A Hybrid Method for Online Trajectory Planning of Mobile Robots in Cluttered Environments

Leobardo Campos-Macías, *Student Member, IEEE,* David Gómez-Gutiérrez, *Member, IEEE,*
Rodrigo Aldana-López, *Student Member, IEEE,* Rafael de la Guardia and José I. Parra-Vilchis

***Abstract*—This paper presents a method for online trajectory planning in known environments. The proposed algorithm is a fusion of sampling-based techniques and model-based optimization via quadratic programming. The former is used to efficiently generate an obstacle-free path while the latter takes into account the robot dynamical constraints to generate a time-dependent trajectory. The main contribution of this work lies on the formulation of a convex optimization problem over the generated obstacle-free path that is guaranteed to be feasible. Thus, in contrast with previously proposed methods, iterative formulations are not required. The proposed method has been compared with state-of-the-art approaches showing a significant improvement in success rate and computation time. To illustrate the effectiveness of this approach for online planning, the proposed method was applied to the fluid autonomous navigation of a quadcopter in multiple environments consisting of up to two hundred obstacles. The scenarios hereinafter presented are some of the most densely cluttered experiments for online planning and navigation reported to date.**

***Index Terms*—Aerial Robotics, Autonomous Agents, Autonomous Vehicle Navigation, Collision Avoidance, Motion and Path Planning**

I. Introduction

THE development of algorithms to enable mobile robotic systems to navigate in complex dynamic environments are of paramount importance towards fully autonomous systems performing complicated tasks. Recently, there has been great progress in online motion planning in structured environments, most notably for autonomous cars [1], [2]. However, to achieve similar levels of autonomy in three dimensional unstructured environments further development is required. Such capabilities are critical, for instance, when involved in search and rescue missions inside collapsed buildings.

Among the most efficient methods available for motion planning of mobile robots are sampling-based methods [3]–[6] and optimization-based methods [7], [8]. In the former category, algorithms such as Rapidly-exploring Random Trees (*RRT*) [3], [4] or Batch Informed Trees (*BIT**) [5], [6], have demonstrated to be effective in determining obstacle-free paths online. However, their extension to trajectory planning taking into account the robot dynamical constraints often results in time consuming algorithms that are no longer applicable online [9], [10]. In the latter category, optimization-based methods take advantage of the underlying mathematical model and have proven to be effective in handling the robot dynamical constraints [7], [11], [12]. The downside is that optimization problems often result in nonlinear programs [7], [8] or Sequential Convex Programs (*SCP*) [12]–[14] for which no efficient solvers exist. This characteristic makes them unsuitable for online planning in cluttered environments.

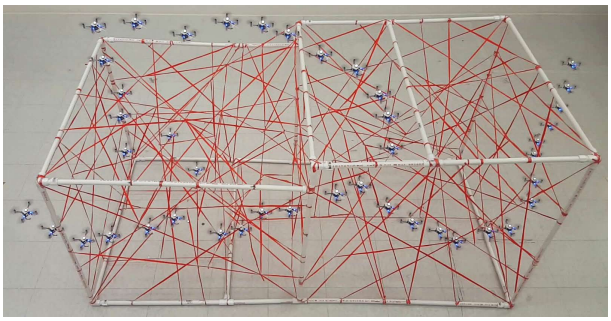


Fig. 1. Composite image of a quadcopter executing a trajectory planned in a fraction of a second for an obstacle-dense environment. See video at https://youtu.be/DJ1IZRL5t1Q.

To take advantage of the efficiency of sampling-based algorithms and the inclusion of the dynamical constraints in optimization-based methods, hybrid approaches have been proposed [11], [15]–[17]. Among the most promising are *CHOMP* [16] and *STOMP* [17]. Both methods rely on the optimization of an objective function with smoothness and collision costs solved by functional gradient descent in *CHOMP* and by gradient-free candidate sampling in *STOMP*. Unfortunately, in cluttered environments reported results show low success rate, often converging to local minima or altogether failing to find feasible solutions. While hybrid approaches are promising, the main challenge for their broader use in online planning is creating formulations and heuristics that translate into optimization problems that can be solved efficiently, avoiding incremental or iterative solutions.

Regarding methods for trajectory planning in cluttered environments, in [8] semidefinite programming is used to segment the space into convex regions while a mixed-integer convex program determines the obstacle-free trajectory. Unfortunately results show infeasibility for online implementation since reported solutions are in the order of minutes. In [15] a free-space flight corridor is generated using an octree-based structure followed by a Quadratic Program (*QP*) formulation using

The authors are with the Multi-Agent Autonomous Systems Lab, Intel Labs, Intel Tecnología de México. (e-mail: l.e.camposmacias@ieee.org, david.gomez.g@ieee.org, a.rodrigo.aldanalopez@ieee.org, rafael.de.la.guardia@intel.com, jose.i.parra.vilchis@intel.com).

A video of the experiments can be found at https://youtu.be/DJ1IZRL5t1Q

Digital Object Identifier (DOI):10.1109/LRA.2017.2655145

connected polynomial functions for each overlapping convex region. The method assumes that each segment end-time is given, though, no methodology for automatically generating such times is provided. This is a major drawback as a bad selection may result in infeasible problems which will require iterative *QP* formulations, failing to provide an online solution. A similar approach is presented in [11], in which collision-free paths are obtained using the asymptotically optimal version of the *RRT* algorithm (*RRT**) proposed in [3], followed by smooth trajectories generation using a *QP* that has to be solved iteratively to determine the total time of the trajectory. Since iterative solutions are undesirable for online implementations and furthermore, as noted in [15], collisions may still occur as the trajectory deviates from the original path. To overcome this problem, the authors proposed to add intermediate waypoints, but no method to determine the number of waypoints needed is provided. In [14] a trajectory optimization algorithm inspired in *CHOMP* was presented, which organizes the workspace in convex regions to perform a *SCP*, improving the computation time of *CHOMP* but with the drawback that convex regions are hard to compute online. Also, inspired in *CHOMP*, in [18] an online trajectory optimization method for local replanning was presented. Unfortunately, it often converges to local minima or fails to find feasible solutions, obtaining a low success rate compared to sampling-based methods.

In this paper, a hybrid method for online trajectory planning in cluttered environments is proposed. Based on a path generated by a sampling-based planner a *QP* is defined taking into account the robot dynamical constraints. The proposed formulation ensures feasibility of the *QP*, avoiding the need to solve multiple optimization problems in an iterative or sequential fashion [11]–[14]. Comparisons with state-of-the-art approaches, such as [13] and [14], are presented showing superior performance in computation time and success rate. To illustrate its effectiveness, the proposed method was applied for the online planning of a palm-sized quadcopter in multiple scenarios consisting of up to two hundred static obstacles. These scenarios are some of the most densely cluttered for online planning of a quadcopter reported in the literature to date. One of these experiments is illustrated in Fig. 1.

II. Outline of the Proposed Method

A. Notation

Let $\mathscr{B}(p)$ be the smallest ball, centered in the robot's centroid with position p, containing the mobile robot. The center of $\mathscr{B}(p)$ is modeled as a free particle in a non-rotating frame with state $x = \begin{bmatrix} p^T & v^T & a^T \end{bmatrix}^T$, where $[\cdot]^T$ is the transpose operator, p is the position, v is the velocity and a is the acceleration of the particle. The configuration space of the particle is denoted by $\chi = [0,1]^d$, with $d \in \{2,3\}$ as its dimension. The obstacle region is χ_{obsReal} and the obstacle-free space is defined as $\chi_{\text{free}} = \{p \in \chi | \mathscr{B}(p) \cap \chi_{\text{obsReal}} = \emptyset\}$.

