

Realtime Safety Control for Bipedal Robots to Avoid Multiple Obstacles via CLF-CBF Constraints

Jinze Liu*, Minzhe Li*, Jessy W. Grizzle, and Jiunn-Kai Huang¹

Abstract

This paper presents a reactive planning system that allows a Cassie-series bipedal robot to avoid multiple non-overlapping obstacles via a single, continuously differentiable control barrier function (CBF). The overall system detects an individual obstacle via a height map derived from a LiDAR point cloud and computes an elliptical outer approximation, which is then turned into a CBF. The QP-CLF-CBF formalism developed by Ames *et al.* is applied to ensure that safe trajectories are generated. Liveness is ensured by an analysis of induced equilibrium points that are distinct from the goal state. Safe planning in environments with multiple obstacles is demonstrated both in simulation and experimentally on the Cassie biped.

Keywords: Motion Planning, Autonomous Navigation, Obstacle Avoidance, Control Barrier Function, Control Lyapunov Function

1. Introduction and Contributions

Bipedal robots are typically conceived to achieve agile-legged locomotion over irregular terrains, and maneuver in cluttered environments[1, 2, 3]. To explore safely in such environments, it is critical for robots to generate quick, yet smooth responses to any changes in the obstacles, map, and environment. In this paper, we propose a means to design and compose control barrier functions (CBFs) for multiple non-overlapping obstacles and evaluate the system on a 20-degree-of-freedom (DoF) bipedal robot.

In an autonomous system, the task of avoiding obstacles is usually handled by a planning algorithm because it has access to the map of an entire environment. Given the map, the planning algorithm is then able to design a collision-free path from the robot's current position to a goal. If the map is updated due to a change in the environment, the planner then needs to update the planned path, so-called replanning, to accommodate the new environment. Such maps are typically large and contain rich information such as semantics, terrain characteristics, and uncertainty, and thus are slow to update. This raises a concern when obstacles either move into the planned path but the map has not been updated or a robot's new pose allows the detection of previously unseen obstacles. The slow update rate of the map leads to either collision or abrupt maneuvers to avoid collisions. The non-smooth aspects arising from the map updates or changes in the perceived environment can be detrimental to the stability of the overall system.

Research on obstacle avoidance has been studied for several decades as pioneered in classic probabilistic roadmap approaches (PRM) [4] and cell decomposition [5, Chapter 6].

*equal contribution.

¹Jinze Liu, Minzhe Li, Jessy W. Grizzle, and Jiunn-Kai Huang are with the Robotics Department, University of Michigan, Ann Arbor, MI 48109, USA. {jzliu, minzlee, grizzle, bjhuang}@umich.edu.

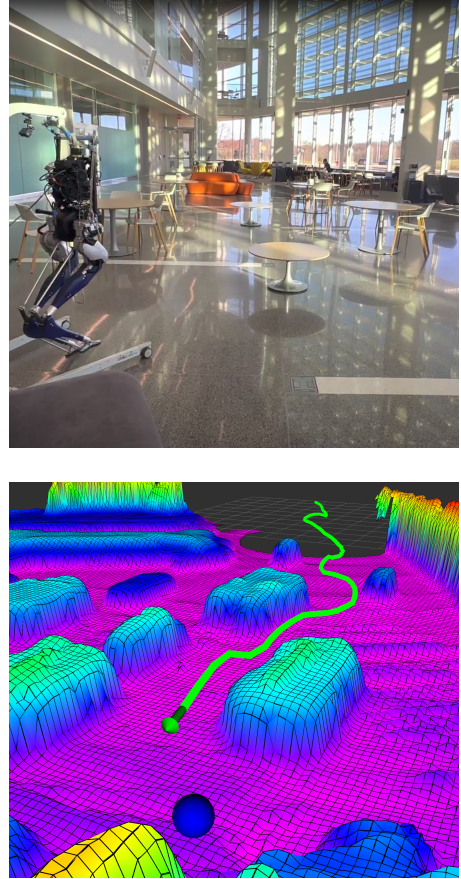


Figure 1: In the top figure, Cassie Blue autonomously avoids multiple obstacles via the developed CLF-CBF-QP obstacle avoidance system, comprised of an intermediate goal selector, obstacle detection, and a CLF-CBF quadratic programming solver. The bottom figure is the elevation map built in real time. The blue and cyan blobs are obstacles that Cassie detects and avoids in real time. A gantry is used in the experiments because the lab-built perception system that has been added to the robot is unprotected in case of falls.

However, the omission of robot dynamics and the extra computation for map discretization make these methods hard to use in real-time applications. Artificial potential fields [6, 7, 8, 9, 10, 11, 12, 13, 14, 15] stand out for their simplicity, extendability, and efficiency, leading to their wide adoption for real-time obstacle avoidance planning problems. A drawback of potential field methods is that they require the entire map of an environment to be available when designing a potential function that will render attractive one or more goal points in the map. Moreover, unwanted local minima and oscillations in the potential field have limited their deployment in the field. A control barrier function (CBF) [16], on the other hand, enables real-time controller synthesis with provable safety for mobile robots operating in a continuous (non-discretized) space and can work with a partial (or incomplete) map.

Control Lyapunov functions (CLFs) are positive definite functions such that at any given time instance, there exists a control input that renders the derivative of the function along the system dynamics negative definite. A CLF is typically designed to vanish at a desired goal state or pose.

The main theme of [16] is that a real-time quadratic program (QP) can be used to combine a CLF and a CBF in such a way that closed-loop trajectories induced by the CLF are minimally modified to provide provable safety, that is, non-collision with obstacles. This design philosophy has been explored in [16, 17, 18].

One means of avoiding obstacles is to come to a complete stop, though it is at the cost of not reaching the goal state. The papers [19, 20, 21, 22] showed that such behavior can be an unintended outcome of the CLF-CBF-QP design approach of [16]. Specifically, the inequality constraints (of the QP) associated with the derivatives of the control Lyapunov and control barrier functions can induce equilibria in the closed-loop system that are distinct from the equilibrium of the CLF. Reference [19] characterizes these equilibria via the KKT conditions associated with the QP, while reference [20] emphasizes that if an induced equilibrium is unstable, then “natural noise” in the environment will avoid the robot being deadlocked at an unstable equilibrium.

Inspired by the above-cited works on CLF-CBF-QPs for planning and control, we incorporate high-bandwidth obstacle avoidance into the CLF-RRT* reactive planner of [1]. The CLF in [1] takes into account features specific to bipeds, such as the limited lateral leg motion that renders lateral walking more laborious than sagittal plane walking. This paper seeks to utilize the CLF designed specifically for bipedal robots in tandem with a CBF to avoid multiple, non-overlapping obstacles in a smooth fashion, while ensuring progress to a goal state.

The main contributions of the new proposed CLF-CBF system are the following:

1. We propose a novel CLF-CBF-QP obstacle avoidance system specifically adapted for bipedal robots locomoting in the presence of multiple non-overlapping obstacles. The full system provides for real-time obstacle detection, CBF design, and safe control input generation through a QP.
2. We mathematically prove the validity of the proposed CBF

for both single and multiple obstacles. We also analytically analyze the existence of spurious equilibrium points induced by the CLF-CBF constraints on the QP.

3. We propose a simple means to smooth and interpolate the discontinuous reference control variables, which are introduced when switching from one target to another.
4. We provide simulations that support the mathematical analysis for obstacle avoidance while reaching a goal.
5. The overall reactive planning system is demonstrated experimentally on a 20-degree-of-freedom Cassie-series bipedal robot.
6. We open-source the implementations of the entire CLF-CBF system in C++ with Robot Operating System (ROS) [23] and associated videos of the experiments; see https://github.com/UMich-BipedLab/multi_object_avoidance_via_clf_cbf.

The rest of the paper is organized as follows. Section 2 overviews related work. The design and validation of the proposed CBF is presented in Sec. 3. We analyze equilibrium points of the proposed CBF in Appendix A. Section 4 proposes a novel and simple method to combine CBFs for non-overlapping obstacles. Section 5 introduces a method to smooth the reference control variables while switching target positions. Simulation and experimental results are given in Sec. 6 and Sec. 7, respectively. Section 8 concludes the paper and provides potential future work.

2. Related Work on Control with Safety

A continuously differentiable, proper, positive definite function $V(x)$ that vanishes at a single point is called a candidate Lyapunov function [24]. If the derivative of $V(x)$ along the trajectories of a control system can be rendered negative definite by proper choice of the control input, it is called a *control Lyapunov Function*, or CLF for short [25, 26, 27]. CLFs are widely used in the design of controllers to asymptotically drive a system to a goal state. Safety involves steering a control system to a goal state while avoiding self-collisions, obstacles, or other undesirable states, collectively referred to as *unsafe states*. The set complement of the unsafe states is the set of *safe states*.

2.1. Artificial Potential Fields and Navigation Functions

The first systematic method for real-time control and obstacle avoidance was introduced by Khatib in [28]. Called the method of *Artificial Potential Functions*, it revolutionized feedback control for manipulators in that hard constraints could be enforced in both the robot’s task space and joint space in real time. Prior to this seminal work, obstacle avoidance, or more generally the generation of safe paths, was relegated to a path planner operating at a much slower time scale. A survey of the method of potential functions can be found in [29].

Potential functions seek to construct “repulsive fields” around obstacles that are active throughout the entire state space

of the robot’s dynamical system, without destroying the presence of an attractive field steering the system to a goal state. It has been recognized that superimposed attracting and repelling fields can create undesired spurious equilibria, which prevent a robot from reaching its goal state [30]. In addition, potential fields have been observed to introduce trajectory oscillations as a robot passes near obstacles. Heuristic modifications have been proposed to avoid local minima [11, 12, 13], while potential fields have been combined with other gradient-based functions to reduce oscillations [14, 15].

The method of *Navigation Functions* by Koditschek and Rimon [31] sought to design a single function whose gradient produces trajectories avoiding multiple obstacles while asymptotically converging to a single goal state from *almost all initial conditions* [32, 33, 34, 35]; specifically, all equilibria except the goal state should be unstable. Because the design of a navigation function takes into account the global topology of the method of navigation functions is not appropriate for problems requiring the online identification and avoidance of obstacles; in addition, there are topological restrictions to the existence of navigation functions.

2.2. Control Barrier Functions and Control Lyapunov Functions

Barrier Functions provide Lyapunov-like conditions for proving a given set of safe states is forward invariant, meaning that trajectories starting in the safe set remain in the safe set. The natural extension of a barrier function to a system with control inputs is a *Control Barrier Function* or CBF for short, first proposed by [36]. CBFs parallel the extension of Lyapunov functions to CLFs, in that the key point is to impose inequality constraints on the derivative of a candidate CBF (resp., CLF) to establish entire classes of controllers that render a given set forward invariant (resp., asymptotically stable).

Importantly, barrier functions and CBFs focus solely on safety and do not seek to simultaneously steer a system to any particular point in the safe set. This allows CBFs to be combined with other “goal-oriented” control methods as a (maximally permissive) supervisor that only modifies a trajectory when it is in conflict with the safety criteria established by the CBF. The papers [37, 38] introduced the notion of using a real-time quadratic program (QP) to combine a CBF with a CLF to achieve convergence to a goal state while avoiding unsafe states. The overall method goes by the acronym CLF-CBF-QP.

