Trajectory Generation Using Semidefinite Programming For Multi-Rotors

Hu, Xiao; Olesen, Daniel; Knudsen, Per

Published in:
Proceedings of the European Control Conference 2019

Link to article, DOI:
10.23919/ECC.2019.8795662

Publication date:
2019

Document Version
Peer reviewed version

Trajectory Generation Using Semidefinite Programming For Multi-Rotors

Xiao Hu, Daniel Olesen, Per Knudsen

***Abstract*— Trajectory generation is one of the most popular techniques for multi-rotors to achieve autonomous navigation capabilities. In this paper, we present an optimization-based approach for generating smooth, feasible and collision-free trajectories. Our method does not require a prior proper time allocation but attempts to optimize time allocation automatically. This method first optimizes trajectories subjected to dynamic constraints and 3D corridor constraints using a quadratic program. Then time allocation is optimized as a semidefinite program by leveraging semidefinite relaxation. A feasible solution is obtained using the rank-one approximation and used to update previous time allocation. This method run iteratively until a feasible trajectory is obtained. Our approach is validated with simulation results. Experimental results show the proposed method ensures to generate feasible trajectories in clutter environments.**

I. Introduction

With the technological advancements in all aspects of unmanned aerial vehicles (UAVs), more applications based on UAVs have emerged, i.e. rescue, environmental monitoring and surveying, etc. Among various types of UAVs, quadcopters have received great popularity because of their superior maneuverability, simplicity, and robustness, which enable them to navigate autonomously in cluttered environments. The challenging environments require the quadcopter to efficiently generate trajectories and then take actions to avoid obstacles, which attract lots of research interests in recent years.

A considerable number of contributions on trajectory generation for quadcopters have been proposed since 2011. The pioneering work by [1] introduces the property of differentiable flatness for quadcopter and proposes a trajectory generation approach via Quadratic Programming (QP) in order to minimize the control effort, which demonstrates great performance in tightly constrained indoor environment. For QP problems only subjected to equality constraints, [2] proposes an analytical solution by reformulating the prime problem with a mapping matrix. Lately, [3] extends their work for moving target tracking by adding a 2 norm regularization of the tracking difference between target trajectory and generated trajectory. In order to deal with moving obstacles, [4] generalize the QP problem to a nonconvex Quadratic Constrained Quadratic Problem (QCQP). By Semi-Definite Relaxation (SDR), the nonconvex QCQP is further reformulated as a standard Semi-Definite Programming (SDP) which can be efficiently solved. A feasible solution may finally be recovered using randomization technique. Another typical solution for quadcopter trajectory generation is based on motion primitives, which is detailed and demonstrated in [5]. Thanks to its computation efficiency, a large number of motion primitives can be generated onboard in real time and then the optimal trajectory can be obtained using brute force search. It has been successfully applied for quadcopter ball juggling [5], autonomous landing in [6], etc.

Although the aforementioned methods have demonstrated great performances, they all require a prior time allocation which includes a series of traversing time from one waypoint to the next. An inappropriate time allocation would make the generated trajectory infeasible, which is shown in Fig. 1. The most intuitive solution for this issue is to try to estimate a proper time allocation a prior, which is often been done with a constant or trapezoid velocity profile [7] [8]. However, the shape of the generated trajectory depends on the choice of time allocation, which makes those coarse estimations inaccurate in most case. A trade-off solution is often taken practically at the expense of underestimating the agility of quadcopters, i.e. constraining the maximum traversing velocity and acceleration to a conservative value [7] [9]. Other non-trivial approaches attempt to find a proper time using optimization. By relaxation of the arrival time for intermediate waypoints, the traversing time for each segment can be optimized [8]. However, an estimation of total traversing time is still required. [2] proposes an approach using nonlinear optimization using NLopt [10]. They start with an initial time allocation for trajectory coefficients computation and then optimize time allocation with precomputed coefficients. The final solution requires iteratively solve a nonlinear optimization problem.

In this paper, a trajectory generation approach is proposed which facilitate generation from arbitrary initial states to a target state while ensuring the smooth and feasibility of the resultant trajectory. The proposed method iteratively solve trajectory generation problem leveraging QP and SDP. More specifically, the QP routine follows the classical minimum snap trajectory generation approach [1]. A novel 3D corridor constraint is proposed for obstacle avoidance. To deal with time allocation, we leverage an SDP optimization routine which attempts to optimize segments' time automatically. Simulation results demonstrate the performance of the proposed approach. The major contribution of this work are summarized as

This work was financially supported by the Innovation Fund Denmark through the project Unmanned Aerial Vehicle for high-Quality Magnetic Surveying (UAV-QMS).

