Model-Free Safety-Critical Control for Robotic Systems

Tamas G. Molnar, Ryan K. Cosner, Andrew W. Singletary, Wyatt Ubellacker, and Aaron D. Ames

***Abstract*— This paper presents a framework for the safety-critical control of robotic systems, when safety is defined on safe regions in the configuration space. To maintain safety, we synthesize a safe velocity based on control barrier function theory without relying on a – potentially complicated – high-fidelity dynamical model of the robot. Then, we track the safe velocity with a tracking controller. This culminates in *model-free safety critical control*. We prove theoretical safety guarantees for the proposed method. Finally, we demonstrate that this approach is application-agnostic. We execute an obstacle avoidance task with a Segway in high-fidelity simulation, as well as with a Drone and a Quadruped in hardware experiments.**

I. INTRODUCTION

Safety is one of the most fundamental requirements in the control of many robotic systems, including legged [1], flying [2] and wheeled robots [3]. Provable safety guarantees and safety-critical control for robotics have therefore attracted significant attention over the past years. Synthesizing safety-critical controllers, however, typically relies on high-fidelity dynamical models describing the robots, which are often complicated and high-dimensional. The underlying control laws, therefore, are nontrivial to synthesize and implement. For example, control barrier functions (CBFs) [4], [5] are a popular tool to achieve provable safety guarantees, although designing CBFs and calculating the corresponding safe control inputs may be nontrivial if the dynamics are complicated.

To tackle this, [6] proposed model-free barrier functions, wherein a data-driven approach is used to create a barrier function, while [7], [8] used robust CBF formulations to overcome the effects of unmodeled dynamics. Furthermore, many works rely on reduced-order models for planning and control [9]. These include single integrator models for multi-robot applications [10], [11] or unicycle models for wheeled robots [12], [13], which have proven to be extremely useful models despite being overly simplistic. In what follows, we draw inspiration from these models and approaches.

In this paper, we rely on CBFs to synthesize safe controllers for robotic systems, while handling the safety-critical aspect of this problem in a model-free fashion, i.e., without relying on the full-order dynamics of the robot. Namely, we follow the approach of [14], [15], where a safe velocity was designed based on reduced-order kinematics – i.e., without the full dynamic model – and this safe velocity was tracked by a velocity tracking controller. We emphasize that this approach is agnostic to the application domain, although the underlying velocity tracking controllers depend on the specific robotic system. Velocity tracking, however, is well-established in robotics [16] and controllers executing stable velocity tracking are available for many robots. Potentially, tracking can be done by model-free controllers (such as PD or PID), leading to *model-free safety-critical control*.

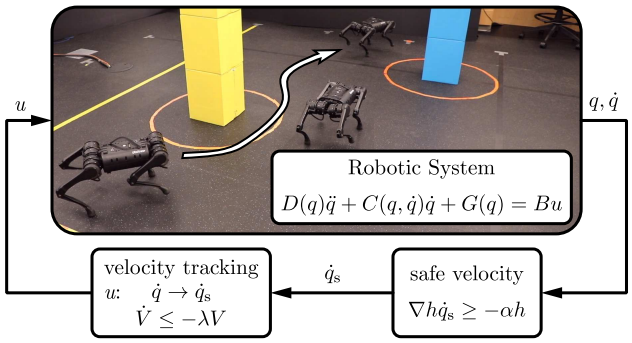

Fig. 1. The proposed control method and its execution on hardware. While the safety-critical controller does not rely on the full dynamical model of the robot, it controls the motion in a provably safe manner.

While the idea underlying this control method was established in [14], the present paper formalizes and generalizes this approach via two main contributions. First, we provide theoretical proof of the safe behavior for robotic systems executing the proposed control approach. Second, we demonstrate the applicability of this method on wheeled, flying and legged robots: a Segway (in simulation), a Drone and a Quadruped (in hardware experiments). This justifies that the method is agnostic to the application domain.

The paper is organized as follows. Section II revisits control Lyapunov and control barrier functions to achieve stability and safety. Section III outlines the proposed control method, states and proves the safety guarantees thereof. Section IV discusses robotic applications through simulations and hardware experiments. Section V concludes the paper.

II. PRELIMINARIES

Our approach relies on stable tracking of a safe velocity to achieve safety for robotic systems. Thus, first we introduce the notions of stability and safety, and the guarantees thereof provided by control Lyapunov functions (CLFs) and control barrier functions (CBFs). CLFs and CBFs are illustrated in Fig. 2 together with a stable and a safe trajectory.

Consider the control-affine system:

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

*This research is supported in part by the National Science Foundation, CPS Award #1932091, Dow (#227027AT) and Aerovironment.

Authors are with the Department of Control and Dynamical Systems and the Department of Mechanical and Civil Engineering, California Institute of Technology, Pasadena, CA 91125, USA. {tmolnar, rkcosner, asinglet, wubellac, ames}@caltech.edu

with state space $X \subseteq \mathbb{R}^n$, state $x(t) \in X$, set of admissible inputs $U \subseteq \mathbb{R}^m$, and control input $u(t) \in U$. We assume $f : X \to \mathbb{R}^n$ and $g : X \to \mathbb{R}^{n\times m}$ are Lipschitz continuous. Hence, given an initial condition $x(0) = x_0 \in X$ and a Lipschitz continuous controller $k : X \to U$, $u = k(x)$, the system has a unique solution $x(t)$ which we assume to exist for all $t \geq 0$. Furthermore, we assume that $x(t) \equiv 0$ is an equilibrium solution of (1) if $u(t) \equiv 0$ (i.e., $f(0) = 0$) and X is an open and connected neighborhood of $x = 0$.

Throughout the paper we use the following notation. $\|.\|$ denotes Euclidean norm, whereas $\|.\|_\infty$ is maximum norm. We say a function is of class-$\mathcal{K}$ (or extended class-$\mathcal{K}$) if it is defined over nonnegative real numbers (or both negative and nonnegative reals), it is continuous, strictly monotonically increasing and zero at zero.

A. Stability and Control Lyapunov Functions

Hereinafter, we rely on the notion of exponential stability, defined as follows.

Definition 1. The equilibrium $x = 0$ of system (1) is *exponentially stable* if there exists $a, M, \beta \in \mathbb{R}_{>0}$ such that $\|x_0\| \leq a \Rightarrow \|x(t)\| \leq M\mathrm{e}^{-\beta t}\|x_0\|$, $\forall t \geq 0$.

An efficient technique to achieve exponential stability is control synthesis via control Lyapunov functions (CLFs) [17], [18]. The definition of CLFs and the stability guarantees provided by them is stated formally below.

Definition 2. A continuously differentiable function $V : X \to \mathbb{R}_{\geq 0}$ is a *control Lyapunov function (CLF)* for (1) if there exists $c, k_1, k_2, \lambda \in \mathbb{R}_{>0}$ such that $\forall x \in X$:

$$\begin{aligned} k_1\|x\|^c \leq V(x) \leq k_2\|x\|^c \\ \inf_{u\in U} \dot{V}(x,u) \leq -\lambda V(x), \end{aligned} \tag{2}$$

where

$$\dot{V}(x,u) = \nabla V(x)(f(x) + g(x)u) \tag{3}$$

is the derivative of V along system (1).