For control systems that are affine in the control variable, CLF-CBF-QPs have proven to be enormously popular in and out of robotics applications [16, 17, 18, 39, 40, 41, 42, 43]. To highlight just a few example, reference [17] uses a CLF-CBF-QP to achieve stable walking for bipedal robots, while trajectory planning under spatiotemporal and control input constraints is presented in [40, 18, 39]. Applications to obstacle avoidance are addressed in [43, 41, 42].

The recent paper [44] shows that CBFs are a strict generalization of artificial potential functions and that in a practical example, a CLF-CBF-QP has reduced issues with oscillations as a robot passes near obstacles and improved *liveness*, mean-

ing the ability to reach the goal state. Hence, we use the method of CLF-CBF-QPs in this paper.

2.3. Combining Multiple CBFs

Usually, a control barrier function is designed for a single obstacle. When there are multiple obstacles in the control system, the barrier functions for each obstacle must be combined in some manner to provide safety guarantees. Reference [45] shows that if the intersection of the set of “allowable controls” of individual CBFs is non-empty, then the CLF-CBF-QP method can be extended to several obstacles; the reference does not show how to check this condition online (in real time). Multiple CBF functions have also been combined to obtain a single CBF so that existing methods can be applied. Reference [46] combines several CBFs into an overall CBF using max-min operations. The resulting CBF is non-differentiable and hence this technique is not used here. Reference [47] combines multiple CBFs for disjoint unsafe sets with a single CLF to produce a new CLF that simultaneously provides asymptotic stability and obstacle avoidance. This work is therefore related to the method navigation functions reviewed above and suffers from the same drawbacks; however, a key technique used in this reference to combine the CBFs before merging them with a CLF will be exploited in the current paper, namely a continuously differentiable saturation function.

2.4. CLF-CBF-QPs and Unwanted Equilibrium Points

The presence of multiple stable equilibrium points introduces “deadlock” in a control system. Reference [19] shows that the use of real-time QPs to combine safety and goal-reaching in navigation problems can lead to unwanted equilibrium points. With this awareness, the authors of [21] modify the cost function in the quadratic program to remove the unwanted equilibria. The modification induces a rotational motion in the closed-loop system that steers it around the obstacle, something a bipedal robot can do naturally. Hence, here we only exploit their analysis method for finding the unwanted equilibria and show that our method introduces at most one undesired equilibrium point when obstacles are disjoint. Moreover, we do not need to remove the unwanted equilibrium using the methods in [48, 22] by transforming the system’s state space into a convex manifold, or by increasing the complexity of the system’s state space.

2.5. Summary

The presence of multiple obstacles is common in practice. While existing works can treat disjoint obstacles, they are not appropriate for use where obstacles are identified in real-time via an onboard perception system. In this work, for a biped-appropriate planning model, we propose a simple means to combine CBFs for disjoint obstacles so that the complexity of the real-time CLF-CBF-QP remains constant and induced equilibrium points are easy to characterize and avoid.

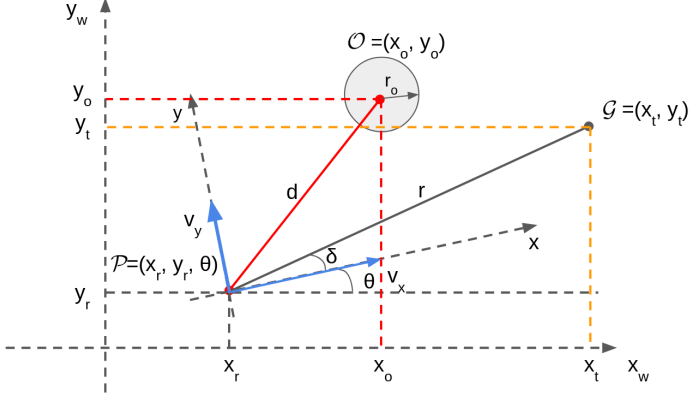


Figure 2: Illustration of the robot state representation. x_w and y_w are the axes of the world frame. The robot's pose and the target position are $\mathcal{P} = (x_r, y_r, \theta)$ and $\mathcal{G} = (x_t, y_t)$ in the world frame, respectively. x and y are the axes of the robot frame where the positive x direction is the robot heading direction. r is the distance between the robot and the target position and δ is the angle between the robot heading and the target direction. (x_o, y_o) is the center of obstacle in the world frame and r_o is the radius of the obstacle. d is the distance between the obstacle and the robot.

3. Construction of Control Lyapunov Function and Control Barrier Function

This section introduces the CLF proposed in [1] and analyzes its trajectories when combined with a quadratic CBF through a real-time QP. The goal is to ensure the closed-loop system is able to reach a goal state while smoothly avoiding a single obstacle. This section lays the foundation for considering multiple obstacles in the next section.

3.1. State Representation

Denote $\mathcal{P} = (x_r, y_r, \theta)$ the robot pose, $\mathcal{G} = (x_t, y_t)$ the goal position in the world frame. We simplify an obstacle $\mathcal{O}$ as a circle (and hence convex) described as its center (x_o, y_o) and its radius r_o . We define the robot state as

$$x = \begin{bmatrix} r \\ \delta \\ \theta \end{bmatrix}, \quad (1)$$

where $r = \sqrt{(x_t - x_r)^2 + (y_t - y_r)^2}$, θ is the heading angle of the robot, and δ is the angle between θ and the line of sight from the robot to the goal, as shown in Fig. 2.

The dynamics of the control system is defined as

$$\begin{aligned} \dot{x} &= f(x) + g(x)u \\ &= \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} -\cos(\delta) & -\sin(\delta) & 0 \\ \frac{\sin(\delta)}{r} & -\frac{\cos(\delta)}{r} & 1 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ \omega \end{bmatrix}, \end{aligned} \quad (2)$$

where we view $u = [v_x, v_y, \omega]^T$ as the control variables in the robot frame, as shown in Fig. 2.

3.2. Design of CLF and CBF for Bipedal Robots

The control Lyapunov function leveraged in the reactive planner proposed in [1] takes into account features specific to

bipeds, such as the limited lateral leg motion that renders lateral walking more laborious than sagittal plane walking. Therefore, we also define the CLF as

$$V(x) = \frac{r^2 + \gamma^2 \sin^2(\beta\delta)}{2}, \quad (3)$$

where γ is the weight on the orientation, and β controls the size of the field of view (FoV). Given $\mathcal{P}$ and $\mathcal{G}$, we have a closed-form solution for control u in (2),

$$\begin{aligned} \omega^{\text{ref}} &= \frac{r \cos(\delta) [rv_\delta \cos(\delta) - v_r \sin(\delta)]}{\alpha + r^2 \cos^2(\delta)} \\ v_y^{\text{ref}} &= \frac{\alpha(v_r \sin(\delta) - rv_\delta \cos(\delta))}{r^2 \cos^2(\delta) + \alpha} \\ v_x^{\text{ref}} &= \frac{v_r \cos(\delta)r^2 + \alpha v_\delta \sin(\delta)r + \alpha v_r \cos(\delta)}{r^2 \cos^2(\delta) + \alpha}; \end{aligned} \quad (4)$$

where v_r and v_δ are defined as:

$$\begin{aligned} v_r &= k_{r1} \frac{r}{k_{r2} + r} \\ v_\delta &= -\frac{2}{\beta} k_{\delta1} \frac{r}{k_{\delta2} + r} \sin(2\beta\delta). \end{aligned} \quad (5)$$

In (4) and (5), $\alpha, \beta, k_{r1}, k_{r2}, k_{\delta1}, k_{\delta2}$ are positive constants. See [1] for more details.

Next, we introduce a candidate CBF as

$$B(x) = \begin{bmatrix} x_r - x_o \\ y_r - y_o \end{bmatrix}^T Q \begin{bmatrix} x_r - x_o \\ y_r - y_o \end{bmatrix} - r_o^2, \quad (6)$$

where (x_o, y_o) gives the center of the obstacle, r_o specifies the “radius” of the obstacle, and Q is positive definite. We next verify that (6) is a valid CBF.

3.3. Proof of CBF Validity

Following [49], we define the sets

$$\begin{aligned} \mathcal{D} &:= \{x \in \mathbb{R}^3 \mid B(x) \neq -r_o^2, \text{ and } r \neq 0\} \\ \mathcal{C} &:= \{x \in \mathcal{D} \mid B(x) \geq 0\} \end{aligned} \quad (7)$$

associated with the candidate CBF (6) and note that $\text{Int}(\mathcal{C}) \neq \emptyset$ and $\overline{\text{Int}(\mathcal{C})} = \mathcal{C}$. From [49], for (6) to be a valid CBF function of (2), there must exist some $\eta > 0$, such that,

$$\forall x \in \mathcal{D}, \exists u \in \mathbb{R}^3, \dot{B}(x, u) + \eta B(x) \geq 0, \quad (8)$$

where $\dot{B}(x, u) := L_f B(x) + L_g B(x)u$ is the time derivative of $B(x)$ along the dynamics of (2), $\eta > 0$ sets the repulsive effort of the CBF, and

$$L_f B(x) := \frac{\partial B(x)}{\partial x} f(x) \quad (9)$$

$$L_g B(x) := \frac{\partial B(x)}{\partial x} g(x). \quad (10)$$

Because the drift term $f(x)$ in (2) is identically zero, the zero control $u \equiv 0$ satisfies (8) for $x \in \mathcal{C}$. Hence, we need to show

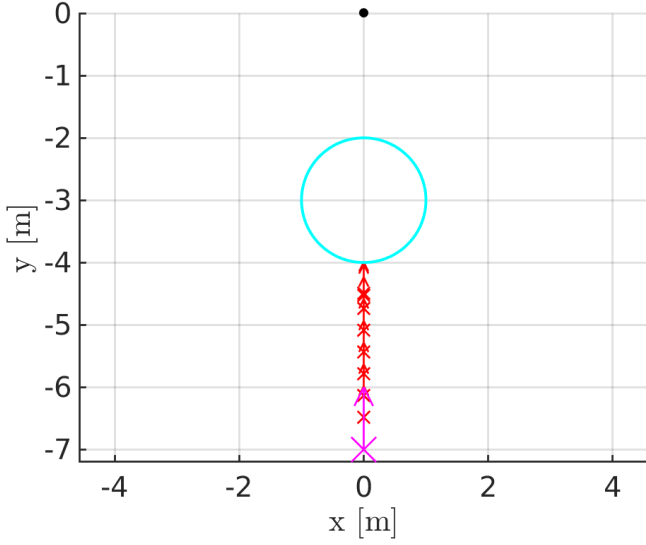


Figure 3: Illustration of a case when the robot directly faces the obstacle and the target creates an equilibrium in the continuous-time system. In a simulation with discrete-time control updates, the robot walks back and forth at the obstacle boundary.

that (8) can be met for $x \in \sim C$, the set complement of C . Direct application of the chain rule gives that

$$L_g B(x) = a(x)b(x)g(x),$$

where

$$\begin{aligned} a(x) &:= 2 \begin{bmatrix} x_t - r \cos(\delta + \theta) - x_o, & y_t - r \sin(\delta + \theta) - y_o \end{bmatrix} Q \\ &= 2 \begin{bmatrix} x_r - x_o, & y_r - y_o \end{bmatrix} Q \\ b(x) &:= \begin{bmatrix} -\cos(\delta + \theta) & r \sin(\delta + \theta) & r \sin(\delta + \theta) \\ -\sin(\delta + \theta) & -r \cos(\delta + \theta) & -r \cos(\delta + \theta) \end{bmatrix} \\ g(x) &= \begin{bmatrix} -\cos(\delta) & -\sin(\delta) & 0 \\ \frac{\sin(\delta)}{r} & -\frac{\cos(\delta)}{r} & 1 \\ 0 & 0 & -1 \end{bmatrix}. \end{aligned} \quad (11)$$

Moreover, $a(x)$ only vanishes at the center of an obstacle, the rows of $b(x)$ are linearly independent for all $r > 0$, and $\det(g(x)) = -\frac{1}{r} \neq 0$ for all $0 < r < \infty$. It follows that for all $x \in \mathcal{D}$, $L_g B(x) \neq 0$ and hence (8) is satisfied, proving that (6) is a valid CBF.

