2017 IEEE 56th Annual Conference on Decision and Control (CDC)
December 12-15, 2017, Melbourne, Australia

Trajectory Optimization with Inter-sample Obstacle Avoidance via Successive Convexification

Daniel Dueri[1], Yuanqi Mao[1], Zohaib Mian[2], Jerry Ding[2], and Behçet Açıkmeşe[1]

***Abstract*— This paper develops a convex optimization-based method for real-time path planning onboard an autonomous vehicle for environments with cylindrical or ellipsoidal obstacles. Obstacles render the state space non-convex, thus a constrained, non-convex optimal control problem must be solved to obtain a feasible trajectory. A technique known as Successive Convexification is used to solve the non-convex optimal control problem via a convergent sequence of convex optimization problems. The paper has two main contributions: first, constraints for ensuring that the computed trajectories do not cross obstacles between discrete states are developed for a class of dynamics; second, the theory of successive convexification is extended to allow for a general class of non-convex state constraints. Finally, simulations for a relevant example are presented to demonstrate the effectiveness of the proposed method.**

I. Introduction

The problem of avoiding obstacles with autonomous vehicles has been attracting immense research interest, and with the proliferation of self-driving cars, autonomous drone delivery systems, and the increasing availability of commercial off–the–shelf UAVs, interest will only continue growing. It is well known that convex optimization problems can be solved to global optimality very efficiently using a variety of methods [1]–[4]. However, incorporating avoidance regions or obstacles inherently makes the state space non-convex. As a consequence, a variety of methods have been used to find feasible solutions to the non-convex optimal control problem of finding obstacle avoidance trajectories [5]–[9].

Random sampling methods have been used to quickly find geometrically feasible trajectories [7]; these trajectories are then smoothed in order to make them feasible with respect to vehicle dynamics and actuator constraints. More recently, sequences of quadratic programs [10], or more generally sequences of convex programs are solved to obtain feasible solutions to non-convex optimal control problems [11]–[13]. These methods have the advantage of dealing with non-convexities while simultaneously ensuring that convex vehicle constraints are satisfied.

The goal of this paper is to develop a framework for obtaining physically feasible trajectories that avoid cylindrical or ellipsoidal obstacles. At a high level, this is accomplished by solving a convergent sequence of convex optimization problems, where the non-convex collision avoidance constraints in each problem are linearized about the solution of the previous iteration (a process called *successive convexification*). Earlier work in this area showed that such sequences of optimization problems converge to a locally optimal fixed point when the system dynamics are nonlinear [13] or when every constraint function is convex [14]. However, no general convergence results exist for problems with non-convex constraint functions solved via successive approximation methods.

There are two main contributions presented in this paper. *First*, the paper develops a method for incorporating inter-sample collision avoidance constraints into discrete trajectory optimization problems. Enforcing collision avoidance constraints solely at the discrete time epochs may lead to computed trajectories passing through obstacles between time epochs. One approach to mitigating this risk is to over-sample the time discretization, thereby reducing the duration of the inter-sample epochs. However, such approaches do not guarantee that the obstacle avoidance constraints will be satisfied between discrete states, and unnecessarily introduce more solution variables. The methods presented in this paper provide a systematic approach for eliminating inter-sample collisions by introducing appropriate state and control constraints. *Second*, the paper extends results in [13] (which focuses solely on non-convex dynamics constraints) to treat a more general class of non-convex state constraints. These theoretical guarantees ensure the successful execution of the successive convexification process, which is critical for safe autonomous operations.

Notation: The set of natural numbers is denoted by $\mathbb{N}$ and for $q, r \in \mathbb{N}$, $\mathcal{N}_q^r := \{q, q+1, \ldots, r\}$ if $0 \leq q \leq r$ and $\mathcal{N}_q^r := \emptyset$ if $q > r$. The set of real numbers is denoted by $\mathbb{R}$, and the set of non-negative reals is denoted by $\mathbb{R}_+$. $\mathcal{C}^q$ for some $q \in \mathbb{N}$ is the set of q continuously differentiable functions. Finally, $I_{n\times n}$ is the multiplicative identity in $\mathbb{R}^{n\times n}$ for $n \in \mathbb{N}$.

II. Problem Formulation

Consider the discrete–time, finite–horizon, constrained trajectory optimization problem specified as follows:

$$\begin{aligned} \underset{z}{\text{minimize:}} \quad & \sum_{k=0}^{N-1} \ell(x_{k+1}, u_k) \\ \text{subject to:} \quad & \forall k \in \mathcal{N}_0^{N-1},\ x_{k+1} = f(x_k, u_k, t_{k+1}), \\ & \forall k \in \mathcal{N}_0^{N-1},\ u_k \in \mathcal{U}, \\ & \forall k \in \mathcal{N}_1^{N},\quad x_k \in \mathcal{X}, \\ & \qquad\qquad\ x_0 \in \mathcal{X}_0 \subset \mathcal{X}, \\ & \qquad\qquad\ x_N \in \mathcal{X}_f \subset \mathcal{X}, \end{aligned} \tag{1}$$