The notation $f(\cdot)$ is used to represent a continuous signal while $f[\cdot]$ represents a discrete signal. A path $\eta : \{0, \cdots, S\} \mapsto \chi_{\text{free}}$ is a sequence of position nodes. Given a discrete path $\eta[s]$ a continuous path $\eta(\mathbf{s})$ can be obtained as the concatenation of the line segments connecting consecutive nodes. A trajectory $p(t)$ is a time-dependent sequence of positions, with velocity $v(t)$, acceleration $a(t)$ and jerk $j(t)$. Continuous-time is represented by t while k represents discrete-time. Thus, for a time step h, $p(t)$ is a continuous-time trajectory while $p[k]$ is a discrete-time trajectory such that $p(kh) = p[k]$.

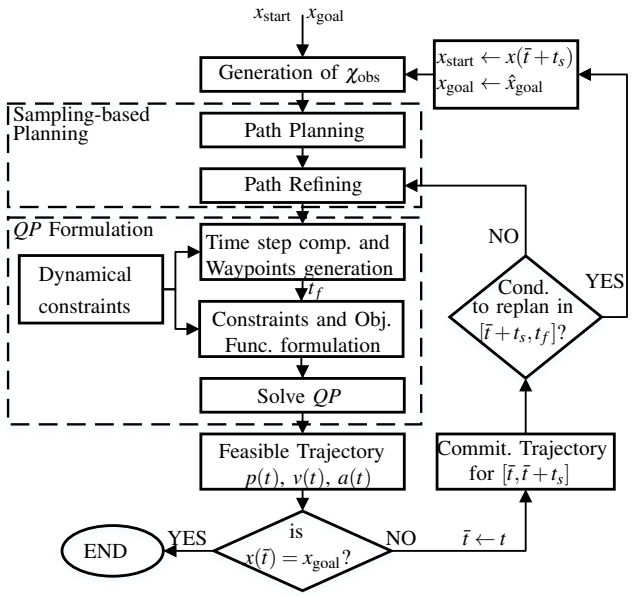


Fig. 2. Sketch of the proposed hybrid method for trajectory generation.

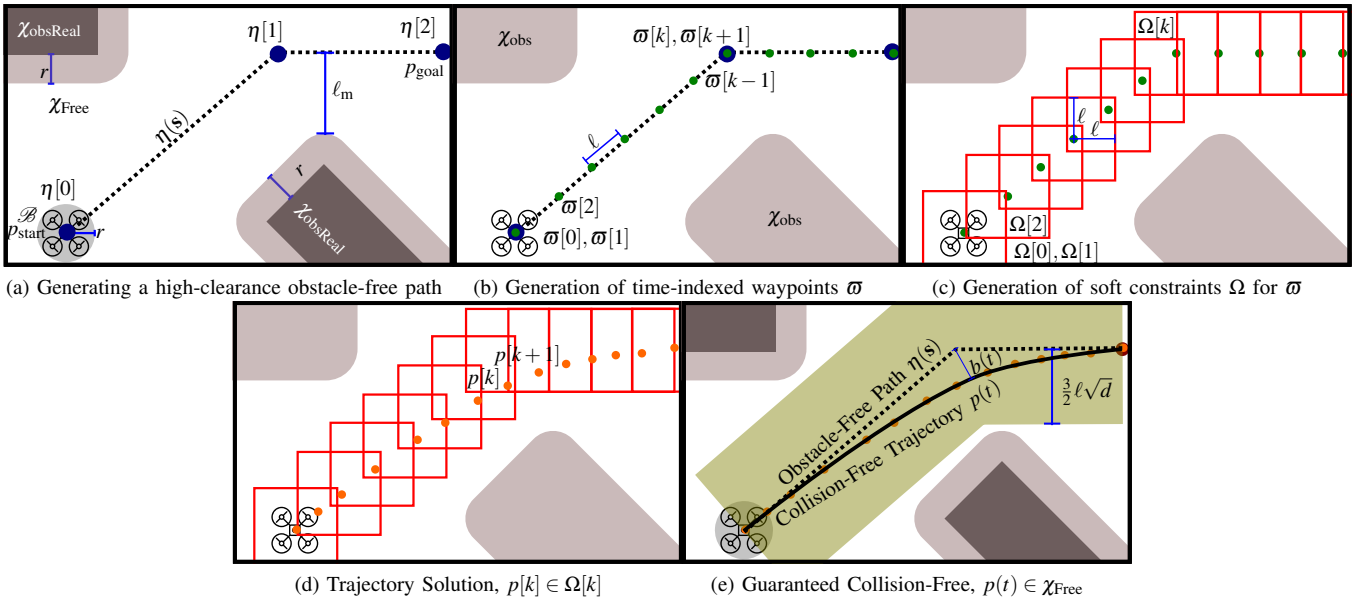
The set of dynamical constraints defines a convex set of allowed states $x = \begin{bmatrix} p^T & v^T & a^T \end{bmatrix}^T$ that is denoted by $\mathscr{X}_{\text{allowed}}$.

B. Problem statement and outline of the method

Problem 1: Consider a robot whose centroid is modeled as a free particle with dynamics described by $\dot{p}(t) = v(t)$, $\dot{v}(t) = a(t)$ where $p, v, a \in \mathbb{R}^d$ and $\|a(t)\|_\infty \leq A_{\max}$. Assuming knowledge of the environment (i.e. the obstacle region χ_{obsReal} is known) and given the initial and final states, $x_{\text{start}} = \begin{bmatrix} p_{\text{start}}^T & v_{\text{start}}^T & a_{\text{start}}^T \end{bmatrix}^T \in \mathscr{X}_{\text{allowed}}$ and $x_{\text{goal}} = \begin{bmatrix} p_{\text{goal}}^T & v_{\text{goal}}^T & a_{\text{goal}}^T \end{bmatrix}^T \in \mathscr{X}_{\text{allowed}}$, respectively, find a trajectory and a time t_f such that, $x(0) = x_{\text{start}}$, $x(t_f) = x_{\text{goal}}$ and for all time $t \in [0, t_f]$, $p(t) \in \chi_{\text{Free}}$ and $x(t) \in \mathscr{X}_{\text{allowed}}$.

Modeling the center of the robot as a free particle is very effective, for instance, for the trajectory planning of quadcopters, since it has being demonstrated that smooth trajectories for each coordinate can be treated as independent reference outputs due to the differentially flat property [19].

Our approach to solve Problem 1 is depicted in Fig. 2 and Fig. 3. The first step, illustrated in Fig. 3a, is to generate an obstacle-free path $\eta(\mathbf{s}) \subset \chi_{\text{free}}$ connecting p_{start} with p_{goal}; where $\eta(\mathbf{s}) \subset \chi_{\text{free}}$. High-clearance paths are desired in order to avoid over-constraining the optimization problem. Hence, a sampling-based path planning technique capable of efficiently generating such paths should be used (e.g. [20], [21]). The minimum distance ℓ_{m} from the path $\eta(\mathbf{s})$ to the obstacle region χ_{obs} is used to set a design parameter ℓ. If a minimum separation ℓ_{m} is required then the obstacles can be further inflated by ℓ_{m} before the path planning algorithm is run.



