

$$\sup_{u \in \mathcal{U}} \; [L_f b_N(x) + L_g b_N(x)u + \alpha_N(b_N(x))] \geq 0 \quad \forall x \in C^* \tag{14}$$

then $b_N(x)$ is an Input Constrained Control Barrier Function (ICCBF).

Consider the dynamical system in *eq.(1)*. Suppose $b_N(x)$ is an ICCBF associated with the class-$\mathcal{K}$ function α_N and with the inner safe set C^*. Let $H : X \to \mathbb{R}_+^{m \times m}$, where $\mathbb{R}_+^{m \times m}$ is the set of real $m \times m$ positive definite matrices, and $F : X \to \mathbb{R}^m$ be Lipschitz continuous cost functions. The solution to the following quadratic optimization problem

$$\begin{aligned} u^*(x) = \underset{u \in \mathbb{R}^m}{\operatorname{argmin}} \; & \frac{1}{2} u^T H(x) u + F(x)^T u \\ \text{subject to } & L_f b_N(x) + L_g b_N(x) u \geq -\alpha_N(b_N(x)) \\ & u \in \mathcal{U} \end{aligned} \tag{15}$$

yields an input-constrained feedback controller $u^* : C^* \to \mathcal{U}$ that renders C^* forward invariant. Thus, $h(x(t)) \geq 0$ for all $t \geq t_0$, i.e. the system trajectories always evolve within the safe set C.

3 Adaptive Cruise Control problem

The Adaptive Cruise Control problem is the problem of finding a suitable control for a vehicle that is following a vehicle in front, which moves in the simplest case at a known constant speed v_0 or in the more general case with a speed changing over time $v(t)$. The controller must prevent the follower from colliding with the leading vehicle (safety hard constraint HC), but should also allow it to accelerate to the speed limit when this requirement does not conflict with safety (performance soft constraint SC).

3.1 Safe Control Synthesis via Input Constrained Control Barrier Functions and CLF-CBF

We started by replicating the results obtained in [2] regarding the control of the following vehicle in an Adaptive Cruise Control problem. We simulated the problem adopting both the ICCBF and the CLF-CBF methods using the same dynamic model, as we will see in the following graphs the ICCBF manages to provide a safe control while the CLF-CBF method fails by letting the system enter in an unsafe region.

3.1.1 Dynamic Model

The formal representation of the dynamical system used in this section is presented as follow

$$\dot{x} = \begin{bmatrix} \dot{d} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} v_0 - v \\ -F(v)/m \end{bmatrix} + \begin{bmatrix} 0 \\ g_0 \end{bmatrix} u \tag{16}$$

The state is composed by $x = (x_1,\ x_2)^T = (d,\ v)^T$ where d represents the distance between the two subjects of the study whereas v is the velocity of the follower. Moreover in the model $F(v) = f_0 + f_1 v + f_2 v^2$ represents the aerodynamic and rolling drag in which the parameters $f_0,\ f_1$, and f_2 are fixed and have been computed empirically (*Tab.(1)*).
Our goal is to find a feasible profile of the control in order to maintain safety (HC) while adapting the velocity trying to reach the desired speed v_{max} (SC), under input limitation $u \in \mathcal{U}$ where $\mathcal{U}$ is defined as

$$\mathcal{U} = \{u : -u_{max} < u < u_{max}\} \tag{17}$$

3.1.2 ICCBF

We built the ICCBF with the iterative procedure described in *eq.(11)* selecting $N = 2$. We built the intermediate functions $b_i(x)$ iteratively starting with the safety function

$$b_0(x) = h(x) = x_1 - 1.8x_2 \tag{18}$$

and at each step computing the new CBF as a function of the previous one and its derivative. *eq.(18)* is constructed considering the desired Time Headway [4] which is expressed as the constraint $d/v_f \geq \tau_d$ specifying the parameter τ_d. A general rule says that the minimum distance between two cars is "half the speedometer", which translates in $\tau_d = 1.8$.
In order to apply this procedure we need to define N class-$\mathcal{K}$ functions, that the authors have set equal to

$$\alpha_0(h) = k_0 h, \qquad \alpha_1(h) = k_1\sqrt{h}, \qquad \alpha_2(h) = k_2 h \tag{19}$$

with parameters of α_i reported in *Tab.(1)*. After the derivation of $b_2(x)$ we need to check if the function is indeed an ICCBF, by solving the optimization problem defined as

$$\begin{gathered} \gamma = \text{minimise}_{x \in X}\ \sup_{u \in \mathcal{U}} [\dot{b}_N(x,u) + \alpha_N(b_N(x))] \\ \text{subject to } x \in C^* \end{gathered} \tag{20}$$

Variable	Description	Value
$d(0)$	initial distance between the vehicles	$100\ m$
$v(0)$	follower initial velocity	$20\ \frac{m}{s}$
m	mass of the following car	$1650\ Kg$
f_0	constant term of friction	$0.1\ N$
f_1	linear term of friction	$5\ N \cdot (\frac{m}{s})^{-1}$
f_2	quadratic term of friction	$0.25\ N \cdot (\frac{m}{s})^{-2}$
g_0	gravitational acceleration	$9.81\ \frac{m}{s^2}$
v_0	leader constant velocity	$13.89\ \frac{m}{s}$
v_{max}	maximum follower velocity	$24\ \frac{m}{s}$
u_{max}	maximum absolute value of input control as a fraction of g_0	0.25
$\gamma(V(x))$	class-$\mathcal{K}$ function for CLF-CBF	$10 \cdot V(x)$
k_0	parameter of α_0	4
k_1	parameter of α_1	7
k_2	parameter of α_2	2

Tab. 1: *Original parameters for simulating this section*

If the result of this optimization problem is a value of $\gamma \geq 0$ then we can conclude that b_N is a valid ICCBF. In our case evaluating *eq.(20)* using $b_2(x)$ gives a result $\gamma = 2.3686$ and this means that we can use it for solving the QP problem. This optimization procedure that is used to find a suitable control has been presented in *eq.(15)* and, in the current framework, assumes the form

$$\begin{aligned} u^* = &\underset{u \in \mathbb{R}}{\operatorname{argmin}} \ \frac{1}{2}\ (u - u_d)^2 \\ \text{subject to } &L_f b_2(x) + L_g b_2(x) u \ \geq -2b_2 \\ &u \in \mathcal{U} \end{aligned} \tag{21}$$

where $u_d(x)$ is the desired acceleration, computed from the Control Lyapunov Function

$$V(x) = (x_2 - v_{max})^2 \tag{22}$$

solving the equation

$$L_f V(x) + L_g V(x) u_d = -\gamma(V(x))$$

From *eq.(21)* we can derive the matrices to be passed to the MATLAB function `quadprog` used for the implementation

$$H = 1 \qquad F = -u_d \qquad A = -L_g b_2 \qquad b = L_f b_2 + k_2 b_2$$

Inside the loop, after the evaluation of the control strategy at the current iteration, the evolution of the system is simulated using the command `ode45` in order to know the values of the state to be used in the successive step.
As a confirmation of the correctness of our implementation, evaluating the problem using the same numerical parameters as [2] (reported in *Tab.(1)*), produces the same results, discussed in *Section 3.1.4*.