Theorem 1 ([17]). *If V is a CLF for (1), then any locally Lipschitz continuous controller $u = k(x)$ satisfying*

$$\dot{V}(x, k(x)) \leq -\lambda V(x), \tag{4}$$

$\forall x \in X$ renders $x = 0$ exponentially stable.

Theorem 1 establishes that synthesizing a control input u while enforcing condition (4) achieves exponential stability.

B. Safety and Control Barrier Functions

We consider system (1) safe if its state $x(t)$ is located within a *safe set* $S \subset X$ for all time, as stated below.

Definition 3. System (1) is *safe* w.r.t. S if S is forward invariant under (1), that is, $x_0 \in S \Rightarrow x(t) \in S$, $\forall t \geq 0$.

The selection of the safe set is application-driven, for example, it may represent robot positions that do not collide with nearby obstacles. In what follows, we define the safe set S as the 0-superlevel set of a continuously differentiable function $h : X \to \mathbb{R}$:

$$S = \{x \in X : h(x) \geq 0\}. \tag{5}$$

Then, control barrier functions (CBFs) can be used as tools to synthesize provably safe controllers in a similar fashion to how CLFs achieve stability.

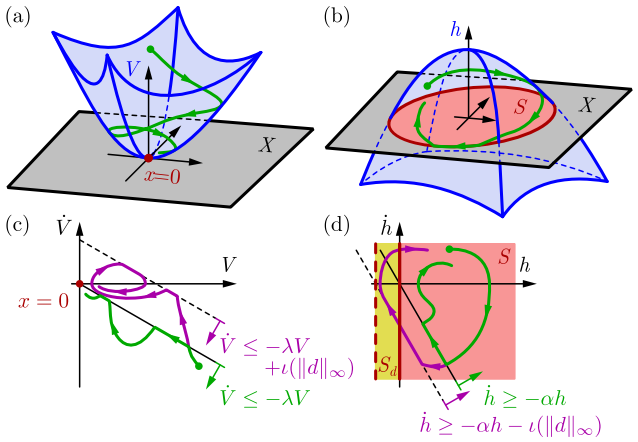

Fig. 2. (a) Illustration of a CLF and a stable trajectory. (b) Representation of a CBF and a safe trajectory. While V is nonnegative, h may take any real value. (c) Representation of the stability condition (4) and a stable trajectory (green), the ISS condition (10) and an input-to-state stable trajectory (purple). For ISS the trajectory converges to a neighborhood of $x = 0$. (d) Illustration of the safety condition (8) and a safe trajectory (green), the ISSf condition (12) and an input-to-state safe trajectory (purple). For ISSf a superset S_d of S is forward invariant.

Definition 4. A continuously differentiable function $h : X \to \mathbb{R}$ is a *control barrier function (CBF)* for (1) if there exists $\alpha \in \mathbb{R}_{>0}$ such that $\forall x \in S$: [1]

$$\sup_{u\in U} \dot{h}(x,u) \geq -\alpha h(x), \tag{6}$$

where

$$\dot{h}(x,u) = \nabla h(x)(f(x) + g(x)u) \tag{7}$$

is the derivative of h along system (1).

Theorem 2 ([19]). *If h is a CBF for (1), then any locally Lipschitz continuous controller $u = k(x)$ satisfying*

$$\dot{h}(x, k(x)) \geq -\alpha h(x), \tag{8}$$

$\forall x \in S$ renders (1) safe w.r.t. S.

Theorem 2 establishes safety-critical controller synthesis by condition (8). For example, a desired but not necessarily safe controller $k_\mathrm{d}(x)$ can be modified in a minimally invasive way to a safe controller by solving the quadratic program:

$$\begin{aligned} k(x) = \operatorname*{argmin}_{u\in U} \ & (u - k_\mathrm{d}(x))^\top (u - k_\mathrm{d}(x)) \\ \text{s.t.} \ & \dot{h}(x,u) \geq -\alpha h(x). \end{aligned} \tag{9}$$

C. Effect of Disturbances

In practice, robotic systems are often subject to unknown disturbances that may compromise stability or safety. For

[1] In general, α can be chosen as an extended class-$\mathcal{K}$ function, while here we use a constant for simplicity.

example, a bounded disturbance $d \in \mathbb{R}^m$ added to the input u leads to the system $\dot{x} = f(x) + g(x)(u + d)$.

In such scenarios, the notion of exponential stability can be extended to *exponential input-to-state stability (ISS)*, by relaxing the condition in Definition 1. Namely, we require that there exists a class-$\mathcal{K}$ function μ such that $\|x_0\| \leq a \Rightarrow \|x(t)\| \leq M\mathrm{e}^{-\beta t}\|x_0\| + \mu(\|d\|_\infty)$, $\forall t \geq 0$. That is, instead of converging to the origin, solutions are required to converge to a neighborhood of the origin which depends on the size of the disturbance. In [20], [21] it was shown that exponential ISS can be achieved by modifying condition (4) in Theorem 1 to

$$\dot{V}(x, u) \leq -\lambda V(x) + \iota(\|d\|_\infty), \tag{10}$$

for some class-$\mathcal{K}$ function ι.

Similarly, in the presence of disturbances safety can be relaxed to *input-to-state safety (ISSf)* by requiring that the system stays within a neighborhood $S_d \supseteq S$ of the safe set S which depends on the size of the disturbance: $x_0 \in S_d \Rightarrow x(t) \in S_d,\ \forall t \geq 0$. Again, we can define this neighborhood as a 0-superlevel set:

$$S_d = \{x \in X : h(x) + \gamma(\|d\|_\infty) \geq 0\}, \tag{11}$$

with some class-$\mathcal{K}$ function γ. It was established in [22] that ISSf is guaranteed by replacing (8) in Theorem 2 with

$$\dot{h}(x, u) \geq -\alpha h(x) - \iota(\|d\|_\infty), \tag{12}$$

for some class-$\mathcal{K}$ function ι.

III. MODEL-FREE SAFETY-CRITICAL CONTROL

Now consider robotic systems with configuration space $Q \subseteq \mathbb{R}^n$, configuration coordinates $q \in Q$, set of admissible inputs $U \subseteq \mathbb{R}^m$, control input $u \in U$, and dynamics:

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

where $D(q) \in \mathbb{R}^{n\times n}$ is the inertia matrix, $C(q, \dot{q}) \in \mathbb{R}^{n\times n}$ contains centrifugal and Coriolis forces, $G(q) \in \mathbb{R}^n$ involves gravity terms and $B \in \mathbb{R}^{n\times m}$ is the input matrix. $D(q)$ is considered to be symmetric, positive definite, whereas $\dot{D}(q, \dot{q}) - 2C(q, \dot{q})$ is skew-symmetric. We further consider control laws of the form $k : Q \times \mathbb{R}^n \to \mathbb{R}^m,\ u = k(q, \dot{q})$, initial conditions $q(0) = q_0,\ \dot{q}(0) = \dot{q}_0$, and assume that a corresponding unique solution $q(t)$ exists for all $t \geq 0$.