Xiao Hu, Daniel Olesen, Per Knudsen are with National Space Institute, Denmark Technical University, Elektrovej Building 328, room 007, 2800 Kongens Lyngby, Denmark. Emails: xiahaa@space.dtu.dk, danole@space.dtu.dk, pk@space.dtu.dk.

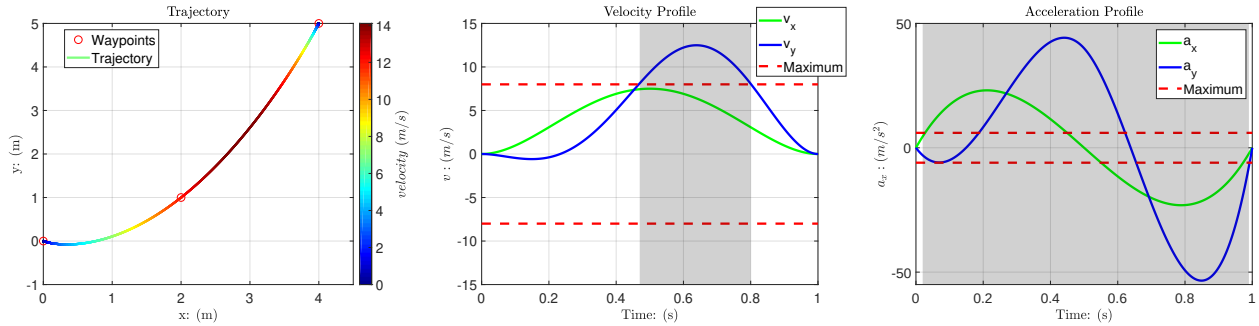

Fig. 1. Trajectory generation result with improper time allocation. The left column shows the generated trajectory connecting 3 assigned waypoints. The middle and right columns plot the corresponding velocity profile and acceleration profile of generated trajectory. Grey areas indicate segments where the trajectory is infeasible for given maximum velocity and acceleration constraints marked with the bold red dashed line.

- We propose a novel 3D corridor constraint that constrains trajectory's deviation from its straight line counterpart, which could be used for collision avoidance. By leveraging two supporting planes, the proposed 3D corridor can be written with 4 convex inequality constraints.
- The proposed method does not require a pre-determined time allocation, but optimize time allocation using SDP and relaxation, which ensures the feasibility without sacrificing agility.

This paper is organized as follows. Section II reviews the preliminaries for trajectory generation. The proposed methodology is detailed in Section III. Section IV describes the experimental results. Conclusion is finally drawn in Section V.

II. Preliminaries

In this section, we briefly introduce the notations and preliminary knowledge used in the paper.

A. Notations

A trajectory is defined in a global frame (i.e. the common used North-East-Ground frame) denoted as $\mathcal{W}$ with 4 dimensions of x, y, z and the yaw angle ϕ. The number of waypoints is denoted with N and their corresponding arriving time is given by $t_i,\ i = 1,\ \cdots,\ N$. To traverse through N waypoints, $M = N - 1$ trajectory segments of r^{th} order are needed. Vectors are indicated with small bold letters (e.g. $\mathbf{p}$) and matrices are represented by bold capital letters (e.g. $\mathbf{Q}$).

B. Differential Flatness

Differential Flatness [11] is an important property for trajectory generation. The proof of differentiable flatness for quadcopters is detailed in [1], which is further refined in [12]. The property of differential flatness allows to plan smooth trajectories in the differentiable space and then appropriate inputs could be mapped from flat outputs, which provides the fundament for the following trajectory generation.

C. Polynomial Representation

Following the same representation of [1], we formulate trajectory with piecewise polynomials. Then, for each dimension, the trajectory $f(t, \mathbf{p})$ is written as:

$$f(t,\mathbf{p}) = \begin{cases} \sum_{i=0}^{r} p_{i,1} t^i, & t_0 \le t < t_1 \\ \sum_{i=0}^{r} p_{i,2} t^i, & t_1 \le t < t_2 \\ \quad\vdots & \\ \sum_{i=0}^{r} p_{i,M} t^i, & t_{N-1} \le t < t_N \end{cases} \tag{1}$$

where $\mathbf{p} = [\mathbf{p_1}^T\ \cdots\ \mathbf{p_1}^T]^T$.

