Polygonal Cone Control Barrier Functions (PolyC2BF) for safe navigation in cluttered environments

Manan Tayal, Shishir Kolathaya

***Abstract*— In fields such as mining, search and rescue, and archaeological exploration, ensuring real-time, collision-free navigation of robots in confined, cluttered environments is imperative. Despite the value of established path planning algorithms, they often face challenges in convergence rates and handling dynamic infeasibilities. Alternative techniques like collision cones struggle to accurately represent complex obstacle geometries. This paper introduces a novel category of control barrier functions, known as Polygonal Cone Control Barrier Function (PolyC2BF), which addresses overestimation and computational complexity issues. The proposed PolyC2BF, formulated as a Quadratic Programming (QP) problem, proves effective in facilitating collision-free movement of multiple robots in complex environments. The efficacy of this approach is further demonstrated through PyBullet simulations on quadruped (unicycle model), and crazyflie 2.1 (quadrotor model) in cluttered environments.**

I. Introduction

Ensuring real-time collision-free navigation of robots within confined, cluttered environments is of paramount importance in critical fields such as mining, search and rescue operations in structurally compromised buildings, and archaeological exploration of sites featuring narrow passageways or constrained spaces. It is imperative to uphold the highest standards of safety and task efficiency in these scenarios. While established optimal path planning algorithms like A* [1], RRT* [2], and PRM [3] offer valuable insights, they grapple with slow convergence rates, rendering them less effective in promptly identifying optimal paths, especially in intricate environments. Additionally, they do not address the dynamic infeasibility challenges encountered during the execution of paths generated by higher-level planners. Approaches like kinodynamic RRT* [4] and LQR-RRT* [5] play a pivotal role in determining appropriate steering inputs for seamless transitions between vertices within the sampling graph. Nevertheless, these techniques fall short in their ability to conduct dynamic collision assessments that accurately accommodate the nonlinear dynamics of the robot.

Alternative collision avoidance techniques like collision cones [6]–[8] are not without notable limitations. Within cluttered environments, the assumption of circular obstacles can result in imprecise collision predictions and sub-optimal path choices due to this oversimplified representation of obstacles. This deficiency is even apparent in scenarios where the robot is endeavoring to evade collision with an extended wall, potentially leading to the incorrect perception that the robot is already within the obstacle, ultimately resulting in a failure state. This phenomenon is visually exemplified in Fig. 1. This drawback highlights the importance of more sophisticated collision avoidance methods that can accurately account for the complex geometry of obstacles in the environment.

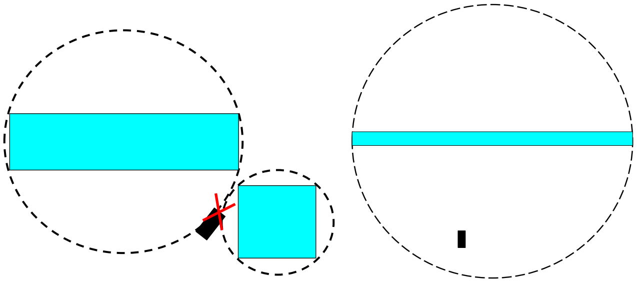


Fig. 1: Disadvantages of Collision Cones: Collision Avoidance in cluttered environment (left) and against a long wall (right)

The Collision Cone Control Barrier Functions [9]–[11], an integration of Collision Cones and Control Barrier Functions (CBFs) [12] [13] principles also confront the challenges observed in collision cones. CBFs are instrumental in establishing formal safety guarantees, particularly in the context of robust collision avoidance with obstacles. These functions delineate a secure state set through inequality constraints and frame it as a quadratic programming (QP) problem to ensure the ongoing invariance of these sets over time.

A notable advantage of employing CBF based QP, when compared to other contemporary techniques like model predictive control (MPC) [14], reference governor [15], reachability analysis [16], and artificial potential fields [17], lies in their efficacy in real-time applications within complex dynamic environments [18], [19]. This translates to the ability to compute optimal control inputs at a notably high frequency using off-the-shelf electronics. Furthermore, CBFs can serve as a rapid safety filter when overlaid onto existing path planning controllers [20].

In scenarios requiring precise obstacle avoidance with minimal clearance, it is imperative to model obstacles as polyhedra. [21], [22] This work accomplishes real-time collision avoidance by leveraging convex and non-convex optimizations in conjunction with control barrier functions. Nonetheless, it is worth noting that optimization-based methodologies typically encounter a significant rise in computational complexity as the number of robots involved increases. This scalability challenge can prove demanding

Manan is supported by the Prime Minister Research Fellowship (PMRF). The research was supported by the SERB grant CRG/2021/008115.

[1]Robert Bosch Center for Cyber Physical Systems (RBCCPS), Indian Institute of Science (IISc), Bengaluru. {manantayal, shishirk}@iisc.ac.in

when implementing such approaches in real-time for distributed multi-robot systems.

