IEEE Robotics and Automation Letters (RAL) paper presented at
2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Safe Control for Navigation in Cluttered Space using Multiple Lyapunov-Based Control Barrier Functions

Inkyu Jang, and H. Jin Kim

***Abstract*— Control barrier functions (CBFs) are powerful tools for ensuring safety in controlled systems, commonly employed through the construction of a safety filter using quadratic programming (QP), known as CBF-QP. However, synthesizing a CBF specifically for the navigation tasks of mobile robots, where safety is crucial, poses challenges due to the complexity of the operating environments. In addition to that, the CBF synthesis should be repeated for every new environment, further escalating the computational burden. In this paper, we introduce Lyapunov-based CBFs, which is a CBF built solely from a control Lyapunov function (CLF). By utilizing multiple Lyapunov-based CBFs as building blocks to create a large control invariant set, we formulate a CBF-QP-like safety filter to ensure safety in cluttered environments. The proposed safety filter inherits the favorable characteristics of CBF-QP such as fast computation and safety guarantee, and can adapt to diverse environments without the need for burdensome resynthesis of a new environment-specific CBF. We demonstrate the effectiveness of the proposed approach through multiple simulation and real-world experiments, whose results show that the proposed safety filter was successful in providing safety for the robot even in complex workspaces with many obstacles.**

I. Introduction

Safety in autonomous mobile robot systems can be defined as satisfying the state constraints (e.g., speed limit and collision avoidance) over an infinite time horizon without exceeding the control limits. Unfortunately, it is generally difficult to directly verify safety in mobile robot systems, mainly due to the difficulty of looking *infinitely far ahead in time*, especially in nonlinear systems. To circumvent direct safety verification, many researchers have proposed alternative approaches using control invariant sets, such as control barrier functions (CBFs) [1]. Safety assurance through CBFs is recently becoming popular, since a valid CBF, once synthesized, offers a real-time safety filter based on quadratic programming (QP) named CBF-QP [2].

Despite the advantages offered by CBFs in terms straightforward safety verification, the process of synthesizing a valid CBF is widely recognized as challenging. The requirements for a function to qualify as a CBF are expressed in the form of partial differential inequalities whose analysis is difficult. Another reason is that the computational burden of synthesizing one typically scales exponentially with the dimension of the state variable, known as the *curse of dimensionality* [3]. Moreover, CBFs in the current form are not suitable for use in navigation tasks of mobile robot systems. The main reason is that mobile robots are typically deployed in large and complex environments with many obstacles, and the burdensome CBF synthesis should be repeated for every environment the robot is deployed in.

In this paper, we propose a CBF-QP-like safety filter for nonlinear mobile robot systems which overcomes the aforementioned shortcomings. For that, we build multiple CBFs only using a single control Lyapunov function (CLF), namely Lyapunov-based CBFs, upon the mild assumption that the system dynamics is invariant under translation in certain directions, which we call translational symmetry in this paper. Dimensionality is partially reduced by patching copies of Lyapunov-based CBFs translated in the symmetric directions. The system is allowed to switch between multiple Lyapunov-based CBFs to track the given reference input without losing safety. The strengths of the proposed method can be summarized as follows:

- The proposed method can adapt to various environments without heavy computation.
- It inherits the similar good characteristics from CBF-QP. It can be run in real-time, and can be used as an augmentation to an existing motion planner, allowing the robot to select its control input from a broader range.
- The synthesis of Lyapunov-based CBFs is inexpensive since it does not require direct analysis of partial differential inequalities.
- One can make use of any off-the-shelf stabilizing feedback control method or a hand-crafted Lyapunov function to construct a Lyapunov-based CBF. This reduces the computational burden of dealing with large complex environments.

The effectiveness of the proposed approach is demonstrated through simulation and real-world experiments with different mobile robot models subject to actuation limits. The results show that the proposed safety filter well filters the reference input and provides persistent constraint satisfaction to the system even in a complex environment with many obstacles.

II. Related Work

Recent works on safety assurance for mobile robots can be roughly categorized into two: one is to find the largest control invariant set within the set of allowed states, and the other is to find the smallest invariant set around the robot's nominal trajectory, typically called *tubes*.

This research was supported by Unmanned Vehicles Core Technology Research and Development Program through the National Research Foundation of Korea (NRF) and Unmanned Vehicle Advanced Research Center (UVARC) funded by the Ministry of Science and ICT, the Republic of Korea (NRF-2020M3C1C1A010864).

Inkyu Jang and H. Jin Kim are with the Department of Aerospace Engineering and Automation and Systems Research Institute (ASRI), Seoul National University, 08826 Seoul, Korea. janginkyu.larr@gmail.com, hjinkim@snu.ac.kr

In the former one's case, CBFs have been widely applied thanks to their straightforward formulation [4]–[6], and many variants of CBFs were also presented [7]–[10]. However, it is widely recognized as challenging to find a valid CBF for systems with actuation limits. Some recent works based on sum-of-squares (SOS) programming [11], [12] or neural networks [13], [14] were successful in synthesizing one, but they only were able to handle simple environments with a few obstacles. Their biggest drawback in navigation tasks is that the barrier certificates are environment-specific and requires resynthesis for every new environment.

The tube-based methods more easily apply to complex environments while not being environment-specific. They typically compute tube certificates in the offline phase and place them around the nominal trajectory in the online phase to guarantee safety. In [15], the robot is required to select from a finite number of *funnels*, where a funnel is a trajectory contained inside tube-shaped bound. The work [16] admits a infinite-sized library of such tube-shaped trajectories, and the robot solves a nonlinear optimization problem to find the best safe one during operation. Tube construction using control contraction metrics also showed to be powerful in obstacle-cluttered environments [17]. Their advantages compared to CBFs are less computational burden in the offline phase and adaptability to various environments. However, they typically come along with a (possibly aggressive) feedback controller to constrain the system within the tube, and may be undesirable in terms of control effort and energy usage.

The proposed method combines the advantages of the two methods for navigation tasks of mobile robots. While being a safety filter without the need of an aggressive feedback controller, it does not require a burdensome environment-specific barrier function synthesis. Additionally, the proposed safety filter can be used with any type of CLF, for example, learning-based [18], reachability-based [19], or even hand-crafted ones. It can also be used with a handful of existing feedback control techniques that come along with a Lyapunov function, such as linear quadratic regulator (LQR), backstepping control, and sliding mode control.