3.1.3 Control Lyapunov Function, Control Barrier Function (CLF-CBF)

In this section we are going to consider the problem of unifying safety (HC) with performance (SC) accordingly to the optimization problem *eq.(9)*.
Considering the Control Lyapunov Function *eq.(22)*, in our case the optimization problem reads as follows:

$$\begin{aligned} u^*(x) = \underset{u\in\mathbb{R},\delta\in\mathbb{R}_+}{\operatorname{argmin}} \quad & \frac{1}{2}u^2 + 0.1\delta^2 \\ \text{subject to} \quad & L_fV(x) + L_gV(x)u \leq -10V(x) + \delta \\ & L_fh(x) + L_gh(x)u \geq -2h(x) \end{aligned} \tag{23}$$

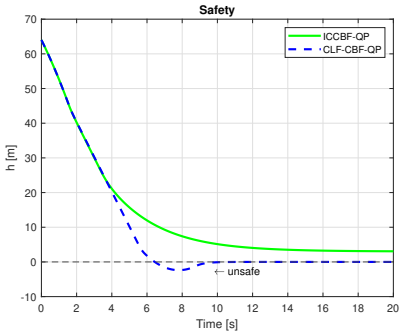
The control is found at each timestep by solving the optimization problem with `quadprog` using the matrices

$$H = \begin{bmatrix} 1 & 0 \\ 0 & 0.2 \end{bmatrix} \qquad F = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \qquad A = \begin{bmatrix} L_gV(x) & -1 \\ -L_gh(x) & 0 \end{bmatrix} \qquad b = \begin{bmatrix} -10V(x) - L_fV(x) \\ L_fh(x) + k_2h(x) \end{bmatrix}$$

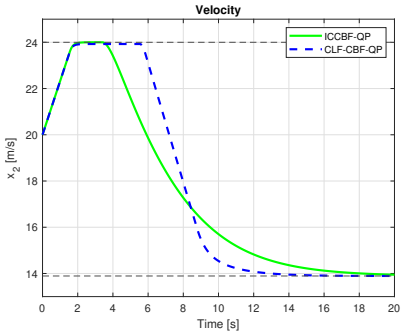
and clamping the solution of the QP such that $u^*(x)$ lies in the range of feasible control inputs, i.e. $u^*(x) \in U$.

3.1.4 Comparison between ICCBF and CLF-CBF

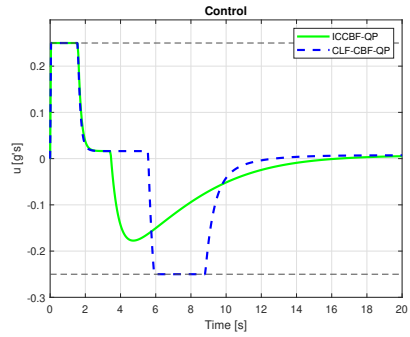
We can see from the graphs reported in *Fig.(2)*, generated using the parameters reported in *Tab.(1)*, that the ICCBF building technique is more conservative w.r.t. CLF-CBF based method since it allows to brake earlier maintaining safety throughout the whole simulation.
Indeed *Fig.(2a)* shows that that CLF-CBF method enters in the unsafe region at time $t \approx 6.6s$.



(a) *profile of the safety function*



(b) *profile of the velocity*



(c) *profile of the control*

Fig. 2: *Comparison between ICCBF and CLF-CBF*

3.1.5 Analysis of the Safe Regions using ICCBF

In this problem, the safe regions are functions of the state and since we are dealing with a $2-$dimensional system we are able to visualize them in a plane where on the $x-axis$ we have the state d and on the $y-axis$ the state v.
As presented in *eq.(13)* the safe set is defined as the intersection between all the positive regions which are the superlevel sets of the computed CBFs. In particular C_i is defined as in *eq.(2)*, so we have that ∂C_i is the portion of the plane defined when $b_i(x) = 0$.
In our implementation we have used the MATLAB command `area` in order to mark the area underneath the curve that, in our case, represents the boundary of the set.
For computing this frontier we have implemented two possible procedures depending on the CBF that we are considering:

1. Looking at the functions defined in *eq.(19)* and at the definition of b_0 we can notice that b_0 and b_1 are linear w.r.t. the state variables. In virtue of this the easiest method for computing the boundary is using linear interpolation to find the equation of the line.

2. The second way is to solve an unconstrained minimization problem. This choice is motivated by the fact that the function $b_i(x)$ can assume numerical values that can be either positive and real, or complex. Since we are looking for the values of the states for which it is equal to 0, we have solved an equivalent problem looking for the minima of $|b_i(x)|$ using the MATLAB command `fminunc` giving a numerical value to one variable of the state.

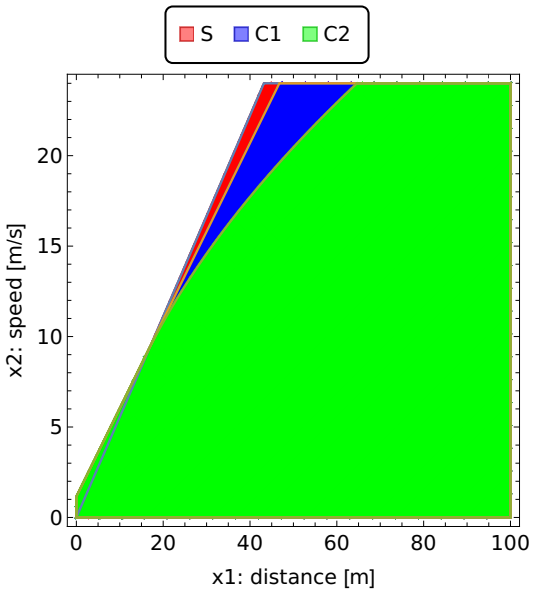


Fig. 3: *Safety regions with original parameters*

Fig.(3) presents the safe region computed by intersecting every superlevel set obtained starting from b_0, b_1, b_2 using the original parameters of *Tab.(1)*.