[1]University of Washington, Department of Aeronautics and Astronautics, 3940 Benton Ln, Seattle, WA, 98105, USA dandueri@uw.edu, yqmao@uw.edu, behcet@uw.edu
[2]United Technology Research Center, 411 Silver Ln, East Hartford, CT, 06118, USA mianzt@utrc.utc.com, dingj@utrc.utc.com

where $z := \{x_1^T, \ldots, x_N^T, u_0^T, \ldots, u_{N-1}^T\}^T \in \mathbb{R}^{N(n_x+n_u)}$, $\ell : \mathbb{R}^{N(n_x+n_u)} \to \mathbb{R}$ is the objective function, N is the number of discrete time steps, each $x_k \in \mathbb{R}^{n_x}$ is the state at time t_k, each $u_k \in \mathbb{R}^{n_u}$ is the control at time t_k, $f(x,u,t)$ maps the state x and control u to the state $x(t)$ (the state at time t), $\mathcal{U} \subset \mathbb{R}^{n_u}$ is the set of permissible controls, $\mathcal{X} \subset \mathbb{R}^{n_x}$ is the set of feasible states, $\mathcal{X}_0 \subseteq \mathcal{X}$ is the set of initial states, and $\mathcal{X}_f \subseteq \mathcal{X}$ is the set of final states. Moreover, we break the feasible set of states $\mathcal{X}$ into

$$\mathcal{X} := \mathcal{X}_C \cap \mathcal{X}_{NC}, \tag{2}$$

where $\mathcal{X}_C$ is a convex, compact set and $\mathcal{X}_{NC}$ is the non-convex set of states that avoids obstacles.

A. Assumptions

This paper partly deals with extending previous trust-region-based convergence results [13] to allow for a more general class of non-convex state and control constraints. We make the following assumptions to obtain these results:

1) $f(x,u,t) \in \mathcal{C}^2$.
2) $\ell(x_{k+1}, u_k)$ is convex with respect to x_{k+1} and u_k.
3) $\mathcal{X}_0$, $\mathcal{X}_f$, and $\mathcal{U}$ are convex, compact sets.
4) Linear Independence Constraint Qualification (LICQ): For any fixed point of the algorithm $(x^*, u^*) \in \mathcal{X} \times \mathcal{U}$, the gradient of active constraint functions evaluated at (x^*, u^*) has full row rank.

The first assumption is quite reasonable, since most dynamical systems of interest have smooth, continuous dynamics. Assumption 2 does not impose any practical limitations because non-convexities in the objective function can be moved to the constraints without loss of generality. The third assumption is also very mild, and a large class of non-convexities in the control space can be handled via lossless convexification [15], [16]. Satisfying LICQ can impose some practical limitations, but LICQ is always satisfied when the boundaries of the obstacles are not in contact with any other constraint boundaries (sufficient condition).

We make one final assumption in order to obtain the inter-sample collision avoidance constraints presented in this paper:

5) For any $(x_k, u_k) \in \mathcal{X} \times \mathcal{U}$, the state transition matrix admits finitely many $t^* \in [t_k, t_{k+1}]$ such that

$$\frac{\partial f(x_k, u_k, t^*)}{\partial t} = 0.$$

Assumption 5 is clearly satisfied for systems with state transition matrices that are polynomial (e.g., double integrator dynamics), sinusoidal, or exponential functions of time since they always have finitely many stationary points over compact sets.

III. Collision Avoidance

For the work presented in this paper, the obstacles are represented as cylinders with infinite height; this has the effect of avoiding a circular region in the projection of the trajectory onto the ground plane. Using this obstacle specification, the set $\mathcal{X}_{NC}$ in (2) is specified by:

$$\mathcal{X}_{NC} := \left\{ x \in \mathbb{R}^{n_x} : \ \|Tx - p_{c,i}\|_2 \geq r_i, \ \forall i \in \mathcal{N}_1^{n_{cyl}} \right\},$$

where $T : \mathbb{R}^{n_x} \to \mathbb{R}^2$ is a projection that maps the position state to the ground plane and satisfies $TT^T = I_{2\times 2}$, $p_{c,i} \in \mathbb{R}^2$ is the position of the center of cylinder i in the ground plane, $r_i \in \mathbb{R}_+$ is the corresponding radius, and n_{cyl} is the number of cylinders that need to be avoided. For the context of this paper, a collision occurs if any $x \notin \mathcal{X}_{NC}$ is a part of a trajectory.

A. Inter-sample Collision Avoidance

In order to avoid collisions for real vehicles, it is not sufficient to impose $x_k \in \mathcal{X}, \ k \in \mathcal{N}_1^N$ since it only ensures that the discrete points x_k do not collide with obstacles. The continuous path between any x_k and x_{k+1} may still intersect an obstacle (see Figure 1).

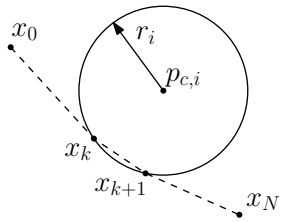

Fig. 1: Inter-sample collision between x_k and x_{k+1}.

Knowledge about the continuous path between any two discrete points is derived from the state transition matrix, $f(x,u,t)$. The function, $h_i(x,u,t)$, can be used to determine whether a collision occurs with cylinder i at any time t, and is defined as follows:

$$h_i(x,u,t) := \|Tf(x,u,t) - p_{c,i}\|_2^2 - r_i^2.$$

Then, the set of states and controls that avoid collisions along the path from time t_A to t_B is given by:

$$\begin{aligned}\mathcal{H}(t_A, t_B) := \big\{ (x,u) \in \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} : \ h_i(x,u,t) \geq 0, \\ \forall t \in (t_A, \ t_B), \ \forall i \in \mathcal{N}_1^{n_{cyl}} \big\}.\end{aligned} \tag{3}$$

Thus, the following constraint guarantees that all continuous sub-arcs between discrete points remain feasible:

$$\forall k \in \mathcal{N}_0^{N-1}, \ (x_k, u_k) \in \mathcal{H}(t_k, t_{k+1}). \tag{4}$$

