2019 19th International Conference on Control, Automation and Systems (ICCAS 2019)
Oct. 15∼18, 2019; ICC Jeju, Jeju, Korea

Integrated Planning and Control for Collision-free Trajectory Generation in 3D Environment with Obstacles

Xiaoxue Zhang[1,2*], Jun Ma[2,3], Sunan Huang[4], Zilong Cheng[1,2], Tong Heng Lee[1,2]

[1]NUS Graduate School for Integrative Sciences and Engineering, National University of Singapore, 119077, Singapore (xiaoxuezhang@u.nus.edu)
[2]Department of Electrical and Computer Engineering, National University of Singapore, 117583, Singapore (xiaoxuezhang@u.nus.edu)
[3]Department of Mechanical Engineering, University of California, Berkeley, Berkeley, CA 94720, USA (jun.ma@berkeley.edu)
[4]Temasek Laboratories, National University of Singapore, 117411, Singapore (tslhs@nus.edu.sg)

Abstract: Control of autonomous robots (ARs) is a hot topic over the past three decades, and motion planning is becoming one of the major challenges in AR control. Substantial works have been done for motion planning in 2-dimensional (2D) environments, but planning in 3D complex environments is still an open problem to be investigated. In this paper, we propose a trajectory planning method to plan a high-quality and safe trajectory in 3-dimensional (3D) environments with multiple obstacles. The proposed method is based on the integration of a global optimal 3D path planner with a local planner. First, a hierarchical path planning algorithm is used as a global path planner to compute a near-optimal path formed by a set of waypoints. Second, a local path trajectory planner is designed to track the path computed by the global path planner with multiple constraints of dynamics and environment satisfied. Finally, simulations are conducted to validate the effectiveness of the proposed trajectory generation approach subject to the constraints of robot physical limits and environmental requirements.

Keywords: 3D path planning, model predictive control (MPC), path tracking, trajectory planning

1. INTRODUCTION

Over the past decades, there has been an increasing trend towards the development and deployment of autonomous robots (ARs). The trajectory generation task is becoming one of the key technologies of ARs and has been widely investigated by researchers around the world [1]. It can be applied on various mobile robots, such as unmanned underwater robots [2], unmanned ground vehicles [3], unmanned aerial vehicles (UAV) [4], etc.. It is not only a path planning problem which is to plan a path for a robot, but also refers to the problem considering how to move along the path, which means trajectory is a set of states that are associated with time [5]. An efficient trajectory can effectively contribute to increasing ARs' intelligence and autonomy. Most of the existing works focus on path planning in 2D environments [7-10]. However, to find an optimal or near-optimal trajectory in a complex and unknown 3D environment has a great prospect though the difficulties increase exponentially. Finding a 3D optimal trajectory planning problem is an NP-hard problem; thus, no common solution exists. In order to plan a collision-free path in a 3D cluttered environment, it is essential to solve the problem of how to model the environment while considering the relevant constraints. Real-time 3D trajectory planning in unknown terrain has not been studied extensively.

Among all of the applications of trajectory generation, UAV is an ideal platform for 3D trajectory generation problem, because trajectory planning may need to work in 3-dimensional state space with more than 3 degrees of freedom and multiple constraints due to its dynamical and physical limits in a typical UAV application. It may involve an optimization problem which may have more than 12 dimensions, where multiple equality and inequality constraints about system state and control input have to be satisfied. it is very difficult to obtain an exact analytic solution. Fortunately, model predictive control (MPC) provides a solution to deal with this issue. In MPC, an optimization problem is solved at each time step to compute a sequence of control input over a prediction horizon, based on the prediction of future state and control of this system by dynamic models and constraints.