3.1.6 Effect of changing the alpha functions on the safe region

Given that there exist an infinite number of possible safe controllers for any given system, if safe control is at all possible, there is a margin for user choice in the iterative algorithm used to obtain the ICCBF starting from the safety function $h(x)$.
More precisely, in the iterative procedure, depicted in *eq.(11)*, the alpha functions (*eq.(19)*) are

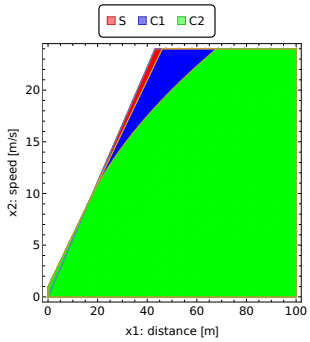
chosen arbitrarily, so we investigated what could happen by **(1.)** substituting the scaled square root function α_1 with a linear one or **(2.)** choosing different parameters $k_0,\ k_1,\ k_2$ for the corresponding α_i.

1. Applying the iterative procedure without the scaled square root function α_1, i.e. using all linear functions for α_i, did not give an ICCBF as output, as the optimization problem seen above *eq.(20)* yielded a negative value for γ so b_2 is not an ICCBF in this case.

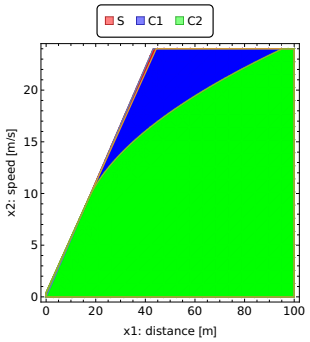
2. In order to see the effects of varying those $k_0,\ k_1,\ k_2$ parameters, we changed them one by one to observe their individual contribution on the definition of the safe regions as shown in the graphs below.

Studying the effect of changing the parameters k_0, k_1 and k_2

From the example in *Fig.(4)* we can see that increasing k_0 reduces the safety region at higher distances, this is reflected in the fact that in order to avoid collision with the vehicle in front, if we have a sufficiently high speed (near to the limit one), we have to keep an higher distance in order to remain safe. *Fig.(5)* shows that the choice of a larger k_1 results in a bigger safe region which is a less conservative situation w.r.t. the case in which we change just the value of k_0, since we are allowing the following vehicle to keep a smaller distance from the other one at higher speed. Finally, accordingly to the iterative construction of the b_i functions (*eq.(11)*), the parameter k_2 has no influence in the definition of the safe regions and so has no meaning to change it.

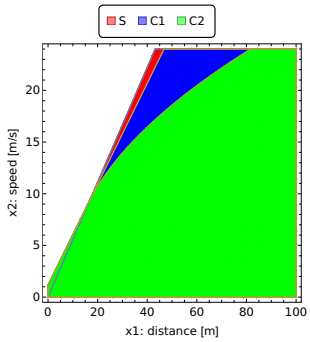


(a) $k_0 = 5\ k_1 = 7\ k_2 = 2$

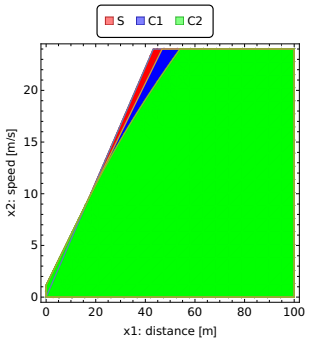


(b) $k_0 = 12\ k_1 = 7\ k2 = 2$

Fig. 4: *Safety regions changing* k_0



(a) $k_0 = 4\ k_1 = 5\ k_2 = 2$



(b) $k_0 = 4\ k_1 = 11\ k_2 = 2$

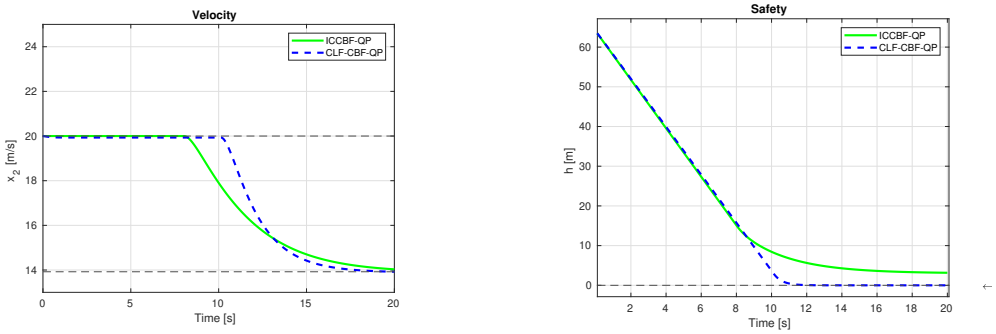
Fig. 5: *Safety regions changing* k_1

3.1.7 Testing the validity limits of the ICCBF

We changed some of the conditions in the simulation to check if the ICCBF method was still able to maintain safety in situations different from the original ones described in [2]. We tested the ICCBF method in two different conditions: alternative values for the target velocity have been used so to check if the case of an higher speed-limit is still handled safely; and we have generalized the problem by letting the leading vehicle to have a variable speed profile, reasonable scenario in real applications, to check if the model responds correctly to changes in the velocity of the vehicle in front.