We consider the robotic system *safe* if its configuration q lies within a *safe set* S for all time: $q(t) \in S,\ t \geq 0$. The safe set is defined as the 0-superlevel set of a continuously differentiable function $h : Q \to \mathbb{R}$:

$$S = \{q \in Q : h(q) \geq 0\}, \tag{14}$$

where we assume finite, nonzero gradient for h: $\exists C_h \in \mathbb{R}_{>0}$ such that $\|\nabla h(q)\| \leq C_h, \forall q \in S$ and $\nabla h(q) \not\equiv 0$. The selection of h is application-specific. Note that we consider cases where safety depends on the configuration only, without restrictions on velocity (h is independent of $\dot{q}$).

Problem Statement. *For the robotic system (13), design a controller $u = k(q, \dot{q})$ that achieves safety with respect to set S in (14), i.e., $q(t) \in S,\ \forall t \geq 0$ given certain initial conditions $q_0 \in Q$ and $\dot{q}_0 \in \mathbb{R}^n$.*

A. Control Method

Following [14], [15], we seek to maintain safety by designing a safe velocity and tracking this velocity. This method reduces the complexity of safety-critical control significantly, while velocity tracking controllers are widely used [16]. The approach will also allow us to realize safety-critical control in a model-free fashion.

We synthesize the *safe velocity* $\dot{q}_\mathrm{s}$ such that it satisfies:

$$\nabla h(q)\dot{q}_\mathrm{s} \geq -\alpha h(q), \tag{15}$$

cf. (8), for some $\alpha \in \mathbb{R}_{>0}$ to be selected. The safe velocity $\dot{q}_\mathrm{s}$ depends on the configuration q. Note that (15) is a kinematic condition that does not depend on the full dynamics (13).

To track the safe velocity, we define the tracking error:

$$\dot{e} = \dot{q} - \dot{q}_\mathrm{s}. \tag{16}$$

and assume it is differentiable, i.e., $\ddot{q}_\mathrm{s}$ and $\ddot{e}$ exist. First, we consider the scenario that a velocity tracking controller $u = k(q, \dot{q})$ is able to drive the error $\dot{e}$ to zero exponentially, then we address the effect of disturbances.

In particular, we consider velocity tracking controllers for which there exists a continuously differentiable Lyapunov function $V : Q \times \mathbb{R}^n \to \mathbb{R}_{\geq 0}$ such that $\forall (q, \dot{e}) \in Q \times \mathbb{R}^n$:

$$k_1\|\dot{e}\| \leq V(q, \dot{e}) \leq k_2\|\dot{e}\|, \tag{17}$$

for some $k_1, k_2 \in \mathbb{R}_{>0}$, and there exists $\lambda \in \mathbb{R}_{>0}$ such that u satisfies:

$$\dot{V}(q, \dot{e}, \dot{q}, \ddot{q}_\mathrm{s}, u) \leq -\lambda V(q, \dot{e}), \tag{18}$$

$\forall (q, \dot{e}, \dot{q}, \ddot{q}_\mathrm{s}) \in Q \times \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}^n$; cf. (4).

Before discussing its safety guarantees, we demonstrate the applicability of this method on a motivating example.

Example 1 (Double integrator system)**.** Here we revisit the example in [14]. As the simplest instantiation of (13), consider a double integrator system in two dimensions:

$$\ddot{q} = u, \tag{19}$$

where $q \in \mathbb{R}^2$ is the planar position of the robot and $u \in \mathbb{R}^2$. Our goal is to navigate the system from a start position q_0 to a goal q_g while avoiding obstacles. A simple solution is to realize the desired velocity $\dot{q}_\mathrm{d} = -K_\mathrm{P}(q - q_\mathrm{g})$ that is based on a proportional controller with gain $K_\mathrm{P} \in \mathbb{R}_{>0}$.

We can avoid an obstacle of radius r centered at q_o by the help of the distance $d = \|q - q_\mathrm{o}\|$ and the CBF:

$$h(q) = d - r, \tag{20}$$

with gradient $\nabla h(q) = (q - q_\mathrm{o})^\top / \|q - q_\mathrm{o}\| = n_\mathrm{o}^\top$ equal to the unit vector n_o pointing from the obstacle to the robot. Then, the safe velocity can be found by using condition (15). Specifically, we modify the desired velocity $\dot{q}_\mathrm{d}$ in a minimally invasive fashion by solving the quadratic program:

$$\begin{aligned} \underset{\dot{q}_\mathrm{s}\in\mathbb{R}^2}{\operatorname{argmin}}\ & (\dot{q}_\mathrm{s} - \dot{q}_\mathrm{d})^\top(\dot{q}_\mathrm{s} - \dot{q}_\mathrm{d}) \\ \text{s.t.}\ & n_\mathrm{o}^\top \dot{q}_\mathrm{s} \geq -\alpha(d - r), \end{aligned} \tag{21}$$

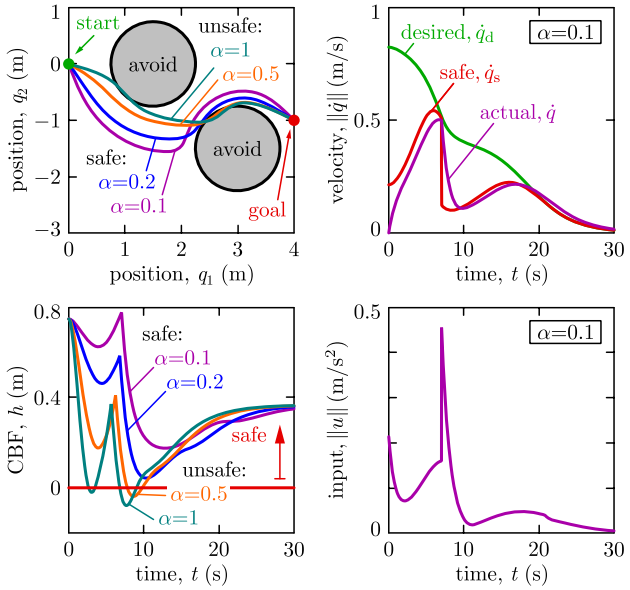

Fig. 3. Numerical simulation of the double integrator system (19) tracking the safe velocity (22). The controller is able to keep the system safe if parameter α is selected to be small enough.

cf. (9). Based on the KKT conditions [23], it has the solution:

$$\dot{q}_\mathrm{s} = \dot{q}_\mathrm{d} + \max\{-n_\mathrm{o}^\top \dot{q}_\mathrm{d} - \alpha(d-r), 0\} n_\mathrm{o}. \tag{22}$$

The safe velocity can be tracked for example by the controller $u = -K_\mathrm{D}(\dot{q} - \dot{q}_\mathrm{s})$ with gain $K_\mathrm{D} \in \mathbb{R}_{>0}$.

Fig. 3 shows four simulation results for avoiding two obstacles with parameters $K_\mathrm{P} = 0.2\,\mathrm{s}^{-1}$, $K_\mathrm{D} = 1\,\mathrm{s}^{-1}$ and $\alpha = 0.1, 0.2, 0.5$ and $1\,\mathrm{s}^{-1}$, respectively. With the proposed approach, the double integrator system avoids the obstacles, although the second-order dynamics was not directly taken into account during the CBF and control design. The condition for safety, however, is picking a small enough α value (e.g. 0.1 or 0.2), while safety is violated for larger α (e.g. 0.5 or 1). We remark that for multiple obstacles we considered the closest one at each time. This results in a nonsmooth CBF which has been analyzed in [24]. Accordingly, the safe velocity $\dot{q}_\mathrm{s}$ is only piecewise differentiable; for simplicity, our constructions are restricted to the differentiable segments.