Hence, this paper presents a novel category of control barrier functions referred to as Polygonal Cone Control Barrier Function (PolyC2BF). This advancement builds upon the existing Collision Cone Control Barrier Function (C3BF) and rectifies the issue of overestimation of obstacle and robot dimensions when represented as circles. Additionally, the proposed PolyC2BF mitigates the computational complexity associated with optimization-based approaches. It is demonstrated to facilitate collision-free movement of multiple robots in complex, confined environments.

Main contributions of our work are as follows:

- We formulate a direct method for safe trajectory tracking of non-holonomic ground and aerial vehicles based on polygonal cone control barrier functions expressed through a quadratic program to traverse in cluttered environment and tight spaces.
- We verify the polygonal cone CBF (PolyC2BF) based safety filter in simulations and compare it with the collision cone CBF (C3BF), to show how the former is better in terms of safe navigation in cluttered environment.

II. Background

In this section, we will introduce the necessary definitions and the relevant background for Control Barrier Functions (CBFs), the collision cone control barrier function and CBF-based safety filters.

Consider a nonlinear control system in affine form:

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

where $x \in \mathcal{D} \subseteq \mathbb{R}^n$ is the state of system, and $u \in \mathbb{U} \subseteq \mathbb{R}^m$ the input for the system. Assume that the functions $f : \mathbb{R}^n \to \mathbb{R}^n$ and $g : \mathbb{R}^n \to \mathbb{R}^{n \times m}$ are locally Lipschitz continuous functions.

The safe set $\mathcal{C}$ is defined as the *super-level set* of a continuously differentiable function $h : \mathcal{D} \subseteq \mathbb{R}^n \to \mathbb{R}$: $\mathcal{C} = \{x \in \mathcal{D} : h(x) \geq 0\}$ where the boundary of the safe set is $\partial\mathcal{C} = \{x \in \mathcal{D} : h(x) = 0\}$ with $\frac{\partial h}{\partial x}(x) \neq 0 \; \forall x \in \partial\mathcal{C}$ and the interior of the safe set is $\text{Int}\,(\mathcal{C}) = \{x \in \mathcal{D} \subset \mathbb{R}^n : h(x) > 0\}$. It is assumed that $\text{Int}\,(\mathcal{C})$ is non-empty and $\mathcal{C}$ has no isolated points, i.e. $\text{Int}\,(\mathcal{C}) \neq \phi$ and $\overline{\text{Int}\,(\mathcal{C})} = \mathcal{C}$. The goal is to minimally modify a given control policy $\pi(x, t)$ with a safety based on a continuous time-CBF filter such that the system is safe (i.e. the system's state x stays in safe set $\mathcal{C}$ if it starts inside $\mathcal{C}$).

Definition 1 (Extended class-$\mathcal{K}$ functions): A continuous function $\kappa : [0, d) \to [0, \infty)$ for some $d > 0$ is said to belong to class-$\mathcal{K}$ if it is strictly increasing and $\kappa(0) = 0$. Here, d is allowed to be ∞. The same function can be extended to the interval $\kappa : (-b, d) \to (-\infty, \infty)$ with $b > 0$ (which is also allowed to be ∞), in which case we call it the extended class $\mathcal{K}$ function.

Definition 2 (Control barrier function (CBF)): Given the set $\mathcal{C} \subset \mathcal{D}$ be the superlevel set of a continuously differentiable function $h : \mathcal{D} \subseteq \mathbb{R}^n \to \mathbb{R}$ with $\frac{\partial h}{\partial x}(x) \neq 0 \; \forall x \in \partial\mathcal{C}$, the function h is called the control barrier function (CBF) defined on the set $\mathcal{D}$, if there exists an extended class $\mathcal{K}$ function κ such that for all $x \in \mathcal{D}$:

$$\underbrace{sup}_{u \in \mathbb{U}} \left[\underbrace{\mathcal{L}_f h(x) + \mathcal{L}_g h(x) u}_{\dot{h}(x,u)} + \kappa\left(h(x)\right) \right] \geq 0 \tag{2}$$

where $\mathcal{L}_f h(x) = \frac{\partial h}{\partial x} f(x)$ and $\mathcal{L}_g h(x) = \frac{\partial h}{\partial x} g(x)$ are the Lie derivatives.

Given this definition of a CBF, we know from [13] and [23] that any Lipschitz continuous control law $k(x)$ satisfying the inequality: $\dot{h} + \kappa(h) \geq 0$ ensures safety of $\mathcal{C}$ if $x(0) \in \mathcal{C}$, and asymptotic convergence to $\mathcal{C}$ if $x(0)$ is outside of $\mathcal{C}$.