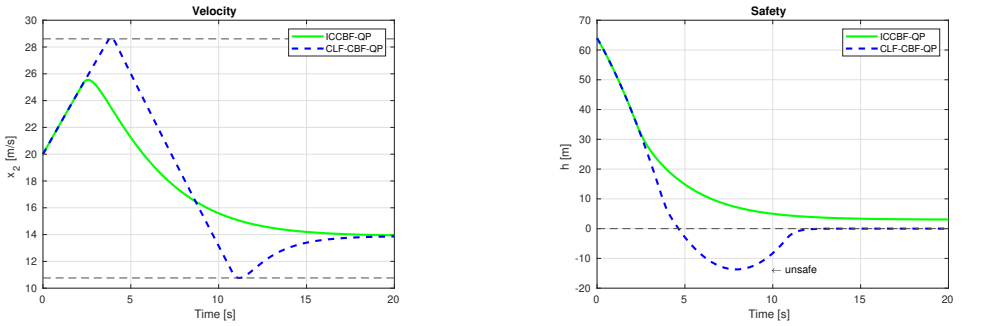
Changing the target velocity

An interesting experiment is changing the target velocity (speed limit) to a lower or higher value to compare the behaviour of the controller and check if the validity of the ICCBF is maintained (i.e. checking that safety is always guaranteed) under different conditions.
In the case of $v_{max} = 20$, *Fig.(6)*, ICCBF and CLF-CBF satisfy the safety condition, the latter one improves w.r.t. the original case, as we should expect, since lowering the maximum speed that the following vehicle can assume, renders the evolution of the system safer.
On the other hand, in the case of $v_{max} = 40$, *Fig.(7)*, the method with CLF-CBF fails in guaranteeing safety, we have worsened its performance since w.r.t the original case it enters in the unsafe region earlier at $t \approx 4.7$, instead of at $t \approx 6.6$ when $v_{max} = 24$, (*Fig.(2a)*).



(a) *Speed profile* (b) *Safety profile*

Fig. 6: *Target speed* $v_{max} = 20\ m/s$



(a) *Speed profile* (b) *Safety profile*

Fig. 7: *Target speed* $v_{max} = 40\ m/s$

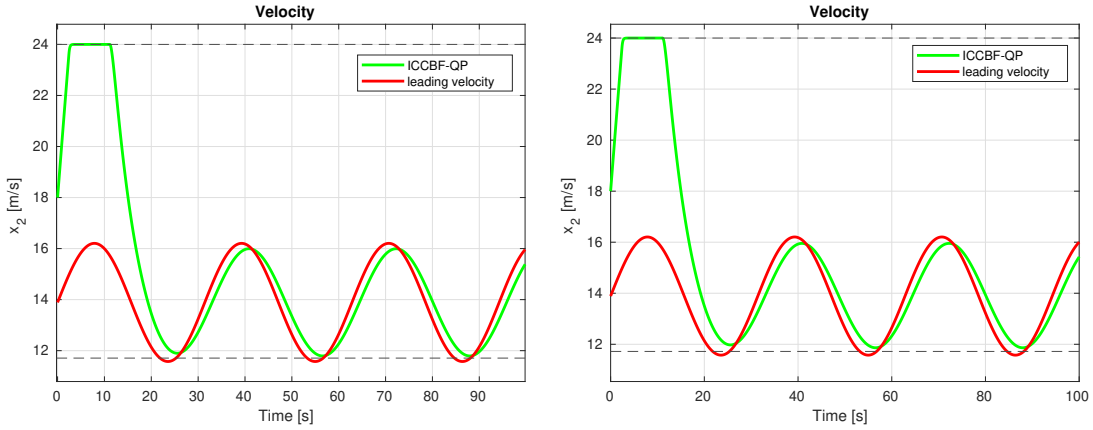
Generalization to a variable leader velocity

The original paper only studied the case of a constant speed for the leading car, we generalized the method to a varying $v(t)$ for the second vehicle by considering the augmented model *eq.(24)* that takes into account the possibility of accelerating or decelerating for the car in front. In the framework of the ICCBF, the presence of the leading acceleration of the model is reflected in the expression of the function $b_2(x)$, since it implicitly comes from the second derivative of the safety function $h(x)$. In order to see the effect of a time dependent $v(t)$ w.r.t the original situation we have considered two different scenarios:

1. **Sinusoidal leading velocity**
   Here we are going to assign to the vehicle in front a sinusoidal speed profile, that oscillates around v_0, which reflects in a cosinusoidal acceleration

$$v_\ell(t) = v_0 + \frac{v_0}{6}\sin(\frac{t}{5})$$
$$a_\ell(t) = \frac{v_0}{30}\cos(\frac{t}{5})$$

   where v_0 is listed in *Tab.(1)*.



(a) *two-state model* (b) *three-state model*

Fig. 8: *Sinusoidal speed profile* v_ℓ

In *Fig.(8a)* we have the evolution of the speed using the model in *eq.(16)*, while in *Fig.(8b)* there is the evolution of the velocity using the one in *eq.(24)*. The presence of the leading acceleration in the ICCBF $b_2(x)$ may suggest that the additional term a_ℓ could act as a "feedforward", but in this case there is no appreciable difference in the velocity profile.
We decided to investigate whether there exist situations where the augmented dynamical system (*eq.(24)*), that includes the leading acceleration, modifies the control. We put ourselves in a more conservative scenario by reducing all the original k_i parameters by a factor of 4.