B. Main Result

In what follows, our main result proves that tracking the safe velocity achieves safety for the full dynamics if parameter α is selected to be small enough. Specifically, for velocity tracking controllers satisfying the stability property (18) stability translates into safety for the full system (13) if $\lambda > \alpha$. This result is agnostic to the application domain. Realizing velocity tracking controllers, however, does depend on the application. Later we give examples for such controllers and corresponding CLFs. These may include PID-type controllers that achieve tracking in a model-free fashion, culminating in *model-free safety-critical control*.

The following theorem summarizes the safety guarantees provided by tracking the safe velocity.

Theorem 3. *Consider system (13), safe set (14), safe velocity satisfying (15), and velocity tracking controller satisfying (18). If $\lambda > \alpha$, safety is achieved such that $(q_0, \dot{e}_0) \in S_V \Rightarrow q(t) \in S$, $\forall t \geq 0$, where:*

$$\begin{aligned} S_V &= \{(q, \dot{e}) \in Q \times \mathbb{R}^n : h_V(q, \dot{e}) \geq 0\}, \\ h_V(q, \dot{e}) &= -V(q, \dot{e}) + \alpha_\mathrm{e} h(q), \end{aligned} \tag{23}$$

with $\alpha_\mathrm{e} = (\lambda - \alpha) k_1 / C_h > 0$ and C_h, k_1 defined at (14, 17).

Proof. Since $V(q, \dot{e}) \geq 0$, the implication $h_V(q, \dot{e}) \geq 0 \Rightarrow h(q) \geq 0$ holds. Thus, $h_V(q(t), \dot{e}(t)) \geq 0$, $\forall t \geq 0$ is sufficient to prove. We prove this by noticing that the initial conditions satisfy $h_V(q_0, \dot{e}_0) \geq 0$ and we also have:

$$\begin{aligned} \dot{h}_V(q, \dot{e}, \dot{q}, \ddot{q}_\mathrm{s}, u) &= -\dot{V}(q, \dot{e}, \dot{q}, \ddot{q}_\mathrm{s}, u) + \alpha_\mathrm{e} \nabla h(q) \dot{q} \\ &\geq \lambda V(q, \dot{e}) + \alpha_\mathrm{e} \nabla h(q) \dot{q}_\mathrm{s} + \alpha_\mathrm{e} \nabla h(q) \dot{e} \\ &\geq \lambda V(q, \dot{e}) - \alpha_\mathrm{e} \alpha h(q) + \alpha_\mathrm{e} \nabla h(q) \dot{e} \\ &\geq (\lambda - \alpha) V(q, \dot{e}) - \alpha_\mathrm{e} \|\nabla h(q)\| \|\dot{e}\| - \alpha h_V(q, \dot{e}) \\ &\geq (\lambda - \alpha) k_1 \|\dot{e}\| - \alpha_\mathrm{e} C_h \|\dot{e}\| - \alpha h_V(q, \dot{e}) \\ &\geq -\alpha h_V(q, \dot{e}). \end{aligned} \tag{24}$$

Here we used the following properties in the 6 lines of the inequality: (i) definition (23) of h_V, (ii) stability condition (18) and definition (16) of $\dot{e}$, (iii) condition (15) on the safe velocity, (iv) definition (23) of h_V and the Cauchy-Schwartz inequality, (v) lower bound of V in (17) and upper bound C_h of $\|\nabla h(q)\|$, (vi) definition of α_e. This guarantees $h_V(q(t), \dot{e}(t)) \geq 0$, $\forall t \geq 0$ by Theorem 2. □

Remark 1. Condition $\lambda > \alpha$ means the controller tracks the safe velocity fast enough (characterized by λ) compared to how fast the boundary of the safe set may be approached by the safe motion (characterized by α). In practice, one can pick a small enough α for a given velocity tracking controller, for example, by gradually increasing α from 0. The existence of such α is guaranteed by the Theorem.

Remark 2. Condition (15) is equivalent to designing a safe control input $\dot{q}_\mathrm{s}$ for the single integrator system $\dot{q} = \dot{q}_\mathrm{s}$. Thus, this approach is a manifestation of control based on reduced-order models. While h is a CBF for the reduced model, h_V is a CBF for the full system (13) as a dynamic extension of h, similar to the energy-based extension in [15]. We remark that other reduced-order models of the form $\dot{q} = A(q)\mu_\mathrm{s}$ with control input $\mu_\mathrm{s} \in \mathbb{R}^k$ and transformation $A(q) \in \mathbb{R}^{n \times k}$ can also be used. This, for example, includes the unicycle model for wheeled robots with $q = (x, y, \psi) \in \mathbb{R}^3$ containing Cartesian positions and yaw angle and $\mu_\mathrm{s} = (v_\mathrm{s}, \omega_\mathrm{s}) \in \mathbb{R}^2$ containing forward velocity and yaw rate:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} \cos\psi & 0 \\ \sin\psi & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_\mathrm{s} \\ \omega_\mathrm{s} \end{bmatrix}. \tag{25}$$

The safe velocity μ_s is given by $\nabla h(q) A(q) \mu_\mathrm{s} \geq -\alpha h(q)$ based on (15), and the proof of Theorem 3 carries over with substitution $\dot{q}_\mathrm{s} = A(q)\mu_\mathrm{s}$. The tracking controller u, however, must provide the stability property (18) with respect to the error $\dot{e} = \dot{q} - A(q)\mu_\mathrm{s}$.

Remark 3. Theorem 3 requires initial conditions to satisfy $(q_0, \dot{e}_0) \in S_V \iff h(q_0) \geq V(q_0, \dot{e}_0)/\alpha_{\mathrm{e}}$. This is a stricter condition than $q_0 \in S \iff h(q_0) \geq 0$ that is usually required in safety-critical control (cf. Definition 3). The additional conservatism is reduced when the initial tracking error $\dot{e}_0$ is smaller (since $V(q_0, \dot{e}_0)$ is smaller) and when the tracking is faster, i.e., $\lambda - \alpha$ is larger (since α_{e} is larger).

C. Effect of Disturbances

Now consider that ideal exponential tracking of the safe velocity in not possible due to a bounded disturbance d in the system. Then, instead of safety, one can guarantee input-to-state safety, i.e., the invariance of the larger set $S_d \supseteq S$:

$$\begin{aligned} S_d &= \{q \in Q : h_d(q) \geq 0\}, \\ h_d(q) &= h(q) + \gamma(\|d\|_\infty), \end{aligned} \tag{26}$$

where γ is a class-$\mathcal{K}$ function to be specified. We also introduce the dynamic extension $S_{Vd} \supseteq S_V$ of set S_d:

$$\begin{aligned} S_{Vd} &= \{(q, \dot{e}) \in Q \times \mathbb{R}^n : h_{Vd}(q, \dot{e}) \geq 0\}, \\ h_{Vd}(q, \dot{e}) &= h_V(q, \dot{e}) + \gamma(\|d\|_\infty). \end{aligned} \tag{27}$$

We show that in order to guarantee input-to-state safety, one needs input-to-state stable tracking of the safe velocity. Instead of (18) we require the tracking controller to satisfy:

$$\dot{V}(q, \dot{e}, \dot{q}, \ddot{q}_{\mathrm{s}}, u) \leq -\lambda V(q, \dot{e}) + \iota(\|d\|_\infty), \tag{28}$$

for some class-$\mathcal{K}$ function ι. The connection between ISS and ISSf is summarized in the following Corollary of Theorem 3.

