Safety-Critical Control and Planning for Obstacle Avoidance between Polytopes with Control Barrier Functions

Akshay Thirugnanam*, Jun Zeng*, and Koushil Sreenath

***Abstract*— Obstacle avoidance between polytopes is a challenging topic for optimal control and optimization-based trajectory planning problems. Existing work either solves this problem through mixed-integer optimization, relying on simplification of system dynamics, or through model predictive control with dual variables using distance constraints, requiring long horizons for obstacle avoidance. In either case, the solution can only be applied as an offline planning algorithm. In this paper, we exploit the property that a smaller horizon is sufficient for obstacle avoidance by using discrete-time control barrier function (DCBF) constraints and we propose a novel optimization formulation with dual variables based on DCBFs to generate a collision-free dynamically-feasible trajectory. The proposed optimization formulation has lower computational complexity compared to existing work and can be used as a fast online algorithm for control and planning for general nonlinear dynamical systems. We validate our algorithm on different robot shapes using numerical simulations with a kinematic bicycle model, resulting in successful navigation through maze environments with polytopic obstacles.**

I. Introduction

Obstacle avoidance in optimization-based control and trajectory planning has received significant attention in the robotics community. When a tight-fitting obstacle avoidance motion is expected, the robot and the obstacles need to be considered as polyhedral. In this paper, we propose an optimization formulation to consider obstacle avoidance between polytopes using discrete-time control barrier function (DCBF) constraints with dual variables. The proposed formulation is shown to be a computationally fast algorithm that can serve as a local planner to generate dynamically-feasible and collision-free trajectories, or even directly as a safety-critical controller for general dynamical systems.

A. Related Work

1) Graph Search-based and Sampling-based Approaches: Motion planning techniques in real-world applications often consider high-level path planning and low-level control synthesis, given safety requirements and dynamical constraints. Graph search-based and sampling-based approaches such as PRM [1], A* [2], RRT* [3] have been explored, and many variant approaches have also been proposed based on

*Authors have contributed equally and are listed alphabetically.

This work is supported in part by National Science Foundation Grant CMMI-1931853.

All authors are with Hybrid Robotics Group at the Department of Mechanical Engineering, UC Berkeley, USA. {akshay_t, zengjunsjtu, koushils}@berkeley.edu

The animation video can be found at https://youtu.be/2hKlihdERog and the implementation code can be found at https://github.com/HybridRobotics/cbf.

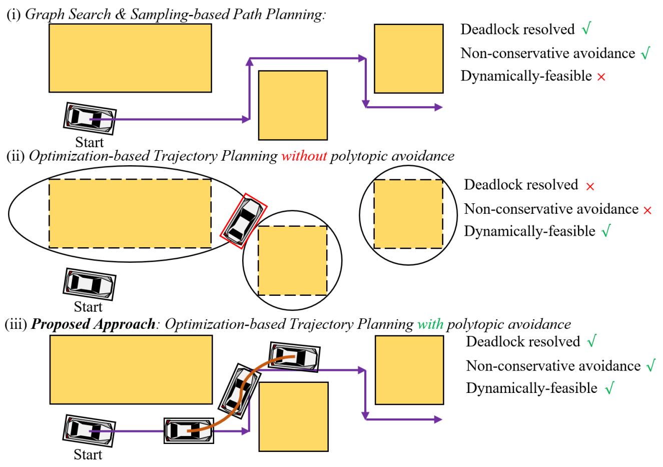

(i) Graph Search & Sampling-based Path Planning:
Start
Deadlock resolved ✓
Non-conservative avoidance ✓
Dynamically-feasible ×
(ii) Optimization-based Trajectory Planning without polytopic avoidance
Start
Deadlock resolved ×
Non-conservative avoidance ×
Dynamically-feasible ✓
(iii) Proposed Approach: Optimization-based Trajectory Planning with polytopic avoidance
Start
Deadlock resolved ✓
Non-conservative avoidance ✓
Dynamically-feasible ✓

Fig. 1: Comparison of approaches using optimization-based trajectory planning with obstacle avoidance. The proposed algorithm developed in this paper allows fast optimization, which can be used as an optimal controller or a trajectory planner for general nonlinear systems with obstacle avoidance between polytopes.

them, which could be applied as efficient strategies for high-dimensional kinematic planning. However, generally, these algorithms assume that a low-level controller exists, and is able to track kinematically feasible trajectories in real time. This leads to trajectories that are dynamically infeasible and results in large tracking errors on dynamical systems. Other approaches such as kinodynamic RRT* [4], LQR-RRT* [5] try to bridge the gap between path planning and control synthesis by finding appropriate steering inputs to go between two vertices in the sampling graph. However, these approaches cannot do dynamic collision checking with respect to the exact nonlinear dynamics of the robot. For general dynamical systems, we still need to locally generate a dynamically feasible and collision-free trajectory.

2) Optimization-based Control and Trajectory Planning: We now narrow down our discussion to optimization-based approaches for generating collision-free trajectories. The existing methods in this sub-area can be classified under two categories: those that generate obstacle avoidance behaviors with additional cost terms, and those that apply constraints to achieve a similar behavior. Additional cost terms were first introduced under the philosophy of potential fields [6], and were later generalized to be named as "barrier function" [7]. This approach has been applied to solve optimal control and trajectory generation problems with broad applications [8]–[12]. Other methods consider the obstacle avoidance criteria as constraints in the optimization problem. An example of such a constraint is the distance constraint, enforced using inequality constraints on the distance function between the robot and obstacles, where the robots and the obstacles are usually approximated as points [13], lines [14], paraboloids [15], ellipsoids [16], or hyper-spheres [17]. The distance functions for these shapes have analytical expressions and are differentiable so that nonlinear optimization (NLP) solvers can easily compute the gradients.

3) Obstacle Avoidance between Polytopes: When a tight-fitting obstacle avoidance motion is expected, the above over-approximations of the shape of the robot can lead to deadlock maneuvers, shown in Fig. 1. A tight polytopic approximation of the shape of the robot enables obstacle avoidance maneuvers that are less conservative, see [18]. However, the distance function between two polytopes is implicit and not analytic [19], and requires a large amount of numerical computation [20], [21]. Moreover, this distance function between polytopes is also non-differentiable with respect to the robot's configuration, which makes it hard to be treated as a constraint directly for a nonlinear programming problem. For collision avoidance between two polytopes, mixed-integer programming [19], [22], [23] applies well for linear systems but cannot be deployed as a real-time controller or trajectory planner for general nonlinear systems due to the added complexity from integer variables [24]. To handle the non-differentiability of the distance function between two polytopes, a duality-based approach [25] is introduced to reformulate constraints as a set of smooth non-convex ones. However, obstacle avoidance behavior with this method can only be achieved with a relatively long horizon and needs to be solved offline for nonlinear systems [26]–[30]. Recently, a dual optimization formulation [31] was introduced to construct a differentiable control barrier function (CBF) [32] for polytopes, but it only optimizes one-step control input and is only applicable for continuous-time affine systems with a relative-degree of one. This formulation could also run into a deadlock for general high relative-degree systems.