D. Semidefinite Relaxation & Semidefinite Programming

According to [13], the real-valued quadratically constrained quadratic program (QCQP) is defined as

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n}\ & \mathbf{x}^T\mathbf{Q}\mathbf{x} \\ \text{subject to:}\ & \mathbf{x}^T\mathbf{A}_i\mathbf{x} \preceq \mathbf{b}_i, i = 1,\ \cdots,\ m. \end{aligned} \tag{2}$$

where $\mathbf{Q},\ \mathbf{A}_i \in \mathbb{S}^n$ with $\mathbb{S}^n$ denoting the set of real symmetric $n \times n$ matrices. Using the property $\mathbf{x}^T\mathbf{C}\mathbf{x} = trace(\mathbf{C}\mathbf{x}\mathbf{x}^T)$ and a new variable $\mathbf{X} = \mathbf{x}\mathbf{x}^T$, Eq. (2) can be rewritten as

$$\begin{aligned} \min_{\mathbf{X}\in\mathbb{S}^n_+}\ & trace(\mathbf{Q}\mathbf{X}) \\ \text{subject to:}\ & \mathbf{A}_i\mathbf{X} \preceq \mathbf{b}_i, i = 1,\ \cdots,\ m. \\ & rank(\mathbf{X}) = 1 \end{aligned} \tag{3}$$

where $\mathbb{S}^n_+$ denotes the set of real positive semidefinite symmetric matrices. The nonconvex rank-one constraint could be dropped to obtain an SDR of (3) as

$$\begin{aligned} \min_{\mathbf{X}\in\mathbb{S}^n_+}\ & trace(\mathbf{Q}\mathbf{X}) \\ \text{subject to:}\ & \mathbf{A}_i\mathbf{X} \preceq \mathbf{b}_i, i = 1,\ \cdots,\ m. \end{aligned} \tag{4}$$

which turns out to be a standard SDP [14]. With regarding to convert a globally optimal solution $\mathbf{X}^*$ of (4) to a feasible solution of (2), a rank-one approximation can be applied as

$$\tilde{\mathbf{x}} = \sqrt{\lambda_1}\mathbf{v}_1 \tag{5}$$

where λ_1 is the largest eigenvalue of $\mathbf{X}^*$ and $\mathbf{v}_1$ is the respective eigenvector. Better approximation result can be obtained via randomization, which is described in [13].

III. Methodology

The proposed methodology is explained in details in this section. We begin with the classical QP based trajectory generation and discuss the novel 3D corridor constraint. The optimization of time allocation leveraging SDP and SDR is then described.

A. Trajectory Generation Using QP

If the time allocation is known, trajectory generation can be solved as a QP problem [1]. The objective function is defined with

$$\mathcal{J} = \frac{1}{2}\int_{t_0}^{t_N} ||f^{(k)}(t,\mathbf{p})||^2 dt \tag{6}$$

where $f^{(k)}(t,\mathbf{p})$ denotes the k^{th} derivative of trajectory. With (1), for each segment l of polynomial, the k^{th} derivative is given by

$$f_l^{(k)}(t,\mathbf{p}_l) = [\underbrace{0 \ \cdots \ 0}_{k} \ \underbrace{\beta_j t^{j-k} \ \cdots \beta_n t^{n-k}}_{j=k,\ \cdots,\ n}]\mathbf{p}_l \tag{7}$$

$$\beta_j = \frac{j!}{(j-k)!} \tag{8}$$

The objective function $\mathcal{J}$ can then be written as

$$\mathcal{J} = \frac{1}{2}\mathbf{p}^T\mathbf{Q}(t)\mathbf{p} \tag{9}$$

where $\mathbf{Q} = blkdiag(\mathbf{Q_1},\ \cdots,\ \mathbf{Q_M})$ is a block diagonal matrix. Here t is omitted since t is known. The QP can be formulated as

$$\begin{aligned} \text{minimize: } & \mathcal{J} = \frac{1}{2}\mathbf{p}^T\mathbf{Q}\mathbf{p} \\ \text{subject to: } & \mathbf{A}\mathbf{p} = \mathbf{b} \\ & \mathbf{C}\mathbf{p} \preceq \mathbf{d} \end{aligned} \tag{10}$$

$\mathbf{A}$ and $\mathbf{b}$ represent the equality constraints including:

- **Waypoints constraints** with designated position, velocity and acceleration that can be written as $f_l^{(k)}(t,\mathbf{p}_l) = b_l^{(k)}$, where $b_l^{(k)}$ represents the given constraints for the k^{th} derivative of l^{th} segment, i.e. $k=0$ equals to enforce a position constraint and $k=2$ means a desired acceleration constraint. From (8), this kind constraint can be written as affine function $\mathbf{a}_{l,k}\mathbf{p} = b_{l,k}$.
- **Continuity constraint** for maintaining smoothness, a trajectory is assumed to be at least C^2 continuous which means the derivatives up to 2^{ord} should be equal for adjoint segments. If the derivative is designated, then 2 equality constraints are enforced. Otherwise, we add the continuity constraint as $f_l^{(k)}(t_{l+1},\mathbf{p}_l) - f_{l+1}^{(k)}(t_{l+1},\mathbf{p}_{l+1}) = 0$. Similarly, it can be written as affine function $\mathbf{a}_{l,l+1,k}\mathbf{p} = 0$.