Corollary 1. *Consider system (13), sets S_d and S_{Vd} in (26) and (27), safe velocity satisfying (15), and velocity tracking controller satisfying (28). If $\lambda > \alpha$, input-to-state safety is achieved such that $(q_0, \dot{e}_0) \in S_{Vd} \Rightarrow q(t) \in S_d$, $\forall t \geq 0$, where α_{e} is given in Theorem 3 and $\gamma(\|d\|_\infty) = \iota(\|d\|_\infty)/\alpha$.*

The proof follows the same steps as those in the proof of Theorem 3, by replacing h and h_V with h_d and h_{Vd}. Corollary 1 concludes that input-to-state stable tracking of a safe velocity implies input-to-state safety for the full system.

D. Velocity Tracking Controllers

Finally, we consider examples of velocity tracking controllers and show that they satisfy stability requirement (18) or ISS requirement (28), respectively. In particular, we consider a model-free D controller as the simplest choice:

$$u = -K_{\mathrm{D}} \dot{e}, \tag{29}$$

where $K_{\mathrm{D}} \in \mathbb{R}^{m \times n}$ is selected so that $K = BK_{\mathrm{D}}$ is positive definite. While this controller is model-free, model-dependent terms – if they are well-known – can also be added to the control law. If $n = m$ and B is invertible (i.e., the system is fully actuated), one may use a D controller with gravity compensation:

$$u = B^{-1}(G(q) - K\dot{e}), \tag{30}$$

with a positive definite gain $K \in \mathbb{R}^{n \times n}$. Furthermore, one can also use a heavily model-dependent extension:

$$u = B^{-1}(D(q)\ddot{q}_{\mathrm{s}} + C(q, \dot{q})\dot{q}_{\mathrm{s}} + G(q) - K\dot{e}). \tag{31}$$

While this controller may achieve better tracking, it requires knowledge of inertia and Coriolis terms ($D(q)$ and $C(q, \dot{q})$), which may have complicated expressions and may be computationally expensive to compute in practice.

For the above controllers, the parameters that can be selected during control design are α and K_{D} or K, respectively. We characterize how fast these controllers are able to track the safe velocity by the constant $\lambda \in \mathbb{R}_{>0}$:

$$\lambda = \frac{\sigma_{\min}(K)}{\sup\limits_{q \in Q} \sigma_{\max}(D(q))}, \tag{32}$$

where $\sigma_{\min}$ and $\sigma_{\max}$ denote the smallest and largest eigenvalue. The eigenvalues are positive real numbers due to the positive definiteness of $D(q)$ and K. Accordingly, λ represents the smallest gain divided by the largest inertia, hence characterizes how fast controllers may track. We associate the controllers with the Lyapunov function candidate:

$$V(q, \dot{e}) = \sqrt{\frac{1}{2}\dot{e}^\top D(q)\dot{e}}, \tag{33}$$

that has the bound (17) with $k_1 = \inf_{q \in Q} \sqrt{\sigma_{\min}(D(q))/2}$ and $k_2 = \sup_{q \in Q} \sqrt{\sigma_{\max}(D(q))/2}$. We also define the linear class-$\mathcal{K}$ function $\iota(\|d\|_\infty) = \|d\|_\infty/(2k_1)$.

Now we state that the above three controllers possess the required stability properties.

Proposition 1. *Consider system (13), Lyapunov function V defined by (33), and constant λ given by (32).*

(i) *Controller (29) satisfies the ISS condition (28) with respect to $d = -D(q)\ddot{q}_{\mathrm{s}} - C(q, \dot{q})\dot{q}_{\mathrm{s}} - G(q)$.*

(ii) *Controller (30) satisfies the ISS condition (28) with respect to $d = -D(q)\ddot{q}_{\mathrm{s}} - C(q, \dot{q})\dot{q}_{\mathrm{s}}$ when $\dot{q}_{\mathrm{s}} \not\equiv 0$ and the stability condition (18) when $\dot{q}_{\mathrm{s}} \equiv 0$.*

(iii) *Controller (31) satisfies the stability condition (18).*

Proof. The proof follows that in Section 8.2 of [16]. Here we prove case (i) only. The proof of case (ii) is the same when $\dot{q}_{\mathrm{s}} \not\equiv 0$, whereas the proofs of case (ii) when $\dot{q}_{\mathrm{s}} \equiv 0$ and case (iii) can be obtained by substituting $d \equiv 0$.

We differentiate V given by (33):

$$\dot{V}(q, \dot{e}, \dot{q}, \ddot{q}_{\mathrm{s}}, u) = \frac{1}{2V(q, \dot{e})}\left(\frac{1}{2}\dot{e}^\top \dot{D}(q, \dot{q})\dot{e} + \dot{e}^\top D(q)\ddot{e}\right), \tag{34}$$

and substitute the error dynamics corresponding to (13, 16):

$$D(q)\ddot{e} = -C(q, \dot{q})\dot{e} - D(q)\ddot{q}_{\mathrm{s}} - C(q, \dot{q})\dot{q}_{\mathrm{s}} - G(q) + Bu. \tag{35}$$

For controller (29) this leads to:

$$\dot{V}(q, \dot{e}, \dot{q}, \ddot{q}_{\mathrm{s}}, u) = \frac{-\dot{e}^\top K\dot{e} + \dot{e}^\top d}{2V(q, \dot{e})}, \tag{36}$$

where the term $\dot{e}^\top(\dot{D}(q, \dot{q}) - 2C(q, \dot{q}))\dot{e}$ dropped since $\dot{D}(q, \dot{q}) - 2C(q, \dot{q})$ is skew-symmetric.

Based (36), now we show (28) holds. The definition (32) of λ implies $\dot{e}^\top K\dot{e} - \lambda\dot{e}^\top D(q)\dot{e} \geq 0$, which, based on the definition (33) of V, leads to:

$$\frac{-\dot{e}^\top K\dot{e}}{2V(q, \dot{e})} \leq -\lambda V(q, \dot{e}), \tag{37}$$