The set described in (3) imposes a constraint for each of the infinitely many time instances between t_A and t_B, and must therefore be re-expressed into a form that is suitable for numerical optimization. This is accomplished by identifying the times at which $h_i(x,u,t)$ attains its minima with respect to time. The set $\mathcal{T}_i^*(x,u,t_A,t_B)$ describes the times in the range $(t_A, \ t_B) \in \mathbb{R}_+$ at which $h_i(x,u,t)$ attains a minimum with respect to time, and is defined as

$$\begin{aligned}\mathcal{T}_i^*(x,u,t_A,t_B) := \Bigg\{ t \in (t_A, \ t_B) : \frac{\partial h_i(x,u,t)}{\partial t} = 0, \\ \frac{\partial^2 h_i(x,u,t)}{\partial t^2} > 0 \Bigg\}.\end{aligned}$$

Note that $\mathcal{T}_i^*(x,u,t_A,t_B)$ is a finite set by Assumption 5. Moreover, if $\mathcal{T}_i^*(x,u,t_A,t_B)=\emptyset$, then no minima occur on the continuous arc from state x using control u over the horizon $(t_A,\ t_B)$ and no additional constraints are necessary (since $x_k\in\mathcal{X},\ k\in\mathcal{N}_1^N$ ensures the endpoints are feasible). On the other hand, if $\mathcal{T}_i^*(x,u,t_A,t_B)\neq\emptyset$, a constraint must be imposed on the value of each of the finitely many distinct minima, leading to the following equivalent reformulation:

$$\begin{aligned}\mathcal{H}(t_A,t_B)=\big\{&(x,u)\in\mathbb{R}^{n_x}\times\mathbb{R}^{n_u}:\ h_i(x,u,t^*)\geq 0,\\ &\forall\, t^*\in\mathcal{T}_i^*(x,u,t_A,t_B),\ \forall i\in\mathcal{N}_1^{n_{cyl}}\big\}.\end{aligned}\tag{5}$$

As a result of the above reformulation, only a finite number of (non-convex) constraints need to be imposed in order to guarantee that the continuous path between two consecutive discrete states avoids collisions. Note that finding the set $\mathcal{T}_i^*(x,u,t_A,t_B)$ may involve using numerical techniques (e.g., Newton-Raphson). Also, note that there are analytic expressions for the roots of polynomials of degree four or less (see Abel–Ruffini theorem).

B. Non-Convex Obstacle Avoidance Formulation

Combining the problem formulation in (1) and the inter-sample constraints in (4) and (5), we arrive at the non-convex obstacle avoidance problem formulation in (6).

$$\begin{aligned}\underset{z}{\text{minimize:}}\quad & \sum_{k=0}^{N-1}\ell(x_{k+1},u_k),\\ \text{subject to:}\quad & \forall k\in\mathcal{N}_0^{N-1},\ x_{k+1}=f(x_k,u_k,t_{k+1}),\\ & \forall k\in\mathcal{N}_0^{N-1},\ u_k\in\mathcal{U},\\ & \forall k\in\mathcal{N}_1^{N},\quad x_k\in\mathcal{X}_C,\\ & \forall k\in\mathcal{N}_1^{N},\quad x_k\in\mathcal{X}_{NC},\\ & \forall k\in\mathcal{N}_0^{N-1},\ (x_k,u_k)\in\mathcal{H}(t_k,t_{k+1}),\\ & x_0\in\mathcal{X}_0\subset\mathcal{X},\\ & x_N\in\mathcal{X}_f\subset\mathcal{X},\end{aligned}\tag{6}$$

where the fourth and fifth constraints in (6) represent non-convex state and control inequality constraints.

IV. Successive Convexification

The objective here is to capture the non-convex inequality constraints in (6) within the successive convexification framework (which previously only treated non-convex *equality* constraints stemming from non-convex dynamics). We begin by expressing non-convex optimization problems in a more general setting. Since non-convex state and control constraints are treated in the same fashion, we utilize the joint variable $z:=\{x_1^T,\ldots,x_N^T,u_0^T,\ldots,u_{N-1}^T\}^T\in\mathbb{R}^M$ where $M=N(n_x+n_u)$, and let $\mathcal{Z}$ be the appropriately repeated Cartesian product of $\mathcal{X}_C$, $\mathcal{U}$, $\mathcal{X}_0$, and $\mathcal{X}_f$ such that every $z\in\mathcal{Z}$ satisfies the convex constraints in (6). Note that $\mathcal{Z}$ is a convex, compact set by Assumption 3. Similarly, we denote $g_0(z)=\sum_{k=0}^{N-1}\ell(x_{k+1},u_k)$ and stack the non-convex constraints to form the column vector $g(z)$. Using these definitions, a general non-convex optimal control problem can be expressed as:

$$\begin{aligned}\underset{z}{\text{minimize:}}\quad & g_0(z)\\ \text{subject to:}\quad & g_j(z)=0,\quad j\in\mathcal{N}_1^e,\\ & g_j(z)\leq 0,\quad j\in\mathcal{N}_{e+1}^s,\\ & g_j(z)=0,\quad j\in\mathcal{N}_{s+1}^r,\\ & g_j(z)\leq 0,\quad j\in\mathcal{N}_{r+1}^q,\end{aligned}\tag{7}$$

where $g_j(z),\ j\in\mathcal{N}_1^s$ represent non-convex constraints, the indices e,s,r,q separate equality and inequality constraints, and $\mathcal{Z}$ can be equivalently expressed as

$$\mathcal{Z}=\left\{z\in\mathbb{R}^M:\, g_j(z)=0, j\in\mathcal{N}_{s+1}^r, g_j(z)\leq 0, j\in\mathcal{N}_{r+1}^q\right\}.$$

Clearly, the non-convex inequality constraints in (6) fall into the constraint $g_j(z)\leq 0,\ j\in\mathcal{N}_{e+1}^s$ in (7).