$\mathbf{C}$ and $\mathbf{d}$ are used for inequality constraints, such as:

- **Physical constraints**, e.g. maximum velocity, maximum acceleration, etc, which ensures the feasibility of the generated trajectory. Usually, the quadratic constraint $||v||_2^2 = vx^2 + vy^2 + vz^2 \leq ||v_{max}||_2^2$ is simplified with three conservative separate constraints $|v_i| \leq v_{max,i},\ i \in x,\ y,\ z,\ ||v_{max,x}||_2^2 + ||v_{max,y}||_2^2 + ||v_{max,z}||_2^2 \leq ||v_{max}||_2^2$, which also applies to acceleration, etc. With the polynomial representation, these constraints can be formulated with affine function, e.g. the maximum velocity constraint $f^{(1)}(t,\mathbf{p}_x) \leq v_{max}$ can be formulated as $\mathbf{c}_x \leq d_x$.
- **Corridor constraints**. Corridor constraints are preferred for collision avoidance in the decoupled motion planning framework where a collision-free path is generated by a global planner. If trajectory can be bounded within a range of the collision-free path, then trajectory is guaranteed to be collision-free which will boost the efficiency since collision check is normally computationally expensive [15]. Corridor constraints are imposed as nonconvex constraints in [16], which would result in a nonconvex optimization problem. We herein propose a 3D corridor constraint which ensures the generated trajectory being bounded within the corridor by using 4 linear inequality constraints. Now considering two consecutive waypoints $\mathbf{p}_i$ and $\mathbf{p}_{i+1}$, the normalized vector from $\mathbf{p}_i$ to $\mathbf{p}_{i+1}$ can be easily computed as $\mathbf{v}_1 = \frac{\mathbf{p}_{i+1}-\mathbf{p}_{i+1}}{||\mathbf{p}_{i+1}-\mathbf{p}_{i+1}||}$. Two orthogonal normalized vectors can be obtained by $\mathbf{v}_2 = [0\ -\mathbf{v}_1(3)\ \mathbf{v}_1(2)]^T$, $\mathbf{v}_3 = \mathbf{v}_1 \times \mathbf{v}_2$. Together with those two orthogonal normal vector and the line $line_{\mathbf{p}_1 \to \mathbf{p}_2}$, two orthogonal supporting planes intersected with line $line_{\mathbf{p}_1 \to \mathbf{p}_2}$ could be established. By constraining the distance from trajectory to these two supporting planes with a threshold denoted with $l_{corridor}$, the trajectory is guaranteed to be bounded by a square box with its surfaces being parallel to respective supporting planes and side length being $2l_{corridor}$. Considering the equation of point-to-plane distance $dist = \frac{|ax+by+cz+d|}{\sqrt{a^2+b^2+c^2}}$ and trajectories represented as polynomials $x = f(t,\mathbf{p}_x)$, $y = f(t,\mathbf{p}_y)$, $z = f(t,\mathbf{p}_z)$, the trajectory-to-plane distance can be formulated as

$$dist = \frac{|af(t,\mathbf{p}_x) + bf(t,\mathbf{p}_y) + cf(t,\mathbf{p}_z) + d|}{\sqrt{a^2+b^2+c^2}} \tag{11}$$

which is nothing more than a r^{th} order polynomial. The 3D corridor constraint can be imposed as

$$\begin{cases} af(t,\mathbf{p}_x) + bf(t,\mathbf{p}_y) + cf(t,\mathbf{p}_z) \leq l_{corridor} - d \\ -(af(t,\mathbf{p}_x) + bf(t,\mathbf{p}_y) + cf(t,\mathbf{p}_z)) \leq l_{corridor} + d \end{cases} \tag{12}$$

Similarly, they can be further written as affine functions.

The aforementioned equality and inequality constraints can all be expressed with affine functions which are convex. Moreover, the Hessian matrix Q is positive semi-definite since it is an integration of a positive semi-definite matrix over a nonnegative interval. It is clearly Eq. 10 is a convex optimization problem, which can be solved efficiently.

It is worthy to note that inequality constraints are meant to be satisfied over a whole time interval, theoretically corresponding to an infinite number of inequality constraints. This can be relaxed by discretization and creating finite inequality constraints [1] [16]. However, the discretization density is often tricky to estimate, so we utilize another

iterative method based on creating supporting inequality constraints at extrema points. By that means, a trajectory is firstly solved without inequality constraints. Then violation points can be found by checking the extrema points and enforced back by adding inequality constraints on the extrema instants. It has been proven that a theoretical upper bound existed for the number of the supporting constraints for constraining the polynomial within the boundary range [8]. Thanks to the polynomial representation, extrema points can be analytically found for trajectory lower than 5^{th} order.