(a) Generating a high-clearance obstacle-free path (b) Generation of time-indexed waypoints ϖ (c) Generation of soft constraints Ω for ϖ

(d) Trajectory Solution, $p[k] \in \Omega[k]$ (e) Guaranteed Collision-Free, $p(t) \in \chi_{\text{Free}}$

Fig. 3. Illustration of the proposed hybrid method for trajectory planning.

The next step is to generate a series of time-indexed waypoints $\varpi[k]$ with associated hypercube regions $\Omega[k]$ of edge length 2ℓ, as shown in Fig. 3b and Fig. 3c. These regions will be used later to introduce soft constraints in the formulation of the optimization problem. Note that in addition to the waypoints inserted at regular intervals along the path, an extra waypoint is added in each node $\eta[s]$. While these extra waypoints may be redundant in some cases, i.e., where the adjacent path segments don't present a drastic change of direction, they are used to guarantee existence of a trajectory solution. As illustrated in Fig. 3d, it is required that such a solution satisfies $p[k] = p(hk) \in \Omega[k]$; where the time step h is chosen as a function of ℓ and the maximum acceleration $A_{\max}$ of the robot. Let $K+1$ be the number of waypoints ϖ, then $t_f = Kh$ is the total time of the trajectory.

It is shown that following our formulation, the posed quadratic optimization problem is guaranteed to be feasible. Furthermore, let the separation $b(t)$ between $\eta(\mathbf{s})$ and $p(t)$ be given by

$$b(t) = \min\{\|p(t) - \lambda(\eta[s+1] - \eta[s]) + \eta[s]\}\|_2 : \\ \lambda \in [0,1], s \in \{0,\ldots,S-1\}\}.$$

The proposed method ensures that $b(t)$ is bounded by $\frac{3}{2}\ell\sqrt{d}$, then $p(t) \in \chi_{\text{Free}}$ for all $t \in [0, t_f]$, as illustrated in Fig. 3e. This is also an important contribution since previous proposals reported in the literature either do not guarantee to be collision-free, see e.g. [16], [17] or only guarantee that a discrete trajectory generated by the *QP* formulation is collision-free but not the interpolated trajectory that is actually executed by the robot, see e.g. [12]. Moreover, they require iterative *QP* formulations with decreasing sampling time.

In many important situations, it is necessary to compute a new trajectory before the robot has completed its current plan. For example, a path that was previously open may have become blocked, or a new task may require the robot to move towards a different final destination. Hence, as depicted in Fig. 2 while the robot is executing the last solution of Problem 1 it should continuously check if a change of plan is required. Let $\bar{t}$ be the current time. If the conditions to re-plan are met then a new optimization problem is formulated with initial state $x_{\text{start}} \leftarrow x(\bar{t} + t_s)$ and final conditions x_{goal}, and a new trajectory is generated as described above. Otherwise, the trajectory in the interval $[\bar{t}, \bar{t} + t_s]$ is committed to the trajectory tracking controller while the path related to the interval $[\bar{t} + t_s, t_f]$ is further refined, for a limited number of iterations, in search of improving the cost or the clearance of the solution, using asymptotically optimal methods [3].

III. Preliminaries: Sampling-Based Path Planning

Sampling based, or stochastic search methods to generate obstacle-free paths [4]–[6], are popular due to their effectiveness at finding traversable paths while avoiding discretization of the state space. In this paper, collision-free path generation is illustrated using Informed Optimal Rapidly-exploring Random Trees (*IRRT**) [22]. This algorithm improves the current solution iteratively via incremental rewiring, converging asymptotically to an optimal solution. While any other technique to generate a path could have been used, such as *BIT** [5] or Regionally Accelerated *BIT** (*RABIT**) [6], the *IRRT** algorithm was selected because it allows anytime planning by limiting its solutions [23]. A brief review of the *IRRT** algorithm [22] is presented in this section.

Assume that the initial position $p_{\text{start}} \in \chi_{\text{free}}$ and the final position $p_{\text{goal}} \in \chi_{\text{free}}$ are given. The aim is to find a path η^*, that minimizes a given cost function $c : \Sigma(\eta) \mapsto \mathbb{R}_{\geq 0}$, where $\Sigma(\eta)$ is the set of all feasible paths, i.e. $\eta^* = \arg\min_{\eta \in \Sigma(\eta)} c(\eta)$. Following the standard *RRT** algorithm [3], [4] a new random node in χ_{free} is sampled and its neighbor nodes within a ball of radius r_{RRT^*} are rewired if this provides a path with lower cost function, where $r_{\text{RRT}^*} = \gamma_{\text{RRT}^*} \left(\frac{\log(\delta)}{\delta}\right)^{\frac{1}{d}}$; δ is the number

of states in the tree and γ_{RRT^*} an appropriate constant given in [3]. This paper seeks to minimize the path length in $\mathbb{R}^d$, so the cost function c is based on the Euclidean distance.

After a path has been obtained using *RRT**, the *IRRT** algorithm allows increasing the probability of reducing the cost of the solution in subsequent rounds of *RRT** by restricting the sampling region. In [22] it is shown that the new nodes that may improve the cost of the path are necessarily contained in the ellipse defined by

$$\chi_{\text{inform}} = \{\mathrm{x} \in \chi \quad : \quad \|p_{\text{start}} - \mathrm{x}\|_2 + \|\mathrm{x} - p_{\text{goal}}\|_2 \leq c_{\text{best}}\},$$

where c_{best} is the cost of the current solution. New nodes sampled from χ_{inform} are generated by taking random samples x_{ball} from a unitary d-ball and then mapped to χ_{inform} using $\mathrm{x}_{\text{ellipse}} = \mathbf{C}\mathbf{L}\mathrm{x}_{\text{ball}} + \mathrm{x}_{\text{centre}}$, where

$$\mathbf{L} = \text{diag}\left\{\frac{c_{\text{best}}}{2}, \overbrace{\frac{(c_{\text{best}}^2 - c_{\text{min}}^2)^{1/2}}{2}, \cdots, \frac{(c_{\text{best}}^2 - c_{\text{min}}^2)^{1/2}}{2}}^{d-1}\right\},$$

is a diagonal matrix, $\mathbf{C} \in SO(d)$ is the rotation between the origin of the d-dimensional Euclidean space and the vector $p_{\text{goal}} - p_{\text{start}}$ and $\mathrm{x}_{\text{centre}} = (p_{\text{goal}} + p_{\text{start}})/2$.

IV. Optimization-based Trajectory Planning

The *QP* formulation derived in this section integrates information from the obstacle-free path $\eta(\mathbf{s})$ together with knowledge of the dynamical constraints of the robot. The problem is posed in such a way that the need for iterative solutions is avoided and the resulting trajectory is guaranteed to be collision-free, i.e. $\forall t,\ p(t) \in \chi_{\text{Free}}$.