4) Obstacle Avoidance with DCBFs: To resolve the problems mentioned above, it's required to propose a computationally fast multi-step optimization formulation for systems with nonlinear discrete-time dynamics. Recently, it has been shown that considering discrete-time control barrier function (DCBF) constraints instead of distance constraints can handle this challenge, where the DCBF constraints can regulate the obstacle avoidance behavior with a smaller horizon and prevent local deadlock in trajectory generation, see [33]. The control and planning problems with one-step [34] or multi-step [33] optimization using DCBF constraints have been studied, and various applications on different platforms, including car racing [17], autonomous vehicles [35], and bipedal robots [36] have validated this approach. In the work mentioned above, the robots and the obstacles are only considered as points or hyper-spheres, while obstacle avoidance constraint between polytopes is still an unsolved problem in all previous work by using discrete-time control barrier functions.

B. Contributions

The contributions of our paper are as follows:

- We formulate the dual form of the obstacle avoidance constraint between polytopes as DCBF constraints for safety. These proposed DCBF constraints are incorporated into an NMPC formulation which enables fast online computation for control and planning for general nonlinear dynamical systems.
- The proposed NMPC-DCBF formulation for polytopes is validated numerically. Different convex and non-convex shaped robots are shown to be able to navigate with tight maneuvers through maze environments with polytopic obstacles using fast real-time control and trajectory generation.

II. Background

In this section, we present a brief background on optimization formulations using discrete-time control barrier functions and obstacle avoidance between polytopic sets.

A. Optimization Formulation using DCBFs

Consider a discrete-time dynamical system with states $x \in \mathcal{X} \subset \mathbb{R}^n$ and inputs $u \in \mathcal{U} \subset \mathbb{R}^m$, as

$$x_{k+1} = f(x_k, u_k), \tag{1}$$

where $x_k := x(k)$, $u_k := u(k)$, $k \in \mathbb{Z}^+$, $\mathcal{U}$ is a compact set and f is continuous.

1) Discrete-time CBFs: Obstacle avoidance for safety for this dynamical system is defined in terms of invariance of its trajectories with respect to a connected set. In other words, if the dynamical system (1) is safe with respect to a set $\mathcal{S} \subset \mathcal{X}$, then any trajectory starting inside $\mathcal{S}$ remains inside $\mathcal{S}$. The set $\mathcal{S}$ is defined as the 0-superlevel set of a continuous function $h : \mathcal{X} \to \mathbb{R}$ as:

$$\mathcal{S} := \{x \in \mathcal{X} \subset \mathbb{R}^n : h(x) \geq 0\}. \tag{2}$$

We refer to $\mathcal{S}$ as the safe set and it represents the region outside the obstacle. h is defined as a discrete-time control barrier function (DCBF) if $\forall\, x \in \mathcal{S}, \exists\, u \in \mathcal{U}$ such that

$$h(f(x,u)) \geq \gamma(x)h(x), \quad 0 \leq \gamma(x) < 1, \tag{3}$$

Let $\gamma_k := \gamma(x_k)$. Satisfying (3) implies $h(x_{k+1}) \geq \gamma_k h(x_k)$, i.e., the lower bound of the DCBF decreases exponentially with the decay rate γ_k [34]. Given a choice of $\gamma(x)$, we denote $\mathcal{K}(x)$ as

$$\mathcal{K}(x) := \{u \in \mathcal{U} : h(f(x,u)) - \gamma(x)h(x) \geq 0\}. \tag{4}$$

Then, if $x_0 \in \mathcal{S}$ and $u_k \in \mathcal{K}(x_k)$, then $x_k \in \mathcal{S}$ for $\forall\, k \in \mathbb{Z}^+$, i.e., the resulting trajectory is safe [34].

Given a valid DCBF h [32], imposing DCBF constraint (3) in an optimization problem could guarantee system safety, i.e., collision-free trajectories. If $\gamma(x)$ is close to 1, the system converges to $\partial\mathcal{S}$ slowly but can easily become infeasible. On the other hand, if $\gamma(x)$ is close to 0, the constraint (3) is feasible in a larger domain but can approach $\partial\mathcal{S}$ quickly and become unsafe. The later proposed formulation in [33] introduces a relaxing form of DCBF constraint as follows,

$$h(f(x,u)) \geq \omega(x)\gamma(x)h(x), \quad 0 \leq \gamma(x) < 1. \tag{5}$$

where the relaxing variable ω resolves the tradeoff between feasibility and safety and is optimized with other variables inside an optimization formulation.

When one-step control input is optimized [34], it could lead to a deadlock situation such that the robot is safe but unable to track the reference command. A nonlinear model predictive control formulation [17] can overcome these problems, shown as follows,

NMPC-DCBF [17]:

$$\min_{U,\Omega} p(x_{t+N|t}) + \sum_{k=0}^{N-1} q(x_{t+k|t}, u_{t+k|t}) + \psi(\omega_k) \tag{6a}$$

$$\text{s.t. } x_{t|t} = x_t, \tag{6b}$$

$$x_{t+k+1|t} = f(x_{t+k|t}, u_{t+k|t}), \quad k=0,...,N-1 \tag{6c}$$

$$u_{t+k|t} \in \mathcal{U},\ x_{t+k|t} \in \mathcal{X}, \quad k=0,...,N-1 \tag{6d}$$

$$h(x_{t+k+1|t}) \geq \omega_k \gamma_k h(x_{t+k|t}), \quad \omega_k \geq 0 \quad \text{for } k=0,...,N_{\text{CBF}}-1, \tag{6e}$$

where $x_{t+i|t}$ and $u_{t+i|t}$ denote the predicted state and input at time $t+i$ evaluated at the current time t. N and $N_{\text{CBF}} \leq N$ denote the prediction and safety horizons respectively, which allows us to control the optimization computation complexity, and $U = [u_{t|t}^T, ..., u_{t+N-1|t}^T]^T$ and $\Omega = [\omega_0, ..., \omega_{N_{\text{CBF}}-1}]$ are the joint input and relaxation variables respectively. $p(\cdot)$ and $q(\cdot,\cdot)$ are the terminal and stage costs respectively, and ψ is the penalty function for the relaxation variable.