M. A. Mousavi *et al.* use a linear time-varying model predictive control (LTV-MPC) as the trajectory planning method with model linearization to handle the moving obstacles, then the LTV-MPC problem is transformed into a convex optimization problem to solve the path, but they limit the motion of UAV into surface without vertical movement [10]. L. Zhang *et al*. propose a method to deal with non-convex constraints for optimization problem by rewriting it as a linear program with mixed integer linear constraints for collision avoidance used in MPC for UAV trajectory planning in 3D environment [11]. M. Hehn *et al* work on trajectory generation by adding dynamic constraints to an optimal control problem and testify the existence of optimal trajectories for quadrotors [12]. H.J. Kim *et al.* use nonlinear MPC to generate trajectories for UAVs while maintaining the input and state constraints [13]. These MPC-based trajectory generation methods can effectively address the state and input constraints, but it only ensures local optimality, which

means it is hard to obtain the global optimality in a high-dimensional, complex environment, i.e., MPC tends to be a local trajectory generation. Therefore, an integration of a global optimal 3D path planner with a local planner is required in this paper to generate a high-quality (near-optimal) and safe trajectory in a 3D environment with multiple obstacles.

In this paper, we introduce an integrated motion planning and control method for autonomous robots in 3D environments. The proposed method uses a hierarchical structure to integrate a path planning with local dynamic optimization. At the higher level, an HLT* algorithm is introduced for global path planning, and a set of waypoints is given, while at the lower level, based on the obtained waypoints, a smooth path is generated by a local planner and it is used as a reference for MPC control. Here, MPC computes a sequence of collision-free control inputs by considering a robot's constraints and obstacles. Therefore, the proposed method gives a near-optimal solution for motion planning. Simulation results validate the effectiveness of the proposed method is validated.

2. GLOBAL PATH PLANNING

In this paper, a new path planning algorithm is used, called Hierarchical Lazy Theta* (HLT*) [14]. HLT* algorithm utilizes the hierarchical approach to accelerate the computation by abstracting the environment map into multiple levels and utilizes an any-angle path-finding algorithm, i.e., Lazy Theta* [15] to find short any-angle paths and to reduce the number of the usage of the line-of-sight check. Further details of the HLT* algorithm and related matters are available in [14] and [15], and these details will also be additionally described during the conference presentation.

In fact, A* algorithm can find the shortest path which is formed by edges of 26-neighbor cubic grids in a 3D environment. The length of the path found by A* is 1.1281 times the length of the truly shortest path [15]. The any-angle path-finding algorithm, lazy theta*, can find a shorter any-angle path. However, it is not optimal, i.e., cannot guarantee to find the true shortest path, because the parent of vertex should be a neighbor or parent, rather than any point in the map. Besides, the hierarchical approach is to divide a large task into many small subtasks by using different resolution of the environment. Therefore, HLT* is a near-optimal planner for an unknown 3D environment. It is a new heuristic-based planning algorithm to compute near-optimal paths fast enough to operate in real-time in an unknown 3D environment.

Therefore, HLT* algorithm is used as a global path planner to find a set of waypoints in a 3D environment with multiple obstacles, and the set of waypoints will be used as a reference for the local path planner.

3. LOCAL TRAJECTORY GENERATION

3.1 Curve Smoothing

A set of reference waypoints has been computed by global path planner HLT*. Then, connecting and smoothing this waypoint set to form a feasible trajectory will be helpful for local path planner.

Bézier curve is a popular method to fit and smooth the waypoints due to its good geometrical properties [16]. Thus the curves can be represented perfectly. A Bézier curve of degree n is a parametric curve composed of Bernestein basis polynomials of degree n with Eq. (1) [17].

$$P(t) = \sum_{i=0}^{n} p_i B_{i,n}(t),\ t \in [0,1]. \tag{1}$$

where $B_{i,n}(t) = C_n^i t^i (1-t)^{n-i}$; n is the degree of Bézier curve, and $p_i = (p_{xi}, p_{yi}, p_{zi})$ is the position coordinates of the ith point, given $n+1$ points p_i in 3-dimensional environment, $i = 0, 1, \cdots, n$.