3.4. Quadratic Program of the Proposed CLF-CBF System

A quadratic program (QP) is set up to optimize the control u with the slack variable s while enforcing both the CLF and CBF constraints. Let $\mathfrak{L}(x, u, s)$ be the CLF constraints

$$\mathfrak{L}(x, u, s) := L_f V(x) + L_g V(x)u + \mu V(x) - s \leq 0, \quad (12)$$

where $L_p q(x) := \nabla q(x) \cdot p(x)$ is the Lie derivative, μ serves as a decay rate of the upper bound of $V(x)$. Next, we denote $\mathfrak{B}(x, u)$ the CBF constraints

$$\mathfrak{B}(x, u) := -L_f B(x) - L_g B(x)u - \eta B(x) \leq 0, \quad (13)$$

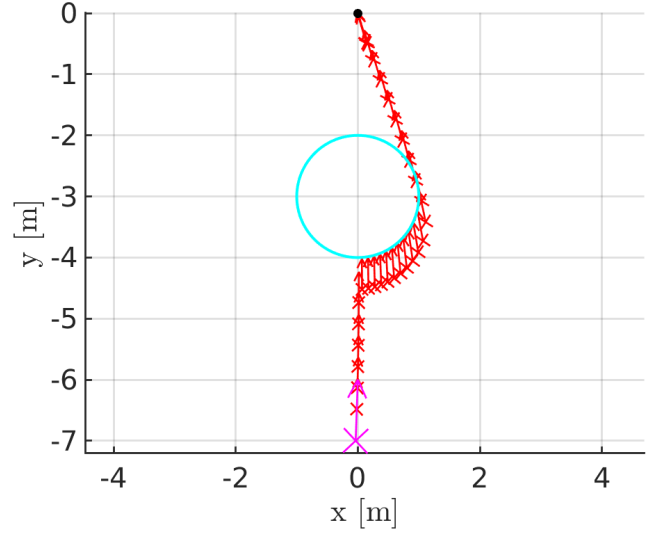


Figure 4: Illustration of breaking the equilibrium by using u_2^{ref} : $[v_x^{\text{ref}} \ v_y^{\text{ref}} \ \omega^{\text{ref}} + \epsilon]^T$ when $\delta = 0$. The robot successfully reaches the target position without colliding with the obstacle.

where η serves as a decay rate of the lower bound of $B(x)$.

Finally, the QP for the control values is formulated as

$$u^*, s^* = \arg \min_{\substack{\mathfrak{L}(x, u, s) \leq 0 \\ \mathfrak{B}(x, u) \leq 0}} \mathfrak{J}(u, s), \quad (14)$$

where the cost function $\mathfrak{J}(u, s)$ is defined as

$$\mathfrak{J}(u, s) := \frac{1}{2}(u - u^{\text{ref}})^T H (u - u^{\text{ref}}) + \frac{1}{2}ps^2, \quad (15)$$

the positive definite, diagonal matrix $H := \text{diag}([h_1, h_2, h_3])$ weights the control variables, $u^{\text{ref}} := [v_x^{\text{ref}} \ v_y^{\text{ref}} \ \omega^{\text{ref}}]^T$ is the control vector from the CLF (3) without obstacles, and $p \geq 0$ is the weight of the slack variable, s .

In the proposed CLF-CBF-QP system, u^{ref} is the closed-form solution obtained from the CLF without obstacles, and H assigns weights for different control variables. The proposed CLF-CBF-QP cost function captures inherent features of a Cassie-series robot, such as the low-cost of longitudinal movement and high-cost of lateral movement, while guaranteeing safety. We next look at liveness, that is, the ability of the system to reach the desired goal.

3.5. Analysis for Unwanted Equilibria

Paper [19] points out very clearly that the CLF-CBF-QP formulation of Sec. 3.4 can introduce unwanted equilibria that may prevent the robot from reaching a goal state. The paper [20] also considered this problem and noted that if the equilibria are unstable, then liveness is preserved for almost all initial conditions. In Appendix A, we follow the KKT-analysis of the CLF-CBF-QP presented in [19] and show that only one equilibrium point is created by the QP. Moreover, the equilibrium occurs at an obstacle boundary for $\delta = 0, d_y = 0, d_x > 0$, in

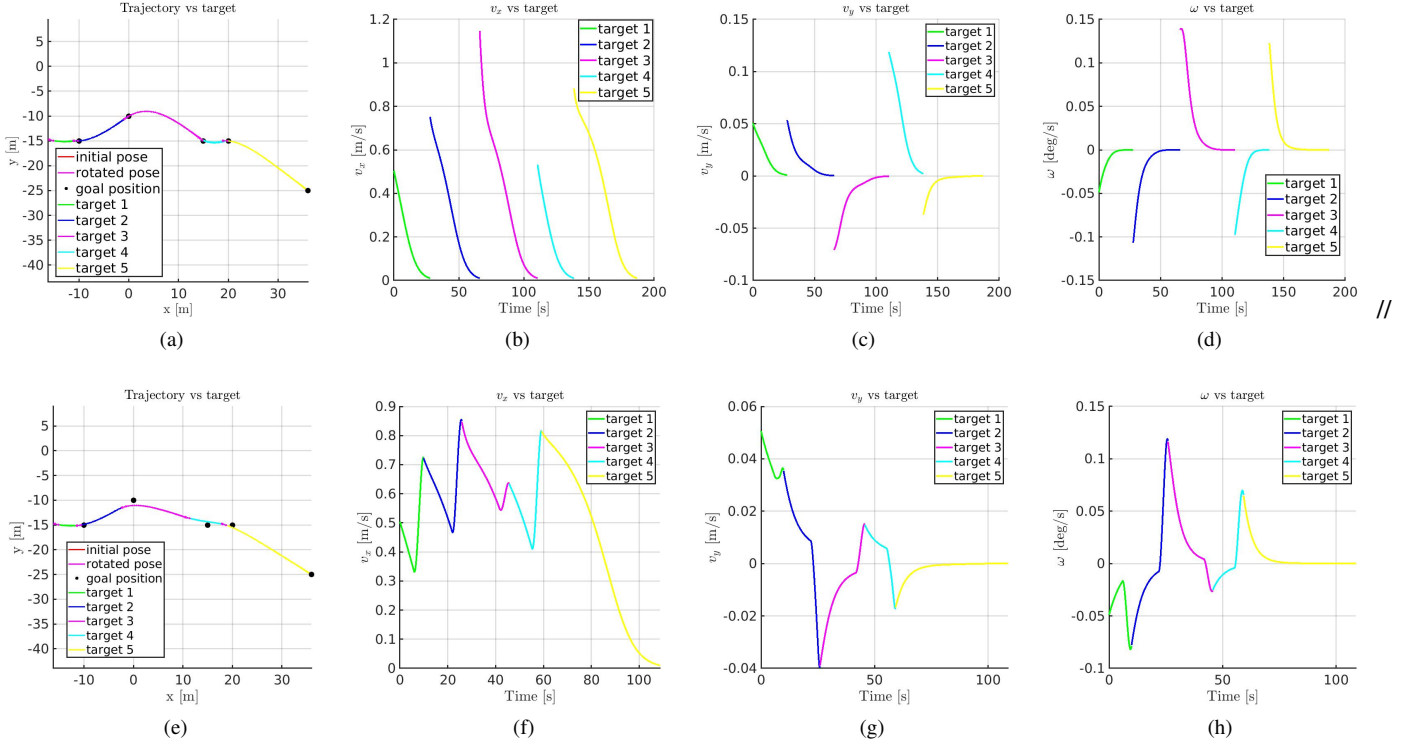


Figure 5: The top and bottom rows are the resulting trajectories and control variables generated by (4) and (30), respectively. The robot starts at $(-15, -15, -15^\circ)$, and the list of targets to reach is $(-10, -15)$, $(0, -10)$, $(15, -15)$, $(25, -15)$, and the final destination is $(40, -30)$. Different colors represent the trajectory with different targets and also correspond to control variables for the trajectory. The control variables generated by (4) decrease to zero each goal is approached, which leads to discontinuous control variables. On the other hand, with the smoothing process, the control variables are continuous and much smoother, while the resulting trajectories are similar.

other words, when the robot's heading faces directly to the obstacle and the target, as shown in Fig. 3. The robot will move directly toward the obstacle and stop at the obstacle boundary.

Remark 1. When the robot encounters the above equilibrium state, we can add a constant $\epsilon > 0$ to u^{ref} in (14) such that $u^{\text{ref}} = [v_x^{\text{ref}} \ v_y^{\text{ref}} \ \omega^{\text{ref}} + \epsilon]^T$. As is shown in Fig. 4, the robot breaks its equilibrium state, avoids the obstacle, and reaches the target position. This is related to, but distinct from, the method presented in [19] for resolving unwanted equilibria.

4. Combining CBFs for Multiple Obstacles

So far, we have assumed there is only one obstacle perceived by the robot. In this section, we will discuss how to handle multiple obstacles in the environment when each obstacle is a positive distance apart from the others [47]. Specifically, for $i \in \{1, 2, \dots, M\}$, suppose that

$$B_i(x) := \begin{bmatrix} x_r - x_{o,i} \\ y_r - y_{o,i} \end{bmatrix}^T Q_i \begin{bmatrix} x_r - x_{o,i} \\ y_r - y_{o,i} \end{bmatrix} - r_{o,i}^2 \quad (16)$$

$$\mathcal{D}_i := \{x \in \mathbb{R}^3 \mid B_i(x) \neq -r_{o,i}^2, \text{ and } r \neq 0\}$$

$$C_i := \{x \in \mathcal{D}_i \mid B_i(x) \geq 0\}$$

are valid CBF functions for the dynamics (2). For $i \neq j$, the obstacles corresponding to $B_i : \mathbb{R}^3 \rightarrow \mathbb{R}$ and $B_j : \mathbb{R}^3 \rightarrow \mathbb{R}$ are a

positive distance apart if

$$\Delta_{ij} := \inf_{\substack{x \in C_i \\ y \in C_j}} \|x - y\| > 0. \quad (17)$$

A key innovation with respect to [46] is that we will compose the associated CBFs in a smooth (C^1) manner. A potential drawback with respect to [46] is that we will assume the obstacles giving rise to the CBFs are a positive distance apart. Similar to [47], we saturate standard quadratic CBFs before seeking to combine them. Distinct from [47], we multiply the saturated CBFs instead of creating a weighted sum. This greatly simplifies the analysis of the composite CBF with respect to all previous works.