To guide the solution of the *QP* and to ensure that the separation $b(t)$ between the planned path $\eta(\mathbf{s})$ and the final trajectory $p(t)$ remains adequately bounded, $K+1$ time-indexed waypoints $\varpi[k]$ are placed along the path $\eta(\mathbf{s})$. Each waypoint $\varpi[k]$ has an associated set of soft constraints defining a hypercube region $\Omega[k]$, as illustrated in Fig. 3c. These constraints lead to $p[k] = p(kh) \in \Omega[k]$, $k = 0, \ldots, K$. The time step h and the maximum velocity V_{max} are obtained based on the value of the parameter ℓ and the maximum acceleration A_{max}. This method ensures that a trajectory $p(t)$ can always be found which travels from one hypercube region to the next in time h while the separation $b(t)$ remains bounded by $\frac{3}{2}\ell\sqrt{d}$, as illustrated in Fig. 4. Since there is a minimum clearance of ℓ_{m}, this property guarantees a collision-free trajectory. In the following, this procedure is explained in greater detail.

A. Problem constraints formulation

Let $\kappa_0 = \kappa_S = 0$ and $\kappa_s = \left\lceil \frac{1}{\ell}\|\eta[s+1] - \eta[s]\|_2 \right\rceil$, $s \in \{1, \ldots, S-1\}$. A sequence of waypoints are obtained such that $\varpi_0[\kappa_0] = \eta[0]$, $\varpi_S[\kappa_S] = \eta[S]$ and $\varpi_s[\mathrm{i}] = \frac{\mathrm{i}}{\kappa_s}(\eta[s+1] - \eta[s]) + \eta[s]$, $\mathrm{i} \in \{0, \ldots, \kappa_s\}$. The time-indexed waypoint $\varpi[k]$ is obtained as the $(k-1)-$th element of the sequence

$$\varpi = \{\varpi_0[\kappa_0], \varpi_1[0], \ldots, \varpi_1[\kappa_1], \ldots, \varpi_{S-1}[0], \ldots, \varpi_{S-1}[\kappa_{S-1}], \varpi_S[\kappa_S]\}.$$

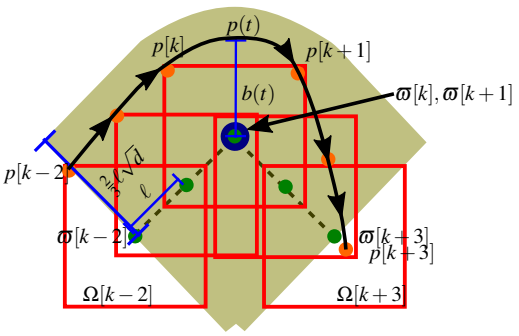


Fig. 4. If V_{max} and h are a solution to Problem 2, then the robot takes time h to navigate from one point in the discrete trajectory to the next (i.e. $p(kh) = p[k]$, $k = 0, \ldots, K$) while the separation $b(t)$ between the resulting trajectory $p(t)$ and the continuous path $\eta(\mathrm{s})$ is bounded by $\frac{3}{2}\ell\sqrt{d}$.

The generation of time-indexed waypoints is illustrated in Fig. 3b; notice that $\varpi_{s-1}[\kappa_{s-1}] = \varpi_s[0] = \eta[s]$ or equivalently $\varpi[k] = \varpi[k+1] = \eta[s]$ for some k. Thus, in addition to the generation of waypoints in-between two consecutive nodes, this procedure also adds two waypoints at each node $\eta[s]$ of the path η, as illustrated in Fig. 3b and Fig. 4 . Furthermore, as illustrated in Fig. 3c, each waypoint $\varpi[k]$ has an associated hypercube of edge length 2ℓ defined by

$$\Omega[k] = \{\rho : \|\rho - \varpi[k]\|_\infty \leq \ell\}. \tag{1}$$

The $\Omega[k]$ regions are used as constraints on the convex optimization problem restricting a solution of the trajectory $p(t)$ to be one satisfying $p[k] = p(kh) \in \Omega[k]$, $k \in \{0, \ldots, K\}$, (or equivalently $\|\varpi[k] - p[k]\|_\infty \leq \ell$) as illustrated in Fig. 3d; where h is the time step hereinafter derived and $K = |\varpi| - 1$. Hence, if such constraints are satisfied, the total time of the trajectory along the path η is $t_f = Kh$.

Problem 2: Based on the design parameter ℓ, the maximum acceleration A_{max} and taking into account particle kinematics for motion with constant acceleration $\|a[k]\|_\infty \leq A_{\text{max}}$ in $t \in (kh, (k+1)h]$, $k \in 0, \cdots, K$, find V_{max} and h such that the following conditions are satisfied:

i) Let $\|\varpi[k+1] - \varpi[k]\|_2 = \ell$ and assume that $p[k] \in \Omega[k]$ and $v[k]$ satisfies $\|v[k]\|_\infty \leq V_{\text{max}}$ and $v[k]^T(\varpi[k+1] - \varpi[k]) \geq 0$. Then, there exists a constant acceleration $\|a[k]\|_\infty \leq A_{\text{max}}$ such that $p[k+1] \in \Omega[k+1]$, $\|v[k+1]\|_\infty \leq V_{\text{max}}$ and $v[k+1]^T(\varpi[k+1] - \varpi[k]) \geq 0$.

ii) Let $\varpi[k] = \varpi[k+1] = \eta[s]$ and assumme that $p[k] \in \Omega[k]$, and $\|v[k]\|_\infty \leq V_{\text{max}}$. Then, there exists a pair of constant accelerations $\|a[k]\|_\infty \leq A_{\text{max}}$ and $\|a[k+1]\|_\infty \leq A_{\text{max}}$, such that $p[k+1] \in \Omega[k+1]$, $\|v[k+1]\|_\infty \leq V_{\text{max}}$, $p[k+2] \in \Omega[k+2]$ and $\|v[k+2]\|_\infty \leq V_{\text{max}}$ with $v[k+2]^T(\varpi[k+2] - \varpi[k+1]) \geq 0$.

iii) For $p[k] \in \Omega[k]$ and $\|v[k]\|_\infty \leq V_{\text{max}}$ and with $\|a[k]\|_\infty \leq A_{\text{max}}$ such that $p[k+1] \in \Omega[k+1]$ and $\|v[k+1]\|_\infty \leq V_{\text{max}}$, it holds that for $t \in [kh, (k+1)h]$, the separation $b(t)$ between $p(t)$ and the straight line connecting $\eta[s]$ and $\eta[s+1]$ is bounded by $|b(t)| \leq \frac{3}{2}\ell\sqrt{d}$.

Assuming condition i) is satisfied implies that the particle can move in the direction of the path from $p[k-2] \in \Omega[k-2]$

to $p[k-1] \in \Omega[k-1]$ satisfying the constraints on velocity and acceleration. Condition ii) applies when the particle moves between $p[k]$ and $p[k+1]$ while it undergoes a sharp change of direction in the path. If this condition is satisfied then a trajectory satisfying the constraints on velocity an acceleration can change direction in a time h and continue satisfying such constraints. To account for such cases the formulation places two identical waypoints at the path node $\eta[s]$, i.e. $\varpi[k] = \varpi[k+1] = \eta[s]$. Finally, condition iii) implies that $p(t)$ is also collision free since $b(t)$ is bounded in such way that $p(t) \in \chi_{\text{Free}}$, $\forall t \in [0, t_f]$.