A. Convex Subproblems

A natural way to convexify (7) is to linearize the non-convex constraints $g_j(z),\ j\in\mathcal{N}_1^s$ by using first order Taylor approximations. In general, the solution of the linearized problem will be different from that of the original, non-convex problem. Therefore, the objective is to utilize an algorithm whose solution at least satisfies the first order optimality conditions of the original problem. As it turns out, one can achieve this by conducting the linearization successively. Let the solution of the $(k-1)^{th}$ succession be z^{k-1} and denote $d=z-z^{k-1}$. Then, the (convex) linearized subproblem is given by:

$$\begin{aligned}\underset{d}{\text{minimize:}}\quad & g_0(d+z^{k-1})\\ \text{subject to:}\quad & g_j(z^{k-1})+\nabla g_j(z^{k-1})\,d=0,\quad j\in\mathcal{N}_1^e,\\ & g_j(z^{k-1})+\nabla g_j(z^{k-1})\,d\leq 0,\quad j\in\mathcal{N}_{e+1}^s,\\ & d+z^{k-1}\in\mathcal{Z},\end{aligned}\tag{8}$$

where $\nabla g_j(z)$ is the gradient of the j^{th} constraint evaluated at z. The linearization renders the problem convex, but also introduces two issues: artificial infeasibility and approximation error. These issues are addressed by introducing penalty functions and trust regions, respectively.

Artificial infeasibility refers to cases in which (7) has feasible solutions, yet (8) is infeasible. This issue may arise when z^{k-1} is not a feasible solution of (7), and is addressed by relaxing the linearized non-convex constraints in (8) and penalizing their violation in the objective function [13]. Note that this is an implicit way to introduce slack variables. The 1-norm used in [13] degenerates to a finite sum of absolute values. Therefore, with the addition of inequality constraints, the penalty function used in [13] is extended to become

$$\begin{aligned}L(d,\lambda):=g_0(d+z^{k-1})&+\sum_{j=1}^{e}\lambda_j\left|g_j(z^{k-1})+\nabla g_j(z^{k-1})\,d\right|\\ &+\sum_{j=e+1}^{s}\lambda_j\max\left\{0,g_j(z^{k-1})+\nabla g_j(z^{k-1})\,d\right\},\end{aligned}\tag{9}$$

where the scalars $\lambda_j \geq 0$ are penalty weights and $\lambda := [\lambda_1, \lambda_2, \ldots, \lambda_s]$. Note that the corresponding penalty function prior to linearization is specified by

$$J(z,\lambda) := g_0(z) + \sum_{j=1}^{e} \lambda_j |g_j(z)| + \sum_{j=e+1}^{s} \lambda_j \max\{0, g_j(z)\}. \tag{10}$$

$J(z,\lambda)$ is also the most commonly used exact penalty function in constrained optimization literature.

As previously mentioned, the second issue introduced by the linearization is the approximation error. If left unchecked, it has the potential to render the optimization unbounded. To mitigate this risk, we ensure that z^k does not deviate significantly from z^{k-1}, via a *trust region* on the new step, $\|d\| \leq \Delta$. The rationale is that the linear approximation is only accurate near the point the constraints were linearized about. Thus, the final formulation of the convex subproblem is specified in (11).

$$\begin{aligned} \underset{d}{\text{minimize:}} \quad & L(d,\lambda) \\ \text{subject to:} \quad & \|d\| \leq \Delta, \\ & d + z^{k-1} \in \mathcal{Z}. \end{aligned} \tag{11}$$

B. Collision Avoidance Algorithm

The overall successive-convexification-based collision avoidance algorithm is presented in this section. The goal here is to obtain feasible, locally optimal trajectories to the non-convex trajectory optimization problem.

Data: $\alpha, \epsilon, \Delta, \rho_0, \rho_1, \rho_2 \in \mathbb{R}_+$ such that $\rho_0 < \rho_1 < \rho_2$ and $\alpha > 1$, N, λ, and the implicit form of the sets $\mathcal{X}_C$, $\mathcal{X}_{NC}$, $\mathcal{U}$, $\mathcal{X}_0$, and $\mathcal{X}_f$
Result: a feasible, locally optimal sequence of states and controls, z^*
begin
  Solve (1) with $\mathcal{X} = \mathcal{X}_C$ to get z^0 (ignore obstacles);
  if $x_k \in \mathcal{X}_{NC},\ k \in \mathcal{N}_1^N$ *and* $(x_k, u_k) \in \mathcal{H}(t_k, t_{k+1}),\ k \in \mathcal{N}_0^{N-1}$ **then**
    No collisions occurred, return $z^* = z_0$;
  else
    $\Delta L^1 = 10\epsilon$;
    $k = 1$;
    while $\Delta L^k \geq \epsilon$ **do**
      Solve (11) to get d and $z^k = z^{k-1} + d$;
      Compute $J(z^k, \lambda)$ using (10);
      Compute $\Delta L^k := J(z^{k-1}, \lambda) - L(d, \lambda)$;
      Compute $\Delta J^k := J(z^{k-1}, \lambda) - J(z^k, \lambda)$;
      Compute $\rho := \Delta L^k / \Delta J^k$;
      if $\rho < \rho_0$ **then**
        Reject this step and $\Delta = \Delta/\alpha$;
      else
        $\Delta = \begin{cases} \Delta/\alpha, & \rho < \rho_1, \\ \Delta, & \rho_1 \leq \rho < \rho_2, \\ \alpha\Delta, & \rho_2 \leq \rho. \end{cases}$;
        $k = k + 1$;
    return $z^* = z^k$;

Here, the initial reference states and controls are obtained by solving (1) without considering the obstacles, thus the initial references may be infeasible with respect to the obstacles. After each successful iteration of the loop, the reference states and controls are updated to the most recent solution of (11). The success of each iteration is based on the criteria commonly used in the trust region methods literature (e.g. [17]).