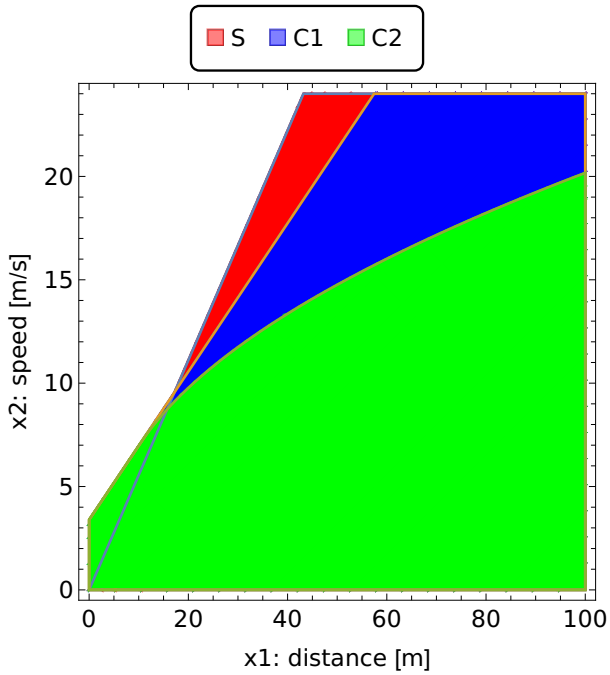
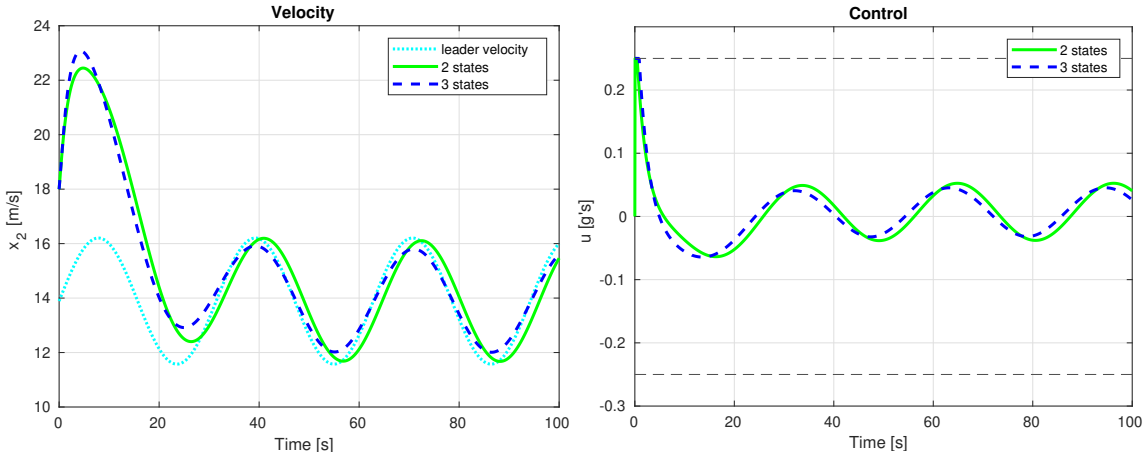


Fig. 9: *Safe region:* $k_0 = 1,\ k_1 = 1.75,\ k_2 = 0.5$

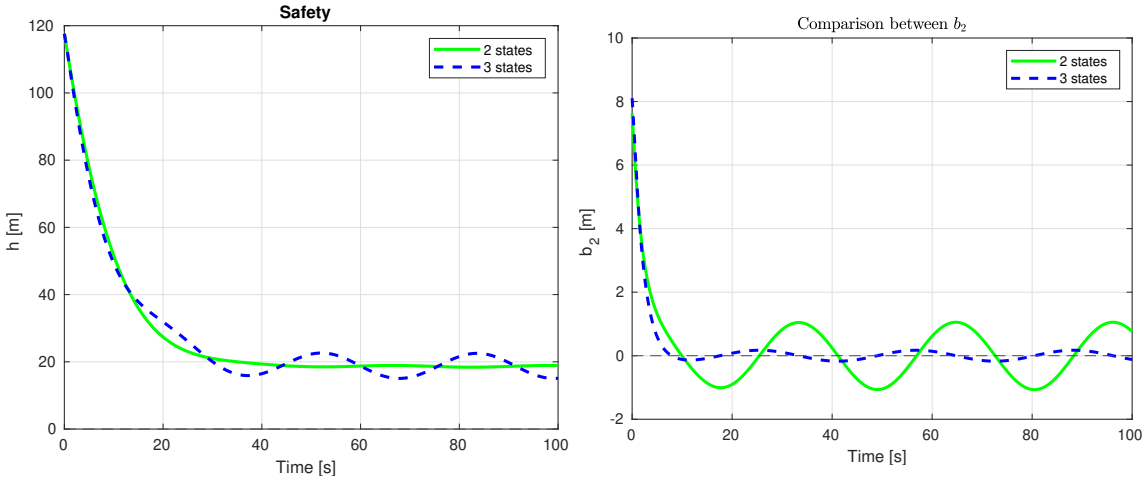
In the 3-state case the speed profiles of leader and follower are in phase, *Fig.(10a)*, because the control takes into account the acceleration of the leader. Instead in the 2-state case the follower starts braking only after it has acknowledged that the leader is decelerating.



(a) *Speed profile* (b) *Control profile*

Fig. 10: *Sinusoidal speed profile* v_ℓ*: more conservative case*

The controller based on the 3-state dynamical system is able to reduce the oscillations of $b_2(x)$ around zero of an order of magnitude, *Fig.(11b)*, but the oscillations of the original safety function $h(x)$ are bigger instead.



(a) *Safety profile* (b) $b_2(x)$ *profile*

Fig. 11: *Comparison between* $h(x)$ *and* $b_2(x)$

2. **Trapezoidal leading velocity**
   Here we have considered a trapezoidal speed profile for the leading car and the simulation results are presented in *Section 3.3.2*, compared with the ones obtained with the other methods.

3.2 Optimal Control Barrier Function for Control Lyapunov Function Method

In this section we are going to cover the final results presented in [3] which solves the ACC problem using Zeroing Control Barrier Functions (ZCBF). The authors present an optimization procedure that allows to define the optimal CBF comparing instantaneous velocity and maximal deceleration of leader and follower. This is used, together with a Control Lyapunov Function $V(x)$, for solving the quadratic programming procedure and computing the control needed in order to remain safe as presented in *Section 2.2*.

3.2.1 Model

The dynamical system used by the authors considers also the acceleration of the leading vehicle.

$$\dot{x} = \begin{bmatrix} \dot{v_f} \\ \dot{v_\ell} \\ \dot{D} \end{bmatrix} = \begin{bmatrix} -F(v_f)/m \\ a_\ell \\ x_2 - x_1 \end{bmatrix} + \begin{bmatrix} 1/m \\ 0 \\ 0 \end{bmatrix} u \tag{24}$$

From *eq.(16)* we can notice that we have an additional state with respect to the one used in *Section 3.1*. In particular we have that $x = (x_1,\ x_2,\ x_3)^T = (v_f,\ v_\ell,\ D)^T$ where v_f is the velocity of the follower, v_ℓ is the velocity of the leader and D is the relative distance between the vehicles.
Also the input vector field is modified since it considers the inverse of the follower mass instead of the acceleration of gravity. In fact in this case the control, which is computed solving the QP problem, has the physical dimension of a force, and this means that, in order to know the corresponding acceleration expressed as fraction of g_0, we need to multiply it by $(m \cdot g_0)^{-1}$.
This problem naturally considers that the leading vehicle may change its acceleration, within certain limits, and the control action is suitably modified by the optimization procedure.