4.1. Smooth Saturation Function

We introduce a continuously differentiable saturation function that will allow us to compose in a simple manner CBFs corresponding to obstacles that are a positive distance apart. Consider $\sigma : \mathbb{R} \rightarrow \mathbb{R}$ by

$$\sigma(s) := \begin{cases} s & s \leq 0 \\ s(1 + s - s^2) & 0 < s < 1 \\ 1 & s \geq 1. \end{cases} \quad (18)$$

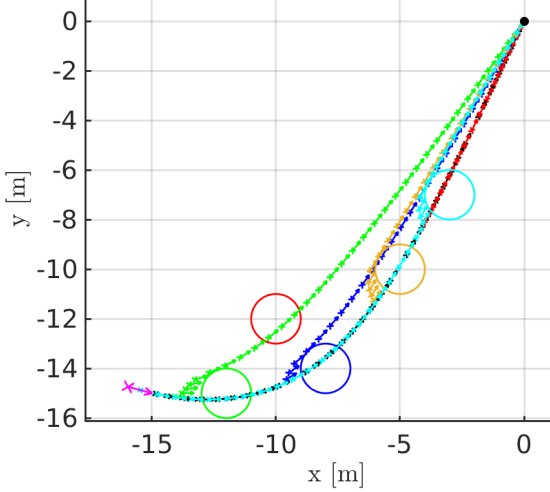


Figure 6: Illustration of how the trajectories vary as a function of different obstacle positions. The target (marked in black) and the robot pose $(-15, -15, -15^\circ)$ are fixed throughout all of the simulations. The different colors correspond to simulations with a single (different) obstacle present at a time. The black trajectory is generated without any obstacle present by only including the CLF constraint in the QP.

Then straightforward calculations show that for all $s \in \mathbb{R}$, $\frac{d\sigma(s)}{ds}$ exists and satisfies

$$\frac{d\sigma(s)}{ds} := \begin{cases} 1 & s \leq 0 \\ 1 + 2s - 3s^2 & 0 < s < 1 \\ 0 & s \geq 1. \end{cases} \quad (19)$$

Upon noting that $\frac{d\sigma(s)}{ds}|_{s=0} = 1$, $\frac{d\sigma(s)}{ds}|_{s=1} = 0$ and for all $0 < s < 1$, $0 < \frac{d\sigma(s)}{ds}$, it follows that $\sigma : \mathbb{R} \rightarrow \mathbb{R}$ is continuously differentiable and monotonic.

Remark 2. For $0 \leq s \leq 1$, σ is constructed from a degree-three Bézier polynomial $p : [0, 1] \rightarrow \mathbb{R}$ such that $p(0) = 0$, $\frac{dp(0)}{ds} = 1$, $p(1) = 1$, $\frac{dp(1)}{ds} = 0$. Moreover, for $0 < s < 1$, $\frac{dp(s)}{ds} > 0$.

Definition 1. For $\kappa > 0$, we define $\sigma_\kappa : \mathbb{R} \rightarrow \mathbb{R}$ by

$$\sigma_\kappa(s) := \sigma\left(\frac{s}{\kappa}\right). \quad (20)$$

Proposition 1. Suppose that $\kappa > 0$ and $B : \mathcal{D} \rightarrow \mathbb{R}$ is a candidate CBF with $\mathcal{D}$ and $\mathcal{C}$ defined as in (7). Then $\sigma_\kappa \circ B : \mathcal{D} \rightarrow \mathbb{R}$ is a valid CBF for the system (2) if, and only if, $B : \mathcal{D} \rightarrow \mathbb{R}$ is a valid CBF.

Proof. For $x \in \mathcal{C}$, $\sigma_\kappa \circ B(x) > 0$ and hence satisfies (8) for $u = 0$. For $x \in \mathcal{D} \setminus \mathcal{C}$, by the chain rule and the construction of $\sigma : \mathbb{R} \rightarrow \mathbb{R}$,

$$\frac{\partial \sigma_\kappa \circ B(x)}{\partial x} = \frac{d\sigma(s)}{ds} \Big|_{s=\frac{B(x)}{\kappa}} \frac{\partial B(x)}{\partial x} = \frac{1}{\kappa} \frac{\partial B(x)}{\partial x}. \quad (21)$$

Hence, the proof of Sect. 3.3 applies. $\square$

Proposition 2. Suppose for $1 \leq i \leq M$, the CBFs $B_i(x) : \mathbb{R}^3 \rightarrow \mathbb{R}$ are a positive distance apart. Then there exist $\kappa_1 > 0$, $\kappa_2 > 0$, $\dots$, $\kappa_M > 0$, such that for all $i \neq j$,

$$\{x \in \mathbb{R}^3 \mid \sigma_{\kappa_i} \circ B_i(x) < 1\} \cap \{x \in \mathbb{R}^3 \mid B_j(x) < 0\} = \emptyset. \quad (22)$$

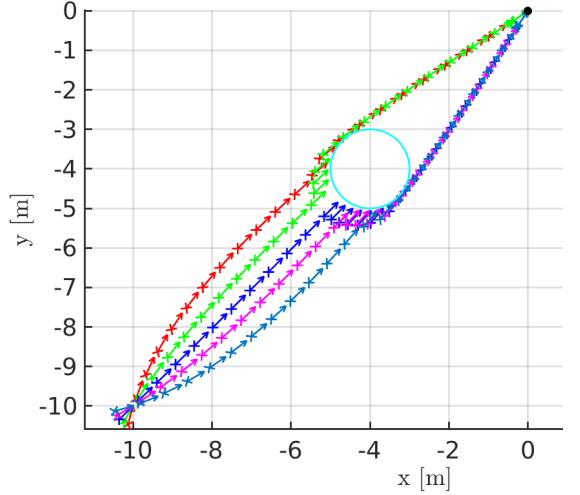


Figure 7: Illustration of how the trajectories vary as a function of different robot orientations with a fixed obstacle location. The target (marked in cyan) and the obstacle at $(-4, -4)$ are fixed throughout all simulations. Each color stands for a different initial robot orientation.

Proof. By the disjointness property, $\Delta_i := \min_{j \neq i} \Delta_{ij} > 0$.

For $S \subset \mathbb{R}^3$ and $x \in \mathbb{R}^3$, define the distance from x to S by

$$d(x, S) := \inf_{y \in S} \|x - y\|. \quad (23)$$

Then, because (i) B_i is continuous, (ii) the set complement of $\mathcal{C}_i$ is bounded, and (iii) $d(x, \sim \mathcal{C}_i) > 0 \implies B_i(x) > 0$, it follows that

$$m_i^* := \sup_{d(x, \sim \mathcal{C}_i) \leq \Delta_i} B_i(x) \quad (24)$$

is a finite positive number. Therefore, for all $0 < \kappa_i < m_i^*$,

$$\{x \in \mathbb{R}^3 \mid \sigma_{\kappa_i} \circ B_i(x) < 1\} \subset \{x \in \mathbb{R}^3 \mid d(x, \sim \mathcal{C}_i) \leq \Delta_i\}, \quad (25)$$

and hence (22) holds. $\square$

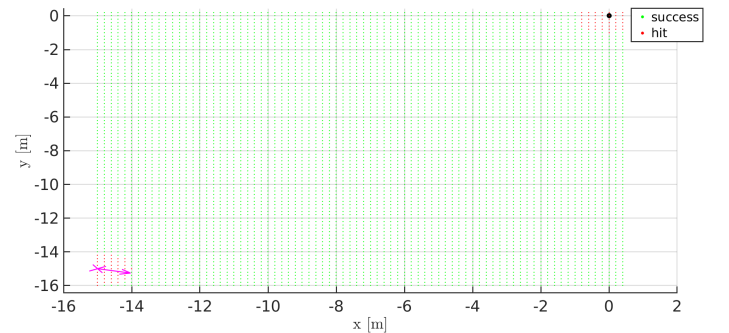


Figure 8: Liveness analysis for the CLF-CBF system. The initial pose is $(-15, -15, -15^\circ)$, and the target is located at $(0, 0)$. Each dot in the figure represents the center of an object with radius $(r = 1)$. The interval between each center dots are 0.2 meter in both x and y direction. Note that all the red points either originally collide with the robot or the target.

4.2. Multiplication Property of Smooth Saturated CBFs

For $M \geq 2$ CBFs corresponding to disjoint obstacles, define the sets

$$\begin{aligned}\mathcal{D}_M &:= \bigcap_{i=1}^M \mathcal{D}_i \\ \mathcal{C}_M &:= \bigcap_{i=1}^M \{x \in \mathcal{D}_M \mid B_i(x) \geq 0\} \\ &= \bigcap_{i=1}^M \mathcal{C}_i.\end{aligned}\quad (26)$$

Theorem 1. *Under the assumed disjointness property, the product of smoothly saturated valid CBFs,*

$$B_M(x) := \prod_{i=1}^M \sigma_{\kappa_i} \circ B_i(x), \quad (27)$$

is a valid CBF for $\mathcal{D}_M$, $\mathcal{C}_M$, and the dynamic system (2).

Proof. For $x \in \mathcal{C}_M$, the zero control $u \equiv 0$ satisfies (8) because the drift term $f(x)$ is zero. We show that for $x \notin \mathcal{C}_M$, (8) can be satisfied.

By the disjoint property of the assumed CBF functions, when $B_M(x) < 0$, we have $\exists i$, such that $\sigma_{\kappa_i} \circ B_i(x) = B_i(x) < 0$, and

$\sigma_{\kappa_j} \circ B_j(x) = 1$ for $j \neq i$. Hence, $B_M(x) = B_i(x)$. Because $B_i(x)$ is assumed to be a valid CBF function, and both $\mathcal{D}_M \subset \mathcal{D}_i$ and $\mathcal{C}_M \subset \mathcal{C}_i$ hold, the CBF property holds for $B_M(x)$. $\square$

Remark 3. *Due to the way we have constructed the multi-obstacle CBF, the equilibrium analysis for a single obstacle carries over here without changes. This is because, when the robot is at a boundary of an obstacle, the values of the saturated CBFs for the other obstacles will all be one.*

5. Smoothing Reference Control Variables

We have created a continuous vector field for a single target position surrounded by multiple obstacles. However, discontinuities are introduced when we switch from one target to another. The jumps appear in two ways: when the attraction points used in the CLF are updated and when the associated reference control variable u^{ref} is updated. The top row of Fig. 5 illustrates the jumps in the control signals associated with a list of targets (black dots).

This section proposes a means to smooth the control signals.

