Journal of Intelligent & Robotic Systems (2019) 94:503–520
https://doi.org/10.1007/s10846-018-0813-9

Concurrent Optimal Trajectory Planning for Indoor Quadrotor Formation Switching

Yang Xu[1] · Shupeng Lai[2] · Jiaxin Li[3] · Delin Luo[1] · Yancheng You[1]

Received: 9 October 2017 / Accepted: 12 March 2018 / Published online: 14 April 2018

Abstract

This paper presents a novel concurrent optimal trajectory planning method for a team of quadrotors to switch between various formation patterns in the confined indoor environment. Using static shape-based, dynamic trajectory-based approaches, different formation patterns of quadrotors can be optimally and rapidly arranged and localized. The modified algorithm of optimal reciprocal collision avoidance (ORCA) is applied to produce the nominal collision-free trajectories for the quadrotor team as the first stage, which is extended to three-dimensional space and the downwash effect induced by the propellers is effectively addressed. Considering the flatness property of the quadrotor dynamics, the jerk-optimized trajectories based on the cubic clamped B-spline are then generated and these corresponding constraints are also satisfied. Moreover, a robust perfect tracking (RPT) outer-loop controller is designed to compensate the poor resolution of the indoor localization system. Finally, simulations are conducted to verify these proposed algorithms, and the real-world flight result was successfully showcased at the opening ceremony of Rotorcraft Asia & Unmanned Systems Asia 2017.

Keywords Multiple quadrotors · Collision avoidance · Path planning · Trajectory generation · Indoor environment

Mathematics Subject Classification (2010) 93C35 · 93C85 · 93C95

JEL Classification (2010) 93 Systems theory; control

Foundation items: National Natural Science Foundation of China (61673327), China Scholarship Council (201606310153) and Aviation Science Foundation of China (20160168001).

✉ Delin Luo
luodelin1204@xmu.edu.cn

Yang Xu
yangxu@u.nus.edu

Shupeng Lai
shupenglai@gmail.com

Jiaxin Li
jli@u.nus.edu

Yancheng You
yancheng.you@xmu.edu.cn

[1] School of Aerospace Engineering, Xiamen University, Xiamen 361005, People's Republic of China

[2] Department of Electrical and Computer Engineering, National University of Singapore, Singapore 117583, Singapore

[3] NUS Graduate School for Integrative Sciences and Engineering, National University of Singapore, Singapore 119077, Singapore

1 Introduction

Benefitting from the development of miniature, high-performance processors in recent decades, unmanned aerial vehicles (UAVs), especially quadrotors, have reached a new stage of becoming smaller, safer, faster and smarter [17]. Moreover, since successful navigation relies on complete interconnected loops of localization, mapping, planning and control, various methods proposed and implemented by researchers are pushing quadrotors to fulfil complex tasks, such as for the multiple vehicle application scenarios in most cases, including package delivery [1], disaster rescue [2], warehouse management [12], and for the entertainment [3]. Due to the requirements of cooperative tasks, it is critical to prevent the inter-quadrotor collision during the switching process of different formation patterns. Thus the focal point of this paper is to make multiple quadrotors track smoothed trajectories robustly in a confined indoor space, which are generated by algorithms satisfying dynamic constraints and collision avoidance.

Multiple quadrotors have unstable and under-actuated dynamic proprieties intrinsically, thus arbitrarily generating collision-free trajectories is often unachievable. In [18], Kuriki proposed an artificial potential field approach that kept collision-free for the linearized quadrotor model, which was built on small-perturbation assumption and near the hovering mode. In addition, collision avoidance questions can be formulated as constrained optimal control problems. In [14], Hehn declared that nonlinear dynamics were suitable for the time-optimal maneuvering trajectories but brought about expensive computation. Just as a remedy, the flatness system theory was imported to generate trajectory for quadrotors, Mellinger in [20] proposed a snap-optimized collision-free trajectory generation method for heterogeneous quadrotors with the help of flat output properties. Rosales developed a new strategy in [23] based on the formation control, which applies the null space of a Jacobian matrix to achieve the different control objectives in a non-conflicting way.

Contrary to the cited approaches, the method proposed in this paper separately generates trajectories of the quadcopters in sequential stages. The nominal trajectories are collision-free, dynamically constrained and smoothed, thereby allowing the robust tracking controller in each quadcopter to accurately track these trajectories. To achieve this objective, necessary algorithms aforementioned need to be modified and integrated. From the geometric perceptive, the terminology of velocity obstacle (VO) was firstly proposed in [10] to define all possible velocities that lead to colliding with observed velocities of dynamic obstacles. Then Berg improved optimal reciprocal collision avoidance (ORCA) in [25], which is a decentralized, real-time algorithm for multi-agent conditions, to assign each agent a safe velocity based on the observed velocities of its neighbors to guarantee collision-free in a known time horizon. However, as ORCA is designed for real-time and does not contain high-order dynamics, it is often unsuitable for real-world implementation. The B-spline interpolation has an advantage of crossing every waypoint of the nominal trajectory exactly, and it uses the control points to fit the given shape and can acquire local refinement. Here, it is necessary to guarantee the initial and terminal boundary conditions of the trajectory, which are usually set to be hovering with zero velocity and acceleration of the quadrotor. The tradeoff between accurate path presentation and smooth curvature within constrains of the B-spline is a necessary issue to be considered. Finally, a robustly stable outer-loop controller is crucial for multiple quadrotors to track reference trajectories by rejecting disturbances from other quadrotors and prevent collisions. Due to the existence of actuator constraints, the maneuver capabilities of quadrotors should be restricted.

The main contributions of this paper are fourfold: 1) Static and dynamic formation patterns are created by optimization methods for goal arrangement; 2) ORCA algorithm is modified that provably ensures the collision avoidance of differential flatness based teams of quadrotors in three dimensional space; 3) a cubic clamped B-spline interpolation method is developed to smooth the nominal trajectory with high-order dynamic constraints and local refinement; 4) a robust tracking controller for each member of the quadrotor team is designed based on the flatness properties, the actuator constraints are converted into the maneuver constrains of quadrotors. It needs to mention that the static obstacle avoidance problem is not considered in this paper.

The remainder of the paper is structured as follows: Section 2 gives an overview of proposed method; Section 3 proposes two goal arrangement approaches for the generation of formation patterns; Section 4 modifies conventional ORCA algorithm to create nominal collision-free trajectories for multiple quadrotors; Section 5 describes the smoothing means to generate feasible dynamic-constrained trajectories; The tracking controller is designed considering measurement uncertainty and actuator constraints in Section 6; Section 7 analyzes the proposed algorithms and demonstrates corresponding simulations; Section 8 shows experimental results of actual quadrotor team flight; Section 9 concludes the paper and summarises the future directions for this work.

Notation Throughout this paper, x represents scalars, $\mathbf{x}$ vectors, $\mathbf{X}$ matrices and $\mathcal{X}$ sets. $\oplus$ denotes the Minkowski sum of two sets, and $\|\cdot\|$ is the Euclidean norm of a vector, 1D, 2D and 3D are respectively short for one-dimensional, two-dimensional and three-dimensional. The quadrotor team members mentioned are homogeneous, for brevity, we assume the same number of quadrotors and goals, thus defining the set of quadrotor index as $\mathcal{S}_p = \{1, 2, \cdots, n\}$, and the set of goal index as $\mathcal{S}_g = \{1, 2, \cdots, n\}$.

2 Method Overview

The approach proposed in this paper pays attention to the switching problem of various formation patterns in 2D or 3D with a team of quadrotors in an indoor environment with confined space, and a whole structure of the method is summarized in Fig. 1.

Given a team of quadrotors with the same radius and height, then adding the corresponding static and dynamic formation patterns, this method can be divided into the following steps. First, the goal positions of all these quadrotors

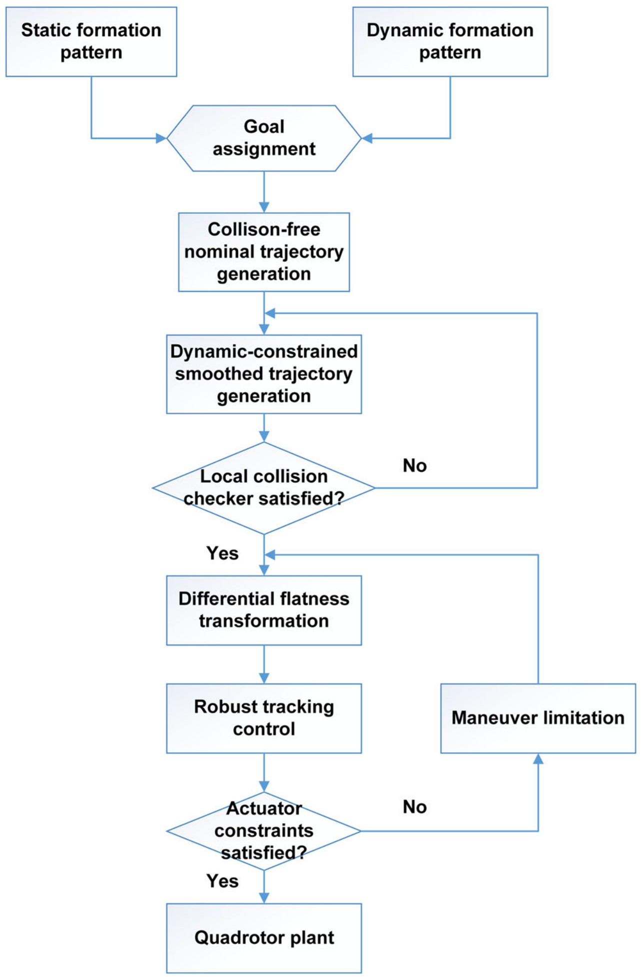

Static formation pattern
Dynamic formation pattern
Goal assignment
Collison-free nominal trajectory generation
Dynamic-constrained smoothed trajectory generation
Local collision checker satisfied?
No
Yes
Differential flatness transformation
Robust tracking control
Maneuver limitation
Actuator constraints satisfied?
No
Yes
Quadrotor plant

Fig. 1 Flowchart of formation switching strategy for a team of quadrotors

are computed and optimized to formulate certain preset formation patterns, which can be switched by the goal assignment method of the Hungarian Algorithm [22] or just the labelled point-to-point assignment. Second, a two-stage trajectory generation approach is used, in which an inter-quadrotor collision avoidance algorithm is designed firstly during the process of formation switching, and then the generated nominal trajectories are smoothed subject to the dynamic constraints. Successively, these smoothed safe trajectories are transformed into quadrotor states and controls using the differential flatness property. Finally, the outer-loop tracking controller is designed to enable synchronous trajectory following and allow for real-time adaptation to disturbances.

It is essential to mention the function of two checkers. Regarding the generation of smooth trajectories, a local collision checking is an indispensable component. If colliding condition is activated, minor refinement is needed to tune this piece of trajectory. After tracking controller has been designed, the full states and control inputs of the quadrotor are sent into the checker to guarantee that actuator limits are not disturbed. Otherwise, the limitation of maneuver is activated to rectify the reference trajectory. This process can be repeated until the appropriate flight trajectories obtain both safety and actuator constraints.

3 Formation Pattern Generation

3.1 Static Pattern Generation: Shape-based Approach

In 2D plane, the contours of various shapes can always be simplified into a polygon. If the number of quadrotors is fixed, the goal arrangement problem can be considered as uniformly distributing these quadrotors inside a polygon region. Decomposition of 2D polygons can be done using Delaunay triangulation [21] or Voronoi tessellation [7]. As the former method always results in silver triangles, the latter one would be a better choice. Therefore, a Voronoi tessellation method using iterative Lloyd's algorithm is modified and used to optimize the goal locations. Take the convex regular octagon and concave regular pentagram for example, the number of six goals are uniformly spread in these two polygons be projected into unit grid area $\mathcal{U} = [0, 1] \times [0, 1]$ as shown in Fig. 2a and b.