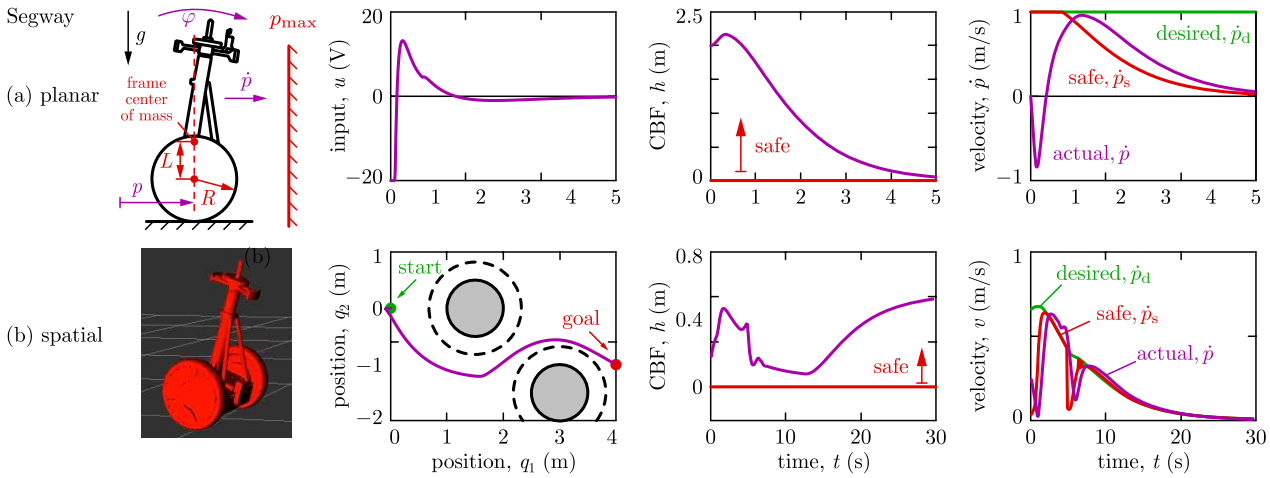

Fig. 4. High-fidelity simulation of a Ninebot E+ Segway platform. (a) Planar dynamical model (13, 39) with the model-free safety-critical controller (41, 42). (b) Spatial dynamical model with the model-free controller (44, 45). The controllers keeps the system safe (the CBF h is positive for all time).

$\forall q \in Q, \dot{e} \in \mathbb{R}^n$. Furthermore, the Cauchy-Schwartz inequality, the bound (17) on V and the definition of ι yield:

$$\frac{\dot{e}^\top d}{2V(q,\dot{e})} \leq \frac{\|\dot{e}\|\|d\|_\infty}{2k_1\|\dot{e}\|} = \iota(\|d\|_\infty). \tag{38}$$

Substituting inequalities (37, 38) into (36) yields (28). □

IV. APPLICATIONS TO WHEELED, FLYING AND LEGGED ROBOTS

Now we apply the proposed control method to robotic platforms, including high-fidelity simulations of a Segway and hardware experiments on a Drone and a Quadruped.

A. Numerical Simulation of Segway

We consider a Ninebot E+ Segway platform described in [26], [27]. Its high-fidelity dynamical model was built in [26]. First we consider planar, then spatial dynamics.

Example 2 (Segway in plane)**.** Consider the two-degrees of freedom planar Segway model in Fig. 4(a) with configuration $q = [p, \varphi]^\top \in \mathcal{Q} = \mathbb{R} \times [0, 2\pi]$ including the position p and pitch angle φ. The dynamics are in form (13), where:

$$D(q) = \begin{bmatrix} m_0 & mL\cos\varphi \\ mL\cos\varphi & J_0 \end{bmatrix}, \quad G(q) = \begin{bmatrix} 0 \\ -mgL\sin\varphi \end{bmatrix},$$
$$C(q,\dot{q}) = \begin{bmatrix} b_\mathrm{t}/R & -b_\mathrm{t} - mL\dot{\varphi}\sin\varphi \\ -b_\mathrm{t} & b_\mathrm{t}R \end{bmatrix}, \quad B = \begin{bmatrix} K_\mathrm{m}/R \\ -K_\mathrm{m} \end{bmatrix}, \tag{39}$$

with parameters given in Table I and $u \in U = [-20, 20]\,\mathrm{V}$.

Our goal is to realize a desired forward velocity $\dot{p}_\mathrm{d}$ until reaching a wall at position p_max, then stop automatically and safely in front of the wall. This is captured by the CBF:

$$h(q) = p_\mathrm{max} - p, \tag{40}$$

which, by condition (15), leads to the safe forward velocity:

$$\dot{p}_\mathrm{s} = \min\{\dot{p}_\mathrm{d}, \alpha(p_\mathrm{max} - p)\}, \tag{41}$$

similar to (22). This safe velocity is tracked by the controller:

$$u = K_{\dot{p}}(\dot{p} - \dot{p}_\mathrm{s}) + K_\varphi\varphi + K_{\dot{\varphi}}\dot{\varphi} \tag{42}$$

TABLE I
PARAMETERS OF THE SEGWAY MODEL

Description	Parameter	Value	Unit
gravitational acceleration	g	9.81	m/s^2
radius of wheels	R	0.195	m
mass of wheels	M	2×2.485	kg
inertia of wheels	J_C	2×0.0559	kgm^2
distance of wheel center to frame CoM	L	0.169	m
mass of frame	m	44.798	kg
inertia of frame	J_G	3.836	kgm^2
lumped mass $m_0 = m + M + J_C/R^2$	m_0	52.710	kg
lumped inertia $J_0 = mL^2 + J_G$	J_0	5.108	kgm^2
torque constant of motors	K_m	2×1.262	Nm/V
damping constant of motors	b_t	2×1.225	Ns

with gains $K_{\dot{p}} = 50\,\mathrm{Vs/m}$, $K_\varphi = 150\,\mathrm{V/rad}$ and $K_{\dot{\varphi}} = 40\,\mathrm{Vs/rad}$, which also stabilizes the pitch angle of the Segway to the upright position.

Fig. 4(a) shows numerical simulation results where the Segway executes the task starting at $p_0 = 0$, $\varphi_0 = -0.138\,\mathrm{rad}$ (where the frame of the Segway is vertical), $\dot{p}_0 = 0$, $\dot{\varphi}_0 = 0$, for $\dot{p}_\mathrm{d} = 1\,\mathrm{m/s}$, $p_\mathrm{max} = 2\,\mathrm{m}$ and $\alpha = 0.5\,\mathrm{s}^{-1}$. We highlight that the controller (41, 42) is model-free, it does not rely on the full dynamics (13, 39). The gains $K_{\dot{p}}$, K_φ and $K_{\dot{\varphi}}$, however, must be tuned so that the full dynamics achieves stable velocity tracking.

Example 3 (Segway in space)**.** Consider the spatial model of the Segway in Fig. 4(b) that has a 7-dimensional state space with 2 control inputs. The task is to navigate it from a start point to a goal (left panel) while avoiding obstacles of radius 0.5 m (solid black), similar to Example 1. The obstacle radius is buffered by the size of the Segway (dashed black) and the Segway's center must be kept outside this zone.

This task is accomplished by tracking a safe velocity obtained from the unicycle model (25); cf. Remark 2. We set the desired forward velocity and yaw rate $\mu_\mathrm{d} = (v_\mathrm{d}, \omega_\mathrm{d})$ based on the distance $d_\mathrm{g} = \|(x_\mathrm{g} - x, y_\mathrm{g} - y)\|$ to the goal as $v_\mathrm{d} = K_v d_\mathrm{g}$ and $\omega_\mathrm{d} = -K_\omega(\sin\psi - (y_\mathrm{g} - y)/d_\mathrm{g})$. To avoid obstacles, we use the following CBF which accounts for the