3.2.2 Selection of the Optimal CBF

The authors solved an optimization problem in order to obtain the best CBF (h^o) to be used depending on the situation. It can be interpreted in the sense that if the leading car brakes using its maximal deceleration force, then the best response of the controlled car is to brake using its maximal deceleration force. The resulting function can be written as

$$h^o(x) = D - \Delta^{o*} \tag{25}$$

where Δ^{o*} is the result of the optimization process and depends on the instantaneous velocity (v_ℓ and v_f) and maximal deceleration of the two vehicles (a_ℓ and a_f). Three equations for Δ^{o*} have been computed

$$\begin{cases} \Delta_1^{o*} = 1.8v_f \\ \Delta_2^{o*} = \frac{1}{2}\frac{(1.8a_f g_0 - v_f)^2}{a_f g_0} + 1.8v_f - \frac{v_\ell^2}{2a_\ell g_0} \\ \Delta_3^{o*} = \frac{1}{2}\frac{(v_\ell + 1.8a_f g_0 - v_f)^2}{(a_f - a_\ell)g_0} + 1.8v_f \end{cases} \tag{26}$$

1. When $a_\ell = a_f$

$$\Delta^{o*} = \begin{cases} \Delta_1^{o*} & \text{if } 0 < v_f < v_\ell + 1.8a_f g_0 \\ \Delta_2^{o*} & \text{otherwise} \end{cases}$$

2. When $a_\ell < a_f$

$$\Delta^{o*} = \begin{cases} \Delta_1^{o*} & \text{if } 0 < v_f < v_\ell + 1.8a_f g_0 \\ \Delta_2^{o*} & \text{if } v_f \geq \frac{a_f}{a_\ell} v_\ell + 1.8a_f g_0 \\ \Delta_3^{o*} & \text{otherwise} \end{cases}$$

3. When $a_\ell > a_f$

$$\Delta^{o*} = \begin{cases} \Delta_1^{o*} & \text{if } 0 < v_f < \sqrt{\frac{a_f}{a_\ell}} v_\ell + 1.8a_f g_0 \\ \Delta_2^{o*} & \text{otherwise} \end{cases}$$

3.2.3 QP problem formalization

The QP problem, solved in MATLAB using the command `quadprog`, has the cost function which is selected in view of achieving the control objective encoded in the CLF with matrices H and F defined as

$$H_{acc} = 2\begin{bmatrix} \frac{1}{m^2} & 0 \\ 0 & p_{sc} \end{bmatrix} \qquad F_{acc} = -2\begin{bmatrix} F(v_f)/m^2 \\ 0 \end{bmatrix}$$

The controller is computed as

$$u^*(x) = \underset{u=[u\ \ \delta]^T \in U_{acc} \times \mathbb{R}}{\operatorname{argmin}} \frac{1}{2} u^T H_{acc} u + F_{acc}^T u \tag{27}$$

$$\begin{aligned} \text{subject to } & L_g V(x)u - \delta \leq -LfV(x) - cV(x) \\ & L_g h(x)u \leq -L_f h(x) + 2h(x) \\ & u \leq a'_f m g_0 \\ & -u \leq a_f m g_0 \end{aligned}$$

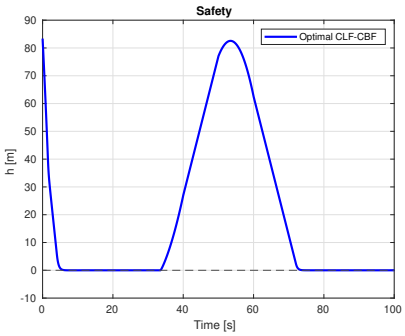
where $U_{acc} := [-a_f m g_0,\ a'_f m g_0]$.

3.2.4 Computational Results

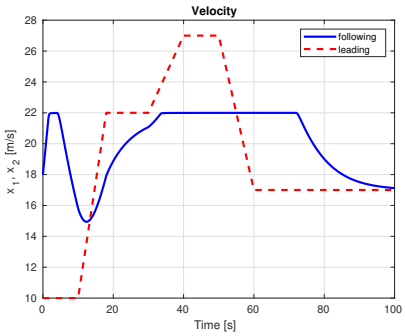
The simulations are conducted, as in the previous problem, using MATLAB command `ode45` in order to compute the evolution of the system. The numerical parameters of the problem are reported in *Tab.(2)*.

Variable	Description	Value
$D(0)$	initial distance between the vehicles	$150\ m$
$v_f(0)$	follower initial velocity	$18\ \frac{m}{s}$
$v_\ell(0)$	leader initial velocity	$10\ \frac{m}{s}$
m	mass of the following car	$1650\ Kg$
f_0	constant term of friction	$0.1\ N$
f_1	linear term of friction	$5\ N \cdot (\frac{m}{s})^{-1}$
f_2	quadratic term of friction	$0.25\ N \cdot (\frac{m}{s})^{-2}$
g_0	gravitational acceleration	$9.81\ \frac{m}{s^2}$
v_{max}	maximum follower velocity	$22\ \frac{m}{s}$
$a'_f = a_f$	maximum follower acceleration/deceleration	0.25
p_{sc}	weight for δ in `quadprog`	100
c	CLF constant	10

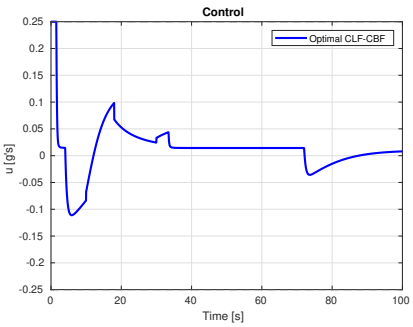
Tab. 2: *Original parameters for simulation of this section*