V. Convergence Analysis

In this section, we extend the convergence result from [13] to include a broader class of state constraints. The objective is to prove that the algorithm in IV-B converges to a local optimum of (7), and therefore a local optimum of (6). The procedure for showing the convergence of this successive convexification method is largely the same as [13]. Thus, in the interest of compactness, only statements that need modification are presented here.

First, the non-convex penalty problem is expressed as

$$\begin{aligned} \text{minimize:} \quad & J(z,\lambda) \\ \text{subject to:} \quad & g_j(z) = 0, \quad j \in \mathcal{N}_{s+1}^r, \\ & g_j(z) \leq 0, \quad j \in \mathcal{N}_{r+1}^q. \end{aligned} \tag{12}$$

The difference between (12) and the conventional penalty formulation is that the convex constraints are not relaxed. As a result, (12) is still a constrained problem, but since $g_j(z) = 0$ and $g_j(z) \leq 0$ for $j \in \mathcal{N}_{s+1}^q$ are convex constraints, they can be incorporated into the convex programming framework without any approximation.

The first step in the proof is to show the exactness of the penalty function, $J(z,\lambda)$, that is, to show that the penalty problem recovers the optimality of the original problem. First, given a feasible point of (12) $z^* \in \mathcal{Z}$, we define the index set of active convex inequality constraints at z^* to be

$$\mathcal{I}_{ac}(z^*) = \left\{ j \in \mathcal{N}_{r+1}^q : \ g_j(z^*) = 0 \right\}.$$

Then, since LICQ is satisfied by Assumption 4, the tangent cone of these active convex constraints is represented by

$$\begin{aligned} T(z^*) = \big\{ \alpha d \in \mathbb{R}^M : \ \alpha \geq 0; \nabla g_j(z^*)^T d = 0, \ j \in \mathcal{N}_{s+1}^r; \\ \nabla g_j(z^*)^T d \leq 0, \ j \in \mathcal{I}_{ac}(z^*) \big\}. \end{aligned}$$

With $T(z^*)$ defined, the first order necessary conditions for a point z^* to be a local optimum of the original, non-convex problem in (7) are given by the very well-known Theorem 1 (shown here only to introduce necessary notation).

Theorem 1 (Karush–Kuhn–Tucker (KKT)). *If the original problem in* (7) *satisfies LICQ, and z^* is a local optimum of that problem, then there exist Lagrange multipliers μ_j such that*

$$\mu_j \geq 0 \text{ and } \mu_j g_j(z^*) = 0, \ \forall\, j \in \mathcal{N}_{e+1}^s,$$

and $\forall\, d \in T(z^*)$,

$$\left(\nabla g_0(z^*) + \sum_{j=1}^{e} \mu_j \nabla g_j(z^*) + \sum_{j=e+1}^{s} \mu_j \nabla g_j(z^*) \right) d \geq 0.$$

We call such a z^* a KKT point.

We now address the changes to [13] introduced by incorporating non-convex inequality constraints. To start, the set of constrained stationary points of the penalty problem (12) is defined as

$$S := \{z : z \in \mathcal{Z}, \text{ and } 0 \in \mathcal{D}(z)\},$$

where

$$\mathcal{D}(z) := \partial J(z) \oplus \left\{ y : \mu_j \geq 0, y = \sum_{j=s+1}^{r} \mu_j \nabla g_j(z) + \sum_{j \in \mathcal{I}_{ac}(z)} \mu_j \nabla g_j(z) \right\}, \tag{13}$$

where $\partial J(z)$ is the Generalized Directional Derivative (GDD) as in Definition 2 of [13], and $\oplus$ denotes the Minkowski sum of two sets.

Now, we are ready to state our next theorem, which provides the exactness of the penalty function $J(z, \lambda)$. The theorem itself is well established in the literature [18] and its proof follows the same reasoning as that used in [13], so it is omitted.

Theorem 2. *[18] If z^* is a KKT point of the original problem in* (7) *with multipliers $\bar{\mu}_j$, and if the penalty weight λ satisfies*

$$\lambda_j \geq |\bar{\mu}_j|, \forall\, j \in \mathcal{N}_1^s,$$

then z^* is a constrained stationary point of the penalty problem in (12)*, i.e. $z^* \in S$. Conversely, if a constrained stationary point of the penalty problem z^* is feasible for the original problem, then it is also a KKT point of the original problem.*

From Theorem 2, one can see that the objective of the successive convexification algorithm is to find such constrained stationary points. In fact, the algorithm produces a sequence of points $\{z^k\}$, and the goal is to prove that this sequence converges to constrained stationary points. This convergence result has been established in Theorem 3 and Theorem 4 of [13], and their proofs are not affected by the introduction of the non-convex inequality constraints and convex equality constraints added in this paper. To be precise, the added term $\max\{0, \cdot\}$ in (10) has the same properties as $|\cdot|$, and the additional term $\sum_{j=s+1}^{r} \mu_j \nabla g_j(z)$ in (13) doesn't change the fact that $\mathcal{D}(z^*)$ is a convex set, which is part of the proof. Thus, we present the final result in the next theorem without re-stating its proof.