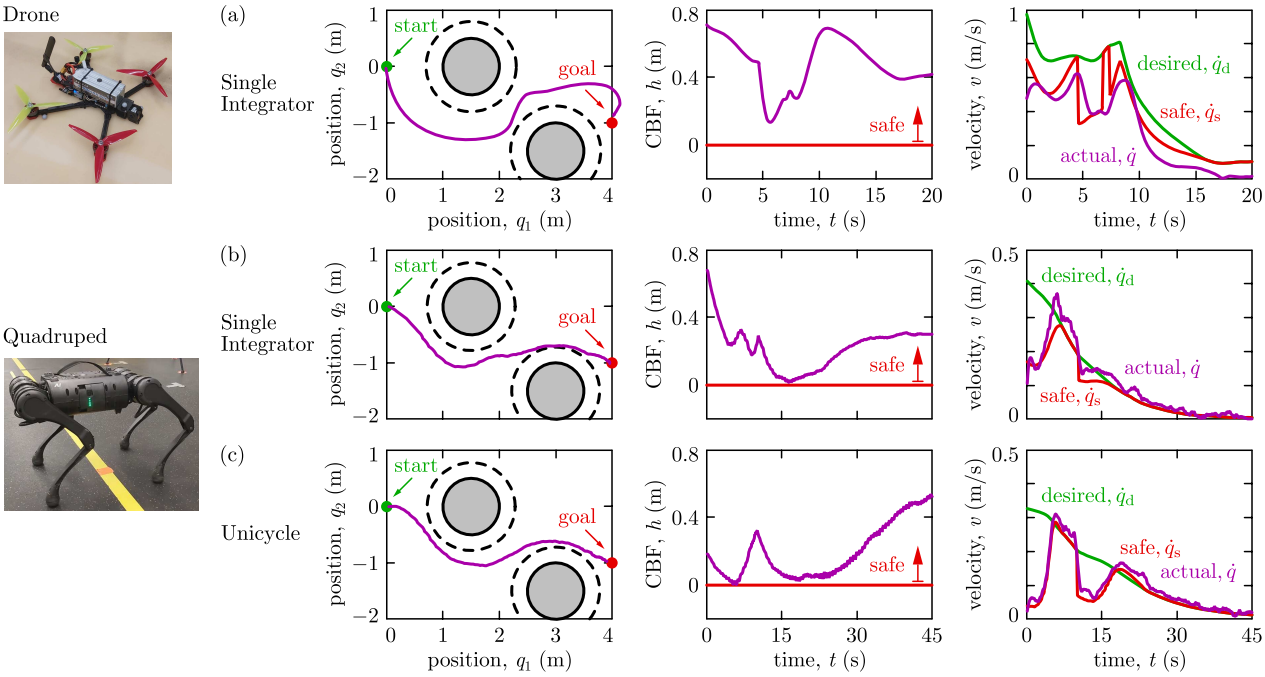

Fig. 5. Hardware experiments using the proposed model-free safety-critical control method. An obstacle avoidance task is accomplished by two fundamentally different robots: a custom-made racing Drone (top) and a Unitree A1 Quadruped (bottom). (a) The Drone is tracking a safe velocity determined based on single integrator model. (b) The Quadruped is tracking a safe velocity based on single integrator model via side-stepping and (c) based on unicycle model via turning. Both robots executed the task with guaranteed safety. A video of the experiments can be found at [25].

direction the robot is heading to:

$$h(q) = d - r - \delta\cos(\psi - \theta), \tag{43}$$

where $d = \|(x_\mathrm{o} - x, y_\mathrm{o} - y)\|$ is the distance from the obstacle, $\theta = \arctan((y_\mathrm{o} - y)/(x_\mathrm{o} - x))$ is the angle towards the obstacle, and $\delta \in \mathbb{R}_{>0}$ is a tunable parameter.

This CBF is incorporated into the quadratic program:

$$\begin{aligned} \underset{\mu_\mathrm{s}\in\mathbb{R}^2}{\operatorname{argmin}}\ & (\mu_\mathrm{s} - \mu_\mathrm{d})^\top \Gamma (\mu_\mathrm{s} - \mu_\mathrm{d}) \\ \text{s.t.}\ & \nabla h(q) A(q) \mu_\mathrm{s} \geq -\alpha h(q), \end{aligned} \tag{44}$$

cf. (21), where $\Gamma = \mathrm{diag}\{1, R^2\}$ is a weight between forward velocity and yaw rate with parameter $R \in \mathbb{R}_{>0}$. Similarly to (22), we can solve (44) analytically to obtain μ_s. The safe velocity $\mu_\mathrm{s} = (v_\mathrm{s}, \omega_\mathrm{s})$ is tracked by the controller:

$$u_{1,2} = K_{\dot{p}}(\dot{p} - v_\mathrm{s}) + K_\varphi \varphi + K_{\dot{\varphi}}\dot{\varphi} \pm K_{\dot{\psi}}(\dot{\psi} - \omega_\mathrm{s}) \tag{45}$$

used at the two wheels with the same gains as in Example 2 and a gain $K_{\dot{\psi}} = 10\,\mathrm{Vs/rad}$ on the yaw rate $\dot{\psi}$.

With this approach, the Segway is able to move to the goal safely, while its controller (44, 45) is model-free. Fig. 4(b) shows the safe motion for $K_v = 0.16\,\mathrm{s}^{-1}$, $K_\omega = 0.8\,\mathrm{s}^{-1}$, $\alpha = 0.2\,\mathrm{s}^{-1}$, $\delta = 0.5\,\mathrm{m}$ and $R = 0.25\,\mathrm{m}$.

B. Hardware Experiments on Drone and Quadruped

We executed the obstacle avoidance task of Example 3 on two fundamentally different hardware platforms: a Drone and a Quadruped; see Fig. 5. The obstacle locations were known to the robots, sensory information was used to determine the robot position only. We performed two classes of experiments: by synthesizing safe velocities based on the single integrator and unicycle models, respectively; cf. Remark 2. A video of the experiments can be found at [25].

First, we considered the single integrator model, and we tracked the associated safe velocity with the Drone and the Quadruped by platform-specific tracking controllers. We used CBF (20) and safe velocity (22). The desired velocity was $\dot{q}_\mathrm{d} = -K_\mathrm{P}(q - q_\mathrm{g})$ with saturation; cf. Example 1.

The Drone was a custom-built racing drone [28], shown in Fig. 5(a). It has 6 degrees of freedom and 4 actuators. The state of the Drone (position, orientation and corresponding velocities) were measured by IMU and an OptiTrack motion capture system. State estimation and control action computation ran at 400 Hz. The safe velocity was commanded to the drone wireless from a desktop computer, while velocity tracking was done using an on-board betaflight flight controller. The safe velocity was calculated with $K_\mathrm{P} = 0.7\,\mathrm{s}^{-1}$ and $\alpha = 0.2\,\mathrm{s}^{-1}$. Fig. 5(a) shows the Drone reaching the goal safely, as guaranteed by Theorem 3 since α was selected small enough for the available tracking performance.

The Quadruped was a Unitree A1 quadrupedal robot, shown in Fig. 5(b), which has 18 degrees of freedom and 12 actuators. Its position was measured based on odometry assuming the feet do not slip, while joint states were available via built-in encoders. An ID-QP walking controller was realized at 1 kHz loop rate on this robot to track a stable walking gait with prescribed forward and lateral velocities and yaw rate, designed using the concepts in [29]. Individual commands were tracked via a motion primitive framework described in [30]. In the single integrator experiments, the yaw rate was set to zero, while the velocity was prescribed

based on the safe velocity (22) with $K_\mathrm{P} = 0.1\,\mathrm{s}^{-1}$ and $\alpha = 0.2\,\mathrm{s}^{-1}$, which the quadruped tracked by forward- and side-stepping. The Quadruped executed the task safely similar to the Drone (see Fig. 5(b)), although it has fundamentally different dynamic behavior. This indicates the application-agnostic nature of our model-free approach.