3.2 Problem Definition

3.2.1 Dynamics Modeling

To demonstrate the trajectory planning process, a quadrotor helicopter is used as the application platform. $p = [p_x, p_y, p_z]^\top$ denotes the position of quadcopter in the world coordinates; $v = [v_x, v_y, v_z]^\top$ represents the velocity of the quadcopter in three dimensions x, y, z; $\zeta = [\phi, \theta, \psi]^\top$ denotes the angle: roll, pitch and yaw; the angular velocity in three dimensions is represented by $\omega = [\omega_x, \omega_y, \omega_z]^\top$. Assume there are no aerodynamic and gyroscopic effects. The dynamic model of quadcopter can be represented by Eq. (2).

$$\begin{aligned}
\dot{p}(t) &= R(\phi,\theta,\psi)^\top v(t) \\
\dot{v}(t) &= -\omega(t) \times v(t) + gR(\phi,\theta,\psi)e + eT/m \\
\dot{\zeta}(t) &= W(\psi,\theta,\phi)\omega(t) \\
\dot{\omega}(t) &= J^{-1}(-\omega(t) \times J\omega(t) + \tau).
\end{aligned} \tag{2}$$

where $e = [0,0,1]^\top$; $\tau = [\tau_x, \tau_y, \tau_z]^\top$ represents the torques of the quadcopter in each dimension; g is the gravitational acceleration; m is the mass of this helicopter; T denotes the total thrust; $J = \text{diag}(J_x, J_y, J_z)$ denotes the moment of inertia of the quadcopter; $R(\psi,\theta,\phi)$ denotes the rotation matrix of the quadcopter. In order to represent the rotation matrix $R(\psi,\theta,\phi)$ in a simple way, we use c, s, t, sc to represent $\sin, \cos, \tan, \sec$ respectively.

$$R(\phi,\theta,\psi) = \begin{bmatrix} c\theta c\psi & c\theta s\psi & -s\theta \\ s\theta c\psi s\phi - s\psi c\phi & s\theta s\psi s\phi + c\psi c\phi & c\theta s\phi \\ s\theta c\psi c\phi + s\psi s\phi & s\theta s\psi c\phi - c\psi s\phi & c\theta c\phi \end{bmatrix}.$$

$$W(\phi,\theta,\psi) = \begin{bmatrix} 1 & s\phi t\theta & c\phi t\theta \\ 0 & c\phi & -s\phi \\ 0 & s\phi sc\theta & c\phi sc\theta \end{bmatrix}.$$

Each quadcopter is equipped four rotors. For each rotor, there are a vertical force due to the rotation of the rotor and a moment perpendicular to the plane of the propeller rotation. Therefore, rotor thrusts of each rotor are chosen as control inputs, i.e. $u = [F_1, F_2, F_3, F_4]^\top$, and the relationship between individual thrusts $[F_1, F_2, F_3, F_4]^\top$ and individual torques

$[\tau_x, \tau_y, \tau_z]^\top$ and total thrusts T is expressed by Eq. (3), where

$$\begin{bmatrix} T \\ \tau_1 \\ \tau_2 \\ \tau_3 \end{bmatrix} = \begin{bmatrix} -1 & -1 & -1 & -1 \\ 0 & -L & 0 & L \\ L & 0 & -L & 0 \\ -c & c & -c & c \end{bmatrix} \begin{bmatrix} F_1 \\ F_2 \\ F_3 \\ F_4 \end{bmatrix}. \quad (3)$$

Define the state vector as

$$\begin{aligned} x &= [p_x, p_y, p_z, v_x, v_y, v_z, \phi, \theta, \psi, \omega_x, \omega_y, \omega_z]^\top \\ &= [p^\top, v^\top, \zeta^\top, \omega^\top]^\top. \end{aligned} \quad (4)$$

The dynamic model of a quadcopter can be formulated by the form of $\dot{x} = f(x) + Bu$. Let $u = u_{eq} + \delta u$, where $u_{eq} = \left[\frac{mg}{4}, \frac{mg}{4}, \frac{mg}{4}, \frac{mg}{4}\right]^\top$ is used to overcome the gravity of quadcopter. Therefore, the quadcopter system can be written as Eq. (5).