It is shown, in Lemma 5–Lemma 7 in the Appendix, that a selection of $V_{\max}$ and h solving Problem 2 is

$$V_{\max}^2 := \ell A_{\max}, \quad h^2 := \frac{4\ell}{A_{\max}}. \tag{2}$$

It follows that there exists a collision-free trajectory $p(t)$ with duration t_f satisfying $p[k] \in \Omega[k]$ and the dynamical constraints $\|v[k]\|_\infty \leq V_{\max}$, $\|a[k]\|_\infty \leq A_{\max}$ such that the separation $b(t)$ between $p(t)$ and $\eta(\mathbf{s})$ is bounded by $\frac{3}{2}\ell\sqrt{d}$.

Theorem 3: Let χ_{free} be the obstacle free space, η be an obstacle-free path connecting p_{start} to p_{goal} and $V_{\max}, h$ defined by (2). Then given ℓ, there exist a trajectory $p(t)$ such that $\forall t \in [0, t_f]$ where $t_f = Kh = (|\varpi| - 1)h$, the following are satisfied:

- $|b(t)| \leq \frac{3}{2}\ell\sqrt{d}$ i.e. $p(t) \in \chi_{\text{Free}}$
- $\|v[k]\|_\infty \leq V_{\max}$
- $\|a[k]\|_\infty \leq A_{\max}$
- $p(0) = p_{\text{start}}$, $v(0) = v_{\text{start}}$, $v(t_f) = v_{\text{goal}}$ for given p_{start},v_{start} and v_{goal}

Proof: The proof follows from Lemma 5 – Lemma 7. ■

B. Quadratic Program Formulation

In the following a *QP* to produce a trajectory for the robot is posed taking into account its dynamical constraints. This formulation, is based of the constraints derived in Section IV-A.

The objective function for the convex problem is formulated as a function of the acceleration by

$$\min_{\mathbf{a}} \quad \mathbf{a}^T H \mathbf{a}, \tag{3}$$

where $\mathbf{a} = [a[0]^T \ \cdots \ a[K]^T]^T \in \mathbb{R}^{dK}$ is a vector of accelerations with $a[k] \in \mathbb{R}^d$ as the acceleration at the discretization time k and the matrix $H \in \mathbb{R}^{dK \times dK}$ is a positive definite constant matrix. By selecting H appropriately a quadratic objective function of (p, v, a, j) can be obtained [13]. For instance, since jerk is given by $j[k] = \frac{a[k+1]-a[k]}{h}$ by taking $H = W^T W$

$$[w]_{\text{ij}} = \begin{cases} -1/h & \text{if i} = \text{j}, \\ 1/h & \text{if i} = \text{j} - 1, \\ 0 & \text{otherwise} \end{cases}$$

we obtain an objective function for minimizing jerk. Thus, the following convex optimization problem is defined:

Problem 4: Taking $p[k]$ and $v[k]$ as

$$p[k] = p[0] + hkv[0] + \frac{h^2}{2}\sum_{\text{i}=0}^{k-1}(2(k-\text{i})+3)a[\text{i}]$$

$$v[k] = v[0] + h\sum_{\text{i}=0}^{k-1} a[\text{i}],$$

obtain a sequence of accelerations $a = [a[0]^T, \ldots, a[K]^T]^T$ that

$$\text{minimizes} \quad \mathbf{a}^T H \mathbf{a}$$

subject to

$$p[0] = p_{\text{start}}, \quad v[0] = v_{\text{start}}, \quad a[0] = a_{\text{start}},$$

$$\|\varpi[k] - p[k]\|_\infty \leq \ell, \quad \|v[k]\|_\infty \leq V_{\max}, \quad \|a[k]\|_\infty \leq A_{\max}$$

for $k = 1, \ldots, K-1$ and

$$p[K] = p_{\text{goal}}, \quad v[K] = v_{\text{goal}}, \quad a[K] = a_{\text{goal}}.$$

Notice that $\|p[k] - \varpi[k]\|_\infty \leq \ell$ is equivalent to $p[k] \in \Omega[k]$.

Since $p[k]$ and $v[k]$ are linear functions of $a[k]$ the optimization problem given in Problem 4 is convex. Moreover, notice that by Theorem 3 the posed *QP* formulation is feasible. This approach, unlike [11]–[13], [24], avoids iterative formulations.

Algorithm	Success Rate	Avg. Path Length [m]	Avg. Max Velocity [m/s]	Avg. Comp. Time [s]
iSCP	239/500	**11.6886**	1.4612	0.4920
CHOMP	83/500	14.9088	**1.7677**	0.4495
Ours	**500/500**	15.8581	1.5266	**0.1519**

TABLE I
COMPARISON IN A POISSON FOREST WITH A DENSITY OF 3.2 trees/m^2.

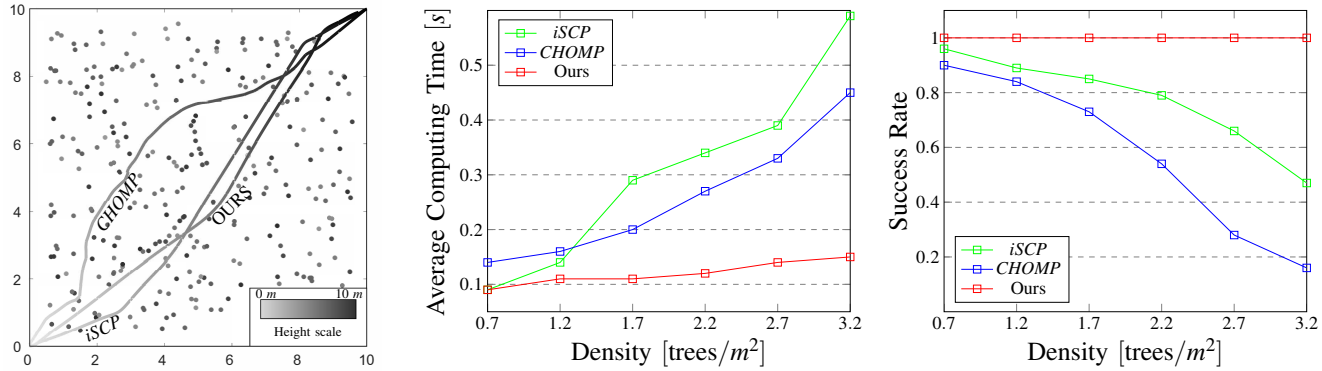
V. RESULTS

The proposed algorithm was programmed in C++ and implemented on a PC running Ubuntu 14.04-LTS with Intel Core i5-3210M @ 2.50GHz and 4 GB of RAM. The *QP* is solved using Mosek [25]. First, benchmarking results are shown comparing our method with state-of-the-art algorithms based on simulations in various forest environments generated using the method in [26]. Then, we present experimental results obtained with a *Crazyflie 2.0* [27] quadcopter in different three dimensional scenarios with varying number of obstacles and complexity - see video at https://youtu.be/DJ1IZRL5t1Q.

A. Benchmark results

The algorithm was compared against the Incremental *SCP* (*iSCP*) algorithm described in [13] and against *CHOMP* [16]. Simulations were run in different Poisson forest environments [26] with varying tree densities in a simulation space of $10m \times 10m \times 10m$. The height of the trees was set following a uniform distribution between $5m$ to $10m$. The start and goal positions for the mobile robot were selected randomly, such that the minimum travel distance was $8m$ starting and ending at rest with a maximum acceleration of $20m/s^2$. For each benchmark 500 tests were performed.