3.2 Dynamic Pattern Generation: Trajectory-Based Approach

The trajectory-based approach gives a series of specific rotational motions to every member of the quadrotor team. For real-world implementation, the trajectories used by guidance modules should contain all acceleration, velocity and position references. By generating the motion elements with rotation, we modify this reference from translation to rotation according to the relation between translational and rotational movements. The trajectories are then generated based on rotational coordination, including references of angular acceleration, angular velocity and angle. Then, this problem can be formulated as a triple integrator Two-Point Boundary Value Problem (TPBVP), which has a typical format similar to that in [26]. We use the point mass model to represent a quadrotor with initial static state constraints, including the angular velocity limit 0.4rad/s, the angular acceleration limit 1rad/s^2 and the angular jerk limit 1rad/s^3. The generated 240° angle-related TPBVP references and part of figure-eight trajectory of a single quadrotor are shown in Fig. 3a and b respectively.

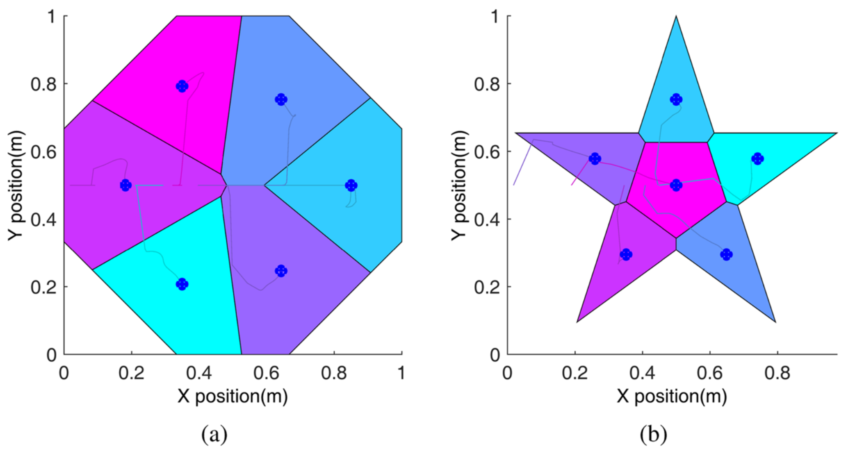


Fig. 2 Visualization and optimization of six goals distributing in an octagon and a pentagram

4 Collision Avoidance Algorithm

4.1 Downwash Effect

From the geometric perspective, the convex hull of the start and goal locations with Minkowski sum can form a closed area, in which the collision-free trajectories are needed. Without reassigning goals or changing the above planned straight-line trajectories, we can use the algorithm of Concurrent Assignment and Planning of Trajectories (CAPT) in [24]. This algorithm works well in 2D plane, we then extend it to 3D as shown in Fig. 4a. The generated straight-line trajectories can be seen as tunnels, therefore CAPT is also valid.

We can easily see that in 3D as shown in Fig. 4a, the trajectories generated by the CAPT algorithm are straight lines. However, the projecting lines of these trajectories on the horizontal plane can not guarantee collision free in the vertical direction as shown in Fig. 4b, indicating that one quadrotor would fly directly below an other one at the same time on the horizon. In reality, the quadrotor has strong downwash effect generated by its four propellers in the vertical direction. The solution in [20] shows that the minimum separation in the altitude must be at least

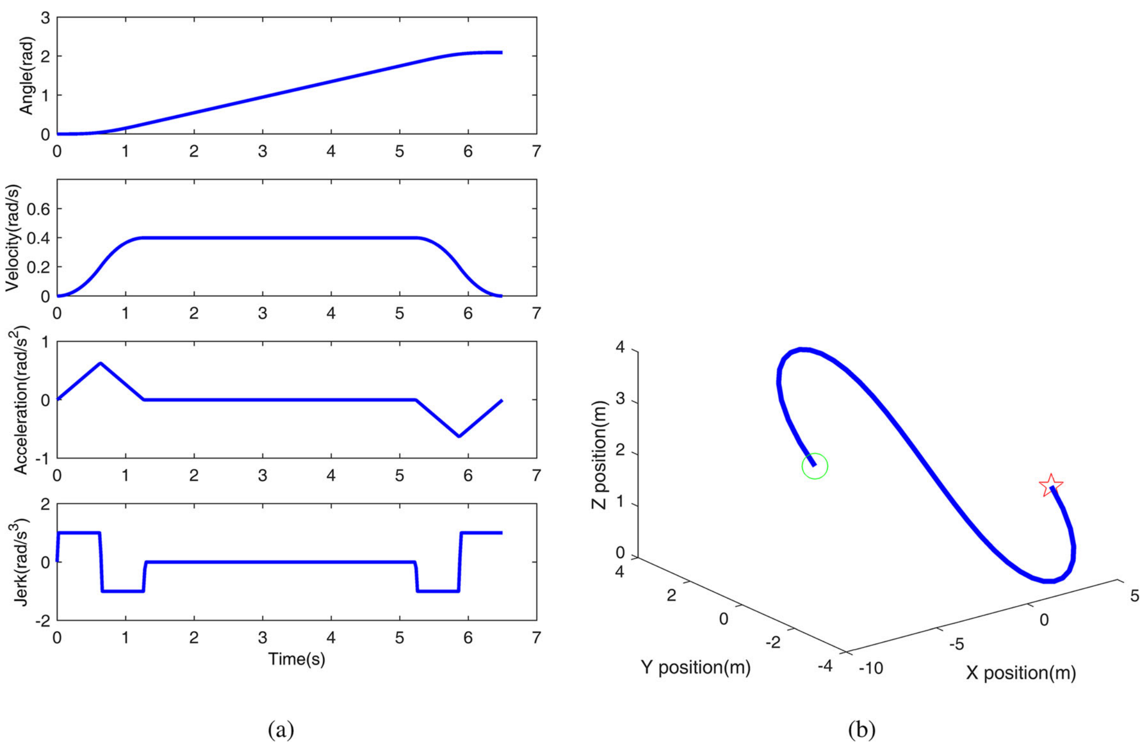


Fig. 3 240° angle-related TPBVP references and trajectory generated of partial figure-eight with a single quadrotor

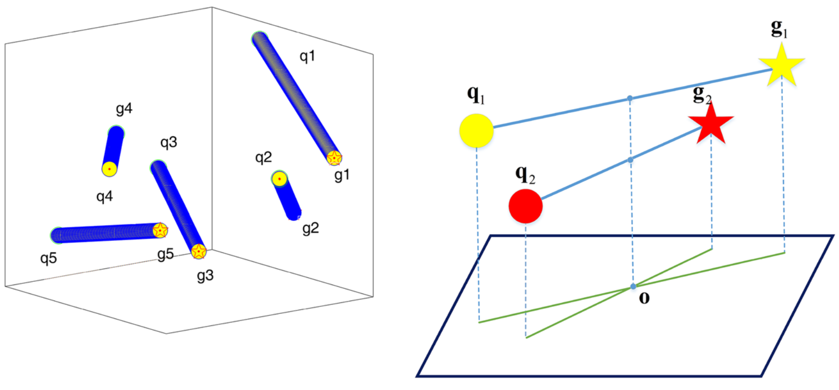


Fig. 4 Straight-line trajectories of five quadrotors generated by 3D CAPT algorithm and downwash condition occurring with two quadrotors in the vertical direction

$8r$, where r is the radius of the quadrotor. In the confined height space such as the indoor environment, it is not always possible to meet this requirement.

4.2 Velocity Obstacle Computation

To account for the downwash, we can model a single quadrotor i as an omnidirectional cylinder $\mathcal{C}(\mathbf{p}_i, r_i, h_i)$ with a center $\mathbf{p}_i$, radius r_i, and cylindrical half height of h_i in 3D space. In addition, the pitch and roll movements are not considered at the beginning.

Modelling as a cylinder in 3D space, we take a pair of quadrotors, say i and j, for example as shown in Fig. 5a. Assuming this pair of quadrotors are localized in positions $\mathbf{p}_i$ and $\mathbf{p}_j$, and simultaneously moving with velocity $\mathbf{v}_i$ and $\mathbf{v}_j$ respectively, the velocity obstacle of quadrotor i induced by quadrotor j within time frame $t \in [0, \tau]$ can be defined as $\mathcal{VO}^{\tau}_{i|j}$ in the velocity space, which is a truncated cone within shadow in Fig. 5b. Therefore, for every pair of neighboring quadrotors i and j, the reciprocal collision avoidance condition of 3D space within $t \in [0, \tau]$ can be defined as

$$\mathbf{v}_i \notin \mathcal{VO}^{\tau}_{i|j} = \bigcup_{t=0}^{\tau} \left(\left(\mathcal{C}\left(\mathbf{p}_j, r_j, h_j\right) \oplus \mathcal{C}\left(-\mathbf{p}_i, r_i, h_i\right)\right) / t \right) + \mathbf{v}_j \tag{1}$$

For the purpose of analysis, this $\mathcal{VO}^{\tau}_{i|j}$ can be projected to horizontal and vertical planes separately. In this paper, the superscript H and V denote the projections of velocities or positions in the horizontal and vertical plane respectively. Thus the constraints can also be projected into two planes, the horizontal and vertical projecting constraints of avoiding $\mathcal{VO}^{\tau}_{i|j}$ within $t \in [0, \tau]$ are

$$\begin{cases} \left\| \mathbf{p}^H_j - \mathbf{p}^H_i + \left(\mathbf{v}^H_j - \mathbf{v}^H_i\right) t \right\| \geq r_i + r_j \\ \left\| \mathbf{p}^V_j - \mathbf{p}^V_i + \left(\mathbf{v}^V_j - \mathbf{v}^V_i\right) t \right\| \geq h_i + h_j \end{cases} \tag{2}$$

From Fig. 5b above, it is easy to see that $\mathcal{VO}^{\tau}_{i|j}$ in 3D space contains five linear constraints, the collision avoidance from left, right, top, bottom and heading directions respectively. As safety is a primary concern in an indoor environment, adding constraints is necessary to ensure the reliability at the cost of performance. Therefore, the velocity obstacle from top and bottom directions should be enlarged to remove downwash effect.