$$\begin{bmatrix} \dot{p} \\ \dot{v} \\ \dot{\zeta} \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} R(\phi, \theta, \psi)^\top v \\ -\omega \times v + gR(\phi, \theta, \psi)e \\ W(\phi, \theta, \psi)\omega \\ J^{-1}(-\omega \times J\omega) \end{bmatrix} + B(u_{eq} + \delta u). \quad (5)$$

where

$$B = -\begin{bmatrix} O_{(5,4)} \\ \begin{bmatrix} \frac{1}{m} & \frac{1}{m} & \frac{1}{m} & \frac{1}{m} \end{bmatrix} \\ O_{(3,4)} \\ \begin{bmatrix} 0 & \frac{L}{J_x} & 0 & -\frac{L}{J_x} \\ -\frac{L}{J_y} & 0 & \frac{L}{J_y} & 0 \\ \frac{\gamma}{J_z} & -\frac{\gamma}{J_z} & \frac{\gamma}{J_z} & -\frac{\gamma}{J_z} \end{bmatrix} \end{bmatrix}$$

where $O_{(5,4)}$ and $O_{(6,4)}$ are 5-by-4 and 6-by-4 zeros matrices, L denotes the distance from the center of this quadcopter to its rotor, and γ is the ratio of rotor angular momentum to lift the quadcopter.

3.2.2 Constraints and Objective Function

Since the control input of the quadcopter is not allowed to change arbitrarily, it should be constrained according to its physical limits, which means the difference of the control input δu should be limited within a desired range, shown as Eq. (6).

$$\delta u_{\min} \leq \delta u \leq \delta u_{\max}. \quad (6)$$

where $\delta u_{\min}$ and $\delta u_{\max}$ are the lower and upper bounds of the control input difference δu.

Moreover, the path tracking error should be limited within a desired range, which can be represented by Eq. (7).

$$\epsilon_{\min} \leq \epsilon \leq \epsilon_{\max}. \quad (7)$$

where $\epsilon = p - p_{ref}$; p is the real position of UAV, and p_{ref} denotes the reference position planned by global path planner (HLT*). $\epsilon_{\min}$ and $\epsilon_{\max}$ are the allowed minimum and maximum tracking errors.

The generated trajectory should guarantee the distance between current location and obstacles is greater than the pre-defined safe distance d_{safe} by

$$||p - p^k_{obs}||_2 \geq d_{safe}. \quad (8)$$

where p^k_{obs} is the 3D position coordinates of nearest kth obstacle within the detection radius r_{det} of this UAV in the 3D environment.

The objective function is defined as Eq. (9) to penalize on the tracking error and difference of control input in a unit time step for generalizing a high-quality trajectory, where

$$\begin{aligned} J_{\text{cost}}(k) = &\sum_{i=1}^{N_p} [x(k+i|k) - x_{ref}(k+i|k)]^\top \\ &Q\,[x(k+i|k) - x_{ref}(k+i|k)] + \\ &\sum_{i=1}^{N_c} [\delta u(k+i|k)^T\, R\, \delta u(k+i|k)]. \end{aligned} \quad (9)$$

where N_p and N_c are prediction and control horizons with $N_p \geq N_c$, Q and R are weighting matrices to penalize on tracking error and control input, respectively.

3.3 Problem Formulation and Solution

According to the dynamic model and constraints, a nonlinear MPC (NMPC) problem can be formulated as Eq. (10).

$$\begin{aligned} \text{minimize} \quad & J_{\text{cost}}(k) \\ \text{subject to} \quad & \dot{x} = f(x) + Bu \\ & \epsilon_{\min} \leq (p - p_{ref}) \leq \epsilon_{\max} \\ & \delta u_{\min} \leq \delta u \leq \delta u_{\max} \\ & ||p - p^k_{obs}||_2 \geq d_{safe} \\ & -\pi \leq \phi, \psi \leq \pi \\ & -\frac{\pi}{2} \leq \theta \leq \frac{\pi}{2} \\ & x(0) = x(t_0). \end{aligned} \quad (10)$$

At time step k, the cost function is optimized under these constraints starting from $x(k|k)$ to obtain an optimal control sequence

$$u(\cdot|k) = [u(k+1|k), u(k+2|k), \cdots, u(k+N_c|k)]^\top. \quad (11)$$

It is obvious that this is a nonlinear optimization problem with multiple constraints. To solve this NMPC problem, a sequential quadratic programming (SQP) method is used. The pseudocode of the process of solving this NMPC problem is shown in Algorithm 1. Note that if there is no solution, slack factors will be added to relax the constraints to obtain a solution.