III. Preliminaries

A. Problem Setup

We start by considering a continuous-time nonlinear time-invariant control-affine system in the following form:

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

where $x \in \mathbb{R}^n$ is the state variable, $u \in U \subset \mathbb{R}^m$ is the input, and $f : \mathbb{R}^n \to \mathbb{R}^n$, $g : \mathbb{R}^n \to \mathbb{R}^{n\times m}$ are Lipschitz continuous functions on the domain $\mathbb{R}^n$, so that the state trajectory never blows up in finite time with bounded input. In this paper, we view safety in the context of set invariance. That is, a system can be considered *safe* within a given set $C \subset \mathbb{R}^n$, if there exists a feasible control strategy which is capable of permanently keeping the state trajectory within C. Therefore, we define safety as follows:

Definition 1 (Safety)**.** The set $C \subset \mathbb{R}^n$ is (infinite-time) control invariant for the system (1), if for all $x_0 \in C$, there exists a (possibly feedback) control strategy $k : [0, \infty) \times \mathbb{R}^n \to U$, such that the solution $x(\cdot)$ (in the Caratheodory sense [20]) to the following initial value problem (IVP)

$$\dot{x}(t) = f(x(t)) + g(x(t))k(t, x(t)), \quad x(0) = x_0 \tag{2}$$

uniquely exists and $x(t) \in C$ for all $t \in [0, \infty)$. If such set C consists of *allowable* states only, we call C a *safe set*.

Let $X_{\text{obs}} \subset \mathbb{R}^n$ be the set of *forbidden* states, for example, collision with an obstacle. We assume that X_{obs} is constant with respect to time. The objective of this work is to synthesize a safety-critical controller which *filters* the reference input $u_{\text{ref}}(t) \in \mathbb{R}^m$ ($t \in [0, \infty)$) to give the actual feasible control input, so that the resulting feedback controller ensures that the state never enters X_{obs}. That is, we want to construct a large safe set C such that C and X_{obs} are disjoint, and a feedback controller that renders the set C control invariant.

B. Assumptions

In this subsection, we list three key assumptions that are essential in formulating the proposed safety filter.

1) Linear Input Bound: The first assumption is that the input set U can be represented using a finite number of linear inequalities, i.e.,

$$U = \{u \in \mathbb{R}^m : a_i^\top u \le b_i,\ \forall i \in \{1, \cdots, N\}\}, \tag{3}$$

where $a_i \in \mathbb{R}^m$ and $b_i \in \mathbb{R}$ for all $i \in \{1, \cdots, N\}$, and N is a (possibly zero) nonnegative integer. This assumption is to ensure that the resulting optimization-based safety filter is formulated as a QP which comprises linear inequality constraints only, so that it can be solved in real-time.

2) Stabilizability to the Origin: Let $\beta : \mathbb{R}^n \to \mathbb{R}$ be a differentiable function. The time derivative of β along the trajectory is written as follows:

$$\dot{\beta}(x, u) = \frac{\partial \beta}{\partial x} \cdot \big(f(x) + g(x)u\big). \tag{4}$$

We assume that there exists a continuously differentiable function $V : \mathbb{R}^n \to \mathbb{R}$ and a (possibly infinite) positive real number b, such that V is a CLF in the domain $D = \{x \in \mathbb{R}^n : V(x) \le b\}$; i.e.,

- V is positive definite: $V(x) \ge 0$ for all $x \in D$, $V(x) = 0$ if and only if $x = 0$,
- For every $x \in D$, there exists an input u such that $\dot{V}(x, u) \le 0$.

The existence of such V implies that the system is *stabilizable* to the origin $x = 0$. We assume that $\frac{\partial V}{\partial x} \neq 0$ for all $x \in D \setminus \{0\}$.

3) Translational Symmetry: We lastly assume that the dynamics exhibits the *translational symmetry* property, which is defined as follows.

Definition 2 (Translational Symmetry)**.** The dynamics (1) is *translationally symmetric* if there exists an *embedding matrix* $E \in \mathbb{R}^{n\times n_q}$ ($0 < n_q \le n$) such that

$$f(x) = f(x - Eq), \quad g(x) = g(x - Eq) \tag{5}$$

for any $q \in \mathbb{R}^{n_q}$.

The definition tells that the dynamics remains invariant under translation in the column space of E, $\mathrm{col}(E)$, where q denotes the displacement on $\mathrm{col}(E)$. Many real-world mobile robot systems have this property. For example, the kinodynamic model of a ground rover is usually symmetric under translation in the longitudinal direction.

C. CBFs and Safety-Critical Control

CBF is a powerful tool for ensuring safety of a control system. The main objective of utilizing CBFs is to determine whether a set $C \subseteq \mathbb{R}^n$ is control invariant, and construct a safety-critical controller which makes C invariant. Suppose the set C is given in the form of the superlevel set of a continuously differentiable function $h : \mathbb{R}^n \to \mathbb{R}$:

$$\begin{aligned} C &= \{x \in \mathbb{R}^n : h(x) \geq 0\} \\ \partial C &= \{x \in \mathbb{R}^n : h(x) = 0\} \\ \mathrm{Int}\ C &= \{x \in \mathbb{R}^n : h(x) > 0\}. \end{aligned} \tag{6}$$

Definition 3 (Control Barrier Function (CBF))**.** Let $\alpha : \mathbb{R} \to \mathbb{R}$ be an extended class-$\mathcal{K}_\infty$ function.[1] A continuously differentiable function $h : \mathbb{R}^n \to \mathbb{R}$ is a CBF, if (6) holds, and there exists a control input $u \in U$ such that

$$\dot{h}(x, u) + \alpha(h(x)) \geq 0 \tag{7}$$

for all $x \in C$, and $\frac{\partial h}{\partial x} \neq 0$ for all $x \in \partial C$.

It follows from Nagumo's theorem [21] that the set C is invariant if such h is a CBF.

Now, let $K_{\mathrm{cbf}}(x) = \{u \in U : \dot{h}(x, u) + \alpha(h(x)) \geq 0\}$. It is obvious from Definition 3 that it is nonempty for all $x \in C$. It is known that any feedback k such that $k(t, x) \in K_{\mathrm{cbf}}(x)$ renders the set C invariant [2, Corollary 2], given that it admits a unique solution to the closed-loop ODE.

Thanks to this invariance property, CBF can be used to synthesize a safety-critical controller. If $h : \mathbb{R}^n \to \mathbb{R}$ is a CBF such that $h(x) < 0$ for all $x \in X_{\mathrm{obs}}$, any feedback controller $u = k(t, x) \in K_{\mathrm{cbf}}(x)$ will guarantee $x(t) \notin X_{\mathrm{obs}}$, should one exists, for all $t \in [0, \infty)$.