Theorem 3. *The "predicted" changes ΔL^k defined inside the algorithm in Section IV-B are non-negative for any k.*

Also, $\Delta L^{k^*} = 0$ implies that z^{k^*} is a constrained stationary point of the penalty problem in (12)*.*

Furthermore, if $\forall k$, $\Delta L^k > 0$, the algorithm generates an infinite sequence $\{z^k\}$. Then, $\{z^k\}$ has limit points, and any limit point z^* is a constrained stationary point of the penalty problem in (12)*.*

VI. Illustrative Example

In this section, we use double integrator dynamics to illustrate the process of imposing (4) in practice. For each $k \in \mathcal{N}_0^{N-1}$ & $i \in \mathcal{N}_1^{n_{cyl}}$, the objective is to impose $h_i(x_k, u_k, \tau) \geq 0, \forall\, \tau \in \mathcal{T}_i^*(x_k, u_k, t_k, t_{k+1})$. The first step is to linearize this non-convex constraint about $(\bar{x}_k, \bar{u}_k, \bar{t}^*), \forall\, \bar{t}^* \in \mathcal{T}_i^*(\bar{x}_k, \bar{u}_k, t_k, t_{k+1})$ via a first order Taylor Series approximation:

$$\bar{h}_i(x_k, u_k, t^*) = h_i(\bar{x}_k, \bar{u}_k, \bar{t}^*) + \left.\frac{\partial h_i}{\partial x_k}\right|_{\substack{x_k=\bar{x}_k \\ u_k=\bar{u}_k}} (x_k - \bar{x}_k) + \left.\frac{\partial h_i}{\partial u_k}\right|_{\substack{x_k=\bar{x}_k \\ u_k=\bar{u}_k}} (u_k - \bar{u}_k),$$

and incorporate $\bar{h}_i(x_k, u_k, t^*)$ into (9). Let the state be given by $x = [p^T, v^T]^T$, where $p, v \in \mathbb{R}^2$. Then for $t \in (t_k, t_{k+1})$, the position of the vehicle is given by

$$Tf(x_k, u_k, t) = p(t) = p_k + v_k(t - t_k) + \frac{1}{2}(u_k + g)(t - t_k)^2.$$

For these dynamics, $\mathcal{T}_i^*(\bar{x}_k, \bar{u}_k, t_k, t_{k+1})$ is found by noting that $h_i(\bar{x}_k, \bar{u}_k, t)$ can be expressed as

$$h_i(\bar{x}_k, \bar{u}_k, t) = a_4 t^4 + a_3 t^3 + a_2 t^2 + a_1 t + a_0,$$

where $a_j,\ j \in \mathcal{N}_0^4$ are known coefficients (since $\bar{x}_k, \bar{u}_k$, and t_k are known). Thus, finding the set $\mathcal{T}_i^*(\bar{x}_k, \bar{u}_k, t_k, t_{k+1})$ is equivalent to finding the roots of a cubic polynomial with known coefficients, which can be done analytically using Cardano's formula. As a final example, we denote $\tau = \bar{t}^* - t_k$ and $\delta = \bar{p}_k - p_{c,i}$ to express the partial of h_i with respect to control as

$$\frac{\partial h_i}{\partial u_k} = 2\delta^T \tau^2 + (u_k + g)^T \tau^3 + \frac{\partial h_i}{\partial t^*} \frac{\partial t^*}{\partial u_k},$$

where the last term reflects the dependence of $\mathcal{T}_i^*$ on u_k, and $\frac{\partial t^*}{\partial u_k}$ can be approximated via the finite difference method.

VII. Numerical Results

This section describes several numerical examples which illustrate the obstacle avoidance capability and numerical convergence properties of the proposed planning algorithm. The application scenario considered for these examples is that of trajectory optimization for multi-rotor unmanned aerial vehicles in obstacle-rich environments. For onboard trajectory planning in such applications, the main requirement is that of rapid convergence to obstacle free and dynamically feasible trajectories.

A. Multi-Rotor Trajectory Planning Problem

For trajectory planning purposes, the motion of the multi-rotor is modeled using double integrator dynamics. The state at time t_k is defined to be $x_k = [p_k^T, v_k^T]^T$ where $p_k \in \mathbb{R}^3$ is the vehicle position and $v_k \in \mathbb{R}^3$ is the corresponding velocity. The discrete time model is given by

$$x_{k+1} = f(x_k, u_k, t_{k+1}) = Ax_k + B(u_k + g)$$

TABLE I: Simulation Parameters

Parameter	Value	Parameter	Value
N	17	$p_{c,1}$	$[-3,\ 0]^T$ m
V_{max}	2 m/s	r_1	3 m
T_{max}	40 N	$p_{c,2}$	$[4,\ -1]^T$ m
m	3 kg	r_2	2 m
g	$[0,\ 0,\ -9.81]^T$ m/s^2	$p_{c,3}$	$[8,\ 1]^T$ m
θ_{cone}	30 deg	r_3	1 m
Δ	3.15	α	1.2
ρ_0	0	ρ_1	0.25
ρ_2	2	ϵ	1×10^{-5}

where A and B correspond to double integrator dynamics with time step Δt, and are given by:

$$A = \begin{bmatrix} I_{3\times3} & \Delta t \cdot I_{3\times3} \\ 0_{3\times3} & I_{3\times3} \end{bmatrix},\ B = \begin{bmatrix} \frac{1}{2}\Delta t^2 \cdot I_{3\times3} \\ \Delta t \cdot I_{3\times3} \end{bmatrix}.$$

As in Section VI, the above dynamics render $h_i(x, u, t)$ a quartic with respect to time.