Definition 3 (Collision Cone CBFs [9], [10]): This approach combines the idea of potential unsafe directions given by collision cone as unsafe set to formulate a CBF. Consider the following CBF candidate:

$$h(x, t) = < p_{\text{rel}}, v_{\text{rel}} > + \|p_{\text{rel}}\| \|v_{\text{rel}}\| \cos\phi, \tag{3}$$

where p_{rel} is the relative position vector between the body center of the ego vehicle and the center of the obstacle, v_{rel} is the relative velocity, $< \cdot, \cdot >$ is the dot product of 2 vectors and ϕ is the half angle of the cone, the expression of $\cos\phi$ is given by $\frac{\sqrt{\|p_{\text{rel}}\|^2 - r^2}}{\|p_{\text{rel}}\|}$. The proposed constraint simply ensures that the angle between $p_{\text{rel}}, v_{\text{rel}}$ is less than $180^\circ - \phi$.

Precise mathematical definitions for $p_{\text{rel}}, v_{\text{rel}}$ will be given in the next section.

Using the CBF conditions, we can define an input set:

$$\mathbb{U}_{cbf}(x) = \{u \in \mathbb{U} \subseteq \mathbb{R}^m : \dot{h} + \kappa\left(h(x)\right) \geq 0\} \tag{4}$$

that renders the system safe. This requires the selection of an extended class-$\mathcal{K}$ function κ that yields a non-empty $\mathbb{U}_{cbf}(x)$ for all $x \in \mathcal{C}$. For a given control policy $\pi(x, t)$ that may not be safe, one can formulate a quadratic program (QP) which satisfy the CBF condition (2) to modify this input in a minimal way:

$$\begin{aligned} u^*(x, t) &= \underset{u \in \mathbb{U} \subseteq \mathbb{R}^m}{\arg\min} \|u - \pi(x, t)\|^2 \\ &\text{s.t. } \mathcal{L}_f h(x) + \mathcal{L}_g h(x) u + \kappa\left(h(x)\right) \geq 0 \end{aligned} \tag{5}$$

This is called the Control Barrier Function based Quadratic Program (CBF-QP). If $\mathbb{U} = \mathbb{R}^m$, then the QP is feasible, and the explicit solution is given by

$$u^*(x, t) = \pi(x, t) + u_{safe}(x, t)$$

where $u_{safe}(x, t)$ is given by

$$u_{safe}(x, t) = \begin{cases} 0 & \text{for } \psi(x, t) \geq 0 \\ -\frac{\mathcal{L}_g h(x)^T \psi(x,t)}{\mathcal{L}_g h(x) \mathcal{L}_g h(x)^T} & \text{for } \psi(x, t) < 0 \end{cases} \tag{6}$$

where $\psi(x, t) := \dot{h}\left(x, \pi(x, t)\right) + \kappa\left(h(x)\right)$. The sign change of ψ yields a switching type of a control law.

III. Polygonal Cone Control Barrier Functions (PolyC2BF)

Assuming the polygonal obstacles as circle/ellipses is not a great idea as shown in Fig 1, especially in cases of very long walls or cluttered environment. However, instead if we consider the vertices/edges of the polygon to construct a cone to be avoided, we may be able to resolve the issue. Fig 2 shows the construction of Polygonal Collision Cones for long wall, semi-infinite wall and cluttered environments.

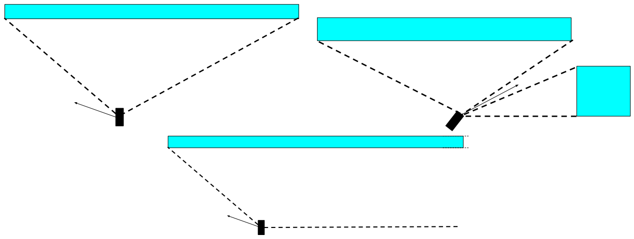


Fig. 2: Polygonal Collision Cone using vertices of the polygonal obstacles

A. Polygonal Cone CBF (PolyC2BF)

Given the outlined limitations associated with assuming obstacles as circles or ellipses, particularly for polygonal obstacles, and the methodology for constructing a cone around a polygonal obstacle, we propose integrating these concepts with Control Barrier Functions to formulate a real-time safe controller tailored for these scenarios.

For polygonal obstacles, the cone can be directly constructed using their vertices. These vertices are selected to ensure comprehensive coverage of the entire polygon. The relative position vector (p_{rel}) is oriented towards the midpoint of the line connecting these two vertices. Let's denote the vector that forms the most substantial angle with p_{rel} as k', and the other one as m'. To account for the width/shape of the ego vehicle (w) [1], the line joining these two vertices is extended by $\frac{w}{2}$ on both sides, forming vectors k and m (corresponding to vectors k' and m', respectively) as depicted in Fig 3.

Remark 1: The vector connecting the vertex which is closer to the ego vehicle, forms the bigger angle with p_{rel}. *Thus, in the analysis shown next, we will try to avoid a cone with half angle* ϕ, *is the angle between vectors* k *(vector connecting the closer vertex) and* p_{rel}.

This novel approach of avoiding the pink cone region using Control Barrier Functions, gives rise to **Polygonal Cone Control Barrier Functions (PolyC2BFs)**. The proposed CBF candidate is as follows:

$$h(x,t) = < p_{\text{rel}}, v_{\text{rel}} > + \|p_{\text{rel}}\| \|v_{\text{rel}}\| \cos\phi \tag{7}$$

where, the expression of $\cos\phi$ is given by $\frac{<p_{\text{rel}}, k>}{\|p_{\text{rel}}\|\|k\|}$.