The optimization formulation (6) can be regarded as a control problem with $U^* = [{u^*}_{t|t}^T, ..., {u^*}_{t+N-1|t}^T]^T$ as the optimized control inputs, as well as a trajectory planning problem with $X^* = [{x^*}_{t|t}^T, ..., {x^*}_{t+N|t}^T]^T$ as the optimized trajectory. From the joint optimal input vector U^* the first input $u^*_{t|t}$ is applied at time $t \in \mathbb{Z}^+$, and the optimization (6) is solved again at time $t+1$ with x_{t+1}.

B. Obstacle Avoidance between Polytopic Sets

In this work, we assume that there are $N_{\mathcal{O}}$ static obstacles together with a single controlled robot. We further assume that the geometry of all the obstacles and the robot can be over-approximated with a union of convex polytopes, which is defined as a bounded polyhedron.

Let the state of the robot be $x \in \mathbb{R}^n$ with its discrete-time dynamics as defined in (1) and the geometry of the robot and obstacles be in a l-dimensional space. We denote the geometry of the i-th static obstacle and the dynamic robot at some state $x \in \mathcal{X}$ by the polytopes:

$$\mathcal{O}_i := \{y \in \mathbb{R}^l : A^{\mathcal{O}_i} y \leq b^{\mathcal{O}_i}\} \tag{7}$$
$$\mathcal{R}(x) := \{y \in \mathbb{R}^l : A^{\mathcal{R}}(x) y \leq b^{\mathcal{R}}(x)\},$$

respectively, where $b^{\mathcal{O}_i} \in \mathbb{R}^{s^{\mathcal{O}_i}}, i \in \{1, ...N_{\mathcal{O}}\}$ and $b^{\mathcal{R}}(x) \in \mathbb{R}^{s^{\mathcal{R}}}$, and $A^{\mathcal{R}}, b^{\mathcal{R}}$ are continuous. Inequalities on vectors are enforced element-wise. We assume that $\mathcal{O}_i$, $i \in \{1, ...N_{\mathcal{O}}\}$ and $\mathcal{R}(x)\ \forall\ x \in \mathcal{X}$ are bounded and non-empty. $s^{\mathcal{O}_i}$, $s^{\mathcal{R}}$ represent the number of facets of polytopic sets for the i-th obstacle and the robot, respectively.

Then $\mathcal{O}_i$, $\forall\ i \in \{1, ..., N_{\mathcal{O}}\}$, and $\mathcal{R}(x)$, $\forall\ x \in \mathcal{X}$, are non-empty, convex, and compact sets, and the minimum distance between any pair $(\mathcal{O}_i, \mathcal{R}(x))$ is well-defined. The minimum distance is 0 if and only if $\mathcal{O}_i$ and $\mathcal{R}(x)$ intersect. The square of the minimum distance between $\mathcal{O}_i$ and $\mathcal{R}(x)$, denoted by $h_i(x)$, can be computed using a QP as follows:

$$h_i(x) = \min_{(y^{\mathcal{O}_i}, y^{\mathcal{R}}) \in \mathbb{R}^{2l}} \quad \|y^{\mathcal{O}_i} - y^{\mathcal{R}}\|_2^2 \tag{8a}$$

$$\text{s.t.} \quad A^{\mathcal{O}_i} y^{\mathcal{O}_i} \leq b^{\mathcal{O}_i},\ A^{\mathcal{R}}(x) y^{\mathcal{R}} \leq b^{\mathcal{R}}(x). \tag{8b}$$

where (8) is a convex optimization problem. To ensure safe motion of the robot, we enforce DCBF constraints (3) pairwise between each robot-obstacle pair. Then, the safe set corresponding to the pair $(\mathcal{O}_i, \mathcal{R})$ is defined as:

$$\mathcal{S}_i := \{x \in \mathbb{R}^n : h_i(x) > 0\}^c, \tag{9}$$

where $(\cdot)^c$ denotes the closure of a set. Enforcing the DCBF constraint for each h_i ensures that the state remains in $\mathcal{S}_i$ for all i, and thus the state remains in $\mathcal{S} := \cap_{i=1}^{N_{\mathcal{O}}} S_i$. Note that, due to the relaxation variable ω, enforcing multiple DCBF constraints for each h_i is equivalent to enforcing a single DCBF constraint on $h(x) := \min_i\{h_i(x)\}$. Thus, we focus on how to enforce the DCBF constraint for a given pair $(\mathcal{O}_i, \mathcal{R})$.

However, (3) requires computation of $h_i(f(x,u))$ via (8), which can only be solved numerically. This results in an optimization formulation with non-differentiable implicit constraints, which results in a significant increase in computation time. In the following section we derive explicit differentiable constraints which guarantee that the DCBF constraint is satisfied without affecting the feasible set of safe inputs $\mathcal{K}(x)\ \forall\ x \in \mathcal{X}$.

III. Optimization with Dual DCBF Constraints

In this section, we derive the duality-based optimization which allows generating obstacle avoidance maneuvers for the controlled robot in tight environments with obstacles.

A. Dual Optimization Problem

Corresponding to the minimization problem (8), we can define a dual problem. The dual problem is always a convex optimization problem, and a maximization problem if the original primal problem is a minimization one. The dual formulation of (8) can be explicitly computed as [37, Chap. 8]:

$$g_i(x) = \max_{(\lambda^{\mathcal{O}_i}, \lambda^{\mathcal{R}})} \quad -\lambda^{\mathcal{O}_i} b^{\mathcal{O}_i} - \lambda^{\mathcal{R}} b^{\mathcal{R}}(x) \tag{10a}$$

$$\text{s.t.} \quad \lambda^{\mathcal{O}_i} A^{\mathcal{O}_i} + \lambda^{\mathcal{R}} A^{\mathcal{R}}(x) = 0, \tag{10b}$$

$$\|\lambda^{\mathcal{O}_i} A^{\mathcal{O}_i}\|_2 \leq 1,\ \lambda^{\mathcal{O}_i} \geq 0,\ \lambda^{\mathcal{R}} \geq 0. \tag{10c}$$

Here $\lambda^{\mathcal{O}_i} A^{\mathcal{O}_i}$ represents the normal vector of the plane of maximum separation between the polytopes.

The Weak Duality Theorem [37, Chap. 5] states that $g_i(x) \leq h_i(x)$ holds for all optimization problems. Since (8) is a convex optimization with linear constraints and has a well-defined optimum solution in $\mathbb{R}^+$, the Strong Duality Theorem [37, Chap. 5] also holds, which states that

$$g_i(x) = h_i(x). \tag{11}$$

B. Obstacle Avoidance with Dual Variables

In order to remove the implicit dependence of $h_i(x)$ on x via (8), we enforce a constraint stronger than (3) which does not require explicit computation of $h_i(x)$ via (8). The dual formulation of (8), (10), can be used to achieve this.