The convex set of feasible states, $\mathcal{X}_C$ is given by:

$$\mathcal{X}_C := \left\{(p,\ v) \in \mathbb{R}^6\ :\ \|v\|_2 \leq V_{max}\right\},$$

where V_{max} is a prescribed velocity norm upper bound. The convex set of controls is given by:

$$\mathcal{U} := \left\{u \in \mathbb{R}^3 : \hat{n}^T u \geq \|u\| \cos(\theta_{cone}),\ \|u\|_2 \leq T_{max}/m\right\},$$

where θ_{cone} defines a thrust cone angle that confines the thrust vector to a cone pointing towards the ceiling, T_{max} is the maximum thrust capacity, and m is the vehicle mass.

For the obstacle avoidance scenario, three cylindrical shapes are used to represent obstacles present in the environment, with centers and radii defined by the parameters $p_{c,i}, r_i,\ i = 1, 2, 3$. The objective function is a minimum time heuristic chosen to promote contact between the trajectory and the boundary of the obstacles. A summary of the simulation parameters selected for the scenario can be found in Table I.

B. Simulation Results

The problem defined in Section VII-A is solved using CVX [19] in MATLAB with the following initial and final conditions as test cases:

$$p_0 = [-7,\ 0,\ 0]^T m,\ v_0 = [0,\ 0,\ 0]^T m/s,$$

$$p_f = [8,\ -0.1,\ 0.7]^T m,\ v_f = [0,\ 0,\ 0]^T m/s.$$

Figure 2 shows a 3-D view of the converged trajectory, note that there is an altitude change. The vehicle path skims the cylinders but does not pass into the circular area, even in the inter-sample regions. Note that the velocity constraint is active for most of the trajectory (since the objective function was minimum time of flight), and the remaining convex constraints are also satisfied (Figure 3).

The algorithm convergence properties, as discussed in Section V, are demonstrated by the plots shown in Figure 4. In particular, the improvement in linearized objective function (ΔL), and the state and control sequence norms converge quickly in a small number of iterations. The state and control sequence norms were determined by finding the norm of the change in state and control between successive iterations of the optimization algorithm, defined as

$$\Delta X^k = \sum_{j=1}^{N} \|x_j^k - x_j^{k-1}\|_2,\ \Delta U^k = \sum_{j=0}^{N-1} \|u_j^k - u_j^{k-1}\|_2,$$

where $\{x_j^k\}$, $\{u_j^k\}$ are the state and control sequences computed in iteration k. For most simulations, algorithm

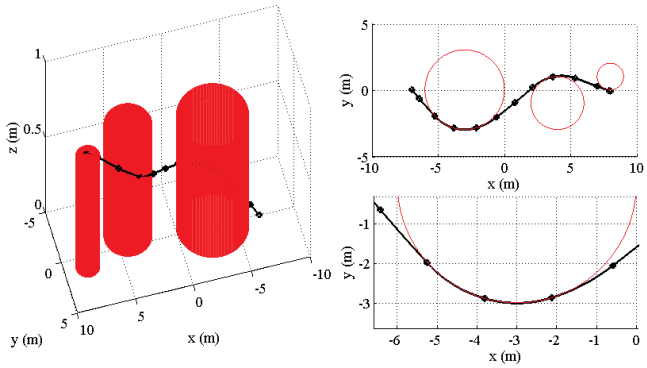

Fig. 2: Converged Feasible Path Views: Isometric, XY-Plane, Inter-sample Region Zoom