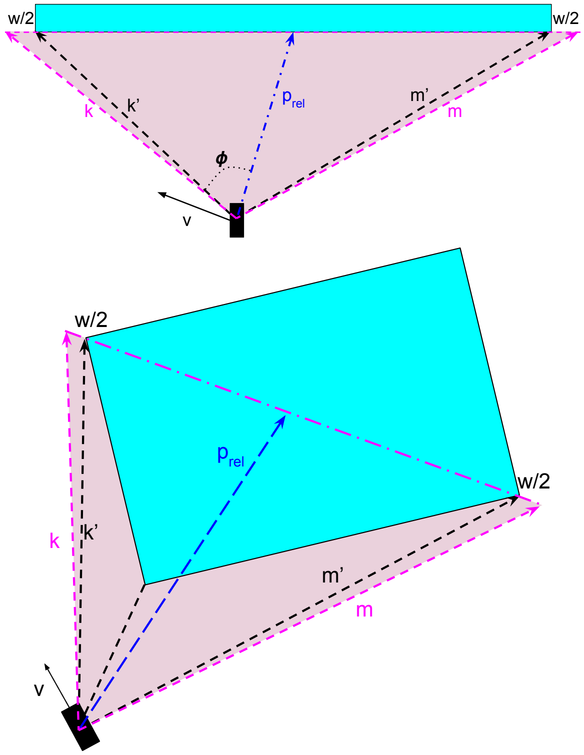


Fig. 3: Construction of Polygonal Cone to avoid polygonal obstacle

We now consider an acceleration controlled non-holonomic ground vehicle (unicycle model) described by

$$\underbrace{\begin{bmatrix} \dot{x}_p \\ \dot{y}_p \\ \dot{\theta} \\ \dot{v} \\ \dot{\omega} \end{bmatrix}}_{\dot{x}} = \underbrace{\begin{bmatrix} v\cos\theta \\ v\sin\theta \\ \omega \\ 0 \\ 0 \end{bmatrix}}_{f(x)} + \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}}_{g(x)} \underbrace{\begin{bmatrix} a \\ \alpha \end{bmatrix}}_{u} \tag{8}$$

where state variables denoted as x_p, y_p, θ, v, and ω, representing pose, linear velocity, and angular velocity, respectively. Linear acceleration (a) and angular acceleration (α) serve as the control inputs.

We first obtain the relative position vector between the body center of the vehicle and the center of the obstacle. Therefore, we have

$$p_{\text{rel}} := \begin{bmatrix} c_x - (x_p + l\cos(\theta)) \\ c_y - (y_p + l\sin(\theta)) \end{bmatrix} \tag{9}$$

Here l is the distance of the body center from the differential drive axis. c_x, c_y represents the center of obstacle as a function of time. Also, since the obstacles are of constant velocity, we have $\ddot{c}_x = \ddot{c}_y = 0$. We obtain its velocity as

$$v_{\text{rel}} := \begin{bmatrix} \dot{c}_x - (v\cos(\theta) - l\sin(\theta) * \omega) \\ \dot{c}_y - (v\sin(\theta) + l\cos(\theta) * \omega) \end{bmatrix}. \tag{10}$$

We have the following first result of the paper:

[1] w can be predetermined (e.g. the diagonal of the vehicle) or can be dynamically adjusted (e.g. w may change as a function of the orientation of the ego-vehicle).

Theorem 1: The proposed CBF candidate (7) *with* $p_{\text{rel}}, v_{\text{rel}}$ *defined by* (9), (10) *is a valid CBF defined for the set* $\mathcal{D}$.

Proof: Taking the derivative of (7) yields

$$\begin{aligned}\dot{h} &= < \dot{p}_{\text{rel}}, v_{\text{rel}} > + < p_{\text{rel}}, \dot{v}_{\text{rel}} > \\ &\quad + < v_{\text{rel}}, \dot{v}_{\text{rel}} > \frac{< p_{\text{rel}}, k >}{\|v_{\text{rel}}\| \|k\|} \\ &\quad + < \dot{p}_{\text{rel}}, \hat{k} > \|v_{\text{rel}}\| + < p_{\text{rel}}, \dot{\hat{k}} > \|v_{\text{rel}}\|. \end{aligned} \tag{11}$$

where, $\hat{k}$ means $k/\|k\|$. Further $\dot{p}_{\text{rel}} = v_{\text{rel}}$ and

$$\dot{v}_{\text{rel}} = \begin{bmatrix} -a\cos\theta + v(\sin\theta)\omega + l(\cos\theta)\omega^2 + l(\sin\theta)\alpha \\ -a\sin\theta - v(\cos\theta)\omega + l(\sin\theta)\omega^2 - l(\cos\theta)\alpha \end{bmatrix}.$$