Some parameters had to be selected for each algorithm. The *iSCP* was configured using a total time of 15 seconds and a discretization step of 0.2 seconds. In the case of *CHOMP* it was configured using a fixed value N of 100 points. These parameters were selected trying to balance the success rate and the computation time. They were set based on trial and error since no methodology is known to calculate them in advance. Finally, for the proposed algorithm, a minimum separation to



(a) 3D Planning in a $10m \times 10m \times 10m$ Poisson forests with density of 3.2 $trees/m^2$.

(b) Computation time vs tree density.

(c) Success rate vs. tree density.

Fig. 5. Algorithm comparison among *iSCP*, *CHOMP* and the proposed approach on different forest environments varying on tree density.

the obstacles is required, as described in Section II. Thus, the obstacles were inflated by ℓ_{m}, resulting in $\ell = 0.05$. For path generation a goal region with a radius of $0.005m$ was specified. The *IRRT** algorithm was run at most four rounds of path generation followed by refinement of the sampling region; the algorithm was stopped early if no reduction to the cost of the solution was obtained in the last round of *RRT** compared to the previous round. Table I shows the benchmark results for a forest density of 3.2 trees$/m^2$. The algorithms are compared in terms of their success rate, average mean path length, average maximum absolute velocity and average computing time. Although, the *iSCP* and *CHOMP* performed slightly better in path length and maximum velocity, respectively, the proposed approach excelled in success rate and its computing time was significantly lower. This success rate is inherited from the sampling-based path planning algorithm (which is probabilistic complete), since our method ensures the feasibility of the *QP* problem.

Fig. 5 shows a benchmark where the tree density was varied between 0.7 and 3.2 trees$/m^2$. Typical trajectories are shown in Fig. 5a. For ease of visualization, the thickness of the paths and the radius of the trees are not drawn at scale. The comparison on computing time and success rate are shown in Fig. 5b and Fig. 5c, respectively.

B. Experimental Results

The test setup consisted of an Optitrack motion capture system with 15 infra-red cameras providing localization with sub-millimeter accuracy. The motion capture software and the trajectory planning algorithm were run on the same computer. The current position of the quadcopter and the corresponding reference point was broadcasted every 6 ms. The firmware of the *Crazyflie 2.0* was modified to implement a custom nonlinear control algorithm for precise trajectory tracking.

The first scenario, shown in Fig. 6, is a $3m \times 5m$ maze where the quadcopter had to navigate from an initial position inside the maze to a final goal. The task was repeated multiple times such that the goal of the last task became the initial position for the next. A new goal was generated by moving a marker inside the maze. The same maze environment was simulated

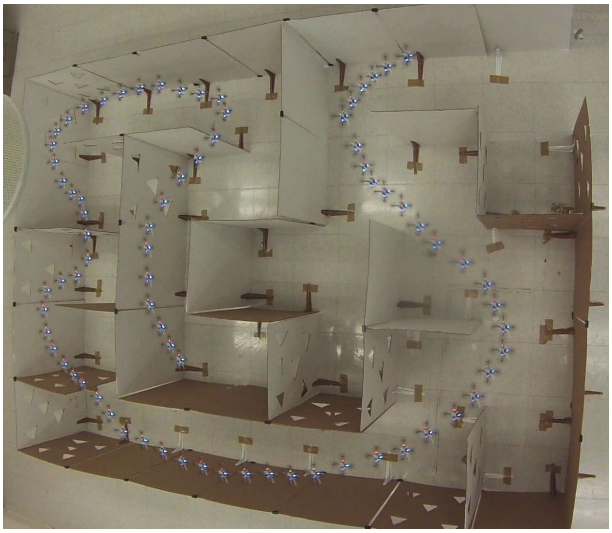


Fig. 6. A composite image of a quadcopter performing a trajectory in a 3x5 meters maze with average planning time of $\sim$ 100ms.

in a computer, obtaining an average planning time of $\sim$ 100ms based on 500 trials with random start and end points.

The algorithm was also tested in a 3-dimensional scenario consisting of two hundred interwoven strings and twenty poles inside a volume of two cubic meters. This scenario, illustrated in Fig. 1, is similar to the one presented in [8] which consisted of 20 interwoven strings. Planning times in the order of minutes were reported in that work compared to a fraction of a second in the present work. The scenario was manually mapped using the motion capture system. During the experiment, new goals were provided online to the quadcopter while it navigated inside the complex environment.

For these experiments, the radius r of $\mathscr{B}(p)$ was selected to be $0.035m$ based on the quadcopter dimensions. Considering also a diameter of $0.005m$ for the strings and an average tracking error of $0.01m$, the maximum ℓ resulted in $0.035m$. The value of ℓ was set to 0.02. The maximum acceleration in these experiments was set to $20m/s^2$ in each axis based on trial and error. Using these parameters, the algorithm took on average $\sim$ 300ms in a benchmark of 500 simulated trials.

Parameter ℓ [m]	**Average Mean Path Length** [m]	**Average Max Velocity** [m/s]	**Average Max Acc** [m/s^2]	**Average Compute Time** [s]
0.010	**3.6899**	0.4391	**2.5828**	0.9472
0.015	3.8931	0.5386	2.6315	0.4750
0.020	3.9431	0.6226	2.6509	0.2922
0.025	4.1916	0.6923	2.7354	**0.2588**
0.030	4.6593	0.7675	2.8683	0.2986
0.035	4.7108	**0.8268**	2.8670	0.3853

TABLE II
COMPARISON OF THE RUN TIME FOR DIFFERENT VALUES OF ℓ.

Since the performance of the algorithm is affected by the desired minimum separation ℓ_{m}, and as a consequence by the parameter ℓ, a benchmark varying the values of ℓ was performed for the same scenario. The results are presented in Table II where it can be seen that a low value of ℓ results in short trajectories, but higher computing time due to increased size of the *QP* problem. On the other hand, a large value of ℓ results in a smaller *QP* problem while a larger separation between the path and the trajectory is allowed and a higher-clearance is obtained, resulting in longer travels for the trajectories. Finally, it can be observed that the sampling-based planner takes longer to find a solution as ℓ is increased. This is expected since χ_{obs} region size depends on ℓ.

VI. CONCLUSIONS AND FUTURE WORK

An algorithm for online trajectory planning and replanning of mobile robots was presented. The proposed method uses a sampling-based planning algorithm to efficiently determine an obstacle-free path. Then, an optimization program takes into account the robot dynamical constraints to generate a time dependent trajectory. The main contribution of this method lies in the formulation of the optimization problem that is guaranteed to be feasible, avoiding iterative formulations. The effectiveness of this method was shown by applying it to the online planning of a quadcopter in multiple scenarios consisting of up to two hundred obstacles. The scenarios presented herein are some of the most obstacle-dense scenarios reported to date for online planning of a quadcopter.

Our approach was contrasted with state-of-the-art algorithms showing a significant improvement in computation time and success rate, which demonstrates its effectiveness for online planning in obstacle-dense environments.

As for future work, we aim to apply the algorithm in complex dynamic environments as well as its extension to multi-agent planning in obstacle-dense environments.