One good way of selecting a control input from $K_{\mathrm{cbf}}(x)$ is to choose one that minimally deviates from a given reference input $u_{\mathrm{ref}}(\cdot) \in \mathbb{R}^m$. Given the assumption on U (3), finding such input can be formulated as a QP, namely CBF-QP [2]:

$$\begin{aligned} k(t, x) = \arg\min_{u \in U} \quad & \left\| u - u_{\mathrm{ref}}(t) \right\|^2 \\ \text{s.t.} \quad & \dot{h}(x, u) + \alpha(h(x)) \geq 0. \end{aligned} \tag{8}$$

Observe that the inequality constraint $\dot{h}(x, u) + \alpha(h(x)) \geq 0$ is affine in u (see (4)), hence (8) is a QP. Note also that this QP is strictly convex, so it admits a unique finite optimal argument, even if the feasible set $K_{\mathrm{cbf}}(x)$ is unbounded. Many modern convex optimization solvers can solve QPs like (8) in real-time.

[1] A function $\alpha : \mathbb{R} \to \mathbb{R}$ is an extended class-$\mathcal{K}_\infty$ function if it is continuous, strictly increasing, unbounded, and $\alpha(0) = 0$.

IV. Safety Filter Design

A. Safety-Critical Control using Multiple CBFs

Let $h_i : \mathbb{R}^n \to \mathbb{R}$ for each $i \in \mathcal{I}$ be a CBF that renders $C_i = \{x \in \mathbb{R}^n : h_i(x) \geq 0\}$ invariant, where $\mathcal{I}$ is a finite index set. For each CBF h_i, consider the following CBF-QP:

$$\begin{aligned} k_i(t, x) = \arg\min_{u \in U} \quad & \left\| u - u_{\mathrm{ref}}(t) \right\|^2 \\ \text{s.t.} \quad & \dot{h}_i(x, u) + \alpha(h_i(x)) \geq 0, \end{aligned} \tag{9}$$

where $u_{\mathrm{ref}}(t)$ is the reference input that is independent of i. We now let $j \in \mathcal{I}$ be an internal state, and set up the following feedback control strategy:

$$\begin{aligned} k(t, j, x) &= k_j(t, x) \\ j^+ &= S\left(\arg\max_{i \in \mathcal{I}} \ h_i(x) \right), \end{aligned} \tag{10}$$

where the internal state j jumps to j^+ whenever the time t is a multiple of the preset CBF-switching interval δt, which can be any positive real number. That is, whenever j jumps, the controller switches to the CBF that gives the biggest value (i.e., the least risk). The *selection function* $S : 2^{\mathcal{I}} \setminus \{\varnothing\} \to \mathcal{I}$ is employed to avoid ambiguity of the $\arg\max_{i \in \mathcal{I}}$ term.[2]

The controller (10) yields the following hybrid closed-loop system: whenever t is multiple of δt, the closed-loop system may experience a sudden *jump* in the control input u. In case of a general hybrid system, a unique Caratheodordy solution to the ODE might be non-existent. However, under a number of mild conditions on the CBFs, and the initial condition, the closed-loop system still possesses a unique Caratheodory solution despite the discontinuities. That is due to the fact that the jumps occur only at countable points in time [22]: the collection of all such times, which is a subset of $\delta t \cdot \mathbb{Z}_{>0}$, is a measure-zero set.

Furthermore, (10) renders the set

$$C = \{x \in \mathbb{R}^n : h(x) \geq 0\} = \bigcup_{i \in \mathcal{I}} C_i \tag{11}$$

invariant, where $h(x) = \max_{i \in \mathcal{I}} h_i(x)$. The following theorem summarizes these two statements. Its proof can be found in the appendix.

Theorem 1. *Let $u_{\mathrm{ref}} : [0, \infty) \to \mathbb{R}^m$ be the reference input. Assume:*

1) *For every $i \in \mathcal{I}$, h_i is a CBF for the system (1).*
2) *For any $t_0 \in [0, \infty)$, and for every $i \in \mathcal{I}$, if $x(t_0) \in C_i$, the CBF-QP (9) provides a unique solution to the closed-loop ODE on the interval $[t_0, t_0 + \delta t]$.*
3) *$x_0 \in C$.*

Then, the closed-loop ODE

$$\begin{aligned} \dot{x} &= f(x) + g(x)k(t, j, x), \quad (t/\delta t \notin \mathbb{Z}_{>0}) \\ j^+ &= S\left(\arg\max_{i \in \mathcal{I}} \ h_i(x) \right), \quad (t/\delta t \in \mathbb{Z}_{>0}) \end{aligned} \tag{12}$$

[2] Note that the maximum argument $\arg\max_{i \in \mathcal{I}} h_i(x)$ in the second line may be not unique. The mapping S selects one among the (possibly multiple) maximum arguments. It can be any mapping that satisfies $S(I) \in I$ for any nonempty $I \subseteq \mathcal{I}$. The existence of the maximum argument and such selecting function is guaranteed, since $\mathcal{I}$ is a finite set.

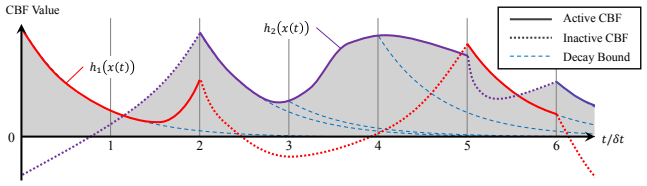


Fig. 1. An example of how CBF value changes with respect to time, when the proposed controller (10) is applied. In this example, $\mathcal{I} = \{1, 2\}$, i.e., two CBFs are used. Theorem 1 states that, at all times, at least one among the two CBFs h_1 (red curve) and h_2 (purple curve) is kept nonnegative, although the *inactive* CBFs are allowed to decay faster than the decay bound given by the class-$\mathcal{K}_\infty$ function α. Therefore, the value of $h(\cdot) = \max\{h_1(\cdot), h_2(\cdot)\}$, shown as a gray shaded region, is also kept nonnegative.

with the initial condition

$$x(0) = x_0, \quad j(0) = S\left(\arg\max_{i\in\mathcal{I}} h_i(x_0)\right) \tag{13}$$

has a unique solution on the interval $[0, \infty)$. *Moreover,* $x(t) \in C$ *for all* $t \in [0, \infty)$.

Fig. 1 shows an example of the change of CBF values when the proposed controller (10) is applied. Although $h(x(t)) \neq h_{j(t)}(x(t))$ for some t, $h(x(\cdot))$ is always kept nonnegative, thus C is an invariant set.