Given $\dot{v}_{\text{rel}}$ and $\dot{h}$, we have the following expression for $\mathcal{L}_g h$:

$$\mathcal{L}_g h = \begin{bmatrix} < p_{\text{rel}} + v_{\text{rel}} \frac{< p_{\text{rel}}, \hat{k} >}{\|v_{\text{rel}}\|}, \begin{bmatrix} -\cos\theta \\ -\sin\theta \end{bmatrix} > \\ < p_{\text{rel}} + v_{\text{rel}} \frac{< p_{\text{rel}}, \hat{k} >}{\|v_{\text{rel}}\|}, \begin{bmatrix} l\sin\theta \\ -l\cos\theta \end{bmatrix} > \end{bmatrix}^T, \tag{12}$$

By employing analogous reasoning as demonstrated in the proof of [9, Theorem 1], it can be ascertained that $\mathcal{L}_g h$ is always a non-zero matrix. Thus, in accordance with [24, Theorem 8], we can deduce that the resultant Quadratic Programming problem (QP), as defined in (6), exhibits Lipschitz continuity. Consequently, we can formulate CBF-QPs utilizing the proposed CBF, as expressed in (7), to ensure collision avoidance. ■

Next, let's consider the aerial vehicle (quadrotor model). In this case we will have to consider construction of polygonal cones in both horizontal and vertical planes thus forming vectors k_h and k_v for horizontal and vertical planes respectively, as shown in Fig 4. In case, $\|k_h\| < \|k_v\|$, we will consider $k = k_h$ and $p_{\text{rel}} = (p_{\text{rel}})_h$, similarly, if $\|k_h\| > \|k_v\|$, we will consider $k = k_v$ and $p_{\text{rel}} = (p_{\text{rel}})_v$. This will be similar to the projection case in [10].

A quadrotor has four propellers, which provides upward thrusts of (f_1, f_2, f_3, f_4) and act as inputs. The states needed to describe the quadrotor system is given by $x = [x_p, y_p, z_p, \dot{x}_p, \dot{y}_p, \dot{z}_p, \phi, \theta, \psi, \omega_1, \omega_2, \omega_3]$. The quadrotor dynamics is as follows:

$$\underbrace{\begin{bmatrix} \dot{x}_p \\ \dot{y}_p \\ \dot{z}_p \\ \ddot{x}_p \\ \ddot{y}_p \\ \ddot{z}_p \\ \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \\ \dot{\omega}_1 \\ \dot{\omega}_2 \\ \dot{\omega}_3 \end{bmatrix}}_{\dot{x}} = \underbrace{\begin{bmatrix} \dot{x}_p \\ \dot{y}_p \\ \dot{z}_p \\ 0 \\ 0 \\ -g \\ \mathbf{W}^{-1} \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix} \\ -I^{-1}\vec{\omega} \times I\vec{\omega} \\ . \end{bmatrix}}_{f(x)} + \underbrace{\begin{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \\ \frac{1}{m_Q}\mathbf{R} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} \\ \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \\ I^{-1}L \begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \\ c_\tau & -c_\tau & c_\tau & -c_\tau \end{bmatrix} \end{bmatrix}}_{g(x)} \underbrace{\begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix}}_{u} \tag{13}$$

x_p, y_p and z_p denote the coordinates of the vehicle's centre of the base of the quadrotor in an inertial frame. ϕ, θ and ψ represents the (roll, pitch & yaw) orientation of the quadrotor. **R** is the rotation matrix (from the body frame to the inertial frame), m_Q is the mass of the quadrotor, **W** is the transformation matrix for angular velocities from the inertial frame to the body frame, I is the inertia matrix and L is the diagonal length of quadrotor. c_τ is the constant that determines the torque produced by each propeller.

We have to obtain the relative position vector between the body center of the quadrotor and the intersection of the axis of the obstacle and the projection plane. Therefore, we have

$$p_{\text{rel}} := \mathcal{P}\left(\begin{bmatrix} c_x \\ c_y \\ c_z \end{bmatrix} - \left(\begin{bmatrix} x_p \\ y_p \\ z_p \end{bmatrix} + \mathbf{R} \begin{bmatrix} 0 \\ 0 \\ l \end{bmatrix} \right) \right). \tag{14}$$

Here l is the distance of the body center from the base. $\mathcal{P} : \mathbb{R}^3 \to \mathbb{R}^3$ is the projection operator, which can be assumed to be a constant[2]. c_x, c_y, c_z represents the center of obstacle as a function of time. Also, since the obstacles are of constant velocity, we have $\ddot{c}_x = \ddot{c}_y = \ddot{c}_z = 0$. We obtain its relative velocity as

$$v_{\text{rel}} := \frac{d(p_{rel})}{dt} = \dot{p}_{\text{rel}} \tag{15}$$

Now, we calculate the $\dot{v}_{\text{rel}}$ term which contains our inputs i.e. (f_1, f_2, f_3, f_4), as follows:

$$\frac{d}{dt}(v_{\text{rel}}) = \mathcal{P}(-\mathbf{R} \begin{bmatrix} 0 & \frac{Ll}{I_{yy}} & 0 & \frac{-Ll}{I_{yy}} \\ \frac{-Ll}{I_{xx}} & 0 & \frac{Ll}{I_{xx}} & 0 \\ \frac{1}{m_Q} & \frac{1}{m_Q} & \frac{1}{m_Q} & \frac{1}{m_Q} \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix} \\ + \text{additional terms}). \tag{16}$$

Please note that the additional terms in the above equation, refer to those terms that do not contain the input terms (f_1, f_2, f_3, f_4) and thus do not contribute to the calculation of $\mathcal{L}_g h$. We have the following result with properties of p_{rel} and v_{rel} similar to those in the projection case of [10].