Let $\bar{g}_i(x, \lambda^{\mathcal{O}_i}, \lambda^{\mathcal{R}})$ be the cost corresponding to any feasible solution $(\lambda^{\mathcal{O}_i}, \lambda^{\mathcal{R}})$ of (10). Since (10) is a maximization problem and the Strong Duality Theorem (11) holds for the primal problem (8),

$$\bar{g}_i(x, \lambda^{\mathcal{O}_i}, \lambda^{\mathcal{R}}) := -\lambda^{\mathcal{O}_i} b^{\mathcal{O}_i} - \lambda^{\mathcal{R}} b^{\mathcal{R}}(x) \leq h_i(x). \tag{12}$$

Then, at time k, we can enforce the stronger constraint

$$-\lambda^{\mathcal{O}_i} b^{\mathcal{O}_i} - \lambda^{\mathcal{R}} b^{\mathcal{R}}(f(x_k, u)) \geq \gamma_k h_i(x_k) \tag{13}$$

which, using (12), implies

$$h_i(f(x_k, u)) \geq \gamma_k h_i(x_k), \tag{14}$$

as required. Hence, the DCBF constraint can be enforced by (13) subject to $(f(x_k, u), \lambda^{\mathcal{O}_i}, \lambda^{\mathcal{R}})$ being feasible, i.e., the stronger DCBF constraint (13) along with the feasibility constraints (10b), (10c) should be satisfied

$$-\lambda^{\mathcal{O}_i} b^{\mathcal{O}_i} - \lambda^{\mathcal{R}} b^{\mathcal{R}}(f(x_k, u)) \geq \gamma_k h_i(x_k), \tag{15a}$$

$$\lambda^{\mathcal{O}_i} A^{\mathcal{O}_i} + \lambda^{\mathcal{R}} A^{\mathcal{R}}(f(x_k, u)) = 0, \tag{15b}$$

$$\|\lambda^{\mathcal{O}_i} A^{\mathcal{O}_i}\|_2 \leq 1, \ \lambda^{\mathcal{O}_i} \geq 0, \ \lambda^{\mathcal{R}} \geq 0. \tag{15c}$$

By the Strong Duality Theorem (11), $\exists\ \lambda^{\mathcal{O}_i *}, \lambda^{\mathcal{R}*}$ satisfying (10b)-(10c) such that for all $x \in \mathcal{X}$,

$$\bar{g}_i(x, \lambda^{\mathcal{O}_i *}, \lambda^{\mathcal{R}*}) = -\lambda^{\mathcal{O}_i *} b^{\mathcal{O}_i} - \lambda^{\mathcal{R}*} b^{\mathcal{R}}(x) = h_i(x). \tag{16}$$

This means that for any fixed $x \in \mathcal{X}$, the input u satisfies the DCBF constraint (3) with the implicit definition of h_i if and only if the tuple $(u, \lambda^{\mathcal{O}_i *}, \lambda^{\mathcal{R}*})$ satisfies (13). So, the feasible set of inputs is not reduced at any $x \in \mathcal{X}$.

C. Optimization Formulation

We adopt the philosophy of the NMPC-DCBF method to enforce safety constraints between polytopes, as shown in Sec. III-B, and construct a multi-step optimization formulation. For simplicity of notation, we drop the explicit dependence of $h_i(x_{t+k|t})$ on $x_{t|t}$ and $[u_{t|t}^T, ..., u_{t+k-1|t}^T]^T$. Since $h_i(x_{t+k|t})$ is also not explicitly known at time t, to impose DCBF constraints along the horizon, we extend the idea in Sec. III-B by enforcing a constraint stronger than $h_i(x_{t+k+1|t}) \geq \gamma_k h_i(x_{t+k|t})$ which does not rely on computations of $h_i(x_{t+k|t})$ and $h_i(x_{t+k+1|t})$ via (8).

The primal optimization problem (8) provides an upper bound to $h_i(x)$, which can be used in the stronger DCBF constraints. Let $(y^{\mathcal{O}_i}, y^{\mathcal{R}})$ be any feasible solution of (8). Since (8) is a minimization problem and its solution is well-defined for all $x \in \mathcal{X}$,

$$\bar{h}_i(x, y^{\mathcal{O}_i}, y^{\mathcal{R}}) := \|y^{\mathcal{O}_i} - y^{\mathcal{R}}\|_2^2 \geq h_i(x). \tag{17}$$

Then at time t, we can enforce

$$-\lambda^{\mathcal{O}_i} b^{\mathcal{O}_i} - \lambda^{\mathcal{R}} b^{\mathcal{R}}(x_{t+k+1|t}) \geq \gamma_k \|y^{\mathcal{O}_i} - y^{\mathcal{R}}\|_2^2 \tag{18}$$

which, using (12) and (17), implies

$$\begin{aligned} h_i(x_{t+k+1|t}) &\geq -\lambda^{\mathcal{O}_i} b^{\mathcal{O}_i} - \lambda^{\mathcal{R}} b^{\mathcal{R}}(x_{t+k+1|t}) \\ &\geq \gamma_k \|y^{\mathcal{O}_i} - y^{\mathcal{R}}\|_2^2 \geq \gamma_k h_i(x_{t+k|t}), \end{aligned} \tag{19}$$

as required. Additionally, we can introduce the relaxation variables without affecting the analysis in this section.

Remark 1. The primal problem (8) and the dual problem (10) provide upper and lower bounds respectively for the distance function h, which is implicit in nature. These bounds can be calculated implicitly since they are equal to the cost functions subject to the corresponding constraints of each optimization problem. Thus, any implicit constraint on h can be converted into a weaker explicit set of constraints using these bounds.