Remark 1. The second condition holds when the controller $k_i(t, x)$ is piecewise continuous and with respect to t, and Lipschitz continuous with respect to x. This might not be true even with smooth reference input u_{ref} in some cases. See [23] for a counterexample and some additional conditions for the continuity conditions to hold.

Remark 2. Note that δt can be an arbitrary positive real number. Thus, we can think of the limit $\delta t \searrow 0$, resulting in the following *naive* version of the controller:

$$k(t, x) = k_{j(x)}(t, x), \tag{14}$$

where $j(x) = S\left(\arg\max_{i\in\mathcal{I}} h_i(x)\right)$. The index $j \in \mathcal{I}$ is now a function of x rather than an internal state, and the controller only depends on x and u_{ref}. We can no longer be certain that a unique solution to the closed-loop ODE exists. In practice, this means that (14) might introduce a severe chattering phenomenon that potentially drives the system to an unsafe state.

Nevertheless, in the simulation experiment, we saw that the proposed controller *usually* generates a piecewise continuous input trajectory, which becomes discontinuous only when $t/\delta t \in \mathbb{Z}$. Furthermore, the naive version (14) also performed well as an appropriate safety filter.

B. Safety-Critical Control using Lyapunov-Based CBFs

We introduce how Lyapunov-based CBFs can be constructed from a CLF using the system's translational invariance property, and how they can be used to synthesize a safety-critical controller. Let $V : \mathbb{R}^n \to \mathbb{R}$ be a CLF on the domain $D = \{x \in \mathbb{R}^n : V(x) \leq b\}$ ($b \in \mathbb{R}_{>0} \cup \{\infty\}$), and consider

$$h_q(x) = W(q) - V(x - Eq), \tag{15}$$

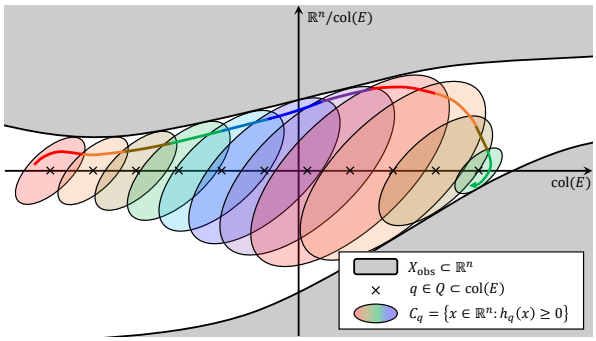


Fig. 2. A graphical explanation of the safe set $C = \bigcup_{q\in Q} C_q$. The stabilization points q are placed along $\text{col}(E)$, the column space of the embedding matrix E. The ellipses with different colors denote where h_q values become nonnegative, and the thick curve denotes a possible trajectory. The colors of the segments denote for which q the $h_q(x)$ value becomes the maximum along the trajectory.

where $E \in \mathbb{R}^{n\times l}$ is the the embedding matrix from Definition 2, $q \in \mathbb{R}^l$, and $0 \leq W(q) \leq b$. An intuitive explanation for $W(q)$ is that it serves as a measure of how far from the state Eq the system can be displaced without violating the safety constraints.

Theorem 2 (Lyapunov-Based CBF)**.** *For any fixed* $q \in \mathbb{R}^l$ *and* $0 \leq W(q) \leq b$, h_q *is a CBF on the domain* $C_q = \{x \in \mathbb{R}^n : h_q(x) \geq 0\}$. *This holds for any class-*$\mathcal{K}_\infty$ *function* α.

In order to use the Lyapunov-based CBF h_q to synthesize a safety-critical controller, we let

$$W(q) < \min\left\{b, \inf_{x\in X_{\text{obs}}} V(x - Eq)\right\}. \tag{16}$$

It is straightforward to find that $W(q) < b$ for any $q \in \mathbb{R}^l$. Moreover, since

$$\begin{aligned} h_q(x_{\text{obs}}) &= W(q) - V(x_{\text{obs}} - Eq) \\ &< \inf_{x\in X_{\text{obs}}} V(x - Eq) - V(x_{\text{obs}} - Eq) \leq 0 \end{aligned} \tag{17}$$

for any $x_{\text{obs}} \in X_{\text{obs}}$, the set $C_q = \{x \in \mathbb{R}^n : h_q(x) \geq 0\}$, if nonempty, is a safe forward invariant set for the given system. Thus, so is $C = \bigcup_{q\in Q} C_q$, which is a union of safe sets [21].

A brief graphical explanation of the proposed safety filter is given in Fig. 2. At any time during operation, the robot can stabilize to at least one $q \in Q$.

V. Experiments

In order to validate the proposed safety filter, we conducted simulation and hardware experiments on three different system dynamics as depicted in Fig. 3. The simulations were run on a desktop computer with 4-GHz CPU and 32-GB RAM, and the hardware experiment ran on an onboard computer with 1.7-GHz CPU and 8-GB RAM. The parameters values used in the simulation and experiments can be found in Table I. For all experiments, we used $\alpha(y) = \gamma \cdot y$ ($y \in \mathbb{R}$) as the extended class-$\mathcal{K}_\infty$ function, where $\gamma > 0$ is a parameter that determines the decay rate. The computation times in the offline and online phases are summarized in Table II which demonstrates real-time applicability of the

TABLE I

THE PARAMETER VALUES USED IN THE SIMULATION AND HARDWARE EXPERIMENTS

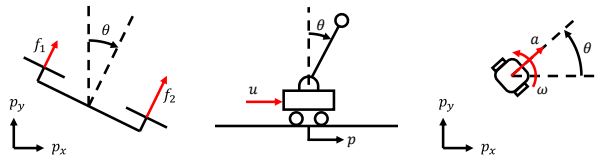
Planar Multirotor			Cart-Pole			Ground Rover			
Name	Symbol	Value	Name	Symbol	Value	Name	Symbol	Value (sim.)	Value (hardware)
CBF switch interval	δt	0 †	CBF switch interval	δt	0.01 s	CBF switch interval	δt	0 †	0.01 s
Robot size	r	0.5	Cart mass	m_1	0.3 kg	Robot size	r	0.3 m	0.25 m
Grid size	d	0.5	Pole tip mass	m_2	0.1 kg	v weight	μ_v	1	1
Obs. size	d_{obs}	2.5	Pole length	l	0.5 m	θ weight	μ_θ	10^{-3}	10^{-3}
Decay rate	γ	10	Gravity	g	9.81 m/s^2	Max. acceleration	a_m	1 m/s^2	0.18 m/s^2
CLF bound	b	6	Grid size	d	0.01 m	Max. turn rate	ω	1 rad/s	1.4 rad/s
Conservatism	ϵ	10^{-6}	Max. force	f_m	2 N	Grid size	d	0.25 m	0.02 m
			Decay rate	γ	2	Obstacle size	-	1 m × 1 m	0.25 m (radius)
			CLF bound	b	0.5	Decay rate	γ	1.5	1
						CLF bound	b	∞	0.1

† See Remark 2.