4. SIMULATION

Parameters of this quadcopter are shown in Table 1, where I_4 and I_{12} denote the identity matrices with the size of 4×4 and 12×12, respectively. All of the simulations are implemented in Python 3.7 environment on a PC with Intel i5 processor at 3.30 GHz. The video can be found in `https://www.youtube.com/watch?v=eKA4H0n-Zj4`.

Algorithm 1 Procedures to solve the NMPC Problem

Require: p_{ref}, p_{obs}^{k}, N_p, N_c, T, Q, R, d_{safe}

Initialize state vector of quadcopter x_0
Compute the initial cost function J_0
for $i = [1, N_p]$ **do**
  if $i = 1$ **then**
    Compute $x(i)$ based on $x(0)$ using Eq. (5)
  else
    Compute $x(i)$ based on $x(i-1)$ using Eq. (5)
  end if
  Generate constraints
end for
Compute cost function J_{cost} using Eq. (9)
Generate constraints
Solve NMPC problem $\min J_{\text{cost}}$
if Exist a solution **then**
  Execute the first action of control sequence
else
  Add slack factors on constraints
  Go to step 1
end if
return control sequence u^*

Table 1 Setting parameters for the NMPC problem

Definition	Notation	Value	Unit
Mass	m	0.8	kg
Gravity acceleration	g	9.81	m/s^2
Moment of inertia	J_x	0.0244	kg· m^2
Moment of inertia	J_y	0.0244	kg· m^2
Moment of inertia	J_z	0.0436	kg·m^2
Dist. from center to rotor	L	0.162	m
Ratio of rotor angular momentum to lift	γ	2.17e-3	m
Detection radius	r_{det}	10	m
Sample time	Δt	0.05	s
State weighting matrix	Q	I_{12}	-
Input weighting matrix	R	I_4	-
Control horizon	N_c	25	-
Prediction horizon	N_p	25	-
Control input difference upper/lower bound	$\delta u_{\max}$, $\delta u_{\min}$	1.96, -1.96	N
Velocity upper/lower bound	$v_{\max}$, $v_{\min}$	5, -5	m/s
Tracking error upper/lower bound	$\epsilon_{\max}$, $\epsilon_{\min}$	2.5, -2.5	m
Angular velocity upper/lower bound	$\omega_{\max}$, $\omega_{\min}$	2, -2	rad/s

4.1 Simulation in 3D environment without obstacles

When there is no obstacle, this problem becomes a pure path tracking problem. In this scenario, the reference trajectory is set to be a spiral curve. Fig. 1 shows the comparison between reference trajectory and resulted trajectory in 3D environment. The comparison between the reference position and resulted position and the tracking errors in three dimensions x, y, z are showed in Fig. 2, and these tracking errors are maintained within $[-2.5, 2.5]$.

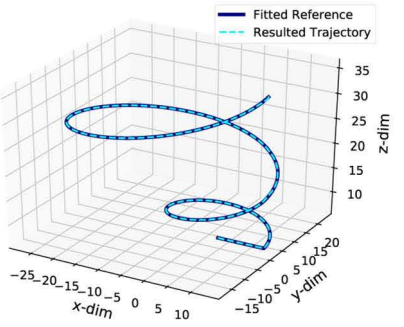

Fig. 1 Comparison between reference and resulted trajectory in 3D environment.

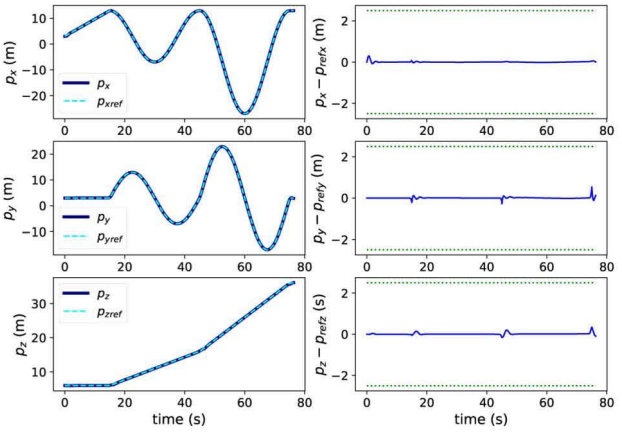

Fig. 2 Comparison between reference and resulted trajectory.