Fortunately, the downwash effect can be eliminated by constructing the $\mathcal{ORCA}^{\tau}_{i|j}$. The projection of which in the horizontal plane is a shaded half plane as shown in Fig. 6. Vector $\mathbf{w}_j$ is the minimum vector added in $\mathbf{v}_i$ to stretch out the $\mathcal{VO}^{\tau}_{i|j}$ with the direction of a normal vector $\mathbf{n}_j$, and the perpendicular half-plane to $\mathbf{w}_j$ at $\mathbf{v}_i + \alpha \mathbf{w}_j$ represents a set

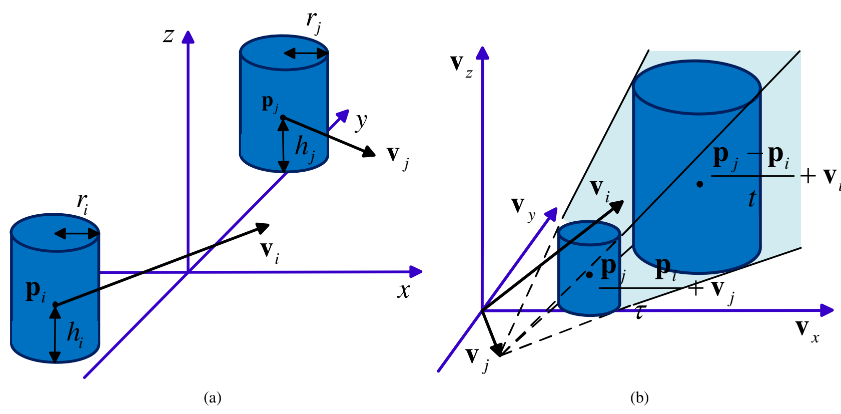


Fig. 5 Cylinder modelling and velocity obstacle of two quadrotors in 3D space

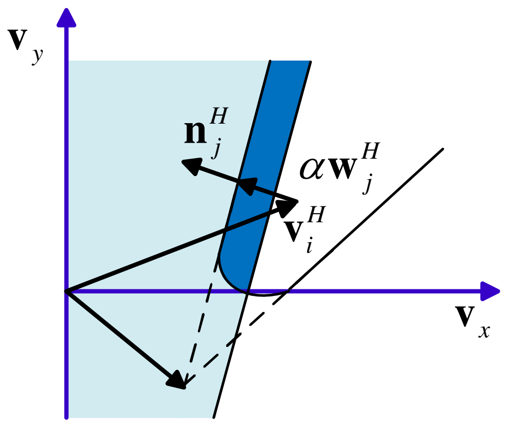


Fig. 6 ORCA region of two quadrotors in horizontal plane of projection

of velocities closest to $\mathbf{v}_i$ and can hence avoid reciprocal collision. Since the ORCA half-plane is perpendicular to the horizontal plane when not considering pitch and roll movements, then it is easy to find $\mathbf{w}_j = \mathbf{w}_j^H$.

$$\mathbf{w}_j = \underset{\mathbf{v}\in\mathcal{VO}_{i|j}^{\tau}}{\arg\min} \left\| \mathbf{v} - \left(\mathbf{v}_i - \mathbf{v}_j\right) \right\| - \left(\mathbf{v}_i - \mathbf{v}_j\right) \tag{3}$$

If the multiple α is selected to be larger than 1, $\mathcal{ORCA}_{i|j}^{\tau}$ can completely contain $\mathcal{VO}_{i|j}^{\tau}$. α can be set as 0.5 so as to guarantee that quadrotors i and j take half of the responsibility for collision avoidance, respectively, according to the proof in [25]. Thus the ORCA space may slightly penetrate into the VO as the darker shaded region shown in Fig. 6.

$$\mathcal{ORCA}_{i|j}^{\tau} = \left\{\mathbf{v} | \left(\mathbf{v} - \left(\mathbf{v}_i + \alpha\mathbf{w}_j\right)\right) \cdot \mathbf{n}_j \geq 0, \alpha = 0.5\right\} \tag{4}$$

After adding another $n-1$ quadrotors in the set $\mathcal{S}_p$, this intersected half-planes $\mathcal{ORCA}_i^{\tau}$ may eventually become a polygonal column, which is perpendicular to the horizontal plane in 3D space. We then introduce the maximum speed constraint of the quadrotor i, and the ORCA space becomes

$$\mathcal{ORCA}_i^{\tau} = \left(\bigcap_{i\neq j\in\mathcal{S}_p}^{n} \mathcal{ORCA}_{i|j}^{\tau} \right) \bigcap \left(\max_{i\in\mathcal{S}_p} \mathcal{E}\left(\mathbf{0}, v_i^H, v_i^V\right) \right) \tag{5}$$

Assuming the maximum vertically climbing and descending speeds of the quadrotors are the same and the maximum speed in the horizon is larger than the vertical one, $\mathcal{E}\left(\mathbf{0}, v_i^H, v_i^V\right)$ is an ellipsoid centred at the position $\mathbf{0}$, with horizontal circle radius v_i^H and vertical elliptic radius v_i^V.

On the basis of the above modelling, the assigned goal location for a quadrotor i with current position $\mathbf{p}_i$ and velocity $\mathbf{v}_i$, the preferred velocity $\tilde{\mathbf{v}}_i$ that is pointed to the goal. The optimal collision-free command velocity $\mathbf{v}_i^*$ in the following time τ should be as close as $\tilde{\mathbf{v}}_i$, and confined with $\mathcal{ORCA}_i^{\tau}$ as

$$\mathbf{v}_i^* = \underset{\mathbf{v}\in\mathcal{ORCA}_i^{\tau}}{\arg\min} \left\| \mathbf{v} - \tilde{\mathbf{v}}_i \right\| \tag{6}$$

This is a quadratic optimization function formulated in 3D space, and the constraints brought by $\mathcal{ORCA}_i^{\tau}$ are all in the horizontal plane, while the vertical direction only contains maximum speed constraints. For the purpose of simplifying this problem, we only consider the vertical constraints separately, and project $\mathcal{ORCA}_i^{\tau}$ onto the horizontal plane. The horizontally projected quadratic optimization can be effectively tackled by a low dimensional linear program as described in [25] with complexity $\mathcal{O}(n)$, regardless of whether the maximum horizontal speed constraint $v_{\max,i}^H$ is included. After tackling the horizontal part, the vertical speed v_i^V can be computed and selected within the constraint of maximum speed $v_{max,i}$ in 3D space.

4.3 Optimal Reciprocal Collision Avoidance Algorithm Modification

Based on the above analysis, the ORCA algorithm is not suitable for real-world application, as it requires the assumption of symmetry and reciprocity. For instance, relative position and velocity observed by a pair of quadrotors should be the same, and the responsibility taken by this pair must be set as half to avoid collision. In practice, the uncertainty of sensors affects this assumption. Thus we use the ORCA algorithm as the first stage to generate a collision-free trajectory for every quadrotor in the simulation environment. The ORCA trajectory generation algorithm of quadrotor i is shown below. In more detail, for each simulation time-step Δt, we assume every single quadrotor knows the current positions and velocities of all the other ones, thus these relative variables also can be calculated. In addition, because the quadrotor is modelled as a cylinder with virtual radius, in order to avoid the impact caused by its pitch and roll movement, the radius of the quadrotor should be extended with a safety threshold for the navigation along a pre-designed trajectory in a real world environment.

In order to meet the performance requirements, within a length of trajectory clamped by the start and goal locations $\mathbf{p}_i$ and $\mathbf{g}_i$, we define a velocity $\bar{v}_i^H$ of the quadrotor i, which denotes the predefined horizontal velocity limit and is smaller than $v_{i,\max}^H$ after projecting into the horizontal plane. In addition, due to the symmetry of the quadrotors, a small random vector $\mathbf{d}_i^H$ needs to be added to $\tilde{\mathbf{v}}_i$ so as to avoid deadlock. Finally, the horizontal arrival time of quadrotor i is t_i^H with the terminal condition of minimum distance threshold $\varepsilon_{g,p}$, and the generated trajectory containing a series of waypoints p_i^H.

Algorithm 1 Algorithm for ORCA Trajectory Generation of Quadrotor i

```
Input: Horizontal start and goal locations $\mathbf{p}_i^H$, $\mathbf{g}_i^H$, horizontal velocity $\bar{v}_i^H$, minimal distance threshold $\varepsilon_{g,p}$
Output: Horizontal arrival time $t_i^H$, horizontal locations of waypoints on the generated trajectory $p_i^H$
  $count_i = 0$
  $p_i^H[0] \leftarrow \mathbf{p}_i^H(x, y)$
  while $\|\mathbf{g}_i^H - \mathbf{p}_i^H\| \geq \varepsilon_{g,p}$ do
      if $\|\mathbf{g}_i^H - \mathbf{p}_i^H\| \geq \bar{v}_i^H$ then
          $\tilde{\mathbf{v}}_i^H = \bar{v}_i^H \cdot (\mathbf{g}_i^H - \mathbf{p}_i^H) / \|\mathbf{g}_i^H - \mathbf{p}_i^H\|$
      else
          $\tilde{\mathbf{v}}_i^H = \mathbf{g}_i^H - \mathbf{p}_i^H$
      end if
      $\tilde{\mathbf{v}}_i^H = \tilde{\mathbf{v}}_i^H + \mathbf{d}_i^H$
      Compute $\mathbf{v}_i^{*H}$
      $\mathbf{p}_i^H = \mathbf{p}_i^H + \mathbf{v}_i^{*H} \cdot \Delta t$
      $count_i{+}{+}$
      $p_i^H[count_i] \leftarrow \mathbf{p}_i^H(x, y)$
  end while
  $t_i^H = count_i \cdot \Delta t$
  return $t_i^H$, $p_i^H$
```

Algorithm 1 generates the single trajectory for horizontal plane. The vertical direction will also need to produce the corresponding series of positions accordingly. In an indoor environment, the height size is much shorter than the lengths of the horizontal area. Meanwhile, as the quadrotor has the ability to fly faster in the horizontal plane, the start and goal locations in the vertical direction can be placed relatively close. Therefore, we generally place these locations under the condition of

$$\frac{\|\mathbf{g}_i^H - \mathbf{p}_i^H\|}{\|\mathbf{g}_i^V - \mathbf{p}_i^V\|} \geq \frac{\bar{v}_i^H}{\sqrt{(v_{\max,i})^2 - (\bar{v}_i^H)^2}} > 1 \tag{7}$$

We can then select the vertical arrival time t_i^V of the quadrotor i as

$$t_i^V = t_i^H \geq \frac{\|\mathbf{g}_i^V - \mathbf{p}_i^V\|}{\sqrt{(v_{\max,i})^2 - (\bar{v}_i^H)^2}} \tag{8}$$

It can be easily found that the relation between t_i^V and t_i^H is not strict. If they are equal, then the quadrotor i arrives at the terminal goal synchronously from the horizontal and vertical directions. The final goal arrival time in 3D space is t_i^H. Therefore, the vertical trajectory of quadrotor i can be generated by Algorithm 2.

By combing Algorithm 1 with 2 can produce the final trajectory of the quadrotor i in 3D space. Additionally, the Algorithm 2 generates a uniform velocity in the vertical direction, for example, if we need a quadrotor to move along a sine wave trajectory, Line 5 then can be modified to be a sine function.

Algorithm 2 Algorithm for Vertical Trajectory Generation of Quadrotor i

```
Input: Vertical start and goal locations $\mathbf{p}_i^V$, $\mathbf{g}_i^V$, vertical arrival time $t_i^V$
Output: Vertical locations of waypoints on the generated trajectory $p_i^V$
  $count_i = 0$
  $total_i = \text{round}\left(t_i^V / \Delta t\right)$
  $p_i^V[0] \leftarrow \mathbf{p}_i^V(z)$
  for $k = 1 : total_i$ do
      $\mathbf{p}_i^V = \mathbf{p}_i^V + k \cdot (\mathbf{g}_i^V - \mathbf{p}_i^V) / total_i$
      $count_i{+}{+}$
      $p_i^V[count_i] \leftarrow \mathbf{p}_i^V(z)$
  end for
  return $p_i^V$
```

Algorithm 1 and 2 can generate the trajectories for each quadrotor in parallel. However, the synchronizing arrival problem of all quadrotors in the set of $\mathcal{S}_p$ can not be guaranteed, as the time lengths in the generated set $\{t_i^H, i \in \mathcal{S}_p\}$ for each quadrotor are not always the same. Before switching to the next movement, the predesigned movement in current process must be achieved. Therefore, Algorithm 3 is used to compensate the time-lag so as to make all quadrotors in the team reach their goals simultaneously, and this will inevitably delay those quadrotors that could take shorter travelling times to their goal locations.