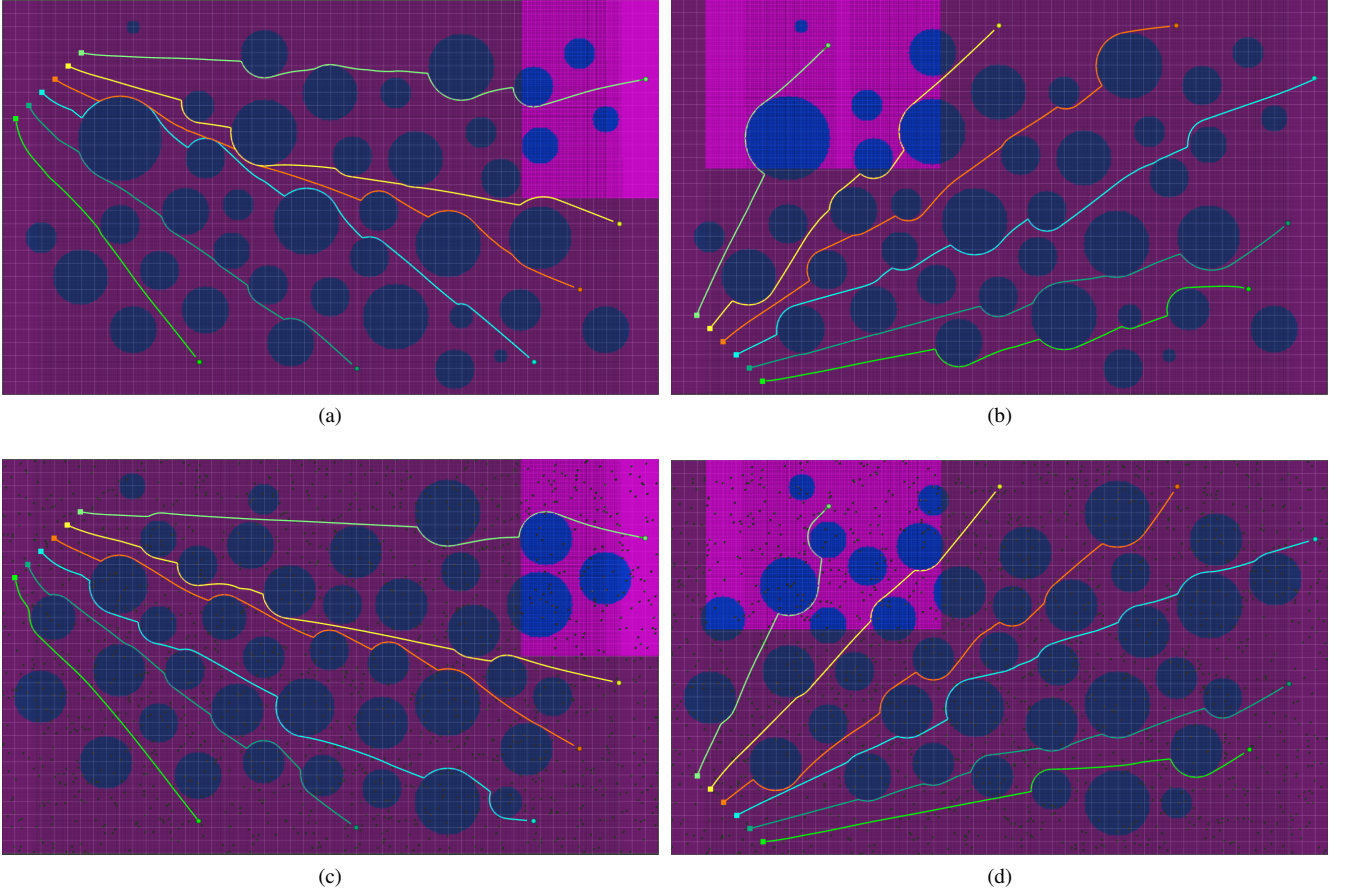


Figure 9: Robot trajectories with 40 obstacles in the noise-free map 1 (top two) and 40 obstacles in the noisy map 2 (bottom two) synthetic maps with the size of 50×30 meters. The highlighted areas are the local map at that specific timestamp. The dark blue circles are the obstacles. Different colors represent different runs in the map.

To begin, we define $\tau : \mathbb{R} \rightarrow \mathbb{R}$ by

$$\tau(s) := \begin{cases} 0 & s \leq 0 \\ s^2(3 - 2s) & 0 < s < 1 \\ 1 & s \geq 1, \end{cases} \quad (28)$$

where we have $\forall s \in \mathbb{R}$, $\frac{d\tau(s)}{ds}$ exists and satisfies

$$\frac{d\tau(s)}{ds} := \begin{cases} 0 & s \leq 0 \\ 6s - 6s^2 & 0 < s < 1 \\ 0 & s \geq 1. \end{cases} \quad (29)$$

Upon noting that $\frac{d\tau(s)}{ds}|_{s=0} = 0$, $\frac{d\tau(s)}{ds}|_{s=1} = 0$ and for all $0 \leq s \leq 1$, $\frac{d\tau(s)}{ds} > 0$, it follows that $\tau(s)$ is continuously differentiable and monotonic.

Consider now two target positions $\mathcal{G}_1$ and $\mathcal{G}_2$ with their reference control variables u_1^{ref} and u_2^{ref} from (4). A smoothed u^{ref} is defined as the convex combination

$$u^{\text{ref}} = \tau\left(\frac{t}{T}\right)u_1^{\text{ref}} + (1 - \tau\left(\frac{t}{T}\right))u_2^{\text{ref}}, \quad (30)$$

where T is a switching time parameter and t is the estimated time to $\mathcal{G}_1$. By the construction of the scaling function τ in (28) and the convex combination in (30), we only execute the interpolation process if the robot is estimated to be less than T seconds away from $\mathcal{G}_1$. The value of t is updated to the estimated time to $\mathcal{G}_1$ when the $\mathcal{G}_1$ is updated. The value of t decreases while the robot is moving. By applying the smoothed u^{ref} and with the same list of goals, the control variables become continuous, as shown in the bottom row of Fig. 5.

6. Simulation Results with single and multiple obstacles

In this section, we first use simulation to study the behavior and liveness of the proposed CLF-CBF system with a single obstacle. Next, we run the system on several synthetic environments with 20 obstacles in Robot Operating System (ROS) [23] with C++.

Remark 4. For the CBF in (6), we take $Q = I$ and in Prop. 1, we take $\kappa_1 = \dots = \kappa_M = \min\{\Delta_i^2\}_{i=1}^M$, which is the minimum of the square of the distance between any of the obstacles.

6.1. Robot Model in Simulation

In MATLAB and ROS, the bipedal robot is represented by the Angular momentum Linear Inverted Pendulum (ALIP) model [2]. The ALIP robot takes piece-wise constant inputs from the CLF-CBF-QP system. Let g, H, τ be the gravitational constant, the robot's center of mass height, and the time interval of a swing phase, respectively. The motion of an ALIP model on the x -axis satisfies

$$\begin{bmatrix} x_{k+1} \\ \dot{x}_{k+1} \end{bmatrix} = \begin{bmatrix} \cosh(\xi) & \frac{1}{\rho} \sinh(\xi) \\ \rho \sinh(\xi) & \cosh(\xi) \end{bmatrix} \begin{bmatrix} x_k \\ \dot{x}_k \end{bmatrix} + \begin{bmatrix} 1 - \cosh(\xi) \\ -\rho \sinh(\xi) \end{bmatrix} p_x, \quad (31)$$

where x_k and $\dot{x}_k$ are the contact position and velocity of the swing foot on the x -axis, p_x is the center of mass (CoM) position on the x -axis of the robot, $\xi = \rho\tau$ and $\rho = \sqrt{g/H}$. The motion of the robot on the y -axis can be similarly defined.

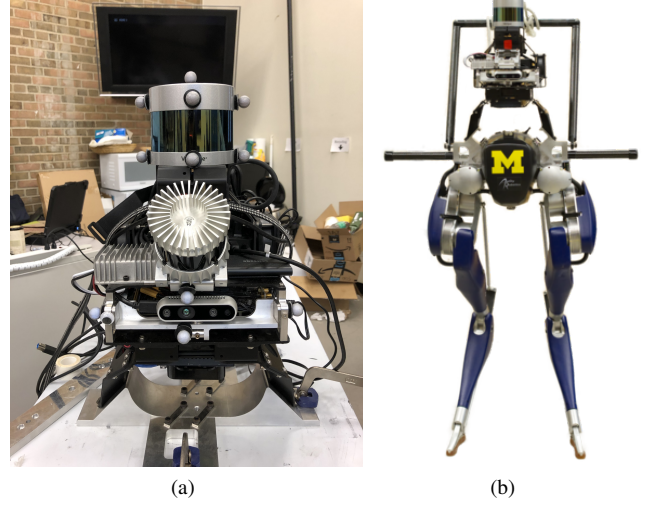


Figure 10: The left shows the sensor suite with different sensors, and the right shows the sensor suite mounted on Cassie Blue.

6.2. Behavior Study with Single Obstacle in MATLAB

The optimal control command of the robot is the solution of the CLF-CBF-QP problem defined in (14). The time interval of a swing phase is set to $\tau = 0.3s$. The robot updates its pose based on the ALIP model and the optimal control command. The updated pose is then fed back to the CLF-CBF-QP system to compute the optimal control for the next iteration. This process continues until the robot reaches the target or collides with an obstacle.

Figure 6 shows how the trajectories vary as a function of a single obstacle's position with a fixed initial robot pose of $(-15, -15, -15^\circ)$, marked as the magenta arrow. The black trajectory is the nominal trajectory without any obstacles present. Each colored trajectory and matching circle represent a distinct simulation result. The robot successfully avoids the obstacle in all cases. In Fig. 7, we show how the trajectories vary as a function of different robot orientations with a fixed obstacle location.

Remark 5. When the robot is within an obstacle, there is also a valid solution that pushes the robot outside of the obstacle. Consider the CBF constraint (13),

$$L_f B(x) + L_g B(x)u + \eta B(x) \geq 0. \quad (32)$$

When the robot is within an obstacle, $B(x) < 0$ and the QP selects u such that $L_f B(x) + L_g B(x)u \geq -\eta B(x)$, causing the robot to leave the obstacle.

6.3. Liveness Analysis in MATLAB

We analyze the liveness by placing an obstacle with a fixed radius ($r = 1$) at different locations. The robot starts at $(-15, -15, -15^\circ)$ and the target is located at $(0, 0)$. The obstacle is placed at every 0.2 meter. If the robot successfully reaches the target without collision, the obstacle location is marked in green otherwise in red, as shown in Fig. 8. All the red points either originally collide with the robot or the target.