TABLE II

SUMMARY OF AVERAGE COMPUTATION TIMES

Scenario	Offline [ms]	Online [ms]
Planar Multirotor (section V-A)	520	2.3
Cart-Pole (section V-B)	< 10	0.5
Ground Rover (Sim., section V-C)	310	7.0
Ground Rover (Exp., section V-D)	820	3.1



(a) Planar Multirotor (b) Cart-Pole (c) Ground Rover

Fig. 3. Illustrations of the systems used in the experiments. The multirotor and the rover have translational invariance in two (horizontal and vertical) directions, and the cart-pole system in one (horizontal) direction.

proposed method. The robot evaluates $W(q)$ for all $q \in Q$ in the offline phase, and solves the QP in the online phase.

A. Planar Multirotor

The first system (Fig. 3 (a)) we use resembles a multirotor drone with two rotors, confined on the vertical plane. Its six dimensional state $x = [p_x, v_x, p_y, v_y, \theta, \omega]^\top \in \mathbb{R}^6$ obeys the control affine dynamics $\dot{x} = f(x) + g(x)u$ as follows:

$$\begin{aligned} \dot{p}_x &= v_x & \dot{v}_x &= (f_1 + f_2)\sin\theta \\ \dot{p}_y &= v_y & \dot{v}_y &= (f_1 + f_2)\cos\theta - 2 \\ \dot{\theta} &= \omega & \dot{\omega} &= f_1 - f_2, \end{aligned} \tag{18}$$

where p_x and p_y are the horizontal and vertical positions, θ is the tit angle, $v_{(\cdot)}$ are the linear velocity components, and ω is the angular velocity. As depicted in Fig. 3 (a), the two-dimensional input $u = [f_1, f_2]^\top \in \mathbb{R}^2$ represent the normalized thrust forces given from the rotors, and we assume they are bounded by $f_{(\cdot)} \in [0, 2]$.

To formulate the safety filter, we first find the CLF for the linearized dynamics around the origin, which is in the quadratic form

$$V(x) = \frac{1}{2} x^\top S x \tag{19}$$

using the computationally inexpensive LQR technique, where $S \in \mathbb{R}^{6\times 6}$ is a positive definite symmetric matrix. To determine the region of attraction of V, we consider the following optimization problem:

$$\begin{aligned} \mathcal{F}(b) = \max_{x\in\mathbb{R}^6} \quad & \min \left\{ \frac{\partial V}{\partial x}(f(x) + g(x)u) : u \in U \right\} \\ \text{s.t.} \quad & V(x) \le b. \end{aligned} \tag{20}$$

Note that the $\min\{\cdot\}$ is a linear program whose optimal value always exists and is finite (U is a compact polytope in this example), the objective function is Lipschitz (recall that we have assumed Lipschitz f and g) within the feasible set $D = \{x \in \mathbb{R}^n : V(x) \le b\}$, and the feasible set is compact for any positive b. We therefore can do a branch-and-bound optimization for fixed b to verify $\mathcal{F}(b) < 0$, which implies D is a valid region of attraction for the original nonlinear system. We examined $\mathcal{F}(b)$ for a number of b values and selected a largest one that gives negative $\mathcal{F}(b)$ value.

We use four different randomly-generated static 2D grid world environments in which the obstacles are represented as squares, and Q is the set of centers of collision-free cells. The shape of the multirotor is assumed to be a circle of radius $r > 0$ centered at $(p_x, p_y) \in \mathbb{R}^2$. Let $q_{\text{obs}} \in Q_{\text{obs}}$ represent the centers of the square-shaped obstacles, d_{obs} their size. We evaluate $W(q)$ as follows:

$$W(q) = \min\left\{ b, \min_{q_{\text{obs}}\in Q_{\text{obs}}} w(q_{\text{obs}} - q) \right\} - \epsilon, \tag{21}$$

where $w(\cdot)$ is given as the solution to the following QP, which searches for the minimum V value on an obstacle:

$$\begin{aligned} w(\delta q) = \min_{x\in\mathbb{R}^6, p\in\mathbb{R}^2} \quad & \frac{1}{2} x^\top S x \\ \text{s.t.} \quad & \|p - \delta q\|_\infty \le \frac{1}{2} d_{\text{obs}} + r,\ Ep = x. \end{aligned} \tag{22}$$

A small positive real number $\epsilon > 0$ enables the strict inequality of (16) to hold.

The reference input is given from the below procedure:

1) Using a grid-based A* path planner, find a path from the current position $[p_x, p_y]^\top$ to the goal position.
2) Depending on the direction of the initial path segment, use the following input as the reference input:

$$u_{\text{ref}} = \begin{cases} [2, 2]^\top & \text{(upward)} \\ [0, 0]^\top & \text{(downward)} \\ [2, 0]^\top & \text{(rightward)} \\ [0, 2]^\top & \text{(leftward).} \end{cases} \tag{23}$$

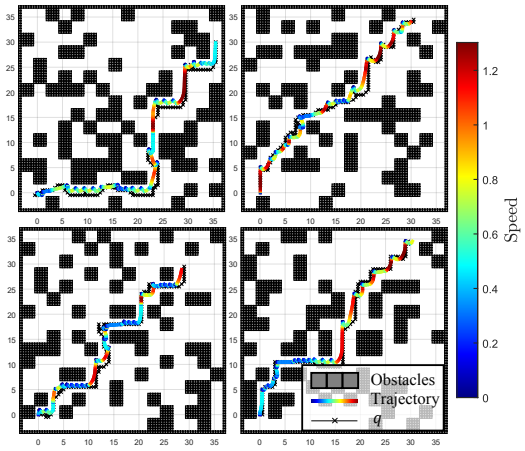


Fig. 4. The simulation environment and the results for the multirotor example. The robots are commanded to move from the bottom left corner to the top right corner through obstacle-cluttered spaces. Different colors of the trajectories denote the robot's speed $\sqrt{v_x^2 + v_y^2}$. Observe that the robot is allowed to move faster through larger spaces between obstacles.