B. Time Allocation Optimization Using SDP

Considering the initial assigned traversing time for the l^{th} segment is t_l and the corresponding updated traversing time $\tilde{t}_l$, their relationship can be formulated as

$$t_l = s\tilde{t}_l \tag{13}$$

Consequently, the k^{th} derivative of the l^{th} trajectory segment to $\tilde{t}_l$ can be obtained using Chain rule as

$$\frac{d^{(k)}f}{d\tilde{t}^{(k)}} = \frac{d^{(k)}f}{dt^k}\frac{dt^{(k)}}{d\tilde{t}^{(k)}} = s^k\frac{d^{(k)}f}{dt^k} \tag{14}$$

Assuming the coefficients of polynomials are known, the cost function (6) is a polynomial function of s

$$\mathcal{J} = \tilde{q}_1 s + \tilde{q}_2 s^2 + \cdots + \tilde{q}_{2k+1}s^{2k+1} \tag{15}$$

Unfortunately, the cost function is nonconvex which may turn out to be difficult to solve directly. Instead, we introduce a new variable $\bar{\mathbf{s}} = [1,\ s,\ s^2,\ \cdots,\ s^{k+1}]^T$, cost function (15) could be written as

$$\mathcal{J} = \bar{\mathbf{s}}^T\bar{\mathbf{Q}}\bar{\mathbf{s}} \tag{16}$$

From section II-D, we know that (16) could be relaxed to a classical SDP with optimization variable being $\bar{\mathbf{S}} = \bar{\mathbf{s}}\bar{\mathbf{s}}^T$. Inequality constraints can be reformulated with $\bar{\mathbf{S}}$ in the form of $trace(\bar{\mathbf{C}}\bar{\mathbf{S}}) \leq \mathbf{d}$. Equality constraints have to be relaxed in SDP optimization.

We summarize the SDP optimization problem as

$$\begin{aligned} &min_{\bar{\mathbf{S}}\in\mathbb{S}_+^{k+2}} \quad trace(\bar{\mathbf{Q}}\bar{\mathbf{S}}) \\ &\text{subject to:} \quad trace(\bar{\mathbf{C}}_i\bar{\mathbf{S}}) \leq \mathbf{d_i}, \quad i = 1, \cdots, \bar{m} \end{aligned} \tag{17}$$

The rank-one approximation is used for recovering a feasible solution from $\bar{\mathbf{S}}^*$. A new time allocation is computed with the feasible solution using (13).

C. The Algorithm

The overall algorithm consists of the following steps:

1: Trajectory generation using initial time allocation using QP.
2: Check feasibility. If feasible, exit.
3: Solve SDP for updating time allocation and go back to 1.

It is worthy to note here that the initial time allocation can be chosen rather trivially, e.g. pick a uniform distribution $t \in [0,\ 1,\ \cdots,\ N]$. Actually, the initial time allocation could be improper, while it will be optimized automatically in this algorithm. The aforementioned feasibility check is to check if the generated trajectory satisfies all designated physical and corridor constraints. Violation points will be used for creating supporting inequality constraints in the next iteration. At the end of iteration, the output of the proposed approach will be the optimized time t^* allocation as well as the trajectory coefficients which minimizes the cost function defined in Eq 10. Compared with [1], the proposed approach releases the prerequisite time allocation.

TABLE I
TRAJECTORY GENERATION PARAMETERS

Parameter	Value
$v_{k,max},\ k \in \{x, y\}$	$10m/s$
$v_{z,max}$	$3m/s$
$a_{k,max},\ k \in \{x, y\}$	$8m/s^2$
$a_{z,max}$	$3m/s^2$
$l_{corridor}$	$0.4m$

IV. EXPERIMENTS

Simulation results are presented in this section. Our trajectory generation approach is implemented in Matlab. The semi-definite problem is solved with CVX[1] which is a package smafor specifying and solving convex programs [17], [18].

We start with an experiment to validate the robustness of the proposed approach. In this experiment, waypoints are manually assigned through a developed GUI program. The initial and target states (velocity and acceleration) and dynamic constraints (i.e. maximum velocity, acceleration, corridor size) could also be designated arbitrarily. A snapshot of one experimental result is shown in Fig. 2 and detailed in Fig. 3. It can be seen from Fig. 2 that the generated trajectory is strictly bounded with 3D corridor (in this case, $l_{corridor} = 1m$). The exact distances from trajectory to the 2 supporting planes are shown in Fig. 3, which shows that the trajectory is constrained within the corridors formed by the two supporting planes. The corresponding velocity and acceleration profiles are demonstrated in the middle and right columns of Fig. 3, where the maximum velocity and acceleration for x, y and z axis are $8m/s$, $5m/s^2$, $3m/s$, $2m/s^2$, respectively. From Fig. 3, the velocity and acceleration of the trajectory is well bounded by the maximum tolerances, which guarantees the feasibility of the planned trajectory.