Algorithm 3 Algorithm for Time Compensation of Quadrotor Team

```
Input: Horizontal and vertical start locations of quadrotor $i$ $\mathbf{p}_i^H$, $\mathbf{p}_i^V$, horizontal arrival time of quadortor $i$ $t_i^H$, a team of quadrotors $i \in \mathcal{S}_p$
Output: Horizontal and vertical locations of waypoints on the generated trajectory $p_i^H$, $p_i^V$
  $count_i = 0$
  $t^H = \max\{t_i^H, i \in \mathcal{S}_p\}$
  $p_i^H[t_i^H] \leftarrow \mathbf{p}_i^H(x, y)$
  $p_i^V[t_i^H] \leftarrow \mathbf{p}_i^V(z)$
  if $t_i^H \leq t^H$ then
      $count_i = \text{round}\left(\left(t^H - t_i^H\right) / \Delta t\right)$
  end if
  for $k = 1 : count_i$ do
      $p_i^H[count_i] = p_i^H[t_i^H]$
      $p_i^V[count_i] = p_i^V[t_i^H]$
  end for
  return $p_i^H$, $p_i^V$
```

From above Algorithm 3, we know the whole quadrotor team arrival time is t^H, and all of these generated trajectories have the same time length. However, through the above analysis, we find that t_i^V may vary due to the different lengths of t_i^H under the condition of Eq. 8. For the purpose of simplification, we can render the whole team's vertical arrival times $\{t_i^V, i \in \mathcal{S}_p\}$ to be the same, which is set as t^V, and let $t_i^V = t^V = t^H$. Then, we modify Line 5 of Algorithm 2 from t_i^V to t^H and remove the vertical part of Algorithm 3. Once we select a quadrotor team, these physical parameters and flight abilities of each one are all the same. Fortunately, the arrival time of the whole team can be successfully synchronized as t^H.

We redesign the trajectory generation into a non-real time one so as to allow manipulation of the time-step Δt of simulation according to the continuity and smoothness of generated trajectories. For the purpose of analysis, we define the generated trajectory by Algorithm 1, modified Algorithm 2 and 3 of each quadrotor $i \in \mathcal{S}_p$ as a nominal one

$$\gamma_{\text{norm},i}(t) = (\mathcal{T}, \mathcal{P}_i) \tag{9}$$

where $t \in \mathcal{T}$, $\mathcal{T} = \{t_0^H, t_1^H, \cdots, t_f^H\}$ is a set of generated time list, and $\mathcal{P}_i = \{(p_{t_0,i}^H, p_{t_0,i}^V), (p_{t_1,i}^H, p_{t_1,i}^V), \cdots, (p_{t_f,i}^H, p_{t_f,i}^V)\}$ is a set of corresponding generated 3D waypoints.

5 Jerk-Optimized B-Spline Trajectory Gneration

5.1 Differential Flatness of Quadrotor Dynamics

During the previous procedure, the nominal trajectories $\gamma_{\text{norm},i}(t), i \in \mathcal{S}_p$ are obtained by ORCA corresponding algorithms. These generated $\gamma_{\text{norm},i}(t)$ may contain sharp turns with large accelerations, which would be impossible for the quadrotors. Taking the higher-order dynamics of the quadrotor into account means the smoother of the generated trajectory. Here, balancing the dynamic feasibility and smoothness is principal, thus the analysis of quadrotor dynamics is the prerequisite.

As shown in [19], the quadrotor can be seen as a differential flat system with flat outputs $\sigma = [\mathbf{p}, \psi]^T$, where $\mathbf{p}$ is the position vector of the center of mass, and ψ denotes the heading angle. The states and inputs of the system can be written in terms of algebraic functions of appropriately chosen flat outputs and their derivatives.

By using the differential flatness property, trajectories can be generated by leveraging the nonlinear dynamics of the quadrotor, rather than simply viewing the system dynamics as constraints. A quadrotor has the ability to behave as an omnidirectional agent, thus for the purpose of simplifying the trajectory generation, ψ is set to be a non-alternating value for the entire duration. Any sufficiently smooth trajectory in the flat output space can then be converted analytically back into a feasible state trajectory and corresponding control input using the endogenous transformation.

5.2 Clamped Normalized B-Spline

The formerly generated $\gamma_{\text{norm},i}(t), i \in \mathcal{S}_p$ has equal time intervals, thus a cubic Clamped Normalized Uniform B-Spline (CNUBS) is designed to regenerate the practical trajectories of the whole quadrotor team.

This kind of cubic B-spline has the knot vector $[0, 0, 0, 0, 1, 2, 3, \cdots, m-1, m, m, m, m]$, where the first and last three elements represent the start and goal boundary conditions. We denote the number of specific control points as m_c, and set $m = m_c - 1$ so as to make the knot vector have $m_c + 6$ elements. Therefore, the whole cubic B-spline of the separated axis $\{x, y, z\}$ can be expressed as

$$S_3(s) = \sum_{k=-2}^{0} c_k N_{k+2,3}(s) + \sum_{k=1}^{m_c} c_k B_3(s-k+1) + \sum_{k=m_c+1}^{m_c+3} c_k N_{k+2,3}(s) \tag{10}$$

where c_k is a projected scalar of 3D control points in a single axis, and s is the path parameter spanning through $[0, 1, 2, \cdots, m-1, m]$. As defined in [15], the middle part of $B_3(*)$ is the common basis element of the B-spline, and the first and last three special bases of $N_{k,3}(*), k \in \{0, 1, 2, m_c+3, m_c+4, m_c+5\}$ are boundary conditions with mutually reversed versions. In addition, this B-spline is a third-order one and thus the subscript of all these bases is 3.

Furthermore, we rearrange the Eq. 10 into a compact form as

$$S_3(s) = \sum_{k=0}^{m_c+5} \bar{c}_k \Gamma_{k,3}(s) \tag{11}$$

where $\bar{c}_k = c_{k-2}$ is a shifted expression, $\Gamma_{k,3}(*) = \{B_3(*), N_{k,3}(*)\}$.

For a dynamic system like the quadrotor, we need to substitute the time t for the path parameter s, because of the relationship between control points and knot vector according to [6]. Thus there is a linear relationship between two variables s and t

$$\frac{s}{t} = \frac{m_c + 6}{t_w} = \beta \tag{12}$$

where t_w is the whole time of the trajectory, and β is a scale factor.

To formulate an optimization problem of the minimum jerk trajectory in 3D space, as defined above, we can simplify it into three independent problems of dealing with the optimization variable c_k separately on the axes $\{x, y, z\}$. Thus the following optimization is built on the single axis presupposition.

5.3 Minimum Jerk Optimization

The trajectories $\gamma_{norm}(t)$ generated by the ORCA algorithm can be seen as a given data set of time-dependent shapes. We represent time set $\mathcal{T}$ and position set $\mathcal{P}$ as two vectors in 1D condition

$$\begin{cases} \mathbf{t}_d = [\mathcal{T}]^T = [t_1, t_2, \cdots, t_l]^T \\ \mathbf{p}_d = [\mathcal{P}]^T = [p_1, p_2, \cdots, p_l]^T \end{cases} \tag{13}$$

where l is an integer and denotes the same length of the both vectors.

Furthermore, the cost function of achieving minimum jerk is formulated as

$$J = w \int_{-\infty}^{\infty} j_d^2(t)dt + \sum_{k=1}^{l} (S_3(\beta t_k) - p_k)^2 \tag{14}$$

where w is a positive weight coefficient, and $j_d(t)$ is the jerk to be minimized, which can be denoted as the third derivative of Eq. 11

$$j_d(t) = \sum_{k=0}^{m_c+5} \frac{d^3}{dt^3} \bar{c}_k \Gamma_{k,3}(s) \tag{15}$$

Although it can be proved based on [16] that the minimum jerk trajectory has a closed solution, the real-world application still needs a large amount of computation. In order to formulate a convex quadratic programming (QP) optimization expression that can be solved by optimization solvers like *quadprog* of MATLAB or *CPLEX* of IBM, we need to rearrange (14) with the vector type $\bar{\mathbf{c}} = [\bar{c}_0, \bar{c}_1, \cdots, \bar{c}_{m_c+5}]^T$ into

$$J_{min} = \min_{\bar{\mathbf{c}}} \left\{ \bar{\mathbf{c}}^T (w\mathbf{G} + \mathbf{H}^T\mathbf{H})\bar{\mathbf{c}} - 2\mathbf{p}_d^T \mathbf{H}\bar{\mathbf{c}} + \mathbf{p}_d^T \mathbf{p}_d \right\} \tag{16}$$

It is easy to find that $w\mathbf{G} + \mathbf{H}^T\mathbf{H}$ is always semi-positive definite (SPD), where

$$\mathbf{H} = \begin{bmatrix} \Gamma_{0,3}(\beta t_1) & \Gamma_{1,3}(\beta t_1) & \cdots & \Gamma_{m_c+5,3}(\beta t_1) \\ \Gamma_{0,3}(\beta t_2) & \Gamma_{1,3}(\beta t_2) & \cdots & \Gamma_{m_c+5,3}(\beta t_2) \\ \vdots & \vdots & \ddots & \vdots \\ \Gamma_{0,3}(\beta t_l) & \Gamma_{1,3}(\beta t_l) & \cdots & \Gamma_{m_c+5,3}(\beta t_l) \end{bmatrix},$$

$$\mathbf{G} = [g_{i,j}]_{i,j\in\{0,1,2,\cdots,m_c+5\}} = \left[\beta^5 \int_{-\infty}^{\infty} \frac{d^3\Gamma_{i,3}(s)}{ds^3} \frac{d^3\Gamma_{j,3}(s)}{ds^3} ds \right].$$

When $j_d(t)$ is not 0, it becomes positive definite (PD), thus a unique minimum of this above QP problem can be guaranteed.

Then, the first and second derivative constraints of the B-spline trajectory, which respectively represent velocity and acceleration, need to be taken into account as

$$\begin{cases} \min_t \frac{dS_3(s)}{dt} \le \mathbf{K}_1(i)\bar{\mathbf{c}} \le \max_t \frac{dS_3(s)}{dt} \\ \min_t \frac{d^2S_3(s)}{dt^2} \le \mathbf{K}_2(j)\bar{\mathbf{c}} \le \max_t \frac{d^2S_3(s)}{dt^2} \end{cases} \tag{17}$$

where $\mathbf{K}_1$ and $\mathbf{K}_2$ are both $(m_c + 6) \times (m_c + 6)$ matrices

$$\mathbf{K}_1 = \beta \begin{bmatrix} -3 & 3 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & -\frac{3}{2} & \frac{3}{2} & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 & 0 & \cdots & 0 \\ \vdots & & & & \ddots & & & & \vdots \\ 0 & 0 & \cdots & 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & -\frac{3}{2} & \frac{3}{2} & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & -3 & 3 \end{bmatrix},$$

$$\mathbf{K}_2 = \beta^2 \begin{bmatrix} 6 & -9 & 3 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & \frac{3}{2} & -\frac{5}{2} & 1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 1 & -2 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & 1 & -2 & 1 & 0 & \cdots & 0 \\ \vdots & & & & \ddots & & & & \vdots \\ 0 & 0 & \cdots & 0 & 1 & -2 & 1 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & \frac{3}{2} & -\frac{5}{2} & 1 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 6 & -9 & 3 \end{bmatrix}.$$

Here, $\mathbf{K}_1(i), i \in \{0, 1, 2, \cdots, m_c + 4\}$ denotes the ith row of $\mathbf{K}_1$, and $\mathbf{K}_2(j), j \in \{0, 1, 2, \cdots, m_c + 3\}$ is the jth row of $\mathbf{K}_2$.