Then at time t, we first calculate the optimal solutions $(y_t^{\mathcal{O}_i *}, y_t^{\mathcal{R}*})$ to the minimum distance QP (8) using $x = x_t$, hence the NMPC-DCBF formulation for polytopes is shown as follows:

NMPC-DCBF for Polytopes:

$$\min_{U,\Omega} p(x_{t+N|t}) + \sum_{k=0}^{N-1} q(x_{t+k|t}, u_{t+k|t}) + \psi(\omega_k) \tag{20a}$$

$$\text{s.t. } x_{t|t} = x_t, \quad y_0^{\mathcal{O}_i} = y_t^{\mathcal{O}_i *}, \quad y_0^{\mathcal{R}} = y_t^{\mathcal{R}*} \tag{20b}$$

$$x_{t+k+1|t} = f(x_{t+k|t}, u_{t+k|t}), \ k=0, ..., N-1 \tag{20c}$$

$$u_{t+k|t} \in \mathcal{U}, \ x_{t+k|t} \in \mathcal{X}, \qquad k=0, ..., N-1 \tag{20d}$$

$$-\lambda_{k+1}^{\mathcal{O}_i} b^{\mathcal{O}_i} - \lambda_{k+1}^{\mathcal{R}} b^{\mathcal{R}}(x_{t+k+1|t}) \geq \omega_k \gamma_k \|y_k^{\mathcal{O}_i} - y_k^{\mathcal{R}}\|_2^2, \quad \text{for } k=0, ..., N_{\text{CBF}}-1 \tag{20e}$$

$$A^{\mathcal{R}}(x_{t+k+1|t}) y_k^{\mathcal{R}} \leq b^{\mathcal{R}}(x_{t+k+1|t}), \ A^{\mathcal{O}_i} y_k^{\mathcal{O}_i} \leq b^{\mathcal{O}_i}, \quad \text{for } k=1, ..., N_{\text{CBF}}-1 \tag{20f}$$

$$\lambda_k^{\mathcal{O}_i} A^{\mathcal{O}_i} + \lambda_k^{\mathcal{R}} A^{\mathcal{R}}(x_{t+k|t}) = 0, \ \|\lambda_k^{\mathcal{O}_i} A^{\mathcal{O}_i}\|_2 \leq 1, \quad \text{for } k=1, ..., N_{\text{CBF}} \tag{20g}$$

$$\lambda_{k+1}^{\mathcal{O}_i} \geq 0, \quad \lambda_{k+1}^{\mathcal{R}} \geq 0, \quad \omega_k \geq 0, \quad \text{for } k=0, ..., N_{\text{CBF}}-1 \tag{20h}$$

The subscripts of $\lambda^{\mathcal{O}_i}, \lambda^{\mathcal{R}}, y^{\mathcal{O}_i}, y^{\mathcal{R}}$ denote the time, (20e) is the DCBF constraint between polytopes, (20f) is the primal feasibility condition, and (20g)-(20h) are the dual feasibility conditions. The initial conditions, system dynamics constraints, and input and state constraints are represented by (20b), (20c), and (20d) respectively. This NMPC-DCBF formulation (20) corresponds to enforcing safety constraints only between the pair $(\mathcal{O}_i, \mathcal{R})$ of polytopes. To enforce DCBF constraints between every pair of robot and obstacle, we introduce corresponding dual and primal variables and enforce the constraints (20e)-(20h) for each pair.

Remark 2. The mathematical intuition behind (20) is different from the previous work in [31] that focuses on duality in the continuous domain. Moreover, the system dynamics is not required to be control affine in our proposed (20). Furthermore, differentiability of dual variables is required in [31] and not needed in this work.

D. Complexity and Performance

1) Exponential DCBF Constraint: To reduce complexity of the NMPC-DCBF shown in (20), we can modify the DCBF constraint $h_i(x_{t+k|t}) \geq \omega_k \gamma_k h_i(x_{t+k-1|t})$ by rolling out time k and removing the dependence on $x_{t+k-1|t}$ from the LHS of the DCBF constraint, thus enforcing the following exponential DCBF constraints:

$$h_i(x_{t+k|t}) \geq \omega_k (\Pi_{j=0}^{k} \gamma_j) h_i(x_{t|t}). \tag{21}$$

The exponential decay rate $\Pi_{j=0}^{k}\gamma_j$ arises due to rolling out the decay rate at time j, γ_j. The DCBF constraint (21) only contains $h_i(x_{t|t})$ on the RHS for all k, which can be explicitly computed at each time step using $x_t = x_{t|t}$. Note that the RHS of the DCBF constraint (19) at time $k \geq 1$ cannot be computed explicitly, since $x_{t+k|t}$ implicitly depends on the control inputs. This leads to modifying (20e) with,

$$-\lambda_k^{\mathcal{O}_i} b^{\mathcal{O}_i} - \lambda_k^{\mathcal{R}} b^{\mathcal{R}}(x_{t+k|t}) \geq \omega_k (\Pi_{j=0}^{k}\gamma_j) h_i(x_{t|t}) \tag{22}$$

Such a formulation speeds up the computational time, as only $h(x_{t|t})$ needs to be calculated at each time t. This change affects neither the feasibility nor the safety of the system.

2) Horizon Length Selection: Compared with distance constraints, DCBF constraints allow effective obstacle avoidance behavior with smaller horizon length. Our NMPC-DCBF formulation does not require the obstacle avoidance horizon length N_{CBF} be equal to N, which additionally reduces the complexity [33]. Additionally, maneuvers such as deceleration for obstacle avoidance or reversing motion for deadlock avoidance are motion primitives that only require a small horizon in the MPC formulation. To sum up, DCBF constraints with dual variables enable fast optimization for obstacle avoidance between polytopes.

IV. Numerical Results

In this section, we consider an autonomous navigation problem. We model the controlled robot with different shapes, including triangle, rectangle, pentagon and L-shape. The proposed optimization-based planning algorithm allows us to successfully generate dynamically-feasible collision-free trajectories even in tight maze environments, shown in Fig. 2. Animation of the navigation problems can be found in the video attachment.

1) Environment: The tight maze environment is described by a combination of multiple convex obstacles, including shapes like triangle, rectangle pentagon, etc. The controlled robot is also modelled with different shapes, including triangle, rectangle, pentagon and L-shape, whose orientation is determined by the yaw angle. The L-shape is non-convex, but can be represented by two convex polytopes. The dimensions for each robot shape are mentioned in Fig. 2.

2) System Dynamics: The controlled robot is described by the kinematic bicycle model, which is typical for testing trajectory planning algorithms in tight environments. The continuous-time dynamics of the robot is given as follows,

$$\dot{c}_x = v\cos(\phi),\ \dot{c}_x = v\sin(\phi),\ \dot{v} = a,\ \dot{\phi} = \frac{v\tan\delta}{l}, \tag{23}$$

where system states are $x = (c_x, c_y, v, \phi)$ with (c_x, c_y) as the center of rear axes, ϕ as yaw angle and v as the velocity, and $u = (a, \delta)$ are inputs with steering angle δ and acceleration a. The wheel base of the robot is $l = 0.1m$. The steering angle and acceleration are limited between $\pm 0.5rad$ and $\pm 1m/s^2$.