APPENDIX
AUXILIARY RESULTS FOR THE CONSTRAINTS DERIVATION

Notice that, since the waypoints connecting $\eta[s]$ and $\eta[s+1]$ are placed in the line segment $\lambda(\eta[s+1]-\eta[s])+\eta[s]$, for some $\eta[s]$ and some $\lambda\in[0,1]$, it is always possible to find a coordinate transformation, for each pair $\eta[s]$, $\eta[s+1]$ with orthogonal axis, in such a way that $\lambda(\eta[s+1]-\eta[s])+\eta[s]$ lie in the x-axis (with the x-axis pointing toward $\eta[s+1]$) of the new coordinate system. The notation p_x, (resp. v_x and a_x) is used to represent the component of the position (resp. velocity and acceleration) along the x-axis.

Notice that, in this new coordinate system $\Omega[k+1]=\left\{\rho:\rho=\gamma+[\ell\ 0\ \cdots\ 0]^T,\gamma\in\Omega[k]\right\}$. Additionally, without loss of generality, the origin is placed at $\varpi[k]$.

Lemma 5: Given the set of time-indexed waypoint regions (1) and taking A_{max} and h as in (2), then for $p[k]\in\Omega[k]$ and $v[k]$ such that $\|v[k]\|_\infty\le V_{\max}$ and $v[k]^T(\varpi[k+1]-\varpi[k])\ge 0$, there exists a constant acceleration $\|a[k]\|_\infty\le A_{\max}$ such that $p[k+1]\in\Omega[k+1]$, $\|v[k+1]\|_\infty\le V_{\max}$ and $v[k+1]^T(\varpi[k+1]-\varpi[k])\ge 0$.

Proof: The proofs will be performed independently for each coordinate of the new coordinate system.

For the x–coordinate. The conditions $v[k]^T(\varpi[k+1]-\varpi[k])\ge 0$, $v[k+1]^T(\varpi[k+1]-\varpi[k])\ge 0$, $\|v[k]\|_\infty\le V_{\max}$ and $\|v[k+1]\|_\infty\le V_{\max}$ imply that $v_x[k]$ and $v_x[k+1]$ lie in $[0,V_{\max}]$.

Let $p[k]\in\Omega[k+1]$ and $v_x[k]\in[0,V_{\max}]$ and notice that (2) implies $A_{\max}=\dfrac{2V_{\max}}{h}$. Then, with the acceleration

$$a_x[k]=\frac{V_{\max}-2v_x[k]}{h}\in\left[-\frac{V_{\max}}{h},\frac{V_{\max}}{h}\right]\subset[-A_{\max},A_{\max}]$$

is obtained that

$$p_x[k+1]=p_x[k]+\ell\quad\text{and}\quad v_x[k+1]=-v_x[k]+V_{\max}\in[0,V_{\max}]$$

For the y–coordinate. With $v_y[k]\in[-V_{\max},V_{\max}]$, and

$$a_y[k]=-\frac{2v_y[k]}{h}\in\left[-\frac{2V_{\max}}{h},\frac{2V_{\max}}{h}\right]=[-A_{\max},A_{\max}]$$

Then $p_y[k+1]=p_y[k]$ and $v_y[k+1]=-v_y[k]$.

For the rest of the coordinates. The argument is equivalent as in the y–coordinate. The above results shows that

$$p[k+1]=p[k]+[\ell\ 0\ \cdots\ ]^T\in\Omega[k+1].$$

Additionally, $v[k+1]]^T(\eta[s+1]-\eta[s])\ge 0$ in the new coordinate system is $[v_x[k+1]\ v_y[k+1]\ \cdots][\alpha\ 0\ \cdots\ 0]^T\ge 0$, for some $\alpha>0$. In this way $[v_x[k+1]\ v_y[k+1]\ \cdots][\alpha\ 0\ \cdots\ 0]^T=\alpha v_x[k+1]\ge 0$, which completes the proof. ∎

Lemma 6: Let $\varpi[k]=\varpi[k+1]=\eta[s]$ and let A_{max} and h be given as in (2). Then given the set of time indexed waypoint regions (1) for $p[k]\in\Omega[k]$, and $v[k]$ satisfying $\|v[k]\|_\infty\le V_{\max}$, there exists a pair of constant accelerations $\|a[k]\|_\infty\le A_{\max}$ and $\|a[k+1]\|_\infty\le A_{\max}$, such that $p[k+1]\in\Omega[k+1]$, $\|v[k+1]\|_\infty\le V_{\max}$, $p[k+2]\in\Omega[k+2]$ and $\|v[k+2]\|_\infty\le V_{\max}$ with $v[k+2]^T(\varpi[k+2]-\varpi[k+1])\ge 0$.

Proof: The proofs will be performed independently for each coordinate of the new coordinate system.

For the x–coordinate. The transformation changes the constraints as follows. For $\varpi[k]$ placed at the origin, $p_x[k]\in[-\alpha\ell,\alpha\ell]$, $v_x[k]\in[-\alpha V_{\max},\alpha V_{\max}]$, $a_x[k]\in[-\alpha A_{\max},\alpha A_{\max}]$ for some $\alpha\in[1,\sqrt{d}]$. Thus, in the new coordinate system $p_x[k]$ can be written as $p_x[k]=\alpha\ell(2\lambda_{x,1}-1)$, $\lambda_{x,1}\in[0,1]$ and $v_x[k]=\alpha V_{\max}(2\lambda_{v,1}-1)$, $\lambda_{v,1}\in[0,1]$ and choosing

$$a_1=\alpha A_{\max}\left(1-\frac{\lambda_{x,1}}{2}-\frac{3\lambda_{v,1}}{2}\right)\in[-\alpha A_{\max},\alpha A_{\max}]$$

$$a_2=\alpha A_{\max}\left(\lambda_{x,3}+\lambda_{x,2}-\frac{1}{2}\right)\in[-\alpha A_{\max},\alpha A_{\max}]$$

with

$$\lambda_{x,2} = \frac{1}{2}(\lambda_{x,1} + \lambda_{v,1}) \in [0,1], \quad \lambda_{v,2} = 1 - \lambda_{x,2} \in [0,1]$$
$$\lambda_{x,3} = \frac{V_f}{2V_{\max}} \in \left[0, \frac{1}{2}\right], \quad V_f \in \left[0, \frac{V_{\max}}{\alpha}\right]$$

is obtained that

$$p_x[k+1] = \alpha\ell(2\lambda_{x,2} - 1) \in [-\alpha\ell, \alpha\ell]$$
$$v_x[k+1] = \alpha V_{\max}(2\lambda_{v,2} - 1) \in [-\alpha V_{\max}, \alpha V_{\max}]$$
$$p_x[k+2] = 2\alpha\ell\lambda_{x,3} \in [0, \ell], \quad v_x[k+2] = \alpha V_f$$

For the rest of the coordinates. The result for the y-coordinate in the proof of Lemma 5 can be used for the interval $[kh,(k+1)h]$ and for $[(k+1)h,(k+2)h]$, so that $p_y[k] = p_y[k+1]$ with feasible velocities and accelerations and the same for the rest of the coordinates. The above results can be used to prove that $p[k+1] \in \Omega[k+1]$ and $p[k+2] \in \Omega[k+2]$ while $v[k+2]^T(\eta[s+1] - \eta[s]) \geq 0$ in a similar way as in the proof of Lemma 5, which completes the proof. ∎

Lemma 7: Given the set of time indexed waypoint regions (1) and taking A_{max} and h as in (2), then for $p[k] \in \Omega[k]$ and $\|v[k]\|_\infty \leq V_{\max}$ and with $\|a[k]\|_\infty \leq A_{\max}$ such that $p[k+1] \in \Omega[k+1]$ and $\|v[k+1]\|_\infty \leq V_{\max}$, then $|b(t)| \leq \frac{3}{2}\ell\sqrt{d}$ for $t \in (kh,(k+1)h)$.