Moreover, the boundary conditions of the trajectory are considered, which are the first and last three fixed elements of $\bar{\mathbf{c}}$. Thus a linear equality constraint can be formulated as

$$\mathbf{A}_e\bar{\mathbf{c}} = \mathbf{b}_e \tag{18}$$

where $\mathbf{A}_e$ is a $(m_c + 6) \times (m_c + 6)$ matrix

$$\mathbf{A}_e = \begin{bmatrix} I_3 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & I_3 \end{bmatrix}.$$

Finally, combining (16), (17) and (18), a convex QP problem is formulated and can be efficiently solved by numerical software toolboxes. After solving this QP problem, we can get $\bar{\mathbf{c}}^*$ on the axes $\{x, y, z\}$ separately. Thus these trajectories can be reconstructed by Eq. 11. As these base functions of CNUBS are defined recursively, and the de Boor Algorithm has the ability to generate the trajectory of arbitrary frequency discretization form, this algorithm is utilized and running independently on each axis.

5.4 Spline Local Refinement

B-splines provide sufficient flexibility to compute a derivative-continuous trajectory and to perform local refinement by adjusting the control points or knots. However, the B-spline constructed by above approach may not have a collision-checking along such generated trajectories as mentioned in [9]. Thus our goal is to check the unsatisfied portions of the B-spline that disturb the minimum distance and pull them near the nominal collision-free trajectory, at the same time to keep the collision-free portions unchanged.

Algorithm 4 Algorithm for Spline Local Refinement

```
Input: Nominal trajectory $\gamma_{\text{norm}}(t), t \in [0, t_l]$, iteration number iteration
Output: Smoothed collision-free B-spline trajectory $\gamma_B(t)$
  for $k = 1$ : iteration do
      Randomly Choose $\gamma_{\text{norm}}(t), t \in [t_a, t_b]$ from the input nominal trajectory
      Construct B-spline $\gamma_B(t)$ to interpolate $m+1$ points on $\gamma_{\text{norm}}(t), t \in [t_a, t_b]$
      Perform spline collision checking for $\gamma_B(t)$
      Resolve the colliding segments in $\gamma_B(t)$ based on spline modification
      Refine $\gamma_B(t)$ locally to satisfy minimum jerk constraints
  end for
```

The whole smoothing algorithm is shown in Algorithm 4. Given an input nominal trajectory $\gamma_{\text{norm}}(t), t \in [0, t_l]$, we can first interpolate it with a smooth spline curve $\gamma_B(t)$. If the number of sampling points is enough, $\gamma_B(t)$ can approximate $\gamma_{\text{norm}}(t)$ with arbitrary precision. Thus, a collision-free and smooth $\gamma_B(t)$ can be always found. Then, we iteratively shortcut the nominal trajectory. At each iteration, we randomly choose a segment $\gamma_{\text{norm}}(t), t \in [t_a, t_b]$ and construct a smoother B-spline trajectory $\gamma_B(t)$ that interpolates the boundary points and corresponding intermediate points. Next, we apply spline collision checking criteria like minimum distance to evaluate whether every portion along the trajectory $\gamma_B(t)$ is collision-free. When a collision happens between neighboring trajectories, we can resolve the collisions by using a recursive method to modify the spline curve. After a collision-free trajectory is computed, we then perform local refinement to satisfy the dynamic bound constraints.

6 Robust Controller Design

6.1 Robust Perfect Tracking

In order to suppress the disturbances generated from other quadrotors and keep collision free, a robust outer loop is crucial to be designed for quadrotors to accurately track reference trajectories in real time. Because the quadrotor body acceleration command $\mathbf{a}_c$ dose not affect the heading direction, the yaw angle reference ψ_c is not considered in the following outer-loop controller design process, which will be generated independently.

As the differential flat characteristics of the quadrotor, after extracting the rotational dynamics, the quadrotor can be seen as a point mass model decoupld in three degree of freedom (DOF) space with constraints of velocity, acceleratioin, jerk, etc. Therefore, for the control law design, in each DOF the outer-loop quadrotor translational dynamics can be simplified as a double integrator as

$$\begin{cases} \dot{\mathbf{p}} = \mathbf{v} \\ \dot{\mathbf{v}} = \mathbf{a} \end{cases} \tag{19}$$

where $\mathbf{p}$, $\mathbf{v}$, $\mathbf{a}$ respectively denote the position, velocity and acceleration vectors of the quadrotor.

The robust perfect tracking (RPT) control approach proposed in [5] accurately meets the requirements. Unlike the nonlinear controllers developed in [4] and [11], in theory, the designed RPT controller can make the linearized system, without any input and state constraints, track any given reference trajectory with arbitrarily fast enough settling time. To use RPT control method, we formulate the linear time invariant (LTI) system as

$$\begin{cases} \dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} + \mathbf{E}\mathbf{w} \\ \mathbf{y} = \mathbf{C}_1\mathbf{x} + \mathbf{D}_1\mathbf{w} \\ \mathbf{h} = \mathbf{C}_2\mathbf{x} + \mathbf{D}_2\mathbf{u} + \mathbf{D}_{22}\mathbf{w} \end{cases} \tag{20}$$

where $\mathbf{x}$, $\mathbf{u}$, $\mathbf{w}$, $\mathbf{y}$, $\mathbf{h}$ are respectively state, input, disturbance, measurement and controlled output. The designed RPT controller is to form a dynamic measurement control law as

$$\begin{cases} \dot{\mathbf{v}} = \mathbf{A}_c(\varepsilon)\,\mathbf{v} + \mathbf{B}_c(\varepsilon)\,\mathbf{y} + \mathbf{G}_0(\varepsilon)\,\gamma + \cdots + \mathbf{G}_{k-1}(\varepsilon)\,\gamma^{k-1} \\ \mathbf{u} = \mathbf{C}_c(\varepsilon)\,\mathbf{v} + \mathbf{D}_c(\varepsilon)\,\mathbf{y} + \mathbf{H}_0(\varepsilon)\,\gamma + \cdots + \mathbf{H}_{k-1}(\varepsilon)\,\gamma^{k-1} \end{cases} \tag{21}$$

where γ represents the reference trajectory, and ε is a parameter need to be selected properly. The controller $\mathbf{v}$ is capable of

1) Asymptotically stabilizing the closed-loop system to follow zero reference.

2) Once $e(t, \varepsilon)$ acts as tracking error, for arbitrary initial condition $\mathbf{x}_0$, existing

$$\|e\|_p = \left(\int_0^\infty \left| e(t)^p \right| dt \right)^{\frac{1}{p}} \to 0 \qquad (22)$$
$$s.t.\ \varepsilon \to 0$$

These derivatives are to create additional inputs in non-zero reference condition. For example, formulating the reference trajectory $\gamma(t)$ as Taylor series expansion at $t = 0$, it can also be effectively tracked by RPT controller.

In order to design a good tracking performance RPT controller, the above nominal system needs to be augmented as

$$\begin{cases} \dot{\mathbf{x}}_a = \begin{bmatrix} 0 & -1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \mathbf{x}_a + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \mathbf{u}_a \\ \mathbf{y}_a = \mathbf{x}_a \\ \mathbf{h}_a = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \mathbf{x}_a \end{cases} \qquad (23)$$

where $\mathbf{x}_a = \begin{bmatrix} \int \mathbf{p}_e & \mathbf{p}_r & \mathbf{v}_r & \mathbf{a}_r & \mathbf{p} & \mathbf{v} \end{bmatrix}^T$, and $\mathbf{p}_e = \mathbf{p}_r - \mathbf{p}$ is the position tracking error, $\mathbf{p}_r$, $\mathbf{v}_r$, $\mathbf{a}_r$ are respectively position, velocity and acceleration control references. Then the linear feedback control law can be obtained as

$$\mathbf{u}_a = \mathbf{F}_a \mathbf{x}_a \qquad (24)$$

where

$$\mathbf{F}_a = \begin{bmatrix} \frac{k_i \omega_n^2}{\varepsilon^3} & \frac{\omega_n^2 + 2\zeta\omega_n k_i}{\varepsilon^2} & \frac{2\zeta\omega_n + k_i}{\varepsilon} & 1 & -\frac{\omega_n^2 + 2\zeta\omega_n k_i}{\varepsilon^2} & -\frac{2\zeta\omega_n + k_i}{\varepsilon} \end{bmatrix}$$

where ε is a design parameter to adjust the settling time, and ω_n, ζ, k_i are the parameters that determine the desired pole locations of the infinite zero structure of Σ_a by

$$p_i(s) = (s + k_i)\left(s^2 + 2\zeta\omega_n s + \omega_n^2\right) \qquad (25)$$

6.2 Actuator Constraints

After the robust tracking controller has been designed, when the collision avoidance maneuver like sharp turn is performed, the quadrotor might act significant deviation from reference trajectory. In this situation, more control effort is required for the quadrotor to get back to the reference point along the trajectory, which might break the constraints of the actuators. A virtual vehicle method proposed in [8] is used to tackle this issue by adjusting the speed of the virtual vehicle as the norm of the tracking error $\|\mathbf{p}_e\|$ changes. Therefore, a virtual time parameter $s(t)$ is defined as $\dot{s}(t) = e^{-k_s\|\mathbf{p}_e\|}$, where k_s is a non-negative virtual gain.

This parameter is fed into the virtual input, which is the third-order derivative of the position of the trajectory. There is $\dot{s} < 1$ when the tracking error is large, and $\dot{s} = 1$ when the tracking error is zero, which indicates that it follows the trajectory at the desired speed. In more details, if $k_s = 0$, then $\dot{s} = 1$ and it will cause a larger pitch angle and thrust, if $k_s = 50$ then $\dot{s} < 1$, both the pitch angle and thrust will decrease. With the help of this mechanism, the instantaneous excessive control effort will be reduced, and the actuator constraints can be satisfied.

7 Numerical Analysis

In order to verify the feasibility of proposed collision avoidance algorithm, we use the simplified point mass model to simulate six quadrotors, set the start locations denoted with circles to form a flat circle shape and the goal locations denoted with stars to build up a tilted regular pentagram shape, both of which are generated by the proposed shape-based approach. Since the time-consuming of computing ORCA velocity can be effectively tackled by a low dimensional linear program with complexity $\mathcal{O}(n)$ of the increasing quadrotor number at the millisecond level. The Hungarian Algorithm is utilized, which generates the nominal trajectories denoted by dashed lines with time-step 0.25s and smoothed trajectories denoted by solid lines with time-step 0.05s as shown in Fig. 7. In this figure, the quadrotors' virtual radius is 1m, horizontal maximum speed is 1m/s and the synchronizing arrival time is 10s.