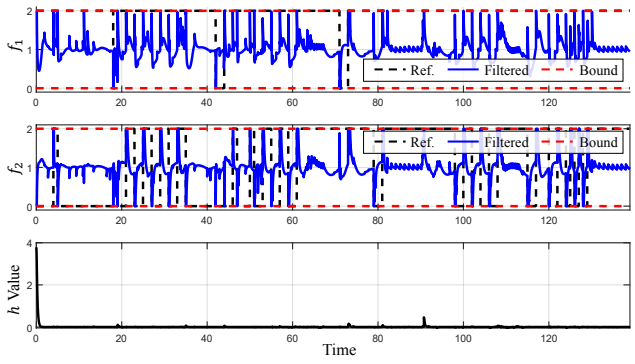


Fig. 5. Simulation result for the multirotor example in the first environment (top left of Fig. 4). Even under an aggressive bang-bang input signal, the safety filter successfully kept the value of h nonnegative while not exceeding the input bound.

3) Keep the reference input constant during duration Δt. If the robot did not reach the goal position, go to 1).

It is a (inherently very dangerous) bang-bang controller, and the system therefore should rely solely on the safety filter to avoid collision.

We implemented the safety filter using MATLAB's `quadprog` QP solver and `ode45` ODE solver. The trajectories of the robot in four different simulation environments are shown in Fig. 4. Fig. 5 depicts the input signals and how the value of h changes along the trajectory for the first environment. It can be clearly seen that h value is kept nonnegative throughout the simulation, except for a very small duration which was mainly caused by numerical integration error of the ODE solver.

B. Cart-Pole

The second target system (Fig. 3 (b)) is an inverted pendulum on a one-dimensional cart, also known as the cart-pole. Let the four-dimensional state vector be written as $x = [p, \theta, v, \omega]^\top \in \mathbb{R}^4$ where p is the horizontal position, θ is the leaning angle of the pendulum, $v = \dot{p}$ is the velocity,

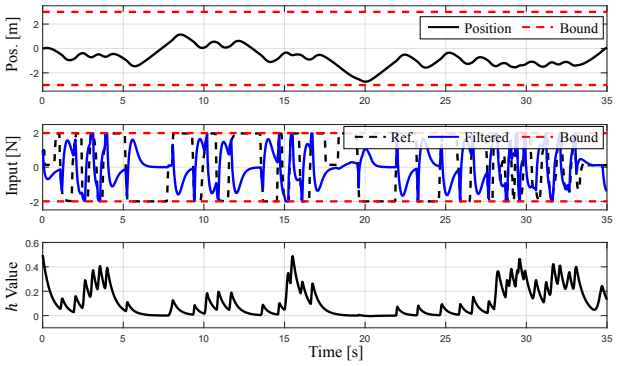


Fig. 6. Simulation result for the cart-pole dynamics. Although given an aggressive input signal, the safety filter keeps the system within the safe region while not exceeding the input bound. (Pos.: Position, Ref.: Reference)

and $\omega = \dot{\theta}$ is the change rate of θ. The dynamics of the system is written as follows:

$$\begin{bmatrix} -\cos\theta & -l \\ m_1 + m_2 & m_2 l \cos\theta \end{bmatrix} \begin{bmatrix} \ddot{p} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} -g \sin\theta \\ u + m_2 l \dot{\theta}^2 \sin\theta \end{bmatrix}. \quad (24)$$

The meanings and values of the physical parameters appearing in this dynamics are summarized in Table I. The safety requirements are to constrain the cart within the region $p \in [-3\text{ m}, 3\text{ m}]$, and the input $u \in [-f_m, f_m]$ ($f_m > 0$) is manually given by a human operator who is instructed to give aggressive input. The LQR-based CLF and $W(q)$ similar to the multirotor system were used. Since the cart-pole system is symmetric in the p direction only, we let Q be the set of 1D grid points within the set $[-3\text{ m}, 3\text{ m}]$ equally spaced by separation d.

We implemented the simulation environment using the C++ programming language, and the safety filter was able to run faster than 1000 Hz. The fast computation speed is due to one-dimensional input, so that the QP of the safety filter has a closed-form solution. The simulation results in Fig. 6 clearly shows that the proposed safety filter was capable of keeping the system within the safety bounds. Also observe that the h values are kept nonnegative throughout the simulation and never decays faster than the prescribed decay rate.

C. Ground Rover (Simulation)

Finally, we conducted simulation and hardware experiments using a ground rover (Fig. 3 (c)) which can be modeled using the following dynamics:

$$\begin{aligned} \dot{p}_x &= v\cos\theta \quad \dot{v} = a \\ \dot{p}_y &= v\sin\theta \quad \dot{\theta} = \omega, \end{aligned} \quad (25)$$

where $x = [p_x, p_y, v, \theta]^\top \in \mathbb{R}^4$ is the state, $u = [a, \omega]^\top \in \mathbb{R}^2$ is the input. The symbols p_x, p_y respectively denote the horizontal and vertical coordinates of the vehicle, v is the velocity, θ is the heading angle. The input components are linear acceleration a and turning rate ω. The input is constrained in a box-shaped set $U = [-a_m, a_m] \times [-\omega_m, \omega_m]$ where a_m and ω_m are positive reals. We use the following

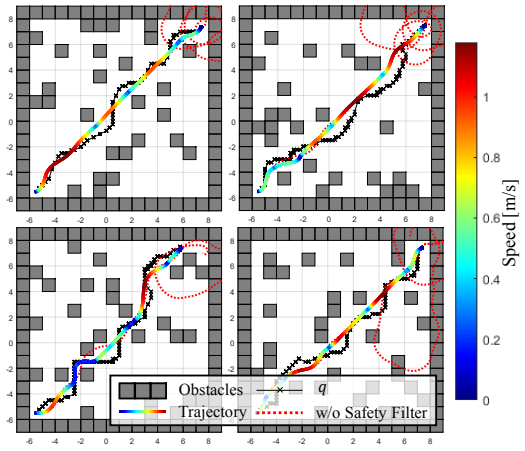


Fig. 7. The simulation environment and the results for the rover example. The robots are commanded to move from the bottom left corner to the top right corner through obstacle-cluttered spaces with an aggressive reference controller (observe the unsafe behavior of the agent without the safety filter shown in red dotted curve).