In the following experiment, numerical simulation of motion planning in a $15m \times 15m \times 2m$ workspace with artificially generated obstacles is carried out. The motion planning is composed of a global planner using the Rapidly-exploring Random Tree Star (RRT*) [19] for generating a collision-free path and a trajectory generator using the proposed method for planning a feasible trajectory corresponding to the path. The workspace is represented with a 3D grid map with the resolution being $0.1m$. Obstacles are generated by using the Perlin noise in the experiment, as shown in Fig. 4. For this experiment, the starting waypoint and ending waypoint are

[1]http://cvxr.com/cvx/

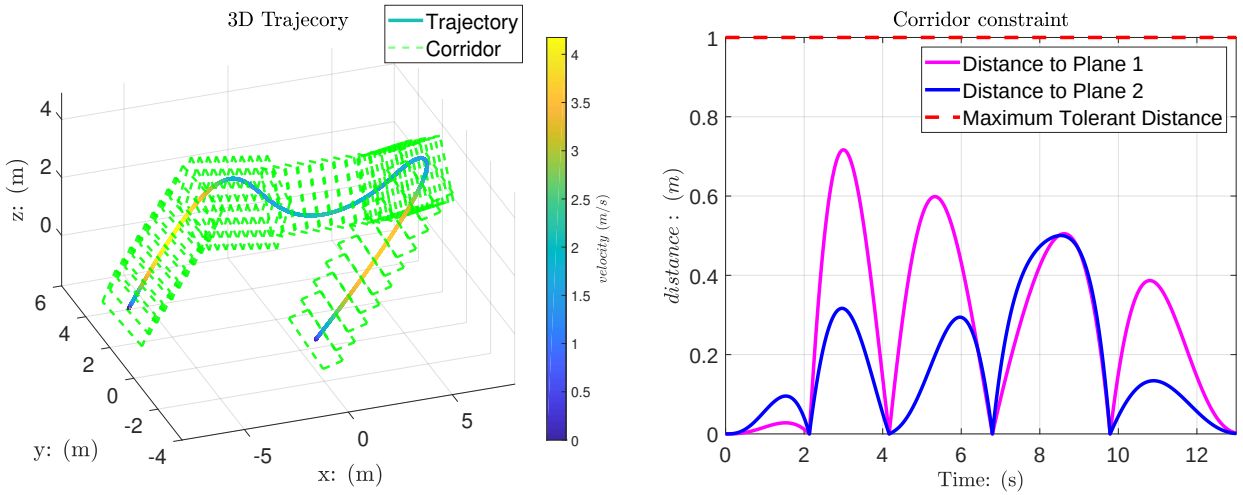

Fig. 2. The left figure shows the trajectory generation result subjected to 3D corridor. For simplicity, discrete samples represented as 3D windows are plotted. Colored solid line shows the generated trajectory constrained with the proposed 3D corridor with the color representing the velocity profile. The right figure details the distances from the trajectory to two supporting hyperplanes. The maximum tolerant distance is shown with the red dashed line.