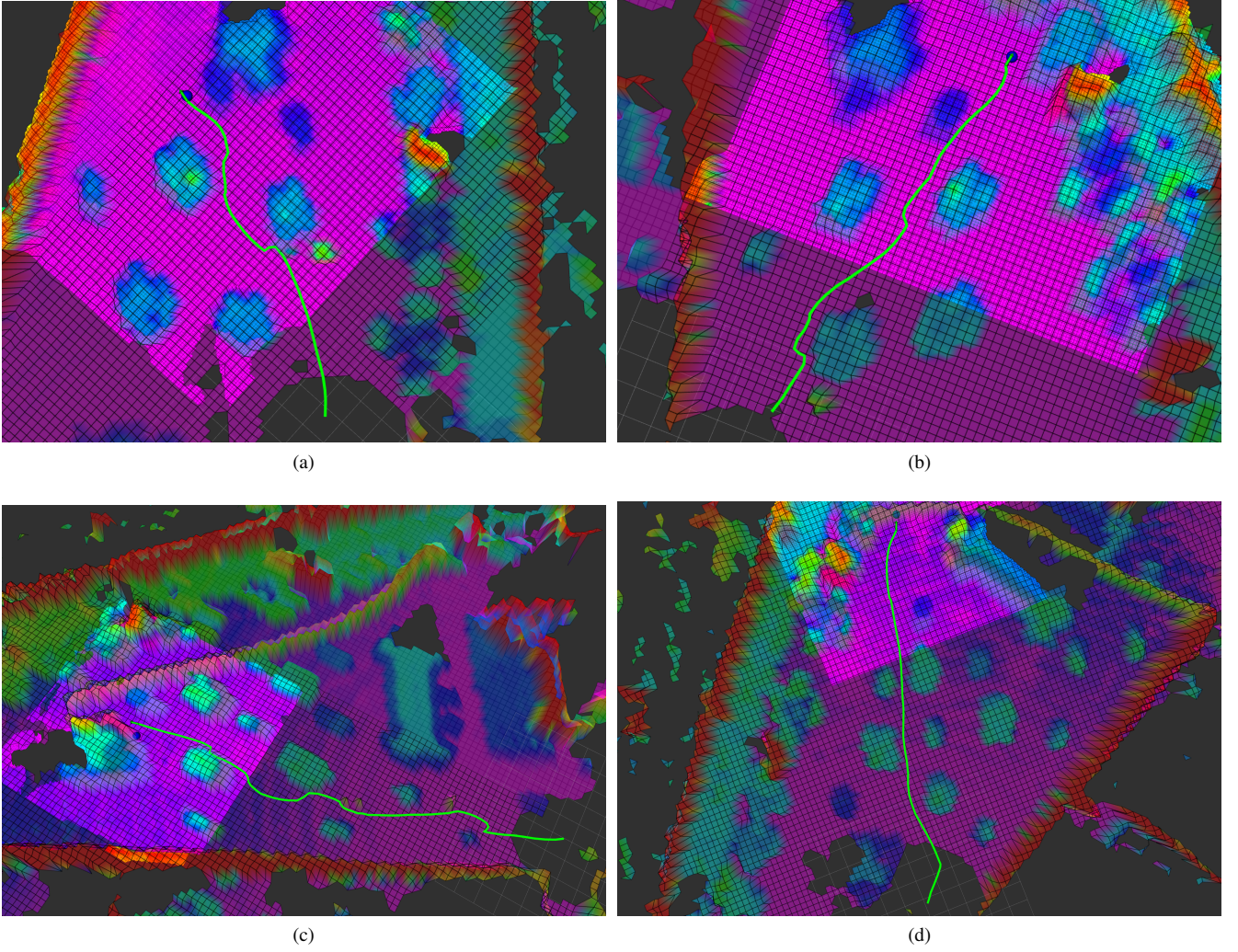


Figure 11: Autonomy experiments with Cassie Blue on the first floor of FRB. The green arrow is Cassie’s pose and the green lines are the resulting trajectories. The blue sphere is the selected target position. The map is colored by height and the highlighted area is the local map.

6.4. Multi-Obstacle Simulation with ROS in C++

In this simulation, we implement a local map centering at robot position with a fixed size and a sub-goal selector to place a target within the local map to achieve long-term planning as not all the obstacles are perceived by the robot at the beginning in practice. Even though the global map is available in simulation but it is not available in practice, therefore, only the information within the local map at the specific timestamp is provided to the robot. The robot model is the same ALIP model in Sec. 6.1.

In Fig. 9, we generate two noise-free and two noisy synthetic maps with the size of 50×30 meters. Each map contains 20 obstacles marked as blue circles. We run six different initial poses and final goals for each map. Different colors represent different runs in the map. The highlighted area is the local map at that specific timestamp. An intermediate goal is chosen at the intersection between the boundary of the local map and the line connecting the robot and the final goal at the current timestamp. If the intermediate goal collides with an obstacle, it is moved back along the line. The intermediate goal is updated when it is reached or becomes inside of an obstacle due to the update of

the local map. The robot with ALIP model successfully reaches the goals in all $6 \times 4 = 24$ runs.

7. Experimental Results on a Bipedal Robot

We perform several experiments of the proposed CLF-CBF-QP system on Cassie Blue, a bipedal robot with 20 degrees of freedom. The entire system integrates elevation mapping, intermediate goal selection, and the low-level CLF-CBF obstacle avoidance system.

7.1. Autonomy System Integration

The following is summarized from [1] for the completeness of the paper. To allow the robot to perceive its surroundings under different lighting conditions and environments, we designed a perception suite that consists of an RGB-D camera (Intel RealSense™ D435) and a 32-Beam Velodyne ULTRA Puck LiDAR, as shown in Fig. 10. The sensor calibrations are performed via [50, 51, 52, 53]. The invariant extended Kalman filter (InEKF) [54] estimates the pose of Cassie at 2k Hz. The raw

point cloud is motion compensated by the InEKF and then used to build an elevation map.

7.2. Autonomy Experiment on Cassie Blue

We conducted several indoor experiments with Cassie Blue on the first floor of the Ford Robotics Building (FRB) where tables and chairs are considered obstacles. To detect obstacles in the environment, an occupancy grid map is updated in real-time using the timestamped elevation map. Grids with heights greater than 0.2 meters are considered occupied. An occupied grid is defined as the boundary of obstacles if there is an unoccupied grid in its neighborhood. The Breadth First Search (BFS) algorithm [55] is utilized to find the separated obstacles in the map. Next, we apply the Gift Wrapping Algorithm [56] to the boundary grids of obstacles to find the convex hulls of the obstacles. Finally, the minimum bounding ball algorithm [57] is applied to the convex hulls to find the minimum bounding circles of the obstacles. The circles are used to represent obstacles in the CBF function (6). The target position is selected by clicking a point in the global map. If the final target is not within the current local map, an intermediate goal will be selected within the local map. When an intermediate goal is reached by Cassie or becomes invalid because of the update of the local map, it is updated. In the experiments, Cassie successfully avoids all the obstacles and reaches the target position, as shown in Fig. 11.

8. Conclusion

This paper presented a reactive planning system that allows a Cassie-series bipedal robot to avoid multiple non-overlapping obstacles via a single, continuously differentiable control barrier function (CBF). The overall system detects an individual obstacle via a height map derived from a LiDAR point cloud and computes an elliptical outer approximation, which is then turned into a quadratic CBF. A continuously differentiable saturation function is presented that preserves the CBF property of a quadratic CBF while allowing the saturated CBFs for individual obstacles to be turned into a single CBF. The CLF-CBF-QP formalism developed by Ames *et al.* can then be applied to ensure that safe trajectories are generated in the presence of multiple obstacles. Liveness is ensured by an analysis of induced equilibrium points that are distinct from the goal state. Safe planning in environments with multiple obstacles is demonstrated both in simulation and experimentally on the Cassie bipedal robot.

Acknowledgment

Toyota Research Institute provided funds to support this work. Funding for J. Grizzle was in part provided by NSF Award No. 1808051. This article solely reflects the opinions and conclusions of its authors and not the funding entities.

References

- [1] J.-K. Huang, J. W. Grizzle, Efficient anytime clf reactive planning system for a bipedal robot on undulating terrain, arXiv preprint arXiv:2108.06699 (2021).
- [2] Y. Gong, J. Grizzle, Angular momentum about the contact point for control of bipedal locomotion: Validation in a lip-based controller, arXiv preprint arXiv:2008.10763 (2020).
- [3] G. Gibson, O. Dosunmu-Ogunbi, Y. Gong, J. Grizzle, Terrain-adaptive, alip-based bipedal locomotion controller via model predictive control and virtual constraints, 2021. URL: <https://arxiv.org/abs/2109.14862>. doi:10.48550/ARXIV.2109.14862.
- [4] L. E. Kavraki, P. Svestka, J.-C. Latombe, M. H. Overmars, Probabilistic roadmaps for path planning in high-dimensional configuration spaces, IEEE transactions on Robotics and Automation 12 (1996) 566–580.
- [5] S. M. LaValle, Planning algorithms, Cambridge university press, 2006.
- [6] J. Borenstein, Y. Koren, Real-time obstacle avoidance for fast mobile robots, IEEE Transactions on systems, Man, and Cybernetics 19 (1989) 1179–1187.
- [7] Y. K. Hwang, N. Ahuja, et al., A potential field approach to path planning, IEEE transactions on robotics and automation 8 (1992) 23–32.
- [8] K. P. Valavanis, T. Hebert, R. Kolluru, N. Tsourveloudis, Mobile robot navigation in 2-d dynamic environments using an electrostatic potential field, IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans 30 (2000) 187–196.
- [9] O. Montiel, U. Orozco-Rosas, R. Sepúlveda, Path planning for mobile robots using bacterial potential field for avoiding static and dynamic obstacles, Expert Systems with Applications 42 (2015) 5177–5191.
- [10] Y. Rasekhipour, A. Khajepour, S.-K. Chen, B. Litkouhi, A potential field-based model predictive path-planning controller for autonomous road vehicles, IEEE Transactions on Intelligent Transportation Systems 18 (2017) 1255–1267. doi:10.1109/TITS.2016.2604240.
- [11] J. Sfeir, M. Saad, H. Saliha-Hassane, An improved artificial potential field approach to real-time mobile robot path planning in an unknown environment, in: 2011 IEEE International Symposium on Robotic and Sensors Environments (ROSE), 2011, pp. 208–213. doi:10.1109/ROSE.2011.6058518.
- [12] W. Lu, G. Zhang, S. Ferrari, An information potential approach to integrated sensor path planning and control, IEEE Transactions on Robotics 30 (2014) 919–934. doi:10.1109/TRO.2014.2312812.
- [13] F. Bounini, D. Gingras, H. Pollart, D. Gruyer, Modified artificial potential field method for online path planning applications, in: 2017 IEEE Intelligent Vehicles Symposium (IV), 2017, pp. 180–185. doi:10.1109/IVS.2017.7995717.
- [14] J. Ren, K. McIsaac, R. Patel, Modified newton’s method applied to potential field-based navigation for mobile robots, IEEE Transactions on Robotics 22 (2006) 384–391. doi:10.1109/TRO.2006.870668.
- [15] K. Biswas, I. Kar, On reduction of oscillations in target tracking by artificial potential field method, in: 2014 9th International Conference on Industrial and Information Systems (ICIIS), 2014, pp. 1–6. doi:10.1109/ICIINFS.2014.7036582.
- [16] A. D. Ames, J. W. Grizzle, P. Tabuada, Control barrier function based quadratic programs with application to adaptive cruise control, in: 53rd IEEE Conference on Decision and Control, 2014, pp. 6271–6278. doi:10.1109/CDC.2014.7040372.
- [17] S.-C. Hsu, X. Xu, A. D. Ames, Control barrier function based quadratic programs with application to bipedal robotic walking, in: 2015 American Control Conference (ACC), 2015, pp. 4542–4548. doi:10.1109/ACC.2015.7172044.
- [18] J. Zeng, B. Zhang, Z. Li, K. Sreenath, Safety-critical control using optimal-decay control barrier function with guaranteed point-wise feasibility, in: 2021 American Control Conference (ACC), 2021, pp. 3856–3863. doi:10.23919/ACC50511.2021.9482626.
- [19] M. F. Reis, A. P. Aguiar, P. Tabuada, Control barrier function-based quadratic programs introduce undesirable asymptotically stable equilibria, IEEE Control Systems Letters 5 (2020) 731–736.
- [20] M. Jankovic, M. Santillo, Y. Wang, Multi-agent systems with cbf-based controllers – collision avoidance and liveness from instability, 2022. URL: <https://arxiv.org/abs/2207.04915>. doi:10.48550/ARXIV.2207.04915.
- [21] X. Tan, D. V. Dimarogonas, On the undesired equilibria induced by control barrier function based quadratic programs, 2021.