Fig. 3 shows the sequence of control input difference, and this figure shows that the control input difference $\delta u = \left[F_1 - \frac{mg}{4},\ F_2 - \frac{mg}{4},\ F_3 - \frac{mg}{4},\ F_4 - \frac{mg}{4}\right]^\top$ is constrained within the range of $[-1.96, 1.96]$.

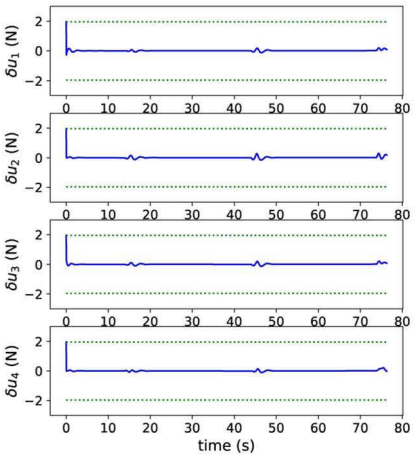

Fig. 3 Bounded control inputs difference.

The velocity $v = [v_x, v_y, v_z]^\top$ and angular velocity $\omega = [\omega_x, \omega_y, \omega_z]^\top$ of quadcopter in three dimensions (x, y, z) are shown in Fig. 4, and they are limited into an interval of $[-5, 5]$ and $[-2, 2]$, respectively.

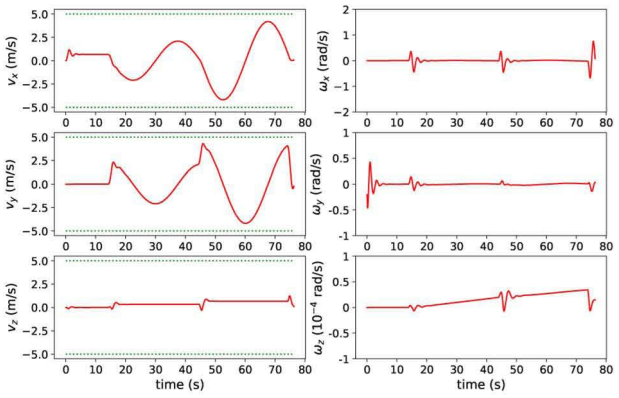

Fig. 4 Bounded velocities and angular velocities in three dimensions.

Simulation results show that the NMPC controller can track a reference path with small tracking errors, and velocity, angular velocity, and control input are successfully constrained into desired ranges.

4.2 Simulation in 3D environment with obstacles

When there are multiple obstacles in a 3D environment, the environment and resulted trajectory are shown in Fig. 5.

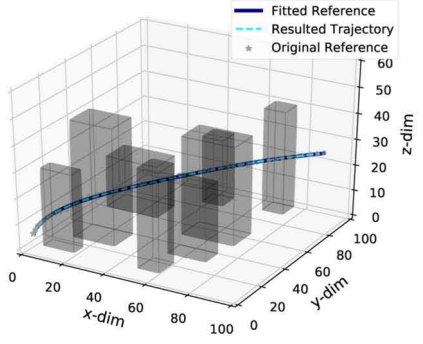

Fig. 5 Comparison between resulted position and reference position in three dimensions.

Fig. 6 shows the difference between reference and resulted positions and the bounded tracking errors in three dimensions. Control input difference computed by the local trajectory planner is shown in Fig. 7 and all the resulted control input differences meet their constraints, i.e. control input δu is in the desired range of $[-1.96, 1.96]$. The velocities and the angular velocities in three dimensions are constrained within their desired range $[-5, 5]$ and $[-2, 2]$, as shown in Fig.8.