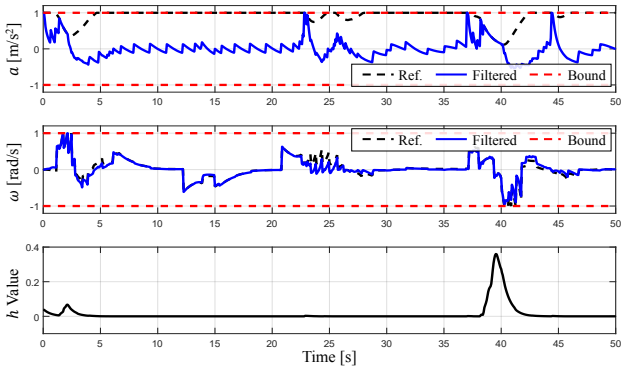


Fig. 8. Simulation result for the ground rover dynamics. The h value is always kept nonnegative by the safety filter. (Ref.: Reference)

hand-crafted CLF for this example:

$$V(x) = \frac{1}{2}(p_x^2 + p_y^2) + \frac{v|v|}{2a_m}(p_x \cos\theta + p_y \sin\theta) + \frac{v^4}{8a_m^2} + \frac{1}{2}\mu_v v^2 + \mu_\theta(1 - \cos\theta), \tag{26}$$

where μ_v an d μ_θ are positive real parameters. To obtain $W(\cdot)$, we used

$$V(x) \le \bar{V}(x) = \begin{cases} \frac{1}{2}(p_x^2 + p_y^2) & (p_x^2 + p_y^2 \le \mu_v^2) \\ -\frac{1}{2}\mu_v^2 + \mu_v\sqrt{p_x^2 + p_y^2} & (p_x^2 + p_y^2 > \mu_v^2) \end{cases} \tag{27}$$

as a conservative approximation of the upper bound of V, then formulated and solved the optimization similar to (22).

We consider a grid world similar to the multirotor example, and the reference input is given from a simple proportional feedback law that tracks the A* path to the goal. Simulation was conducted in four different randomly-generated obstacle configurations shown in Fig. 7. The safety filter was implemented using MATLAB's `quadprog` and `ode45`. Fig. 8 depicts the inputs and the h values during the simulation.

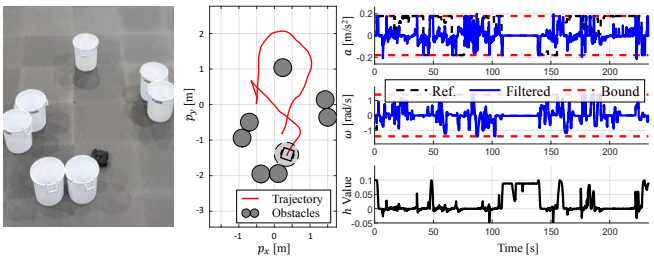


Fig. 9. The hardware experiment result. Left: A snapshot taken from the experiment. Middle: The visualization of the set C_q and the robot when the snapshot was taken. Right: The input and h values, plotted as a function of time. The short intervals where h values are not kept nonnegative mainly result from communication delay, model mismatch, and numerical errors of the optimization solver.

D. Ground Rover (Hardware Experiment)

Using the same rover model but circle-shaped obstacles, we conducted an experiment using Turtlebot 3 [24], a differential-drive mobile robot hardware. In the experiment, the robot is required to track the reference input given by a human operator while not colliding with the obstacles. The operator is instructed to transmit aggressive inputs toward the obstacles.

The experiment results are briefly shown in Fig. 9. The safety filter was implemented using OSQP optimization solver [25]. The robot was able to on-board generate filtered input at 100 Hz. To show fast adaptability to various environments, the positions of the obstacles are given using a motion capture system, and at every update, the robot has to update the $W(q)$ values according to the new configuration. The robot was able to handle obstacle configuration changes and $W(q)$ updates within 1 second.

The results in Fig. 9 well showcases that the safety filter was able to keep the h value nonnegative, except for a short intervals, although these were not significant to cause any collision. A good interpretation for this is that they were mainly caused by communication delay and model mismatch which are unavoidable when deploying real hardware.

VI. Conclusion

This paper introduced a CBF-QP-like safety filter for navigation tasks in mobile robot systems. To guarantee safety even in cluttered environments with many obstacles, the proposed safety filter uses multiple CBFs synthesized out of a single CLF only, namely Lyapunov-based CBFs.

While inheriting favorable characteristics, including real-time performance, from CBF-QP, the proposed safety filter offers versatility to various obstacle configurations without the need for repeating the computationally intensive CBF synthesis process. The proposed method was validated in simulation and hardware experiments using different system dynamics. In all scenarios, even under the presence of many obstacles and dangerous reference input, the safety filter was capable of keeping the system's state away from collision.

Some limitations of this work are as follows. Firstly, how to synthesize an effective CLF given a system dynamics that works well with the proposed method is still an open problem. Additionally, the sudden jumps introduced by CBF

switches might cause unfavorable behavior of the system. These should be addressed in the future works.

ACKNOWLEDGMENT

The authors would like to thank Dabin Kim (Seoul National University) for his help conducting the experiments.

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] 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.

[3] K. P. Wabersich, A. J. Taylor, J. J. Choi, K. Sreenath, C. J. Tomlin, A. D. Ames, and M. N. Zeilinger, "Data-driven safety filters: Hamilton-Jacobi reachability, control barrier functions, and predictive methods for uncertain systems," *IEEE Control Systems Magazine*, vol. 43, no. 5, pp. 137–177, 2023.

[4] A. Singletary, A. Swann, Y. Chen, and A. D. Ames, "Onboard safety guarantees for racing drones: High-speed geofencing with control barrier functions," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2897–2904, 2022.

[5] X. Wang, "Ensuring safety of learning-based motion planners using control barrier functions," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4773–4780, 2022.

[6] Y. Emam, G. Notomista, P. Glotfelter, Z. Kira, and M. Egerstedt, "Safe reinforcement learning using robust control barrier functions," *IEEE Robotics and Automation Letters*, pp. 1–8, 2022.

[7] Q. Nguyen and K. Sreenath, "Exponential control barrier functions for enforcing high relative-degree safety-critical constraints," in *2016 American Control Conference (ACC)*. IEEE, 2016, pp. 322–328.

[8] W. Xiao and C. Belta, "Control barrier functions for systems with high relative degree," in *2019 IEEE 58th conference on decision and control (CDC)*. IEEE, 2019, pp. 474–479.

[9] W. Xiao, C. Belta, and C. G. Cassandras, "Adaptive control barrier functions," *IEEE Transactions on Automatic Control*, vol. 67, no. 5, pp. 2267–2281, 2021.

[10] J. J. Choi, D. Lee, K. Sreenath, C. J. Tomlin, and S. L. Herbert, "Robust control barrier–value functions for safety-critical control," in *2021 60th IEEE Conference on Decision and Control (CDC)*. IEEE, 2021, pp. 6814–6821.