3) Global Planning: The global path from the starting position to the goal is generated using the A* algorithm. The 2-D space is sub-divided into grids and obstacle collision checks are performed at each grid point during the algorithm. A safety margin, which is smaller than at least one dimension of the robot, is used for the collision checks. Finally, the generated optimal path is reduced to fewer waypoints using line-of-sight reductions, similar to the θ^* algorithm [38]. The generated global path is not dynamically feasible, and is only safe at the node points.

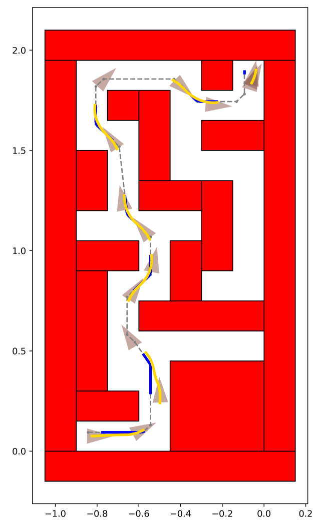
4) Local Trajectory Generation: The local trajectory planning is formulated using the NMPC-DCBF formulation (20) to track the local reference trajectory while avoiding obstacles. The local reference trajectory $\bar{X} = [\bar{x}_{t|t}^T, \bar{x}_{t+1|t}^T, ..., \bar{x}_{t+N|t}^T]^T$ is generated from a start point with a constant speed v_0 and the same orientation as the global planner. The start point is found by local projection from the current robot's position to the global path. For tracking the local reference trajectory, the cost function (20) in the optimizer is constructed with terminal cost, stage cost, and relaxing cost function, respectively as

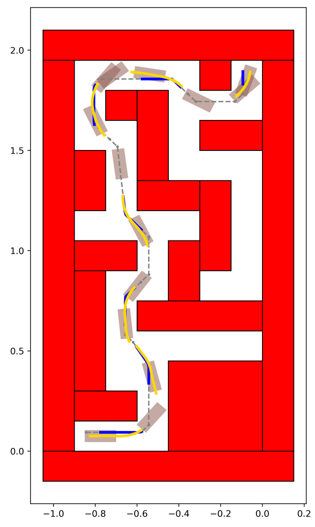
$$p(x_{t+N|t}) = ||x_{t+N|t} - \bar{x}_{t+N|t}||_{Q_T}^2, \tag{24a}$$

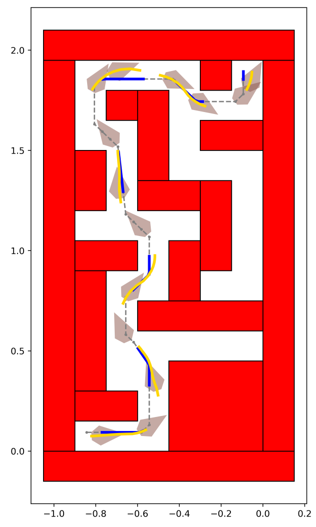
$$\begin{aligned} q(x_{t+k|t}, u_{t+k|t}) =& ||x_{t+k|t} - \bar{x}_{t+k|t}||_Q^2 + ||u_{t+k|t}||_R^2 \\ &+ ||u_{t+k|t} - u_{t+k-1|t}||_{dR}^2, \end{aligned} \tag{24b}$$

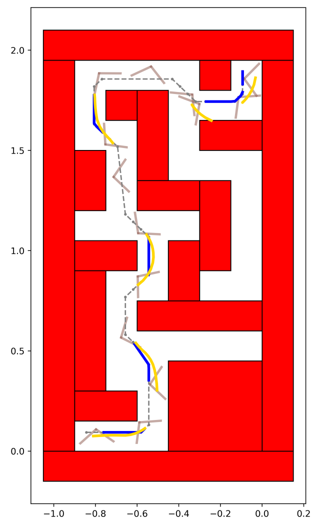
$$\psi(\omega_k) = p_\omega(\omega_k - 1)^2, \tag{24c}$$

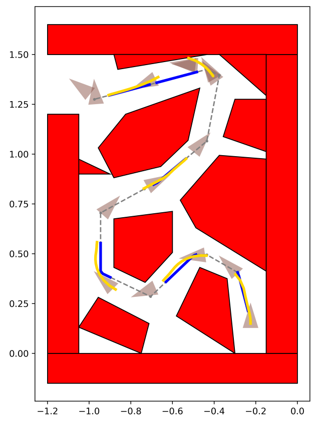
where $u_{t-1|t} = u_{t-1|t-1}^*$ represents the last optimized control input. The dynamics constraints (20c) is applied with the discrete-time forward Euler formulation from the continuous-time dynamics (23). The input constraints (20d) is imposed by the steering angle and acceleration limits mentioned above. A prediction horizon of 11 is used, with the DCBF horizon as 6, and the decay rate as 0.8. The obstacle avoidance constraints (20e)-(20h) can be applied directly to each pair of the convex-shaped (triangle, rectangle, pentagon)

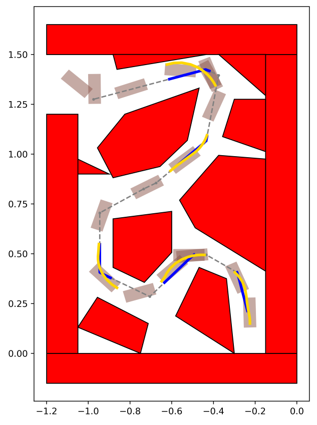

(a) Triangle-shaped robot

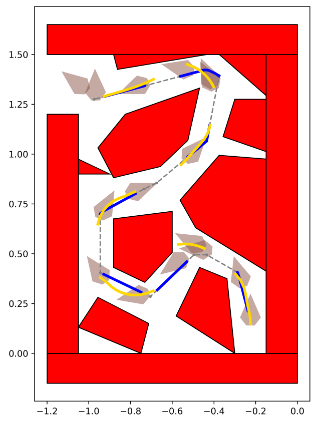

(b) Rectangle-shaped robot


(c) Pentagon-shaped robot


(d) L-shaped robot


(e) Triangle-shaped robot


(f) Rectangle-shaped robot


(g) Pentagon-shaped robot

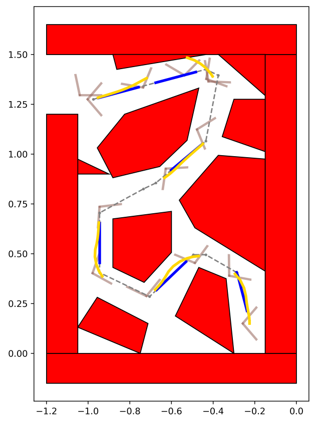

(h) L-shaped robot

Fig. 2: Snapshots from simulation of tight maneuvers of obstacle avoidance with a controlled robot with different shapes in two maze environments. The triangle has side lengths $0.133m, 0.133m, 0.1m$; the rectangle has length $0.15m$ and width $0.06m$; the symmetric pentagon has length $0.16m$ and width $0.1m$; and the L-shape has an arm length $0.114m$, arm width $0.03m$, with included angle of $101\,\text{deg}$. The corridors in both the maze environments have widths $0.15m$, and every robot geometry has at least one dimension larger than the corridor width. The grey dotted lines represent the global reference path, the blue lines represent the local reference trajectory for the optimization, and the yellow lines represent the optimized trajectory at various time instances.

robot and each convex obstacle. When the robot is non-convex shaped (L-shape), these constraints are applied to the convex parts of the robot and each convex obstacle.