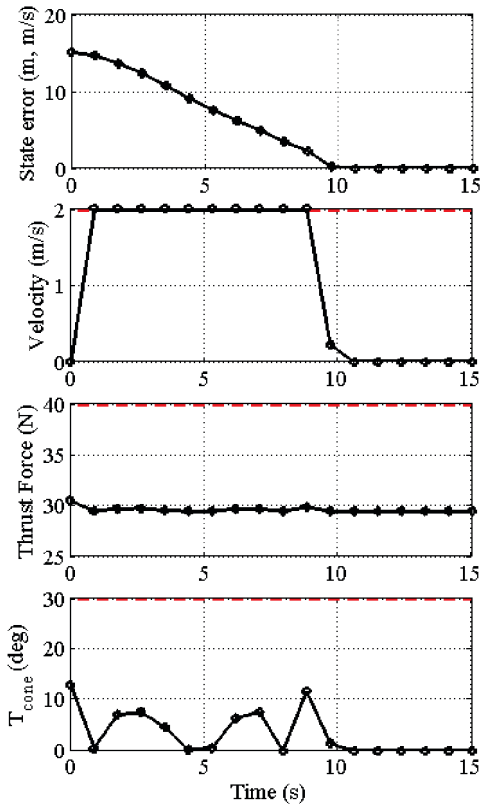

Fig. 3: Constraint Overview

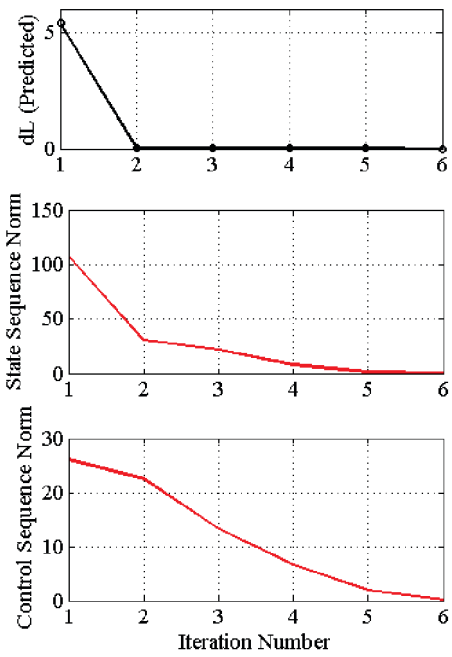

Fig. 4: Convergence Overview

convergence is achieved within 5 to 11 iterations.

VIII. Conclusion

This paper describes a trajectory generation approach for autonomous vehicle motion planning. The approach is well-suited for the onboard generation of dynamically feasible vehicle trajectories in environments with relatively simple obstacle geometries. This could be the case, for example, for unmanned aerial vehicles avoiding no-fly zones in a given airspace. Operation in more complex environments would likely require coupling of this approach with higher-level mission planning using a hierarchical framework such as in [20].

Future work could include extensions to MPC, and experimental implementation and testing (on a UAV). Moreover, the generalization of the constraints to handle more complex obstacle geometries could enable the application of the approach to complex, cluttered environments. Free space and belief space planning can be integrated for addressing uncertainty. Finally, it is also important to investigate conditions which ensure the timely convergence of the optimization algorithm for reliability and safety assurances on aerial vehicle platforms.

References

[1] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[2] Y. Nesterov and A. Nemirovsky, *Interior-point Polynomial Methods in Convex Programming*. SIAM, 1994.

[3] S. Mehrotra, "On the implementation of a primal-dual interior point method," *SIAM Journal on optimization*, vol. 2, no. 4, pp. 575–601, 1992.

[4] D. Dueri, J. Zhang, and B. Açikmese, "Automated Custom Code Generation for Embedded, Real-time Second Order Cone Programming," in *IFAC Proceedings Volumes*, vol. 47, no. 3. Elsevier, 2014, pp. 1605–1612.

[5] S. Karaman and E. Frazzoli, "Incremental sampling-based algorithms for optimal motion planning," *Robotics Science and Systems VI*, vol. 104, 2010.

[6] L. Yang, J. Qi, J. Xiao, and X. Yong, "A literature review of uav 3d path planning," in *Intelligent Control and Automation (WCICA), 2014 11th World Congress on*. IEEE, 2014, pp. 2376–2381.

[7] M. Elbanhawi and M. Simic, "Sampling-based robot motion planning: A review," *IEEE Access*, vol. 2, pp. 56–77, 2014.

[8] A. Richards, T. Schouwenaars, J. P. How, and E. Feron, "Spacecraft trajectory planning with avoidance constraints using mixed-integer linear programming," *Journal of Guidance, Control, and Dynamics*, vol. 25, no. 4, pp. 755–764, 2002.

[9] A. Richards and O. Turnbull, "Inter-sample avoidance in trajectory optimizers using mixed-integer linear programming," *International Journal of Robust and Nonlinear Control*, vol. 25, no. 4, pp. 521–526, 2015.

[10] C. Büskens and H. Maurer, "Sqp-methods for solving optimal control problems with control and state constraints: adjoint variables, sensitivity analysis and real-time control," *Journal of computational and applied mathematics*, vol. 120, no. 1, pp. 85–108, 2000.

[11] X. Liu and P. Lu, "Solving nonconvex optimal control problems by convex optimization," *Journal of Guidance, Control, and Dynamics*, vol. 37, no. 3, pp. 750–765, 2014.

[12] M. Szmuk and B. Açıkmeşe, "Successive convexification for fuel-optimal powered landing with aerodynamic drag and non-convex constraints," in *AIAA Guidance, Navigation, and Control Conference*, 2016, p. 0378.

[13] Y. Mao, M. Szmuk, and A. A. Behcet, "Successive convexification of non-convex optimal control problems and its convergence properties," in *IEEE Conference on Decision and Control*, December 2016.

[14] Y. Mao, D. Dueri, M. Szmuk, and B. Açıkmeşe, "Successive convexification of non-convex optimal control problems with state constraints," in *submitted to International Federation of Automatic Control*, November 2016.

[15] B. Açıkmeşe and S. R. Ploen, "Convex programming approach to powered descent guidance for Mars landing," *AIAA Journal of Guidance, Control and Dynamics*, vol. 30, no. 5, pp. 1353–1366, 2007.

[16] M. Harris and B. Açıkmeşe, "Lossless convexification of non-convex optimal control problems for state constrained linear systems," *Automatica*, vol. 50, no. 9, pp. 2304–2311, 2014.

[17] A. R. Conn, N. I. Gould, and P. L. Toint, *Trust region methods*. Siam, 2000, vol. 1.

[18] S.-P. Han and O. L. Mangasarian, "Exact penalty functions in nonlinear programming," *Mathematical programming*, vol. 17, no. 1, pp. 251–269, 1979.

[19] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming, version 2.1," http://cvxr.com/cvx, Mar. 2014.

[20] X. D. Ding, B. Englot, A. Pinto, A. Speranzon, and A. Surana, "Hierarchical multi-objective planning: From mission specifications to contingency management," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, May 2014, pp. 3735–3742.