[11] X. Xu, J. W. Grizzle, P. Tabuada, and A. D. Ames, "Correctness guarantees for the composition of lane keeping and adaptive cruise control," *IEEE Transactions on Automation Science and Engineering*, vol. 15, no. 3, pp. 1216–1229, 2017.

[12] L. Wang, D. Han, and M. Egerstedt, "Permissive barrier certificates for safe stabilization using sum-of-squares," in *2018 Annual American Control Conference (ACC)*. IEEE, 2018, pp. 585–590.

[13] A. Robey, H. Hu, L. Lindemann, H. Zhang, D. V. Dimarogonas, S. Tu, and N. Matni, "Learning control barrier functions from expert demonstrations," in *2020 59th IEEE Conference on Decision and Control (CDC)*. IEEE, 2020, pp. 3717–3724.

[14] W. Xiao, T.-H. Wang, R. Hasani, M. Chahine, A. Amini, X. Li, and D. Rus, "Barriernet: Differentiable control barrier functions for learning of safe robot control," *IEEE Transactions on Robotics*, 2023.

[15] A. Majumdar and R. Tedrake, "Funnel libraries for real-time robust feedback motion planning," *The International Journal of Robotics Research*, vol. 36, no. 8, pp. 947–982, 2017.

[16] S. Kousik, S. Vaskov, F. Bu, M. Johnson-Roberson, and R. Vasudevan, "Bridging the gap between safety and real-time performance in receding-horizon trajectory design for mobile robots," *The International Journal of Robotics Research*, vol. 39, no. 12, pp. 1419–1469, 2020.

[17] S. Singh, B. Landry, A. Majumdar, J.-J. Slotine, and M. Pavone, "Robust feedback motion planning via contraction theory," *The International Journal of Robotics Research*, vol. 42, no. 9, pp. 655–688, 2023.

[18] Y.-C. Chang, N. Roohi, and S. Gao, "Neural Lyapunov control," *Advances in neural information processing systems*, vol. 32, 2019.

[19] Z. Gong, M. Zhao, T. Bewley, and S. Herbert, "Constructing control Lyapunov-value functions using Hamilton-Jacobi reachability analysis," *IEEE Control Systems Letters*, vol. 7, pp. 925–930, 2022.

[20] J. Cortes, "Discontinuous dynamical systems," *IEEE Control Systems Magazine*, vol. 28, no. 3, pp. 36–73, 2008.

[21] F. Blanchini, "Set invariance in control," *Automatica*, vol. 35, no. 11, pp. 1747–1767, 1999.

[22] H. K. Khalil, *Nonlinear control*. Pearson New York, 2015, vol. 406.

[23] B. J. Morris, M. J. Powell, and A. D. Ames, "Continuity and smoothness properties of nonlinear optimization-based feedback controllers," in *2015 54th IEEE Conference on Decision and Control (CDC)*. IEEE, 2015, pp. 151–158.

[24] "Turtlebot 3," https://emanual.robotis.com/docs/en/platform/turtlebot3/overview/, accessed: 2023-09-15.

[25] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, "OSQP: an operator splitting solver for quadratic programs," *Mathematical Programming Computation*, vol. 12, no. 4, pp. 637–672, 2020.

APPENDIX

A. Proof for Theorem 1

Proof. Let s be an arbitrary nonnegative integer. For brevity, in this proof, the notations $[s]$ and $[s)$ are used as shorthands for $[s \cdot \delta t, (s+1) \cdot \delta t]$ and $[s \cdot \delta t, (s+1) \cdot \delta t)$, respectively.

Recall that $j(t)$ is kept constant on the interval $[s)$, hence we can let $j(t) = j_s$ for all $t \in [s)$. Therefore, the control input during $[s)$ is identical to a CBF-QP using h_{j_s}. From assumption 2), we have that if $h_{j(s\cdot\delta t)}(x(s \cdot \delta t)) \geq 0$, a unique solution for (12) exists on the interval $t \in [s)$. Let $x((s+1) \cdot \delta t) = \lim_{\tau \nearrow (s+1)\cdot\delta t} x(\tau)$, where the existence of the limit also follows from the second assumption. Since we do not permit jumps in the state x, such trajectory $x(t)$ is a unique solution for (12) on $t \in [s]$. Such solution is given from the CBF-QP using h_{j_s}, so the invariance property holds:

$$h_{j_s}(x(s \cdot \delta t)) \geq 0 \Rightarrow h_{j_s}(x(t)) \geq 0, \quad \forall t \in [s]. \tag{28}$$

Recall also that h is defined as the pointwise maximum of all given CBFs, and according to the update rule (10), whenever t is a multiple of δt,

$$h(x(t)) = h_{j(t)}(x(t)) = h_{j_s}(x(t)). \tag{29}$$

Combining with (28), we get

$$h(x(t)) \geq h_{j_s}(x(t)) \geq 0 \quad \forall t \in [s], \tag{30}$$

and

$$h_{j_{s+1}}(x((s+1) \cdot \delta t)) = h(x((s+1) \cdot \delta t)) \geq 0. \tag{31}$$

Therefore, from (28) and (31),

$$h(x(s \cdot \delta t)) \geq 0 \Rightarrow h(x(t)) \geq 0 \quad \forall t \in [s]. \tag{32}$$

Applying mathematical induction on $s \in \mathbb{Z}_{\geq 0}$ starting from assumption 3) completes the proof. □

B. Proof for Theorem 2

Proof. Let $u^*(\cdot) = \arg\min_{u \in U} \dot{V}(\cdot, u)$. Since V is a CLF, $\dot{V}(y, u^*(y)) \leq 0$ for all $y \in D$. Let $y = x - Eq$. If $x \in C_q$, $y \in D$ because $W(q) \leq b$.

Since we have assumed translational invariance of the dynamics, $\dot{\beta}(y, u) = \dot{\beta}(x, u)$ for any differentiable function $\beta : \mathbb{R}^n \to \mathbb{R}$. Therefore,

$$\max_{u \in U} \dot{h}_q(x, u) = -\min_{u \in U} \dot{V}(y, u) \geq -\dot{V}(y, u^*(y)) \geq 0, \tag{33}$$

and

$$\max_{u \in U} \dot{h}_q(x, u) + \alpha(h_q(x)) \geq \alpha(h_q(x)) \geq 0, \tag{34}$$

for all $x \in C_q$, since α is an extended class-$\mathcal{K}_\infty$ function and $h_q(x) \geq 0$ for all $x \in C_q$. □