To reduce the complexity of the optimization formulation, we only consider the obstacles which are within a specified radius from the robot at any given time. The radius is calculated using the reference tracking velocity, prediction horizon and the maximum deceleration of the robots.

5) Warm Start: The NMPC-DCBF formulation is a non-convex optimization, and hence computationally challenging to solve in general. Although the DCBF constraints help to reduce the complexity, as discussed in Sec. III-D.2, it still requires a good initial guess to lead to faster numerical convergence. The initial guess trajectory and control inputs are generated using a braking controller. An acceleration input equal to the maximum deceleration is provided and the steering angle is set to zero. Once the robot comes to a halt, the acceleration input is also set to zero. The braking control inputs, along with the trajectory generated from it are provided as an initial guess to the optimization at each time step. Since at each time step $h_i(x_{t|t})$ is solved using (10), the dual optimal solution from this computation is provided as an initial guess for the dual variables for the entire horizon.

6) Simulation Results: To evaluate the performance of (20), we study the navigation problem with two different maze environments with four choices of robot shapes. The optimization problems are implemented in Python with CasADi [39] as modelling language, solved with IPOPT [40] on Ubuntu 18.04 with Intel Xeon E-2176M CPU with a 2.7GHz clock. From the snapshots we can observe tight-fitting obstacle avoidance motion of the robot, and also reversing motion to avoid deadlock. These examples highlight the safety and planning features of our implementation. We also analyze the computational time of trajectory generation using (20), which is shown in TABLE I. This illustrates that optimization (20) can be solved sufficiently fast to be deployed on different-shaped robots for trajectory generation in different maze environments. The details of hyperparameter selections can be found in the open-source repository.

The specific choice of the parameters in the optimization formulation (20) can influence safety and deadlock behaviors in the robot. Qualitatively, for safety, the reference tracking velocity should be such that the robot can come to a halt within the prediction horizon with the input as the maximum deceleration. There is also a trade-off between deadlock avoidance and safety: Higher value of the terminal cost weight Q_T improves deadlock avoidance, but can also increase velocity of the robot, leading to unsafe motion.

To tune the hyperparameters for a given system, consider the set of feasible trajectories of (20) for some reference trajectory $\bar{X}$. First, to capture the affect of all control inputs on the DCBF h, the safety horizon N_{CBF} must be larger than the relative degree of the system with respect to h. If there exists any safe trajectory at the current state that brings the robot to rest, the resulting control law guarantees that the closed-loop trajectory is also safe. Thus, it is desirable to choose the safety horizon N_{CBF} high enough so that the robot can come to rest within N_{CBF} steps. However, choosing a large safety horizon may result in an increased computation time and safety could be achieved in practice with proper tuning of the decay rate γ. Choosing γ closer to 1 prioritizes safety over reference tracking while choosing γ closer to 0 ensures faithful tracking at the cost of safety. Note that the relaxation variable ω_k is essential for feasibility of (20) especially for γ closer to 1, see [33]. Selecting a large terminal cost parameter p_ω in (24c) is also key to ensure that the effective decay rate does not deviate, thus prioritizing safety of the system. Finally, the stage cost parameter Q in (24b) and the terminal cost parameter Q_T in (24a) present a trade-off between reference tracking and exploration. Larger values of Q result in the closed-loop trajectory closely following the reference trajectory. Notice that since the reference trajectory may not be dynamically feasible, this may still result in deadlocks. Selecting larger values of Q_T mostly penalizes deviation from the terminal state and thus promotes exploration during timesteps $k = 1, ..., N_{\text{CBF}-1}$. This can result in better deadlock avoidance maneuvers as depicted in Fig. 2.

TABLE I: Solver time statistics of NMPC-DCBF with polytopes (20).

Env	Robot Shape	median	std	min	max
Maze 1	triangle (Fig. 2a)	51ms	19ms	14ms	149ms
	rectangle (Fig. 2b)	49ms	25ms	15ms	185ms
	pentagon (Fig. 2c)	71ms	15ms	15ms	272ms
	L-shape (Fig. 2d)	85ms	13ms	13ms	215ms
Maze 2	triangle (Fig. 2e)	33ms	24ms	13ms	121ms
	rectangle (Fig. 2f)	29ms	25ms	13ms	122ms
	pentagon (Fig. 2g)	30ms	29ms	13ms	150ms
	L-shape (Fig. 2h)	29ms	49ms	13ms	233ms

V. Conclusion

We proposed a nonlinear optimization formulation using discrete-time control barrier function based constraints for polytopes. The proposed formulation has been shown to be applied as a fast optimization for control and planning for general nonlinear dynamical systems. We validated our approach on navigation problems with various robot shapes in maze environments with polytopic obstacles.

References

[1] R. Bohlin and L. E. 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. IEEE, 2000, pp. 521–528.

[2] 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.

[3] F. Islam, J. Nasir, U. Malik, Y. Ayaz, and O. Hasan, "Rrt∗-smart: Rapid convergence implementation of rrt∗ towards optimal solution," in *2012 IEEE international conference on mechatronics and automation*. IEEE, 2012, pp. 1651–1656.

[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*. IEEE, 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*. IEEE, 2012, pp. 2537–2542.

[6] Y. K. Hwang, N. Ahuja *et al.*, "A potential field approach to path planning." *IEEE Transactions on Robotics and Automation*, vol. 8, no. 1, pp. 23–32, 1992.

[7] A. G. Wills and W. P. Heath, "Barrier function based model predictive control," *Automatica*, vol. 40, no. 8, pp. 1415–1422, 2004.

[8] J. Schulman, Y. Duan, J. Ho, A. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, and P. Abbeel, "Motion planning with sequential convex optimization and convex collision checking," *The International Journal of Robotics Research*, vol. 33, no. 9, pp. 1251–1270, 2014.