Theorem 2: Given the quadrotor model (13)*, the proposed CBF candidate* (7) *with* $p_{\text{rel}}, v_{\text{rel}}$ *defined by* (14), (15) *is a valid CBF defined for the set* $\mathcal{D}$.

Proof: The derivative of (7) is given by (11).

By substituting for $\dot{v}_{\text{rel}}$ (16) (only the term which contains the input) in $\dot{h}$ (11), we have the following expression for

[2] Note that the obstacles are always translating and not rotating. In addition, it is not restrictive to assume that the translation direction is always perpendicular to the cylinder axis. This makes the projection operator a constant.

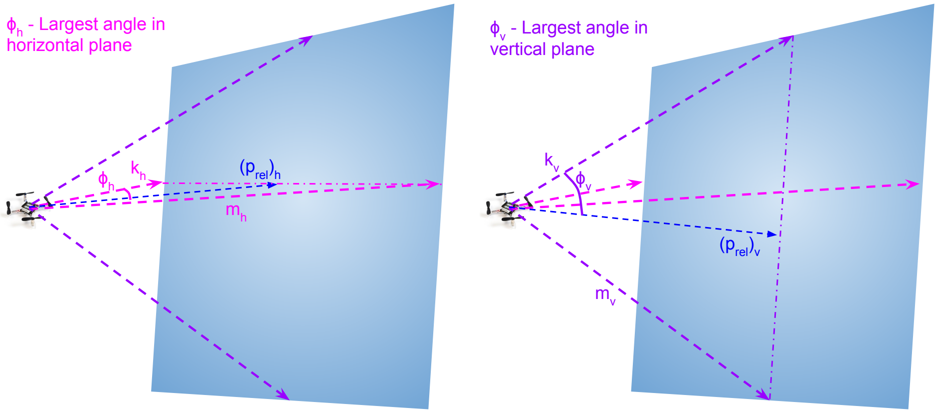

ϕ_h - Largest angle in horizontal plane
k_h
ϕ_h
(p_rel)_h
m_h
ϕ_v - Largest angle in vertical plane
k_v
ϕ_v
(p_rel)_v
m_v

Fig. 4: Construction of Polygonal Cone for an Aerial Vehicle

$\mathcal{L}_g h$:

$$\mathcal{L}_g h = \begin{bmatrix} < p_{\text{rel}} + v_{\text{rel}} \frac{<p_{\text{rel}}, \hat{k}>}{\|v_{\text{rel}}\|}, \mathbf{R} \begin{bmatrix} 0 \\ \frac{-Ll}{I_{xx}} \\ \frac{1}{m_Q} \end{bmatrix} > \\ < p_{\text{rel}} + v_{\text{rel}} \frac{<p_{\text{rel}}, \hat{k}>}{\|v_{\text{rel}}\|}, \mathbf{R} \begin{bmatrix} \frac{Ll}{I_{yy}} \\ 0 \\ \frac{1}{m_Q} \end{bmatrix} > \\ < p_{\text{rel}} + v_{\text{rel}} \frac{<p_{\text{rel}}, \hat{k}>}{\|v_{\text{rel}}\|}, \mathbf{R} \begin{bmatrix} 0 \\ \frac{Ll}{I_{xx}} \\ \frac{1}{m_Q} \end{bmatrix} > \\ < p_{\text{rel}} + v_{\text{rel}} \frac{<p_{\text{rel}}, \hat{k}>}{\|v_{\text{rel}}\|}, \mathbf{R} \begin{bmatrix} \frac{-Ll}{I_{yy}} \\ 0 \\ \frac{1}{m_Q} \end{bmatrix} > \end{bmatrix}^T, \qquad (17)$$

By employing analogous reasoning as demonstrated in the proof of [10, Theorem 1], it can be ascertained that $\mathcal{L}_g h$ is always a non-zero matrix, implying that it is a valid CBF. Consequently, QP formulation of PolyC2BF as expressed in (7), ensures collision avoidance. ■

Consider the point mass model, characterized as follows:

$$\begin{bmatrix} \dot{p} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} 0_{2x2} & I_{2x2} \\ 0_{2x2} & 0_{2x2} \end{bmatrix} \begin{bmatrix} p \\ v \end{bmatrix} + \begin{bmatrix} 0_{2x2} \\ I_{2x2} \end{bmatrix} u, \qquad (18)$$

where $p = [x_p, \ y_p]^T$, $v = [v_x, \ v_y]^T$, and $u = [a_x, \ a_y]^T \in \mathbb{R}^2$ represent the position, velocity, and acceleration inputs, respectively. Analogous to Theorem 1, similar results can be derived for this simplified point mass model. It is worth noting that the proposed PolyC2BF-QP stands as a valid Control Barrier Function (CBF), and its proof is straightforward.