Finally, we used the unicycle model (25) and CBF (43) to achieve safety on the Quadruped. The safe forward velocity and yaw rate in (44) were tracked by the same ID-QP walking controller. Fig. 5(c) shows the Quadruped traversing the obstacle course with $K_v = 0.08\,\mathrm{s}^{-1}$, $K_\omega = 0.4\,\mathrm{s}^{-1}$, $\alpha = 0.2\,\mathrm{s}^{-1}$, $\delta = 0.5\,\mathrm{m}$ and $R = 0.5\,\mathrm{m}$. While safety is maintained, the Quadruped performs the task with different behavior than in the previous experiment: it walks forward and turns instead of forward- and side-stepping. Still, safety is provably guaranteed — and in a model-free fashion.

V. CONCLUSIONS

We considered safety-critical control for robotic systems in a model-free fashion following [14]. Our control method relies on a synthesizing a safe velocity using control barrier functions and tracking this velocity. We stated and proved theoretical guarantees for the safety of our method. Namely, safety is achieved when the safe velocity is tracked faster than how fast the corresponding safe motion may approach the boundary of the safe set. Due to its model-free nature, our approach is application-agnostic. By simulation and hardware experiments we demonstrated that it works for various robots such as a Segway, a Drone and a Quadruped.

While our current method does not rely on the full dynamical model of the robot to achieve safety, it relies on kinematic models such as the single integrator or unicycle models. Our future work includes further exploration of safety-critical control based on reduced-order models which may be beyond simple kinematic ones. We also intend to study the robustness of controlling full dynamical models based on reduced-order dynamics.

REFERENCES

[1] S. Teng, Y. Gong, J. W. Grizzle, and M. Ghaffari, "Toward safety-aware informative motion planning for legged robots," *arXiv preprint*, no. arXiv:2103.14252, 2021.

[2] J. Tordesillas, B. T. Lopez, and J. P. How, "Faster: Fast and safe trajectory planner for flights in unknown environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2019, pp. 1934–1940.

[3] 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.

[4] A. Ames, J. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs with application to adaptive cruise control," in *IEEE Conference on Decision and Control*, 2014, pp. 6271–6278.

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

[6] E. Squires, R. Konda, S. Coogan, and M. Egerstedt, "Model free barrier functions via implicit evading maneuvers," *arXiv preprint*, no. arXiv:2107.12871, 2021.

[7] M. Jankovic, "Robust control barrier functions for constrained stabilization of nonlinear systems," *Automatica*, vol. 96, pp. 359–367, 2018.

[8] P. Seiler, M. Jankovic, and E. Hellstrom, "Control barrier functions with unmodeled dynamics using integral quadratic constraints," *arXiv preprint*, no. arXiv:2108.10491, 2021.

[9] S. Singh, M. Chen, S. L. Herbert, C. J. Tomlin, and M. Pavone, "Robust tracking with model mismatch for fast and safe planning: An SOS optimization approach," in *International Workshop on the Algorithmic Foundations of Robotics*, M. Morales, L. Tapia, G. Sánchez-Ante, and S. Hutchinson, Eds., 2020, pp. 545–564.

[10] D. Pickem, P. Glotfelter, L. Wang, M. Mote, A. Ames, E. Feron, and M. Egerstedt, "The Robotarium: A remotely accessible swarm robotics research testbed," in *IEEE International Conference on Robotics and Automation*, 2017, pp. 1699–1706.

[11] S. Zhao and Z. Sun, "Defend the practicality of single-integrator models in multi-robot coordination control," in *IEEE International Conference on Control Automation*, 2017, pp. 666–671.

[12] A. De Luca, G. Oriolo, and M. Vendittelli, "Control of wheeled mobile robots: An experimental overview," in *Lecture Notes in Control and Information Sciences*, S. Nicosia, S. B., A. Bicchi, and P. Valigi, Eds. Berlin: Springer, 2001, vol. 270, pp. 181–226.

[13] D. Koung, I. Fantoni, O. Kermorgant, and L. Belouaer, "Consensus-based formation control and obstacle avoidance for nonholonomic multi-robot system," in *International Conference on Control, Automation, Robotics and Vision*, 2020, pp. 92–97.

[14] A. Singletary, K. Klingebiel, J. R. Bourne, N. A. Browning, P. Tokumaru, and A. Ames, "Comparative analysis of control barrier functions and artificial potential fields for obstacle avoidance," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2021.

[15] A. Singletary, S. Kolathaya, and A. D. Ames, "Safety-critical kinematic control of robotic systems," *IEEE Control Systems Letters*, vol. 6, pp. 139–144, 2022.

[16] M. W. Spong, S. Hutchinson, and M. Vidyasagar, *Robot Modeling and Control*. New York: John Wiley and Sons, 2005.

[17] H. Khalil, *Nonlinear Systems*, 3rd ed. Upper Saddle River: Prentice Hall, 2002.

[18] A. D. Ames, K. Galloway, K. Sreenath, and J. W. Grizzle, "Rapidly exponentially stabilizing control Lyapunov functions and hybrid zero dynamics," *Transactions on Automatic Control*, vol. 59, no. 4, pp. 876–891, 2014.

[19] 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.

[20] E. D. Sontag and Y. Wang, "On characterizations of input-to-state stability with respect to compact sets," in *Nonlinear Control Systems Design*. Elsevier, 1995, pp. 203–208.

[21] E. D. Sontag, "Input to state stability: Basic concepts and results," in *Nonlinear and Optimal Control Theory*. Springer, 2008, pp. 163–220.

[22] S. Kolathaya and A. D. Ames, "Input-to-state safety with control barrier functions," *IEEE Control Systems Letters*, vol. 3, no. 1, pp. 108–113, 2019.

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

[24] P. Glotfelter, J. Cortes, and M. Egerstedt, "A nonsmooth approach to controller synthesis for Boolean specifications," *IEEE Transactions on Automatic Control*, pp. 1–1, 2020.

[25] Supplementary video: https://youtu.be/vNcc5vgswx0.

[26] T. Gurriet, M. Mote, A. Singletary, P. Nilsson, E. Feron, and A. D. Ames, "A scalable safety critical control framework for nonlinear systems," *IEEE Access*, vol. 8, pp. 187 249–187 275, 2020.

[27] R. K. Cosner, A. W. Singletary, A. J. Taylor, T. G. Molnar, K. L. Bouman, and A. D. Ames, "Measurement-robust control barrier functions: Certainty in safety with uncertainty in state," *arXiv preprint*, no. arXiv:2104.14030, 2021.

[28] A. Singletary, A. Swann, and A. D. Ames, "Onboard safety guarantees for racing drones: A high-speed geofencing solution," *IEEE Robotics and Automation Letters*, 2021, submitted.

[29] J. Buchli, M. Kalakrishnan, M. Mistry, P. Pastor, and S. Schaal, "Compliant quadruped locomotion over rough terrain," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009, pp. 814–820.

[30] W. Ubellacker, N. Csomay-Shanklin, T. G. Molnar, and A. D. Ames, "Verifying safe transitions between dynamic motion primitives on legged robots," *arXiv preprint*, no. arXiv:2106.10310, 2021.