Proof: To show this argument is not required to rotate the coordinate system but only to place the origin at $\varpi[k]$. Taking into account first only the x-coordinate, the biggest separation for $p_x(t)$, $t \in [kh,(k+1)h]$ occurs when $p_x[k] = \pm\ell$ and $v_x[k] = \pm V_{\max}$. The only way that $p_x[k+1] \in [-\ell,\ell]$ and $v_x[k+1] \in [-V_{\max}, V_{\max}]$ for feasible $a_x[k]$ is that $p_x[k+1] = p_x[k]$ and $v[k+1] = -v[k]$. For this condition it is required that $a_x[k] = \mp A_{\max}$. Due to the symmetry of the problem, the maximum value of $p_x(t)$ for $t \in (kh,(k+1)h)$ occurs when $t = t^* = \left(k + \frac{1}{2}\right)h$ so that $x(t^*) = \frac{3}{2}\ell$. If every coordinate has a maximum separation of $\frac{3}{2}\ell$ then $b(t)$ will have a bound given by $|b(t)| \leq \frac{3}{2}\ell\sqrt{d}$. ∎

ACKNOWLEDGEMENTS

The authors would like to thank Kirk Skeba and Maynard Falconer for the fruitful discussions on this research topic and the anonymous reviewers for their constructive comments.

REFERENCES

[1] C. Katrakazas, M. Quddus, W.-H. Chen, and L. Deka, "Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions," *Transportation Research Part C: Emerging Technologies*, vol. 60, pp. 416 – 442, November 2015.

[2] B. Paden, M. Cap, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *IEEE Trans. Intell. Veh.*, vol. 1, no. 1, pp. 33–55, March 2016.

[3] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *Int. J. Robot. Res.*, vol. 30, no. 7, pp. 846–894, June 2011.

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

[5] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot, "Batch informed trees (BIT*): Sampling-based optimal planning via the heuristically guided search of implicit random geometric graphs," in *IEEE Int. Conf. Robot. Autom.*, June 2015, pp. 3067–3074.

[6] S. Choudhury, J. D. Gammell, T. D. Barfoot, S. S. Srinivasa, and S. Scherer, "Regionally accelerated batch informed trees (RABIT*): A framework to integrate local information into optimal path planning," in *IEEE Int. Conf. Robot. Autom.* IEEE, May 2016, pp. 4207–4214.

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

[8] B. Landry, R. Deits, P. R. Florence, and R. Tedrake, "Aggressive quadrotor flight through cluttered environments using mixed integer programming," in *IEEE Int. Conf. Robot. Autom.*, May 2016, pp. 1469–1475.

[9] D. J. Webb and J. van den Berg, "Kinodynamic RRT*: Asymptotically optimal motion planning for robots with linear dynamics," in *IEEE Int. Conf. Robot. Autom.* IEEE, May 2013, pp. 5054–5061.

[10] C. Xie, J. van den Berg, S. Patil, and P. Abbeel, "Toward asymptotically optimal motion planning for kinodynamic systems using a two-point boundary value problem solver," in *IEEE Int. Conf. Robot. Autom.* IEEE, May 2015, pp. 4187–4194.

[11] C. Richter, A. Bry, and N. Roy, "Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments," in *Robotics Research*. Springer, April 2016, pp. 649–666.

[12] F. Augugliaro, A. P. Schoellig, and R. D'Andrea, "Generation of collision-free trajectories for a quadrocopter fleet: A sequential convex programming approach," in *IEEE/RSJ Int. Conf. Intell. Robots Syst.*, October 2012, pp. 1917–1922.

[13] Y. Chen, M. Cutler, and J. P. How, "Decoupled multiagent path planning via incremental sequential convex programming," in *IEEE Int. Conf. Robot. Autom.*, May 2015, pp. 5954–5961.

[14] 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," *Int. J. Robot. Res.*, vol. 33, no. 9, pp. 1251–1270, June 2014.

[15] J. Chen, T. Liu, and S. Shen, "Online generation of collision-free trajectories for quadrotor flight in unknown cluttered environments," in *IEEE Int. Conf. Robot. Autom.*, May 2016, pp. 1476–1483.

[16] M. Zucker, N. Ratliff, A. D. Dragan, M. Pivtoraiko, M. Klingensmith, C. M. Dellin, J. A. Bagnell, and S. S. Srinivasa, "CHOMP: Covariant hamiltonian optimization for motion planning," *Int. J. Robot. Res.*, vol. 32, no. 9-10, pp. 1164–1193, September 2013.

[17] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, "STOMP: Stochastic trajectory optimization for motion planning," in *IEEE Int. Conf. Robot. Autom.* IEEE, May 2011, pp. 4569–4574.

[18] H. Oleynikova, M. Burri, Z. Taylor, J. Nieto, R. Siegwart, and E. Galceran, "Continuous-time trajectory optimization for online UAV replanning," in *IEEE/RSJ Int. Conf. Intell. Robots Syst.*, October 2016, pp. 5332–5339.

[19] G. Loianno, C. Brunner, G. McGrath, and V. Kumar, "Estimation, control, and planning for aggressive flight with a small quadrotor with a single camera and IMU," *IEEE Robot Autom. Lett.*, vol. 2, no. 2, pp. 404–411, November 2017.

[20] L. Jaillet, J. Cortés, and T. Siméon, "Sampling-based path planning on configuration-space costmaps," *IEEE Trans. Robot*, vol. 26, no. 4, pp. 635–646, August 2010.

[21] D. Devaurs, T. Siméon, and J. Cortés, "Optimal path planning in complex cost spaces with sampling-based algorithms," *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 2, pp. 415–424, April 2016.

[22] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot, "Informed RRT*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic," in *IEEE/RSJ Int. Conf. Intell. Robots Syst.*, September 2014, pp. 2997–3004.

[23] S. Karaman, M. R. Walter, A. Perez, E. Frazzoli, and S. Teller, "Anytime motion planning using the RRT*," in *IEEE Int. Conf. Robot. Autom.* IEEE, May 2011, pp. 1478–1483.

[24] M. Neunert, C. de Crousaz, F. Furrer, M. Kamel, F. Farshidian, R. Siegwart, and J. Buchli, "Fast nonlinear model predictive control for unified trajectory optimization and tracking," in *IEEE Int. Conf. Robot. Autom.*, May 2016, pp. 1398–1404.

[25] Mosek ApS, *The MOSEK optimization software*, 2016. [Online]. Available: http://www.mosek.com/

[26] S. Karaman and E. Frazzoli, "High-speed flight in an ergodic forest," in *IEEE Int. Conf. Robot. Autom.*, May 2012, pp. 2899–2906.

[27] Bitcraze AB, *A quadcopter open platform*, 2016. [Online]. Available: https://www.bitcraze.io/crazyflie-2/