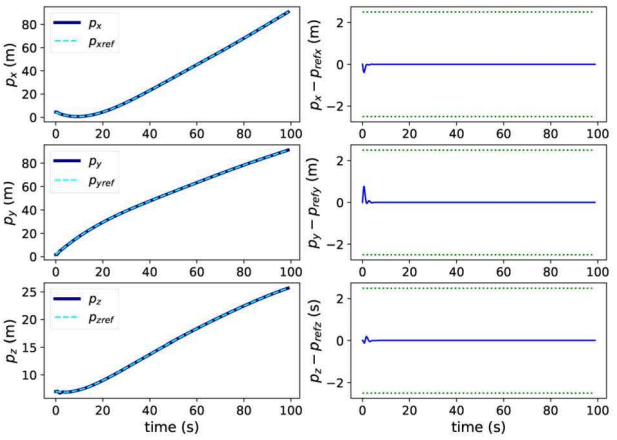

Fig. 6 Comparison between resulted position and reference position in three dimensions.

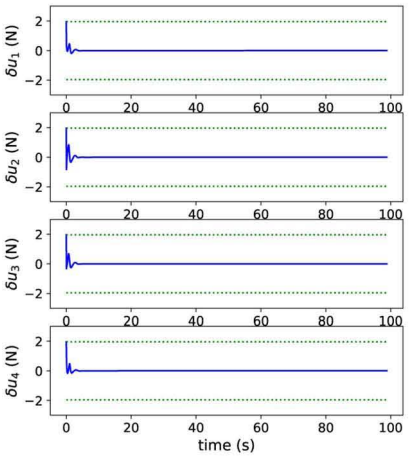

Fig. 7 Bounded control inputs difference.

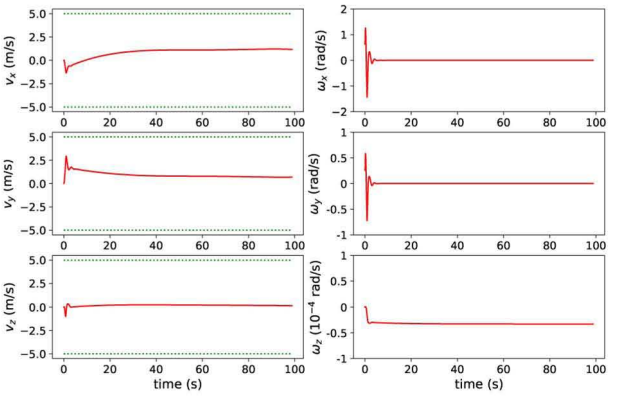

Fig. 8 Bounded velocities and angular velocities in three dimensions.

The shortest distance of the quadcopter from current position $p(i)$ to the nearest obstacles p^k_{obs} is $||p(i) - p^k_{obs}||_2$, and it is bounded in $[1, +\infty)$, shown in Fig. 9. Notably, the initial shortest distance $||p(0) - p^k_{obs}(0)||_2 = 1$. In this figure, we capped the distant by 12 when there is no obstacles detected in this quadcopter's detection radius r_{det}.

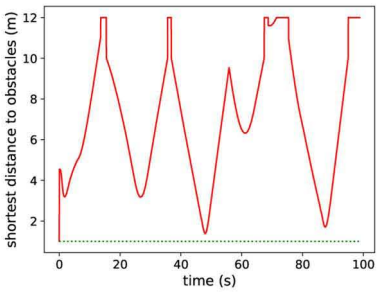

Fig. 9 Shortest distance to obstacles.

5. CONCLUSION

In this paper, an integrated planning and control algorithm is used for UAVs to plan a feasible and collision-free trajectory in a 3D environment with multiple obstacles. In the first path planning process, a hierarchical path planning algorithm, HLT*, is used as a global path planner to compute a set of waypoints. The set of waypoints is approximated to be a smoother and more feasible trajectory as a reference path for the local planner. Then, a nonlinear MPC problem is formulated and solved by sequential quadratic programming to achieve high-quality path tracking performance and avoid collision with obstacles in the 3D environment with multiple constraints. Simulation results show that this method achieves small tracking errors and limit the distance to obstacles into the desired range with satisfying all required constraints.