(a) *profile of the safety function*



(b) *profile of the velocity*



(c) *profile of the control*

Fig. 12: *Optimal ZCBF simulation results*

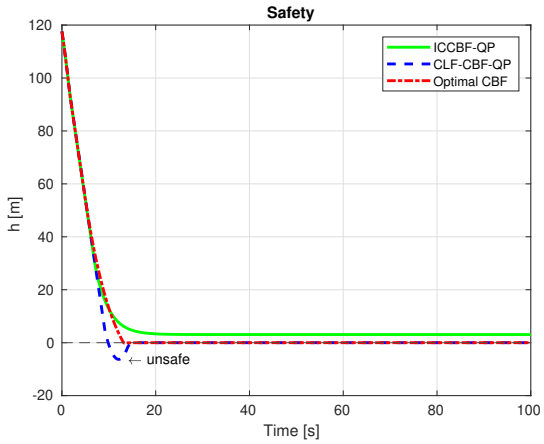
Fig.(12) shows the results of the simulations conducted for this problem where we can find the safety profile (*Fig.(12a)*), the velocity of leader and follower (*Fig.(12b)*) and the control profile that have been applied (*Fig.(12c)*). In particular we can notice that safety is never violated (in fact in the corresponding plot is always $h(x) \geq 0$) and, as expected, is greatly increased when the leading vehicle assumes a velocity which is greater than the maximum value of the follower (defined as Soft Constraint through the CLF) that is never bypassed. The control profile is very smooth except for the first instants where we can see that is in saturation. This corresponds to the fact that we accelerate as much as possible in order cover the gap with the leader, since safety is loosely guaranteed in the initial condition and this allow to focus mainly on satisfy the performance constraint, until we reach a situation where we have to decelerate for not compromising safety.

3.3 Comparison between ICCBF, CLF-CBF and Optimal CBF

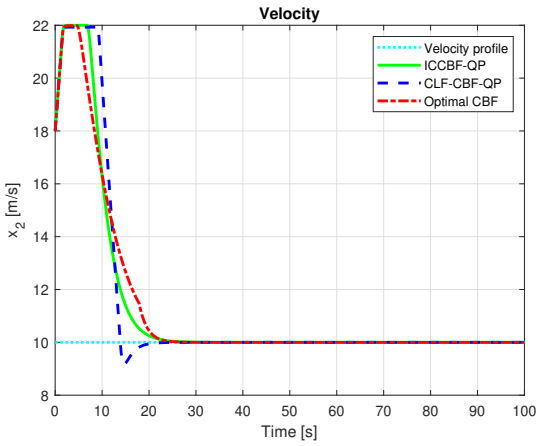
The goal of this section is to make a comparison between the three different methods which have been analysed in this work: ICCBF, CLF-CBF (from *Section 3.1*) and CLF-CBF with optimal CBF (from *Section 3.2*).
The parameters used in order to do the comparison are reported in *Tab.(2)*, the model is the one depicted in *eq.(24)* and two types of velocity profiles have been used: one considering that the leading vehicle moves with constant speed and the other using a trapezoidal profile that is composed by accelerations, decelerations and constant speed sections.

3.3.1 Case 1: Constant velocity

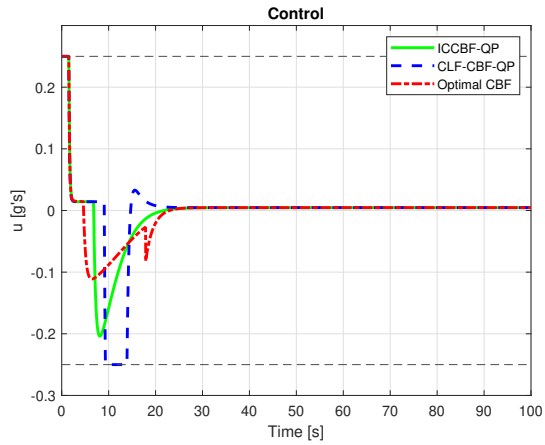
The velocity of the leading vehicle is equal to the initial speed for the whole time of simulation and this implies its acceleration is 0. As a result the dynamical system evolves with the remaining two states, but still we are considering a maximum deceleration (with $a_\ell > a_f$) in order use the optimality results described in *Section 3.2*.
Plots in *Fig.(13)* show the results that we obtained in this framework. From *Fig.(13a)* we clearly see that CLF-CBF method, using the same safety function, is not able to guarantee always safety since the following vehicle does not start to decelerate soon enough (also recognizable by looking at the undershoot in *Fig.(13b)*), entering in the unsafe region. Moreover we can notice that the solution based on the Optimal CBF after $t \approx 10$ is constantly on the boundary of the safety region whereas the ICCBF solution is more conservative. As a result we have that, from the performance point of view, ICCBF is slower w.r.t. the other methods in reaching steady state but it has a margin w.r.t. the boundary of the safety region, whereas CLF-CBF is faster, but we are very close to becoming unsafe.



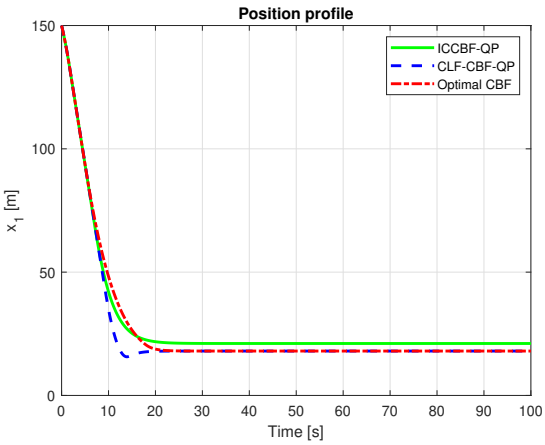
(a) *profile of the safety function*



(b) *profile of the velocity*



(c) *profile of the control*

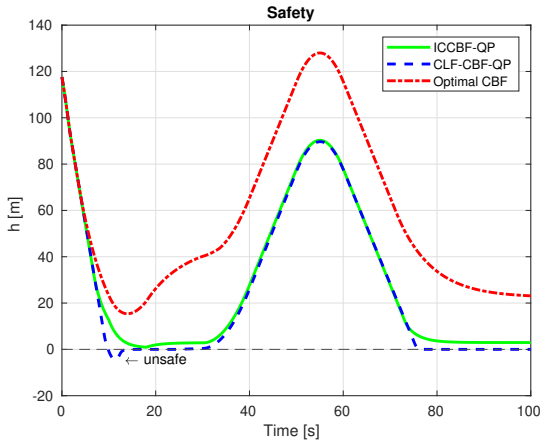


(d) *profile of the position*

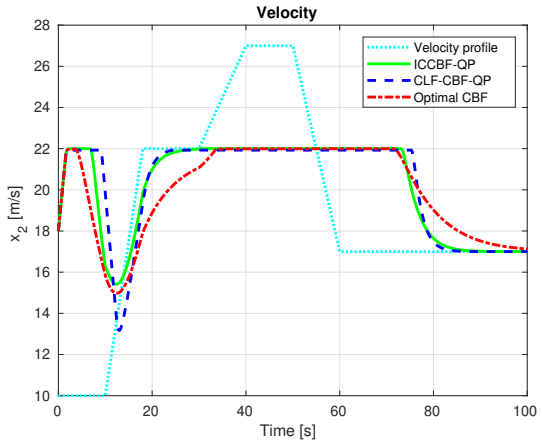
Fig. 13: *Comparison between ICCBF, CLF-CBF and Optimal CBF using a constant velocity profile*