[9] A. Liniger, A. Domahidi, and M. Morari, "Optimization-based autonomous racing of 1: 43 scale rc cars," *Optimal Control Applications and Methods*, vol. 36, no. 5, pp. 628–647, 2015.

[10] M. Z. Romdlony and B. Jayawardhana, "Stabilization with guaranteed safety using control lyapunov–barrier function," *Automatica*, vol. 66, pp. 39–47, 2016.

[11] H. Obeid, L. M. Fridman, S. Laghrouche, and M. Harmouche, "Barrier function-based adaptive sliding mode control," *Automatica*, vol. 93, pp. 540–544, 2018.

[12] Z. Wu, F. Albalawi, Z. Zhang, J. Zhang, H. Durand, and P. D. Christofides, "Control lyapunov-barrier function-based model predictive control of nonlinear systems," *Automatica*, vol. 109, p. 108508, 2019.

[13] R. B. Patel and P. J. Goulart, "Trajectory generation for aircraft avoidance maneuvers using online optimization," *Journal of guidance, control, and dynamics*, vol. 34, no. 1, pp. 218–230, 2011.

[14] L. Blackmore, M. Ono, and B. C. Williams, "Chance-constrained optimal path planning with obstacles," *IEEE Transactions on Robotics*, vol. 27, no. 6, pp. 1080–1094, 2011.

[15] F. Ferraguti, M. Bertuletti, C. T. Landi, M. Bonfè, C. Fantuzzi, and C. Secchi, "A control barrier function approach for maximizing performance while fulfilling to iso/ts 15066 regulations," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5921–5928, 2020.

[16] U. Rosolia, S. De Bruyne, and A. G. Alleyne, "Autonomous vehicle control: A nonconvex approach for obstacle avoidance," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 2, pp. 469–484, 2016.

[17] J. Zeng, B. Zhang, and K. Sreenath, "Safety-critical model predictive control with discrete-time control barrier function," in *2021 American Control Conference (ACC)*. IEEE, 2021, pp. 3882–3889.

[18] M. Kennedy III, D. Thakur, M. Ani Hsieh, S. Bhattacharya, and V. Kumar, "Optimal paths for polygonal robots in se (2)," *Journal of Mechanisms and Robotics*, vol. 10, no. 2, p. 021005, 2018.

[19] I. E. Grossmann, "Review of nonlinear mixed-integer and disjunctive programming techniques," *Optimization and engineering*, vol. 3, no. 3, pp. 227–252, 2002.

[20] E. G. Gilbert and C.-P. Foo, "Computing the distance between general convex objects in three-dimensional space," *IEEE Transactions on Robotics and Automation*, vol. 6, no. 1, pp. 53–61, 1990.

[21] R. Deits and R. Tedrake, "Computing large convex regions of obstacle-free space through semidefinite programming," in *Algorithmic foundations of robotics XI*. Springer, 2015, pp. 109–124.

[22] A. Richards and J. P. How, "Aircraft trajectory planning with collision avoidance using mixed integer linear programming," in *Proceedings of the 2002 American Control Conference (IEEE Cat. No. CH37301)*, vol. 3. IEEE, 2002, pp. 1936–1941.

[23] R. Deits and R. Tedrake, "Efficient mixed-integer planning for uavs in cluttered environments," in *2015 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2015, pp. 42–49.

[24] H. Liu, W. Liu, and L. J. Latecki, "Convex shape decomposition," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, 2010, pp. 97–104.

[25] X. Zhang, A. Liniger, A. Sakai, and F. Borrelli, "Autonomous parking using optimization-based collision avoidance," in *2018 IEEE Conference on Decision and Control (CDC)*. IEEE, 2018, pp. 4327–4332.

[26] X. Zhang, A. Liniger, and F. Borrelli, "Optimization-based collision avoidance," *IEEE Transactions on Control Systems Technology*, vol. 29, no. 3, pp. 972–983, 2020.

[27] J. Zeng, P. Kotaru, M. W. Mueller, and K. Sreenath, "Differential flatness based path planning with direct collocation on hybrid modes for a quadrotor with a cable-suspended payload," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3074–3081, 2020.

[28] X. Shen, E. L. Zhu, Y. R. Stürz, and F. Borrelli, "Collision avoidance in tightly-constrained environments without coordination: a hierarchical control approach," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 2674–2680.

[29] R. Firoozi, L. Ferranti, X. Zhang, S. Nejadnik, and F. Borrelli, "A distributed multi-robot coordination algorithm for navigation in tight environments," *arXiv preprint arXiv:2006.11492*, 2020.

[30] S. Gilroy, D. Lau, L. Yang, E. Izaguirre, K. Biermayer, A. Xiao, M. Sun, A. Agrawal, J. Zeng, Z. Li, and K. Sreenath, "Autonomous navigation for quadrupedal robots with optimized jumping through constrained obstacles," in *2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)*, 2021, pp. 2132–2139.

[31] A. Thirugnanam, J. Zeng, and K. Sreenath, "A duality-based approach for real-time obstacle avoidance between polytopes with control barrier functions," *arXiv preprint arXiv:2107.08360*, 2021.

[32] 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)*. IEEE, 2019, pp. 3420–3431.

[33] J. Zeng, Z. Li, and K. Sreenath, "Enhancing feasibility and safety of nonlinear model predictive control with discrete-time control barrier functions," in *2021 IEEE Conference on Decision and Control (CDC)*. IEEE, 2021.

[34] A. Agrawal and K. Sreenath, "Discrete control barrier functions for safety-critical control of discrete systems with application to bipedal robot navigation," in *Robotics: Science and Systems*, 2017.

[35] H. Ma, J. Chen, S. Eben, Z. Lin, Y. Guan, Y. Ren, and S. Zheng, "Model-based constrained reinforcement learning using generalized control barrier function," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 4552–4559.

[36] S. Teng, Y. Gong, J. W. Grizzle, and M. Ghaffari, "Toward safety-aware informative motion planning for legged robots," *arXiv preprint arXiv:2103.14252*, 2021.

[37] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[38] A. Nash, K. Daniel, S. Koenig, and A. Felner, "Theta^*: Any-angle path planning on grids," in *AAAI*, vol. 7, 2007, pp. 1177–1183.

[39] J. A. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "Casadi: a software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019.

[40] L. T. Biegler and V. M. Zavala, "Large-scale nonlinear programming using ipopt: An integrating framework for enterprise-wide dynamic optimization," *Computers & Chemical Engineering*, vol. 33, no. 3, pp. 575–582, 2009.