REFERENCES

[1] T. M. Howard, and A. Kelly, "Optimal rough terrain trajectory generation for wheeled mobile robots," *The International Journal of Robotics Research*, vol. 26, no. 2, pp. 141-166, 2007.

[2] N. K. Yilmaz, C. Evangelinos, P. F. Lermusiaux, and N. M. Patrikalakis, "Path planning of autonomous underwater vehicles for adaptive sampling using mixed integer linear programming," *IEEE Journal of Oceanic Engineering*, vol. 33, no. 4, pp. 522-537, 2008.

[3] C. K. Kim, Y. M. Han, B. H. Bae, J. H. Kim, "The research of path planning algorithm considering vehicle's turning radius for unmanned ground vehicle," *International Conference on Control, Automation and Systems*, pp. 754-756, 2011.

[4] Y. Chen, G. Luo, Y. Mei, J. Yu, and X. Su, "UAV path planning using artificial potential field method updated by optimal control theory," *International Journal of Systems Science*, vol. 47, no. 6, pp. 1407-1420, 2016.

[5] L. Yang, J. Qi, D. Song, J. Xiao, J. Han, and Y. Xia, "Survey of robot 3D path planning algorithms," *Journal of Control Science and Engineering*, vol. 2016, no. 5, 2016.

[6] X. Liang, G. Meng, H. Luo, and X. Chen, "Dynamic path planning based on improved boundary value problem for unmanned aerial vehicle," *Cluster Computing*, vol. 19, no. 4, pp. 2087-2096, 2016.

[7] T. Lozano-Pérez, and M.A. Wesley, "An algorithm for planning collision-free paths among polyhedral obstacles," *Communications of the ACM*, vol. 22, no. 10, pp. 560-570, 1979.

[8] A. Stentz, "The focussed D* algorithm for real-time replanning," *International Joint Conferences on Artificial Intelligence Organization*," vol. 95, pp. 1652–1659, August 1995.

[9] S.M. LaValle, and J.J. Jr Kuffner, "Randomized kinodynamic planning," *The international journal of robotics research*, vol. 20, no. 5, pp.378–400, 2001.

[10] M. A. Mousavi, Z. Heshmati,and B. Moshiri,"LTV-MPC based path planning of an autonomous vehicle via convex optimization," *Iranian Conference on Electrical Engineering (ICEE)*, pp. 1-7, 2013.

[11] L. Zhang, Z. Zhou, and F. M. Zhang, "Mixed Integer Linear Programming for UAV Trajectory Planning Problem," *Applied Mechanics and Materials*, vol. 541, pp. 1473-1477, 2014.

[12] M. Hehn, and R. D'Andrea, "Quadrocopter trajectory generation and control," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 1485-1491, 2011.

[13] H. J. Kim, D. H. Shim, and S. Sastry, "Nonlinear model predictive tracking control for rotorcraft-based unmanned aerial vehicles," *American Control Conference*, vol. 5, pp. 3576-3581, 2002.

[14] X. Zhang, S. Huang, W. Liang, and T. H. Lee, "HLT*: Real-time and Any-angle Path Planning in 3D Environment," *45th Annual Conference of the IEEE Industrial Electronics Society (IES)*, Manuscript submitted for publication, 2019.

[15] A. Nash, S. Koenig , C. Tovey, "Lazy Theta*: Any-angle path planning and path length analysis in 3D," *Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010.

[16] A. Gálvez, A. Iglesias, A. Cobo, J. Puig-Pey, and J. Espinola, "Bézier curve and surface fitting of 3D point clouds through genetic algorithms, functional networks and least-squares approximation," *International Conference on Computational Science and Its Applications*, pp. 680-693, 2007.

[17] M. Elhoseny, A. Tharwat, A. E. Hassanien, "Bezier curve based path planning in a dynamic field using modified genetic algorithm," *Journal of Computational Science*, vol. 25, pp. 339-350, 2018.

[18] M. Radmanesh, M. Kumar, P. H. Guentert, and M. Sarim, "Overview of path-planning and obstacle avoidance algorithms for UAVs: a comparative study," *Unmanned Systems*, vol. 6, no. 2, pp. 95-118, 2018.