3.3.2 Case 2: Trapezoidal velocity

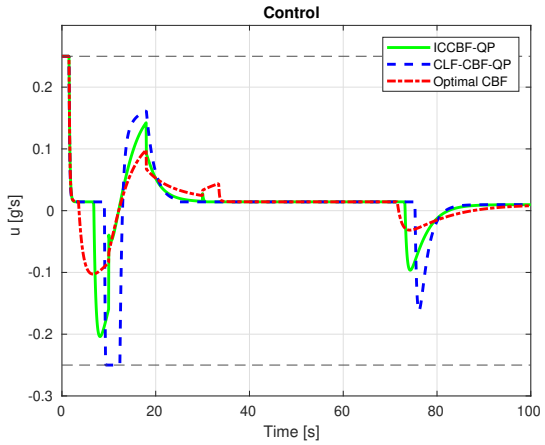
In this context we have used the same velocity profile that have been used in *Section 3.2* and, since the acceleration of the leading vehicle isn't constant, we have used the system with an additional state also for evaluating the ICCBF and CLF-CBF methods.
Fig.(14) shows the results of the evaluation in this framework where we can notice that the optimal CBF method applies a lower acceleration on the following vehicle (*Fig.(14c)*) that is reflected in the safety and velocity profiles, since the plots are different w.r.t. the other methods. In *Fig.(14a)* we have compared all the methods by using the same safety function *eq.(18)*. We can notice that in the case of optimal CBF we have obtained a more conservative situation w.r.t. the others since the optimality is related to the current situation in which we evaluate the system. In addition CLF-CBF still does not guarantee safety while ICCBF is nearer the safety boundary but remains more conservative with respect to CLF-CBF.



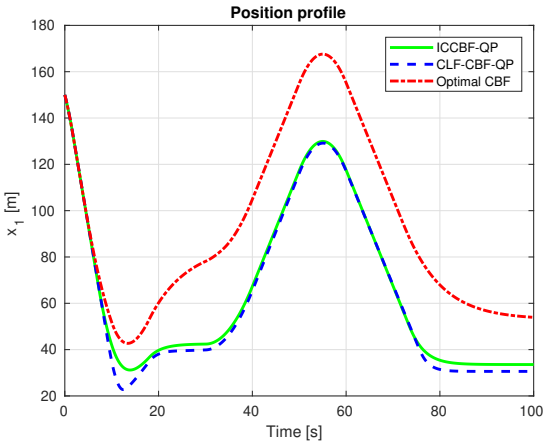
(a) *profile of the safety function*



(b) *profile of the velocity*



(c) *profile of the control*



(d) *profile of the position*

Fig. 14: *Comparison between ICCBF, CLF-CBF and Optimal CBF using a trapezoidal velocity profile*

4 Conclusion

In this project we have analysed the framework for the control of safety-critical systems through the use of the Control Barrier Functions (CBF) in the context of the Adaptive Cruise Control (ACC) problem considering different scenarios: changing the method used to derive the controller and considering alternative velocity profiles of the leading vehicle. In the context of affine control systems, this naturally yields control barrier functions (CBFs) with a large set of available control inputs that renders a set C forward invariant. CBFs are expressed as affine inequality constraints, in the control input, that imply forward invariance of the set, hence safety.
We have first looked at the problem of ACC in the case of the presence of input constraints, which are explicitly included in the construction of control barrier functions (ICCBF) in order to guarantee that safety is maintained with an input constrained controller.
We have analysed how the shape of the safe region changes by varying the parameters that define the class-$\mathcal{K}$ functions α_i, finding out that lowering these values reflects in a smaller safe region, while enlarging them too much corresponds to not finding a proper ICCBF accordingly to the choices done in our simulations, i.e. the number N and the functions α_i. Furthermore, we have tested the validity of this type of CBFs concluding that, once we tuned properly the parameters of α_i, safety is guaranteed.
We have also changed the maximum allowed velocity for the following vehicle, that reflects in a different desired controller $u_d(x)$, and we have shown that the ICCBF controller still guarantees safety and the new speed limit is never violated, whereas the CLF-CBF still does not always respect the hard constraint.
In addition, by looking at the case of a variable leading velocity, we have augmented the dynamical system taking into account also the leading speed evolution. With the new system the acceleration of the vehicle in front shows up in the definition of the ICCBF and we have noticed that the follower accelerates and decelerates thanks to its a-priori knowledge of the leading velocity profile, in fact it moves in phase with the leader.
This solution has been compared with the framework in which we unify safety conditions with control objectives (expressed as Control Lyapunov Functions), by strictly enforcing the safety constraint and relaxing the control objective defining a proper quadratic program problem. Then, in the case of the CLF-CBF, we have considered the QP problem defined in terms of optimal control barrier functions, chosen based on the velocities of the two vehicles. Finally, we have done a comparison among all the solutions obtained with the different frameworks in the ACC problem by analysing differences in term of control effort and safety.

References

[1] Aaron D Ames, Samuel Coogan, Magnus Egerstedt, Gennaro Notomista, Koushil Sreenath, and Paulo Tabuada. Control barrier functions: Theory and applications. In *2019 18th European control conference (ECC)*, pages 3420–3431. IEEE, 2019.

[2] Devansh Agrawal and Dimitra Panagou. Safe control synthesis via input constrained control barrier functions. *arXiv preprint arXiv:2104.01704*, 2021.

[3] Aaron D Ames, Xiangru Xu, Jessy W Grizzle, and Paulo Tabuada. Control barrier function based quadratic programs for safety critical systems. *IEEE Transactions on Automatic Control*, 62(8):3861–3876, 2016.

[4] Katja Vogel. A comparison of headway and time to collision as safety indicators. *Accident analysis & prevention*, 35(3):427–433, 2003.