URL: <https://arxiv.org/abs/2104.14895>. doi:10.48550/ARXIV.2104.14895.

- [22] G. Notomista, M. Saveriano, Safety of dynamical systems with multiple non-convex unsafe sets using control barrier functions, *IEEE Control Systems Letters* 6 (2022) 1136–1141. doi:10.1109/LCSYS.2021.3089097.
- [23] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng, ROS: an open-source Robot Operating System, in: *ICRA workshop on open source software*, 2009.
- [24] H. Khalil, *Nonlinear Systems - 3rd Edition*, Prentice Hall, Upper Saddle River, NJ, 2002.
- [25] E. Sontag, A Lyapunov-like stabilization of asymptotic controllability, *SIAM Journal of Control and Optimization* 21 (1983) 462–471.
- [26] E. Sontag, A 'universal' construction of Artstein's theorem on nonlinear stabilization, *Systems & Control Letters* 13 (1989) 117–123.
- [27] R. A. Freeman, P. V. Kokotović, *Robust Nonlinear Control Design*, Birkhäuser, 1996.
- [28] O. Khatib, Real-time obstacle avoidance for manipulators and mobile robots, in: *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, volume 2, IEEE, 1985, pp. 500–505.
- [29] D. E. Koditschek, Robot planning and control via potential functions, *The robotics review* (1989) 349.
- [30] Y. Koren, J. Borenstein, et al., Potential field methods and their inherent limitations for mobile robot navigation., in: *ICRA*, volume 2, 1991, pp. 1398–1404.
- [31] E. Rimon, D. Koditschek, Exact robot navigation using artificial potential functions, *IEEE Transactions on Robotics and Automation* 8 (1992) 501–518. doi:10.1109/70.163777.
- [32] D. E. Koditschek, E. Rimon, Robot navigation functions on manifolds with boundary, *Advances in applied mathematics* 11 (1990) 412–442.
- [33] O. Arslan, D. E. Koditschek, Exact robot navigation using power diagrams, in: *Proc. IEEE Int. Conf. Robot. and Automation*, 2016, pp. 1–8. doi:10.1109/ICRA.2016.7487090.
- [34] O. Arslan, D. E. Koditschek, Sensor-based reactive navigation in unknown convex sphere worlds, *The International Journal of Robotics Research* 38 (2019) 196–223.
- [35] S. Paternain, D. E. Koditschek, A. Ribeiro, Navigation functions for convex potentials in a space with convex obstacles, *IEEE Transactions on Automatic Control* 63 (2017) 2944–2959.
- [36] P. Wieland, F. Allgöwer, Constructive safety using control barrier functions, in: *Proceedings of the 7th IFAC Symposium on Nonlinear Control System*, 2007, pp. 462–467.
- [37] Z. Li, Comparison between safety methods control barrier function vs. reachability analysis, 2021. URL: <https://arxiv.org/abs/2106.13176>. doi:10.48550/ARXIV.2106.13176.
- [38] A. Singletary, S. Kolathaya, A. D. Ames, Safety-critical kinematic control of robotic systems, *IEEE Control Systems Letters* 6 (2022) 139–144. doi:10.1109/LCSYS.2021.3050609.
- [39] M. Jankovic, Combining control lyapunov and barrier functions for constrained stabilization of nonlinear systems, in: *2017 American Control Conference (ACC)*, 2017, pp. 1916–1922. doi:10.23919/ACC.2017.7963232.
- [40] K. Garg, D. Panagou, Control-lyapunov and control-barrier functions based quadratic program for spatio-temporal specifications, in: *2019 IEEE 58th Conference on Decision and Control (CDC)*, 2019, pp. 1422–1429. doi:10.1109/CDC40024.2019.9029666.
- [41] M. Desai, A. Ghaffari, Clf-cbf based quadratic programs for safe motion control of nonholonomic mobile robots in presence of moving obstacles, in: *2022 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, IEEE, 2022, pp. 16–21.
- [42] E. A. Basso, E. H. Thyri, K. Y. Pettersen, M. Breivik, R. Skjetne, Safety-critical control of autonomous surface vehicles in the presence of ocean currents, in: *2020 IEEE Conference on Control Technology and Applications (CCTA)*, IEEE, 2020, pp. 396–403.
- [43] A. Agrawal, K. Sreenath, Discrete control barrier functions for safety-critical control of discrete systems with application to bipedal robot navigation., in: *Robotics: Science and Systems*, volume 13, Cambridge, MA, USA, 2017.
- [44] A. Singletary, K. Klingebiel, J. Bourne, A. Browning, P. Tokumaru, A. Ames, Comparative analysis of control barrier functions and artificial potential fields for obstacle avoidance, in: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 8129–8136. doi:10.1109/IROS51168.2021.9636670.
- [45] M. Rauscher, M. Kimmel, S. Hirche, Constrained robot control using control barrier functions, in: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2016, pp. 279–285.
- [46] P. Glotfelter, J. Cortés, M. Egerstedt, Nonsmooth barrier functions with applications to multi-robot systems, *IEEE control systems letters* 1 (2017) 310–315.
- [47] M. Z. Romdlony, B. Jayawardhana, Stabilization with guaranteed safety using control lyapunov–barrier function, *Automatica* 66 (2016) 39–47.
- [48] P. Thontepu, B. G. Goswami, N. Singh, S. P. S. S. M. G. S. Sundaram, V. Katewa, S. Kolathaya., Control barrier functions in ugvs for kinematic obstacle avoidance: A collision cone approach, 2022. URL: <https://arxiv.org/abs/2209.11524>. doi:10.48550/ARXIV.2209.11524.
- [49] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, P. Tabuada, Control barrier functions: Theory and applications, in: *2019 18th European control conference (ECC)*, IEEE, 2019, pp. 3420–3431.
- [50] J. Huang, J. W. Grizzle, Improvements to Target-Based 3D LiDAR to Camera Calibration, *IEEE Access* 8 (2020) 134101–134110.
- [51] J. K. Huang, S. Wang, M. Ghaffari, J. W. Grizzle, LiDARtag: A Real-Time Fiducial Tag System for Point Clouds, *IEEE Robotics and Automation Letters* (2021) 1–1. doi:10.1109/LRA.2021.3070302.
- [52] J.-K. Huang, C. Feng, M. Achar, M. Ghaffari, J. W. Grizzle, Global Unifying Intrinsic Calibration for Spinning and Solid-State LiDARs, *arXiv preprint arXiv:2012.03321* (2020).
- [53] J.-K. Huang, W. Clark, J. W. Grizzle, Optimal target shape for lidar pose estimation, *arXiv preprint arXiv:2109.01181* (2021).
- [54] R. Hartley, M. Ghaffari, R. M. Eustice, J. W. Grizzle, Contact-aided invariant extended kalman filtering for robot state estimation, *The International Journal of Robotics Research* 39 (2020) 402–430.
- [55] A. Bundy, L. Wallen, Breadth-first search, in: *Catalogue of artificial intelligence tools*, Springer, 1984, pp. 13–13.
- [56] T. Cormen, C. Leiserson, R. Rivest, C. Stein, *Finding the convex hull. introduction to algorithms*, 2009.
- [57] A. M. N. Alzubaidi, Minimum bounding circle of 2d convex hull, *International journal of Science and research* 3 (2014) 364–367.
- [58] S. I. Gass, C. M. Harris, *Encyclopedia of operations research and management science*, *Journal of the Operational Research Society* 48 (1997) 759–760.

Appendix A. Equilibrium Analysis of Multi-Obstacle Systems

We give the complete analysis for a single obstacle, following the work of [19]. Because the drift term of our model is zero, any equilibrium points are where the optimal control is the zero vector. To avoid this undesirable situation, we seek to find all equilibrium points $\mathcal{E} = \{x|u^* = [0, 0, 0]^T, r > 0, \text{ and } \delta, \theta \in (-\pi, \pi]\}$, where u^* is the optimal control variable. Recall that $f = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T$ in (2), which leads to $L_f V(x) = L_f B(x) = 0$. We denote the following to re-write the CLF and the CBF constraints:

$$\begin{bmatrix} d_x & d_y & 0 \end{bmatrix} = -L_g B(x) \\ \begin{bmatrix} a_x & a_y & a_\omega \end{bmatrix} = L_g V(x), \quad (\text{A.1})$$

where

$$\begin{aligned} a_x &= -r \cos(\delta) + \frac{\beta \gamma^2 \sin(2\beta\delta) \sin(\delta)}{2r} \\ a_y &= -r \sin(\delta) - \frac{\beta \gamma^2 \sin(2\beta\delta) \cos(\delta)}{2r} \\ a_\omega &= \frac{\beta \gamma \sin(2\beta\delta)}{2}. \end{aligned} \quad (\text{A.2})$$

The constraints become

$$\mathfrak{L}(x, u, s) = a_x v_x + a_y v_y + a_\omega \omega - s + \mu V(x), \quad (\text{A.3})$$

$$\mathfrak{B}(x, u) = d_x v_x + d_y v_y - \eta B(x), \quad (\text{A.4})$$

and the cost function (15) of the QP can then be re-written as:

$$\begin{aligned} \mathfrak{J}(u, s) = & \frac{1}{2}h_1(v_x - v_x^{\text{ref}})^2 + \frac{1}{2}h_2(v_y - v_y^{\text{ref}})^2 \\ & + \frac{1}{2}h_3(\omega - \omega^{\text{ref}})^2 + \frac{1}{2}ps^2, \end{aligned} \quad (\text{A.5})$$

where $\{h_i\}_{i=1}^3$ are the diagonal elements of H in (15), the weights of control variables $[v_x, v_y, \omega]$ with $h_i > 0$.

The KKT conditions [58] of this quadratic program are:

$$\frac{\partial L}{\partial u} = Hu^* - Hu^{\text{ref}} + \lambda_1 L_g V^T - \lambda_2 L_g B^T = 0 \quad (\text{A.6})$$

$$\frac{\partial L}{\partial s} = ps - \lambda_1 = 0 \quad (\text{A.7})$$

$$0 = \lambda_1(L_f V + L_g V u^* + \mu V - s) \quad (\text{A.8})$$

$$0 = \lambda_2(-L_f B - L_g B u^* - \eta B) \quad (\text{A.9})$$

$$0 \geq L_f V + L_g V u^* + \mu V - s \quad (\text{A.10})$$

$$0 \geq -L_f B - L_g B u^* - \eta B \quad (\text{A.11})$$

$$0 \leq \lambda_1, \lambda_2, \quad (\text{A.12})$$

where $\lambda_1, \lambda_2 \in \mathbb{R}$, and L is the Lagrangian function and defined as

$$L(u, s, \lambda_1, \lambda_2) = \mathfrak{J}(u, s) + \lambda_1 \mathfrak{L}(x, u, s) + \lambda_2 \mathfrak{B}(x, u). \quad (\text{A.13})$$

Next, we analyze equilibrium points (if any) via four cases depending on whether each CLF or CBF constraint is active or inactive following [19].