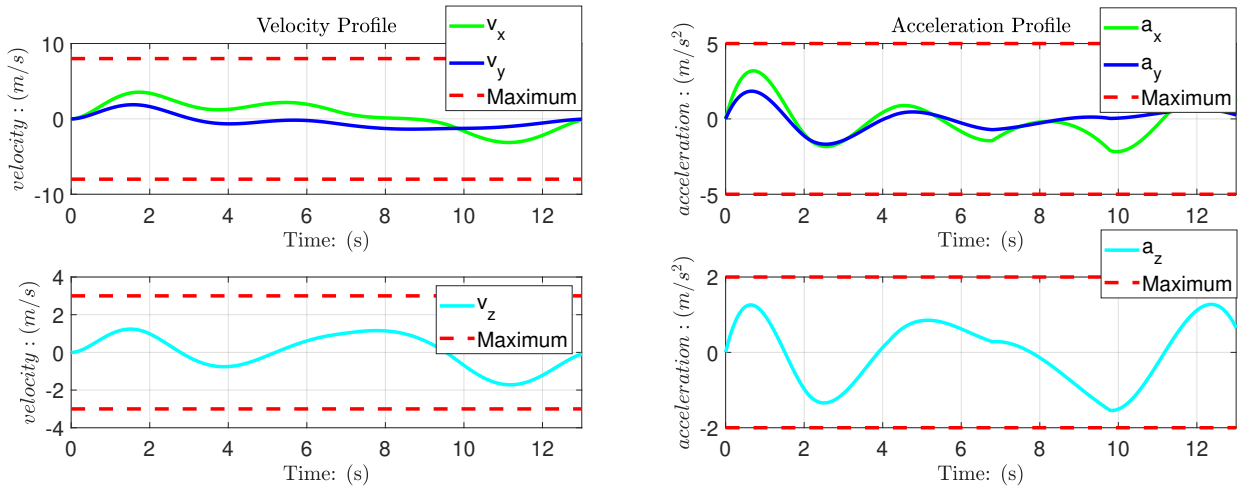

Fig. 3. Detailed profile curves for trajectory shown in Fig. 2. The left and right figures show the velocity profile and acceleration profile corresponding to the generated trajectory. The maximum tolerant constraints are plotted with the red dashed lines.

set to $[-7,\ -7,\ 0]$ and $[8,\ 8,\ 1]$, respectively. Intermediate waypoints are generated automatically using the RRT* algorithm and then pruned with the line-of-sight collision check, where a collision radius of $0.5m$ is used here. The initial and final velocities, accelerations are set to zero. The velocity and acceleration profile for those intermediate waypoints are left free for optimization. Constraints for trajectory generation are defined in Table I. The experimental result is shown in Fig 4. Although the map is challenging, the resultant trajectory is guaranteed to be collision-free thanks to the proposed 3D corridor which controls the deviations of the generated trajectory to the planned path. The velocity and acceleration profiles are shown in the middle and bottom figures of Fig 4. It can be observed that velocity and acceleration are strictly bounded with given constraints, which ensures the feasibility of the generated trajectory.

V. Conclusion And Future Works

In this paper, a trajectory generation approach has been proposed for quadcopter vehicle. The proposed method solves the trajectory generation problem iteratively using quadratic programming and semi-definite programming. Compared with previous works, the proposed method doesn't rely on a proper prior time allocation but optimize the time allocation automatically via relaxation and semi-definite programming. Experiments show that the proposed method can generate a feasible trajectory which satisfies dynamic and environmental constraints even with an improper time allocation.

In the future, we will consider the onboard implementation of the proposed method on quadcopter vehicle to enable