To analyze the downwash effect avoidance of this proposed algorithm, we first adopt collision avoidance algorithm and use the same configuration as defined above to simulate the swap actions of two and four quadrotors. The nominal trajectories are shown in Fig. 8a and b. Here,

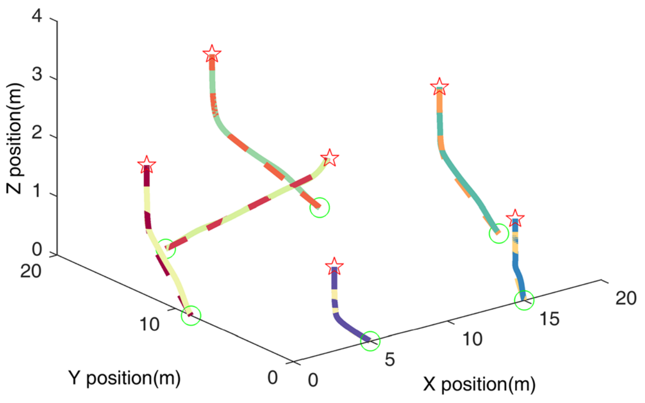
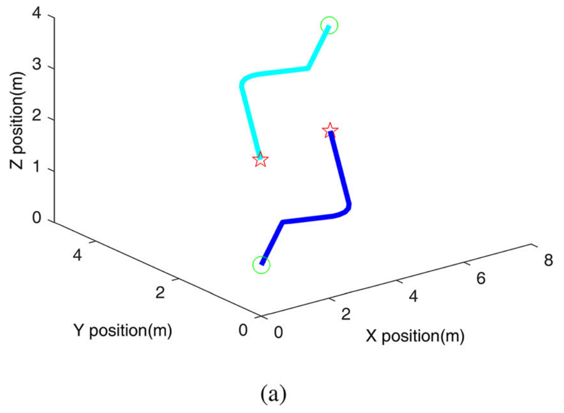


Fig. 7 Trajectories generated of six quadrotors switching from flat circle shape to tilted regular pentagram shape



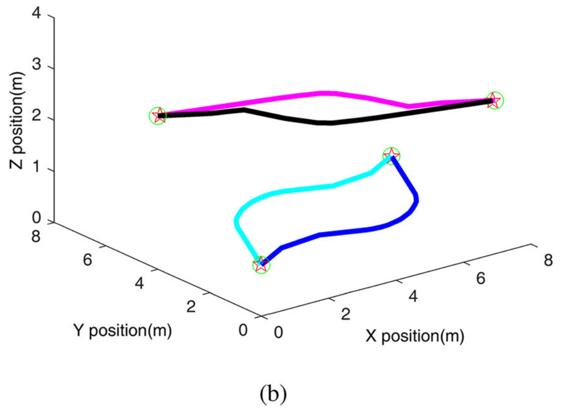


Fig. 8 Trajectories generated of two or four quadrotors swapping positions

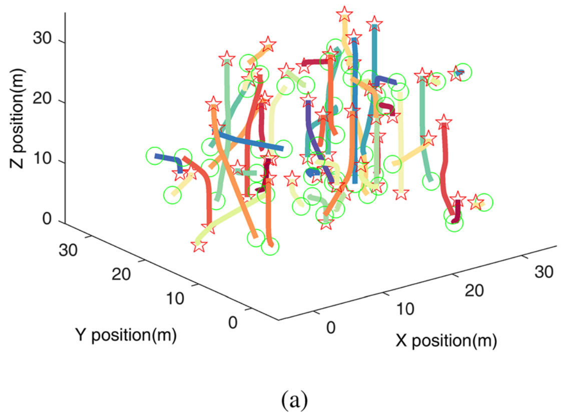
we abandon the Hungarian Algorithm and make quadrotor's start and goal positions localize at the same altitude. From these figures and generated corresponding time list, there are no straight-line trajectories and the vertical overlap does not occur at the same time, thus the downwash effect of these quadrotors can be successfully avoided. Although the generated trajectories may not be smooth enough due to the relatively larger preset time-step, the results are not affected and these nominal trajectories will be further smoothed by jerk-optimized B-spline algorithm considering the downwash effect avoidance.

For the purpose of testing collision avoidance ability, we select the start and goal locations randomly and densely in a confined 30m×30m×30m 3D space with constraints of the CAPT algorithm and vertically non-overlapping condition, and set the same configuration as previous simulations. Using the Hungarian Algorithm, the generated trajectories of 50 quadrotors are shown in Fig. 9a. As the vertical distances are always infinite at each time-step, the corresponding horizontal minimum distance between every pair of quadrotors is shown in Fig. 9b. These figures show that the minimum distance is always larger than the preset virtual radius 1m, thus collision within a large quadrotor team is successfully avoided during the flight of about 10.5s. Moreover, another fifty tests were performed and no collision occurred.

8 Flight Test Results

The proposed quadrotor team performance kicked off the opening ceremony of Rotorcraft Asia & Unmanned Systems Asia 2017 at the Changi Exhibition Centre in Singapore. For the preparation of this ceremony, the designed six quadrotors as shown in Fig. 10 with high control bandwidth are adopted. Each one uses four propellers of 6.5 inches diameter, with a motor-to-motor width of 450mm and weight of 950g.

Many flight tests were carried out before the official ceremony to guarantee the proposed algorithms and



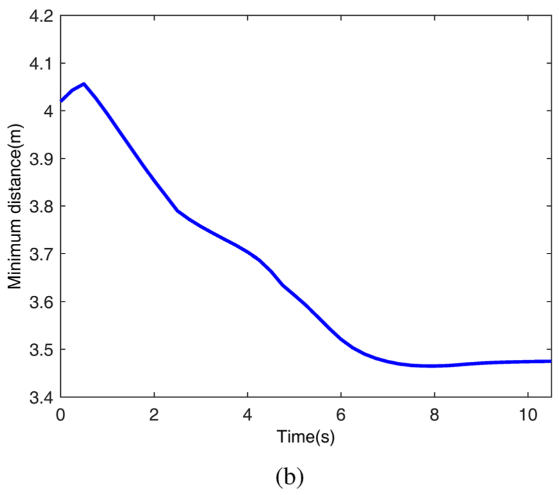


Fig. 9 Trajectories generated and minimum distance in the horizontal plane of fifty quadrotors with randomly-distributed start and goal positions

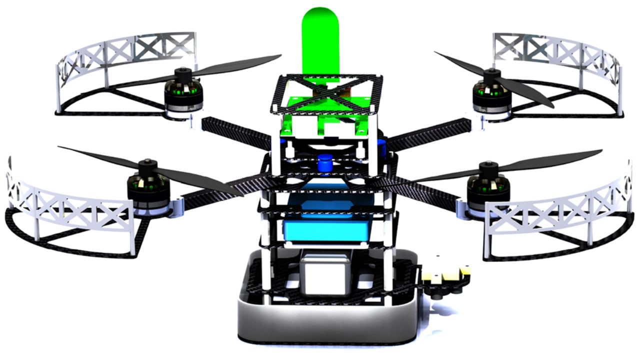


Fig. 10 Rendering of the designed quadrotor

controllers. The whole procedure of the flight test follows Fig. 1 and includes two steps: 1) The first one can be seen as the preparation of the flight test. It is to obtain required static and dynamic patterns and collision-free trajectories. This step is carried out on the ground station with MATLAB and the collision checker should be activated to make these trajectories feasible. This step is similar to Section 7 and omitted here. 2) The second step is to test the flight performance, log and analyze the results. The generated patterns and trajectories need to be uploaded into the on-board computers of PIXHAWK ETH attached to the quadrotors, and the designed tracking controller is used to make each quadrotor follow the given trajectory accurately. In this step, actuator constraints should be checked to ensure that quadrotors move consistently within their maneuver limits. This section is to give these results and corresponding discussion.

To match the theme of "Experience The Drones Of The Future" for this opening ceremony, we designed a special static heart-shaped formation, which is given in Fig. 15a. And other static patterns, such as the circle and the rectangle, are also designed. For these dynamic formation patterns, the figure-eight and spiral are selected and realized, the result of figure-eight is shown in Fig. 16. The generated reference trajectories with the jerk-optimized B-spline algorithm constrains the maximum absolute value of velocity within 1m/s and acceleration within 1m/s^2 in horizontal direction, and 0.5m/s and 0.5m/s^2 in vertical direction. The position references are shown in Fig. 13, and the velocity and acceleration references are given in Figs. 14 and 11 respectively.

After generating required formation patterns and collision-free trajectories. The designed tracking controller needs to be implemented. In practice, due to the constraints of physical system and inner-loop dynamics, we would like to limit the bandwidth of the outer loop to be at least one third of the inner-loop system bandwidth. The outer-loop control frequency is set as 50Hz in PIXHAWK ETH. In

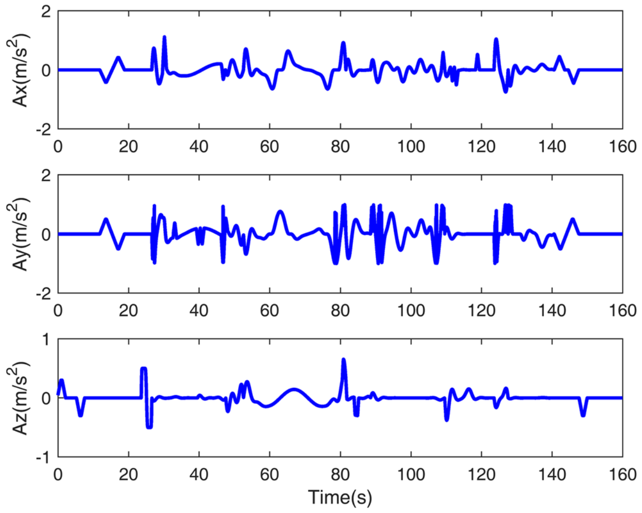


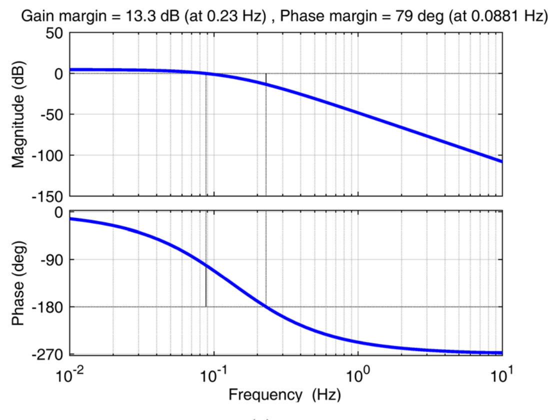
Fig. 11 X, Y, Z(Height) reference acceleration of a single quadrotor

order to verify the robustness of the outer-loop flight control system, we examine the frequency response of each individual channel of the outer-loop system. Then, x and y positions are with parameters of $\omega_n = 0.8$, $\zeta = 0.8$, $\varepsilon = 1$ and $k_i = 0.9$, these two results are the same and shown in Fig. 12a. In height direction, parameters are $\omega_n = 1.0$, $\zeta = 0.8$, $\varepsilon = 1$ and $k_i = 0.55$, the result is shown in Fig. 12b. It is clear that all of the channels have a gain margin larger than 10 dB and a phase margin greater than 75 degrees. The robustness of the outer-loop control system is good.

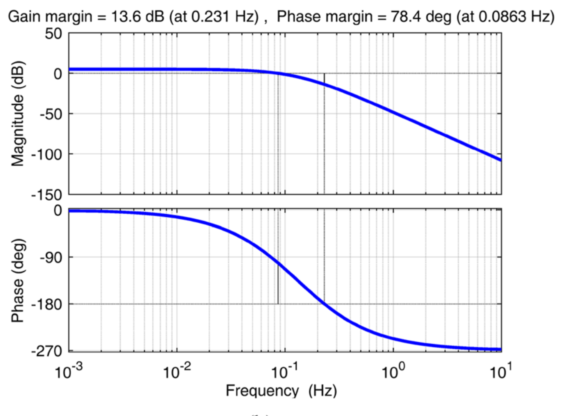
The show environment is a confined space of 7m×16m×6m. To set up the localization system, a relatively low cost and low power method of indoor localization utilizing ultra-wideband (UWB) radio was chosen, of which the mean error for distance measurement is around 0.1m. With the robust performance of outer-loop RPT controller, the X, Y and Z (height) positions track the reference trajectories accurately for the entire duration despite the relatively poor measurement of UWB compared with Vicon even refined by the method in [13]. Figure 13a, b and c show a single quadrotor's reference, actual state positions and their errors in 3D space respectively.