IV. Simulations

PolyC2BF-QP based controller has been validated on both quadruped (which can be assumed as an accelerated non holonomic vehicle (unicycle model) as in [11], [25]) and quadrotor and the simulations were conducted on PyBullet [26], a python-based physics simulation engine. The Simulation runs on a computer with Ubuntu 22.04 with Intel i7-12800 HX CPU, with 16GB RAM and Nvidia 3070Ti GPU.

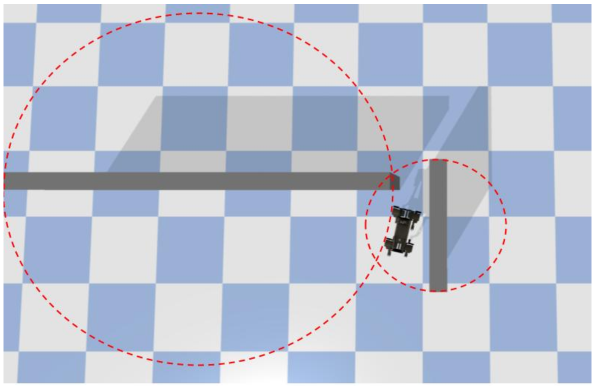


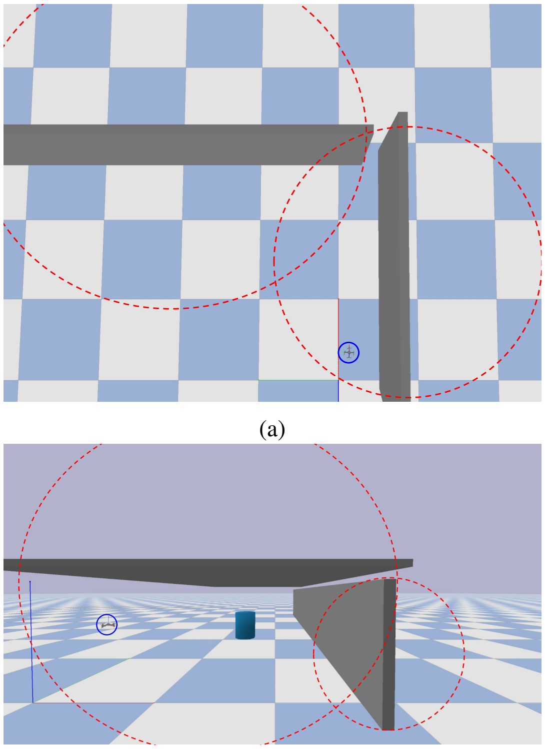
Fig. 5: Navigation in a cluttered environment using PolyC2BF in case of Quadruped

For both the cases, i.e. quadruped and drone, we have considered the reference control inputs as a simple PD controller for simulating which can be replaced with any state-of-the-art path planning/ trajectory tracking controller. Constant target velocities were chosen to verify the PolyC2BF-QP. Here, we select the class-$\mathcal{K}$ function $\kappa(h) = \gamma h$ as the identity map i.e. $\gamma = 1$. In all the experiments, the position and the velocity of the obstacles are assumed to be measured and always available. Fig. 5 and Fig. 6 (a) and (b) shows the quadruped and drone moving in a cluttered environment in front of a very long wall, which would not have been possible with collision cone control barrier function. The red dotted circles in the figures represent the virtual obstacles considered by the collision cone control barrier function (C3BF); thus, employing the C3BF implies that the ego-vehicle's initial position lies within these virtual obstacles, rendering it ineffective in such scenarios. All the simulations experiments can be viewed in the attached webpage[3]

V. Conclusions

We presented a novel CBF formulation for collision avoidance in cluttered environments and tight spaces. Unlike existing works, the proposed Polygonal Collision Cone based

[3].https://tayalmanan28.github.io/PolyC2BF/



(a)

(b)

Fig. 6: Navigation in a cluttered environment using PolyC2BF in case of Quadrotor in 2 different arrangements

QP formulation (PolyC3BF-QP) allows the vehicle to safely maneuver cluttered envioronments and tight spaces, in real time without any complexity of optimization. This is demonstrated in PyBullet simulations of Quadruped and Quadrotor. Future work includes considering more complex 3D shapes and cases for quadrotors and hardware implementation.