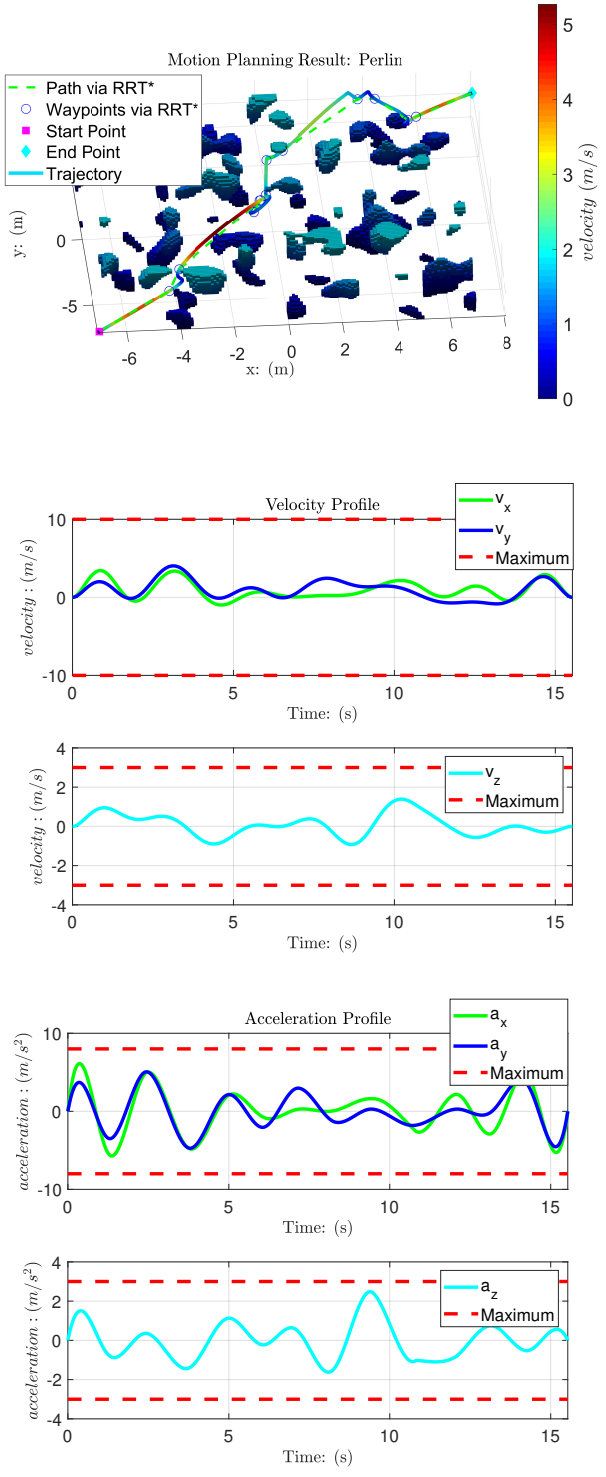

Fig. 4. The Top figure shows the trajectory generation result for the map of Perlin noise. The start point and end point are plotted as the magenta and cyan square and diamond points, respectively. Intermediate waypoints planned by RRT* are shown with blue circular points and linked by the green dotted line. The generated trajectory is demonstrated with the solid colored line with the color representing the velocity profile. The middle and bottom figures detail the velocity profile and acceleration profile corresponding to the trajectory. The maximum velocity constraint and maximum acceleration constraint are shown with red dashed lines.

quadcopter to navigate autonomously in the outdoor environment. The proposed method is also planned to be extended to task-driven trajectory generation problem involving multiple UAVs in the future.

REFERENCES

[1] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 2520–2525.

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

[3] J. Chen and S. Shen, "Using a quadrotor to track a moving target with arbitrary relative motion patterns," in *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*. IEEE, 2017, pp. 5310–5317.

[4] F. Gao and S. Shen, "Quadrotor trajectory generation in dynamic environments using semi-definite relaxation on nonconvex qcqp," in *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE, 2017, pp. 6354–6361.

[5] M. W. Mueller, M. Hehn, and R. D'Andrea, "A computationally efficient motion primitive for quadrocopter trajectory generation," *IEEE Transactions on Robotics*, vol. 31, no. 6, pp. 1294–1310, 2015.

[6] D. Falanga, A. Zanchettin, A. Simovic, J. Delmerico, and D. Scaramuzza, "Vision-based autonomous quadrotor landing on a moving platform," in *IEEE/RSJ International Symposium on Safety, Security and Rescue Robotics (SSRR)*, no. EPFL-CONF-232507, 2017.

[7] M. Burri, H. Oleynikova, M. W. Achtelik, and R. Siegwart, "Real-time visual-inertial mapping, re-localization and planning onboard mavs in unknown environments," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. IEEE, 2015, pp. 1872–1878.

[8] J. Chen, T. Liu, and S. Shen, "Online generation of collision-free trajectories for quadrotor flight in unknown cluttered environments," in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. IEEE, 2016, pp. 1476–1483.

[9] F. Gao, W.Wu, Y. Lin, and S. Shen, "Proc. of the IEEE intl. conf. on robot. and autom." in *Online Safe Trajectory Generation For Quadrotors Using Fast Marching Method and Bernstein Basis Polynomial*, Brisbane, Australia, May 2018.

[10] S. G. Johnson, "The nlopt nonlinear-optimization package," 2014.

[11] M. Van Nieuwstadt, M. Rathinam, and R. Murray, "Differential flatness and absolute equivalence of nonlinear control systems," *SIAM Journal on Control and Optimization*, vol. 36, no. 4, pp. 1225–1239, 1998.

[12] M. Faessler, A. Franchi, and D. Scaramuzza, "Differential flatness of quadrotor dynamics subject to rotor drag for accurate tracking of high-speed trajectories," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 620–626, 2018.

[13] Z.-Q. Luo, W.-K. Ma, A. M.-C. So, Y. Ye, and S. Zhang, "Semidefinite relaxation of quadratic optimization problems," *IEEE Signal Processing Magazine*, vol. 27, no. 3, pp. 20–34, 2010.

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

[15] S. S. Skiena, *The algorithm design manual: Text*. Springer Science & Business Media, 1998, vol. 1.

[16] F. Augugliaro, A. P. Schoellig, and R. D'Andrea, "Generation of collision-free trajectories for a quadrocopter fleet: A sequential convex programming approach," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, 2012, pp. 1917–1922.

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

[18] ——, "Graph implementations for nonsmooth convex programs," in *Recent Advances in Learning and Control*, ser. Lecture Notes in Control and Information Sciences, V. Blondel, S. Boyd, and H. Kimura, Eds. Springer-Verlag Limited, 2008, pp. 95–110, $http://stanford.edu/\ boyd/graph_dcp.html$.

[19] S. Karaman, "Incremental sampling-based algorithms for optimal motion planning," *Robotics Science and Systems VI*, vol. 104, p. 2, 2010.