Meanwhile, these velocity results of X, Y and Z direction are given in Fig. 14. The benefit of the RPT controller is clearly shown here as the quadrotor not only tracks its position well, but also has good velocity control performance. The results also show that the velocity constraint of the quadrotor within −1m/s to 1m/s in horizontal direction is satisfied. The velocity in height direction is also inside the boundary of −0.5m/s to 0.5m/s except some overshoot responses near the ground because of the shaded UWB signals and rotational movements with lager acceleration.

From Figs. 13 and 14, we can see that these positions and velocities are not tracked so accurately in the height

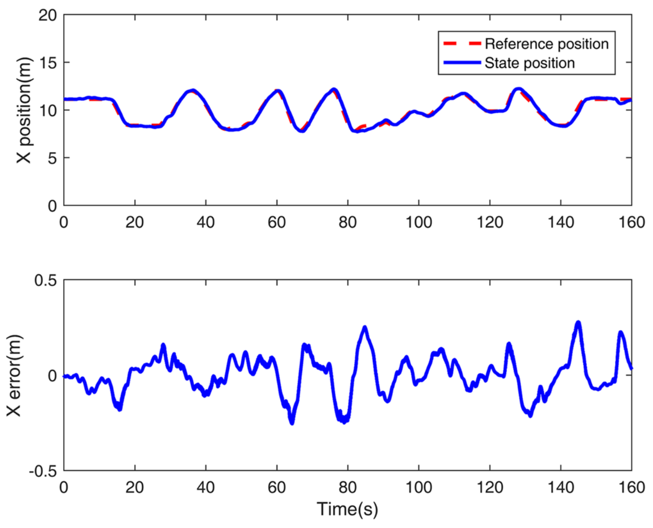


(a)

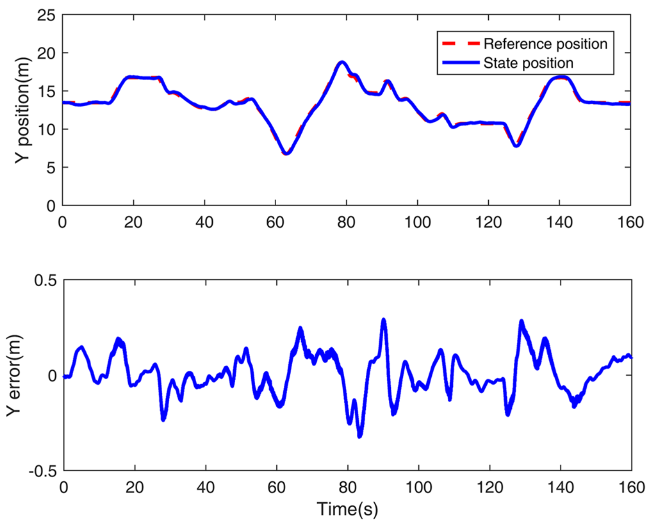


(b)

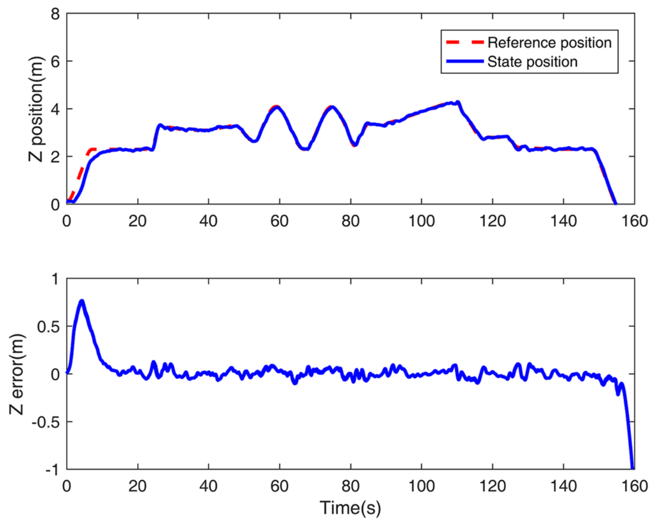
Fig. 12 Gain and phase margins of X, Y, Z (Height) position control



(a)



(b)



(c)

Fig. 13 X, Y, Z(Height) reference, state positions and related tracking errors of a single quadrotor

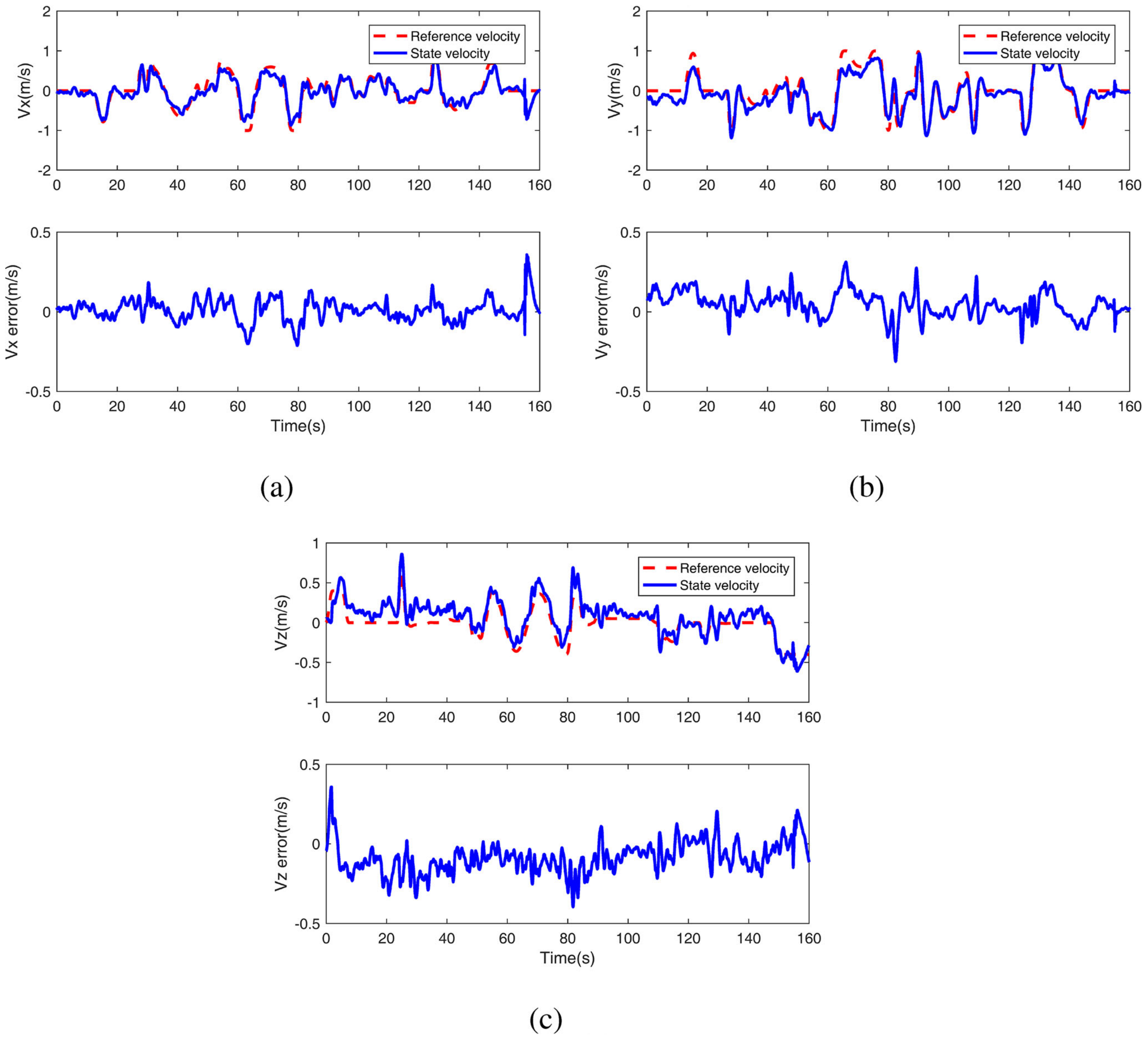


Fig. 14 X, Y, Z(Height) reference, state velocity/acceleration and related tracking errors of a single quadrotor

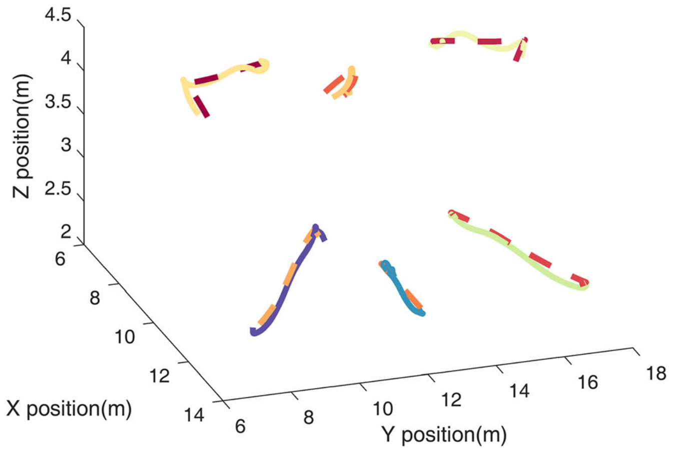
direction compared to the horizontal plane. This is due to the larger horizontal acceleration, especially during the process of tracking rotational trajectories. In addition, the height positions and velocities of UWB measurement near the ground are not precise as signals are partially blocked by the desk on the platform. The actual preset trajectories are activated and invalid from the height of 2m respectively at the start and terminal stages.

Then, the formation switching result is given in Fig. 15a. It shows the reference smoothed by CNUBS and actual flight trajectories of six quadrotors switching from a static tilted heart shape to a larger static tilted rectangle, where dashed lines represent reference trajectories and solid lines denote actual flight ones.

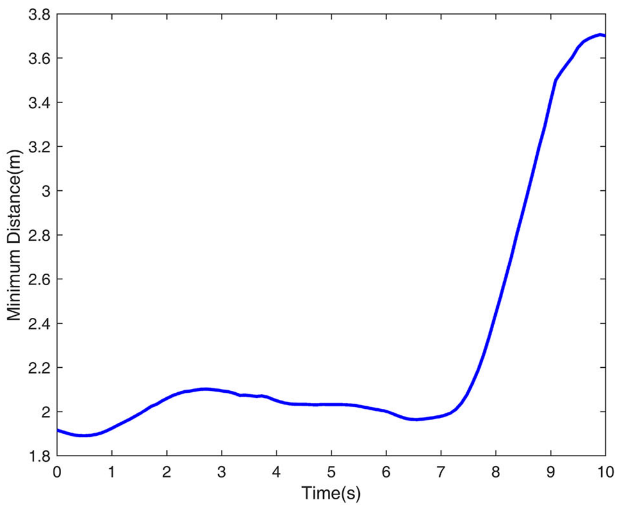
Since there are only six members, the shape of heart is of relatively low resolution. Figure 15b illustrates that the whole swicthing time is about 10s and the minimum distance of quadrotors in the horizontal plane during the actual flight process is always larger than the preset safety region of 1.5m, thus the effect of downwash is successfully avoided.

The partial dynamic figure-eight reference trajectory after being smoothed by CNUBS and actual flight trajectory of one quadrotor are illustrated in Fig. 16a. Meanwhile, Fig. 16b shows the actual trajectories of the whole team flying in a dynamic figure-eight formation.