Appendix A.1. Both CLF and CBF are inactive

When both constraints are inactive, we have

$$\begin{aligned} \lambda_1 &= 0 \\ \lambda_2 &= 0 \\ 0 &> L_f V + L_g V u^* + \mu V - s \\ 0 &> -L_f B - L_g B u^* - \eta B. \end{aligned} \quad (\text{A.14})$$

With (A.6) and (A.7), u^* and s^* in this case are

$$\begin{aligned} u^* &= u^{\text{ref}} \\ s^* &= 0. \end{aligned} \quad (\text{A.15})$$

From (4), as long as the goal is not reached, u^{ref} is not a zero vector. Hence, there is no equilibrium point in this case.

Appendix A.2. CLF constraint inactive and CBF constraint active

We prove that there is no equilibrium point in this case by contradiction. When the CLF constraint is inactive and the CBF constraint is active, we have

$$\begin{aligned} \lambda_1 &= 0 \\ \lambda_2 &\geq 0 \\ 0 &> L_f V + L_g V u^* + \mu V - s \\ 0 &= -L_f B - L_g B u^* - \eta B. \end{aligned} \quad (\text{A.16})$$

With (A.6) and (A.7), u^* , s^* and λ_2 in this case are

$$\begin{aligned} u^* &= u^{\text{ref}} + \lambda_2 H^{-1} L_g B^T \\ s^* &= 0 \\ \lambda_2 &= -\frac{\eta B + L_f B + L_g B u^{\text{ref}}}{L_g B H^{-1} L_g B^T}. \end{aligned} \quad (\text{A.17})$$

If there is an equilibrium point, then u^* is the zero vector. Hence, at the equilibrium point, by $L_f V(x) = 0$, $u^* = 0$ and $s^* = 0$, we have

$$L_f V + L_g V u^* + \mu V - s^* = \mu V > 0, \quad (\text{A.18})$$

which conflicts with (A.16). Therefore, there is no equilibrium point in this case.

Appendix A.3. CLF constraint active and CBF constraint inactive

When the CLF constraint is active and the CBF constraint is inactive, we have

$$\begin{aligned} \lambda_1 &\geq 0 \\ \lambda_2 &= 0 \\ 0 &= L_f V + L_g V u^* + \mu V - s \\ 0 &> -L_f B - L_g B u^* - \eta B. \end{aligned} \quad (\text{A.19})$$

With (A.6) and (A.7), u^* , s^* and λ_1 in this case are

$$\begin{aligned} u^* &= u^{\text{ref}} - \lambda_1 H^{-1} L_g V^T \\ s^* &= \frac{\lambda_1}{p} \\ \lambda_1 &= \frac{p\mu V + pL_f V + pL_g V u^{\text{ref}}}{pL_g V H^{-1} L_g V^T + 1}. \end{aligned} \quad (\text{A.20})$$

Using the variables defined in (A.1), u^* can be rewritten as:

$$u^* = \begin{bmatrix} v_x^* \\ v_y^* \\ \omega^* \end{bmatrix} = \begin{bmatrix} v_x^{\text{ref}} - \frac{\lambda_1 a_x}{h_1} \\ v_y^{\text{ref}} - \frac{\lambda_1 a_y}{h_2} \\ \omega^{\text{ref}} - \frac{\lambda_1 a_\omega}{h_3} \end{bmatrix}. \quad (\text{A.21})$$

We know from (A.2) that

$$(a_y = 0 \ \& \ a_\omega = 0) \iff \delta = 0. \quad (\text{A.22})$$

In addition, we know from (4) that

$$(v_y^{\text{ref}} = 0 \ \& \ \omega^{\text{ref}} = 0) \iff \delta = 0. \quad (\text{A.23})$$

Therefore, we split this case into three cases based on the value of δ .

Appendix A.3.1. $\delta = 0$ (Case I)

Substituting $\delta = 0$ to (A.1), we have $a_x = -r < 0$, $a_y = 0$, $a_\omega = 0$, and to (4), we have $v_x^{\text{ref}} > 0$, $v_y^{\text{ref}} = 0$, $\omega^{\text{ref}} = 0$. Finally, with (A.12), the optimal control command (A.21) can be simplified as:

$$u^* = \begin{bmatrix} v_x^* \\ v_y^* \\ \omega^* \end{bmatrix} = \begin{bmatrix} v_x^{\text{ref}} + \frac{\lambda_1 r}{h_1} > 0 \\ 0 \\ 0 \end{bmatrix}. \quad (\text{A.24})$$

The optimal control command is not a zero vector, and hence there is no equilibrium point in this case.

Appendix A.3.2. $\delta > 0$ (Case II)

When $\delta > 0$, by the definitions in (A.2), we have $a_y < 0$, $a_\omega > 0$, and by (4), we have $v_y^{\text{ref}} > 0$, $\omega^{\text{ref}} < 0$. With (A.12) and (A.21), we have

$$\begin{aligned} v_y^* &= v_y^{\text{ref}} - \frac{\lambda_1 a_y}{h_2} > 0 \\ \omega^* &= \omega^{\text{ref}} - \frac{\lambda_1 a_\omega}{h_3} < 0. \end{aligned} \quad (\text{A.25})$$

The optimal control command is not a zero vector in this case. Therefore, there is no equilibrium points in this case either.

Appendix A.3.3. $\delta < 0$ (Case III)

Similarly, by (A.1) and (4), we have $a_y > 0, a_\omega < 0$ and $v_y^{\text{ref}} < 0, \omega^{\text{ref}} > 0$. With (A.12) and (A.21), we have

$$\begin{aligned} v_y^* &= v_y^{\text{ref}} - \frac{\lambda_1 a_y}{h_2} < 0 \\ \omega^* &= \omega^{\text{ref}} - \frac{\lambda_1 a_\omega}{h_3} > 0 \end{aligned} \quad (\text{A.26})$$

The optimal control command is not a zero vector in this case; hence, there is no equilibrium point in this case.

In summary, there is no equilibrium point when the CLF constraint is active and the CBF constraint is inactive.

Appendix A.4. Both CLF and CBF constraint are active

When the CLF constraint is active and the CBF constraint is active, we have

$$\begin{aligned} \lambda_1 &\geq 0 \\ \lambda_2 &\geq 0 \\ 0 &= L_f V + L_g V u^* + \mu V - s \\ 0 &= -L_f B - L_g B u^* - \eta B. \end{aligned} \quad (\text{A.27})$$

We can rewrite (A.6) and (A.7) as:

$$\begin{aligned} u^* &= u^{\text{ref}} - \lambda_1 H^{-1} L_g V^T + \lambda_2 H^{-1} L_g B^T \\ s^* &= \frac{\lambda_1}{p} \end{aligned} \quad (\text{A.28})$$

Using the variables defined in (A.1), u^* can be rewritten as:

$$u^* = \begin{bmatrix} v_x^* \\ v_y^* \\ \omega^* \end{bmatrix} = \begin{bmatrix} v_x^{\text{ref}} - \frac{\lambda_1 a_x}{h_1} - \frac{\lambda_2 d_x}{h_1} \\ v_y^{\text{ref}} - \frac{\lambda_1 a_y}{h_2} - \frac{\lambda_2 d_y}{h_2} \\ \omega^{\text{ref}} - \frac{\lambda_1 a_\omega}{h_3} \end{bmatrix}. \quad (\text{A.29})$$

When the robot is at an equilibrium point, u^* is the zero vector. By (A.27) and $L_f B = 0$, $u^* = 0$, we have $B = 0$, which implies that the robot is at the boundary of an obstacle. In the following proof of Sec. Appendix A.4, we will assume the robot is at the boundary of obstacles.

The property of $B = 0$ leads to an immediate proposition which is helpful in finding the equilibrium point in the system when one of the components of the optimal control is 0.

Proposition 3. $d_y = 0 \implies v_x^* = 0$.

Proof. By the proof in 3.3, we have $L_g B(x) = \nabla B(x) \cdot g(x) \neq 0$ for $x \in \mathcal{D}$. Therefore, when $d_y = 0$, we have $d_x \neq 0$. Then, we can further have $L_g B(x) u^* = 0 \implies v_x^* = 0$. $\square$

In addition, with the properties (A.22) and (A.23), we split this case into four cases based on whether δ and d_y are zero.

Appendix A.4.1. $d_y = \delta = 0$ (Case I)

Substituting to (A.1), we have $a_x = -r < 0, a_y = 0, a_\omega = 0$, and to (4), we have $v_x^{\text{ref}} > 0, v_y^{\text{ref}} = 0, \omega^{\text{ref}} = 0$. Finally, with Proposition 3, in this case the optimal control command (A.29) can be written as:

$$u^* = \begin{bmatrix} v_x^* \\ v_y^* \\ \omega^* \end{bmatrix} = \begin{bmatrix} v_x^{\text{ref}} - \frac{\lambda_1 a_x}{h_1} - \frac{\lambda_2 d_x}{h_1} \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}. \quad (\text{A.30})$$

λ_1 and λ_2 can be obtained by (A.30), (A.27) and (A.28):

$$\begin{aligned} \lambda_1 &= p\mu V > 0 \\ \lambda_2 &= \frac{h_1 v_x^{\text{ref}} - p\mu V a_x}{d_x} \end{aligned} \quad (\text{A.31})$$

By (A.12) and (A.31), we have

$$\because v_x^{\text{ref}} > 0, a_x < 0, \frac{h_1 v_x^{\text{ref}} - p\mu V a_x}{d_x} \geq 0 \implies d_x > 0. \quad (\text{A.32})$$

Hence, there is an equilibrium point when $B = 0, d_y = \delta = 0$ and $d_x > 0$.

Appendix A.4.2. $d_y \neq 0, \delta = 0$ (Case II)

When $\delta = 0$, by (A.1) and (4), we have $a_x = -r < 0, a_y = 0, a_\omega = 0$ and $v_x^{\text{ref}} > 0, v_y^{\text{ref}} = 0, \omega^{\text{ref}} = 0$. Finally, with (A.12), the optimal control command (A.29) can be simplified as:

$$u^* = \begin{bmatrix} v_x^* \\ v_y^* \\ \omega^* \end{bmatrix} = \begin{bmatrix} v_x^{\text{ref}} - \frac{\lambda_1 a_x}{h_1} - \frac{\lambda_2 d_x}{h_1} \\ -\frac{\lambda_2 d_y}{h_2} \neq 0 \\ 0 \end{bmatrix}. \quad (\text{A.33})$$

Because $v_y^* \neq 0$, the optimal command is not a zero vector in this case. Equilibrium points don't exist when $d_y \neq 0$ and $\delta = 0$.

Appendix A.4.3. $\delta > 0$ (Case III)

When $\delta > 0$, by (A.25), $\omega^* < 0$. Hence, the optimal command is not a zero vector and there are no equilibrium points in this case.

Appendix A.4.4. $\delta < 0$ (Case IV)

When $\delta < 0$, by (A.26), $\omega^* > 0$. Hence, the optimal command is not a zero vector and there are no equilibrium points in this case.