References

[1] F. Duchoň, A. Babinec, M. Kajan, P. Beňo, M. Florek, T. Fico, and L. Jurišica, "Path planning with modified a star algorithm for a mobile robot," *Procedia Engineering*, vol. 96, pp. 59–69, 2014, modelling of Mechanical and Mechatronic Systems. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S187770581403149X

[2] F. Islam, J. Nasir, U. Malik, Y. Ayaz, and O. Hasan, "Rrt star-smart: Rapid convergence implementation of rrt star towards optimal solution," in *2012 IEEE International Conference on Mechatronics and Automation*, 2012, pp. 1651–1656.

[3] R. Bohlin and L. Kavraki, "Path planning using lazy prm," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 1, 2000, pp. 521–528 vol.1.

[4] D. J. Webb and J. van den Berg, "Kinodynamic rrt*: Asymptotically optimal motion planning for robots with linear dynamics," in *2013 IEEE International Conference on Robotics and Automation*, 2013, pp. 5054–5061.

[5] A. Perez, R. Platt, G. Konidaris, L. Kaelbling, and T. Lozano-Perez, "Lqr-rrt*: Optimal sampling-based motion planning with automatically derived extension heuristics," in *2012 IEEE International Conference on Robotics and Automation*, 2012, pp. 2537–2542.

[6] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using the relative velocity paradigm," in *[1993] Proceedings IEEE International Conference on Robotics and Automation*, 1993, pp. 560–565 vol.1.

[7] ——, "Motion planning in dynamic environments using velocity obstacles," *The International Journal of Robotics Research*, vol. 17, no. 7, pp. 760–772, 1998. [Online]. Available: https://doi.org/10.1177/027836499801700706

[8] A. Chakravarthy and D. Ghose, "Obstacle avoidance in a dynamic environment: a collision cone approach," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 28, no. 5, pp. 562–574, 1998.

[9] P. Thontepu, B. G. Goswami, M. Tayal, N. Singh, S. S. P I, S. S. M G, S. Sundaram, V. Katewa, and S. Kolathaya, "Collision cone control barrier functions for kinematic obstacle avoidance in ugvs," in *2023 Ninth Indian Control Conference (ICC)*, 2023, pp. 293–298.

[10] M. Tayal, R. Singh, J. Keshavan, and S. Kolathaya, "Control barrier functions in dynamic uavs for kinematic obstacle avoidance: A collision cone approach," *arXiv preprint arXiv:2303.15871*, 2023.

[11] M. Tayal and S. Kolathaya, "Safe legged locomotion using collision cone control barrier functions (c3bfs)," *arXiv preprint arXiv:2309.01898*, 2023.

[12] A. D. Ames, J. W. Grizzle, and 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.

[13] 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, aug 2017. [Online]. Available: https://doi.org/10.1109%2Ftac.2016.2638961

[14] N. Piccinelli, F. Vesentini, and R. Muradore, "Mpc based motion planning for mobile robots using velocity obstacle paradigm," in *2023 European Control Conference (ECC)*, 2023, pp. 1–6.

[15] I. Kolmanovsky, E. Garone, and S. Di Cairano, "Reference and command governors: A tutorial on their theory and automotive applications," in *2014 American Control Conference*, 2014, pp. 226–241.

[16] S. Bansal, M. Chen, S. Herbert, and C. J. Tomlin, "Hamilton-jacobi reachability: A brief overview and recent advances," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, 2017, pp. 2242–2253.

[17] Y. Hwang and N. Ahuja, "Path planning using a potential field representation," in *Proceedings. 1988 IEEE International Conference on Robotics and Automation*, 1988, pp. 648–649 vol.1.

[18] A. W. Singletary, K. Klingebiel, J. R. Bourne, N. A. Browning, P. T. Tokumaru, and A. D. Ames, "Comparative analysis of control barrier functions and artificial potential fields for obstacle avoidance," *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 8129–8136, 2021.

[19] Z. Li, "Comparison between safety methods control barrier function vs. reachability analysis," 2021. [Online]. Available: https://arxiv.org/abs/2106.13176

[20] A. Manjunath and Q. Nguyen, "Safe and robust motion planning for dynamic robotics via control barrier functions," in *2021 60th IEEE Conference on Decision and Control (CDC)*, 2021, pp. 2122–2128.

[21] A. Thirugnanam, J. Zeng, and K. Sreenath, "Safety-critical control and planning for obstacle avoidance between polytopes with control barrier functions," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 286–292.

[22] ——, "Duality-based convex optimization for real-time obstacle avoidance between polytopes with control barrier functions," in *2022 American Control Conference (ACC)*, 2022, pp. 2239–2246.

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

[24] X. Xu, P. Tabuada, J. W. Grizzle, and A. D. Ames, "Robustness of control barrier functions for safety critical control." *IFAC-PapersOnLine*, vol. 48, no. 27, pp. 54–61, 2015, analysis and Design of Hybrid Systems ADHS. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896315024106

[25] T. G. Molnar, R. K. Cosner, A. W. Singletary, W. Ubellacker, and A. D. Ames, "Model-free safety-critical control for robotic systems," *IEEE robotics and automation letters*, vol. 7, no. 2, pp. 944–951, 2021.

[26] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," http://pybullet.org, 2016–2019.