Our designed staic and dynamic formation patterns switched successfully during the entire show. Figure 17a, b

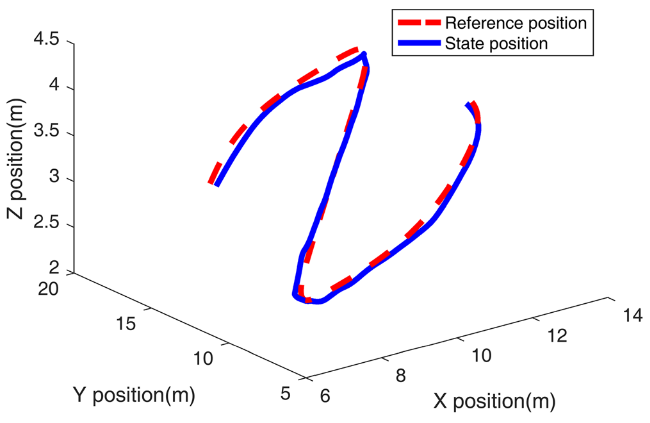


(a)

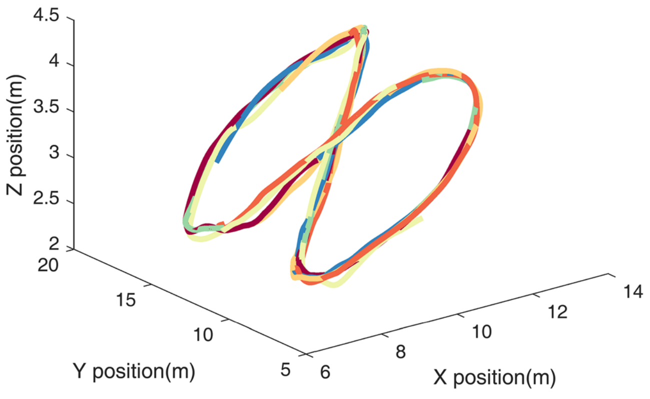


(b)

Fig. 15 Actual flight and minimum jerk references, minimum distance of actual flight in the horizontal plane of six quadrotors switching from tilted heart shape to tilted rectangle shape



(a)



(b)

Fig. 16 Actual flight, minimum jerk reference of partial eight shape with a single quadrotor and actual flight of eight shape with six quadrotors

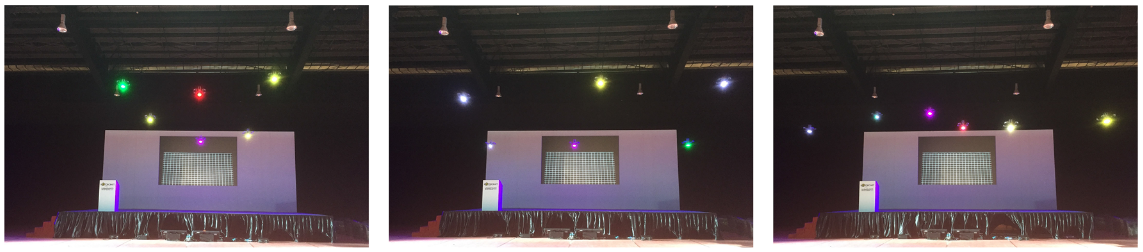

(a) (b) (c)

Fig. 17 The snapshots of switched three different formation patterns with six quadrotors: heart, rectangle and partial figure-eight

and c show the snapshots of the resulting heart, rectangle and partial figure-eight formation patterns of six quadrotors. The full show video can be viewed at https://www.youtube.com/watch?v=eSdghUK-a-8.

The flight test results in this section show that our proposed algorithms and controllers are suitable for a confined indoor space with multiple quadrotors. First, static and dynamic formation patterns can be generated according to the user's requirements adaptively and quickly. Second, these switching trajectories generated by our modified ORCA algorithm can not only keep collision-free between agents, but can also keep off the downwash effect caused by other quadrotors. Third, these dynamic constraints of collision-free trajectories can be satisfied under the jerk-optimized B-spline method. Fourth, the RPT controller is designed to satisfy all the dynamic constraints of quadrotors with good robustness and tracking performance.

9 Conclusion

In this paper, trajectory planning approaches have been developed for a quadrotor team to deal with formation flight challenges in an indoor confined space environment. Firstly, both static and dynamic formation patterns are generated optimally and efficiently. The modified 3D ORCA algorithm is utilized to create the nominal trajectories, which can effectively avoid the adverse downwash effect generated by other quadrotors. A cubic B-spline approach then interpolates these nominal trajectories to satisfy the minimum jerk constraint, thus the smoothed trajectories are generated and locally refined, which is suitable for quadrotor dynamics. Next, considering the actuator limits of quadrotors, the robust outer-loop tracking controller is designed to compensate the low resolution of sensors. Finally, all of these methods are successfully verified by the relevant simulation results and adopted in the real-world flight test. The static obstacle avoidance will be carried out to complement the current work in the near future. Then, the coming research directions will be focused on issues such as distributed formation and swarm control, real-time formation pattern and trajectory generation, fault detection and toleration.

Acknowledgements This work was supported by National Natural Science Foundation of China [grant number 61673327], China Scholarship Council [grant number 201606310153], as well as Aviation Science Foundation of China [grant number 20160168001]. The authors would like to thank Prof. Ben M. Chen and his Unmanned System Research Group at National University of Singapore.

References

1. Agha-mohammadi, A.-a., Ure, N.K., How, J.P., Vian. J.: Health aware stochastic planning for persistent package delivery missions using quadrotors. In: IEEE /RSJ International Conference on Intelligent Robots and Systems, pp. 3389–3396. Chicago (2014)
2. Alvissalim, M.S., Zaman, B., Hafizh, Z.A., Ma'sum, M.A., Jati, G., Jatmiko, W., Mursanto, P.: Swarm quadrotor robots for telecommunication network coverage area expansion in disaster area. In: Annual Conference of the Society of Instrument and Control Engineers of Japan, pp. 2256–2261. Akita (2012)
3. Augugliaro, F., Schoellig, A.P., D'Andrea, R.: Methods for designing and executing an aerial dance choreography. IEEE Robot. Autom. Mag. **20**(4), 96–104 (2013)
4. Basri, M.A.M., Husain, A.R., Danapalasingam, K.A.: Enhanced backstepping controller design with application to autonomous quadrotor unmanned aerial vehicle. J. Intell. Robot. Syst. **79**(2), 295–321 (2015)
5. Chen, B.M., Lee, T.H., Peng, K.M., Venkataramanan, V. Hard Disk Drive Servo Systems, 2nd edn. Springer, London (2006)
6. De Boor, C.: A Practical Guide to Splines, vol. 27. Springer-Verlag, New York (1978)
7. Du, Q., Faber, V., Gunzburger, M.: Centroidal Voronoi Tessellations: Applications and Algorithms. SIAM Rev. **41**(4), 637–676 (1999)
8. Egerstedt, M., Hu, X., Stotsky, A.: Control of mobile platforms using a airtual vehicle approach. IEEE Trans. Autom. Control **46**(11), 1777–1782 (2001)
9. Elbanhawi, M., Simic, M., Jazar, R.N.: Continuous path smoothing for car-like robots using b-spline curves. J. Intell. Robot. Syst. **80**(1), 23–56 (2015)
10. Fiorini, P., Shiller, Z.: Motion planning in dynamic environments using velocity obstacles. Int. J. Robot. Res. **17**(7), 760–772 (1998)
11. Guadarrama-Olvera, J.R., Corona-Sánchez, J.J., Rodriguez-Cortes, H.: Hard real-time implementation of a nonlinear controller for the quadrotor helicopter. J. Intell. Robot. Syst. **73**(1-4), 81–97 (2014)
12. Guérin, F., Guinand, F., Brethé, J.F., Pelvillain, H., et al.: Towards an autonomous warehouse inventory scheme. In: IEEE Symposium Series on Computational Intelligence, pp. 1–8, Athens (2016)
13. Guo, K., Qiu, Z., Miao, C., Zaini, A.H., Chen, C.L., Meng, W., Xie, L.: Ultra-wideband-based localization for quadcopter navigation. Unmanned Syst. **4**(01), 23–34 (2016)
14. Hehn, M., D'Andrea, R.: Real-time trajectory generation for quadrocopters. IEEE Trans. Robot. **31**(4), 877–892 (2015)
15. Kano, H., Nakata, H., Martin, C.F.: Optimal curve fitting and smoothing using normalized uniform b-splines: a tool for studying complex systems. Appl. Math. Comput. **169**(1), 96–128 (2005)
16. Kano, H., Fujioka, H., Martin, C.F.: Optimal smoothing and interpolating splines with constraints. Appl. Math. Comput. **218**(5), 1831–1844 (2011)
17. Kumar, V., Michael, N.: Opportunities and challenges with autonomous micro aerial vehicles. In: Robotics Research, pp. 41–58. Springer (2017)
18. Kuriki, Y., Namerikawa, T.: Consensus-based cooperative formation control with collision avoidance for a multi-UAV system. In: American Control Conference, pp. 2077–2082. Portland (2014)
19. Mellinger, D., Kumar, V.: Minimum snap trajectory generation and control for quadrotors. In: IEEE International Conference on Robotics and Automation, pp. 2520–2525. Shanghai (2011)
20. Mellinger, D., Kushleyev, A., Kumar, V.: Mixed-integer quadratic program trajectory generation for heterogeneous quadrotor teams.

In: IEEE International Conference on Robotics and Automation, pp. 477–483. St Paul (2012)
21. Mücke, E.P., Saias, I., Zhu, B.: Fast randomized point location without preprocessing in two-and three-dimensional Delaunay triangulations. Comput. Geom. **12**(1-2), 63–83 (1999)
22. Munkres, J.: Algorithms for the assignment and transportation problems. J. Soc. Indus. Appl. Math. **5**(1), 32–38 (1957)
23. Rosales, C., Leica, P., Sarcinelli-Filho, M., Scaglia, G., Carelli, R.: 3D formation control of autonomous vehicles based on null-space. J. Intell. Robot. Syst. **84**(1-4), 453–467 (2016)
24. Turpin, M., Michael, N., Kumar, V.: CAPT: concurrent assignment and planning of trajectories for multiple robots. Int. J. Robot. Res. **33**(1), 98–112 (2014)
25. Van Den Berg, J., Guy, S.J., Lin, M., Manocha, D.: Reciprocal n-body collision avoidance. In: Robotics research, pp. 3–19. Springer (2011)
26. Wang, F., Liu, P., Zhao, S., Chen, B.M., Phang, S.K., Lai, S., Pang, T., Wang, B., Cai, C., Lee, T.H.: Development of an unmanned helicopter for vertical replenishment. Unmanned Syst. **3**(01), 63–87 (2015)

Yang Xu received his B.Sc. ans M.Sc. degrees from the College of Automation, Nanjing University of Aeronautics and Astronautics. He is currently an exchange Ph.D. student at National University of Singapore and a Ph.D. candidate at Xiamen University. His research interests lie in control theory and application of multiple robotic systems.

Shupeng Lai received his B.Eng. degree from the Department of Electronic and Electrical Engineering, Nanyang Technological University and his Ph.D. degree from the Department of Electrical and Computer Engineering, National University of Singapore. He is currently a Post-doc research fellow at National University of Singapore. His research interests lie in path planning and navigation of multiple unmanned systems.

Jiaxin Li received his B.Eng. degree from the Department of Mechanical Engineering, Tsinghua University. He is currently a Ph.D. candidate at National University of Singapore. His research interests lie in computer vision and signal fusion.

Delin Luo received his Ph.D. degree from Nanjing University of Aeronautics and Astronautics. He is currently an associate professor of the School of Aerospace Engineering at Xiamen University. His main research interests include aircraft guidance and control, UAV cooperative control, computational intelligence.

Yancheng You received his Ph.D. degree from Nanjing University of Aeronautics and Astronautics and then became a research scientist in German Aerospace Center (DLR). He is currently a full professor and the dean of the School of Aerospace Engineering at Xiamen University. His main research interests include flight control theory and application, numerical turbulence modelling and aerodynamic design of hypersonic vehicles.