PLOS ONE

RESEARCH ARTICLE

Formation flight and collision avoidance for multiple UAVs based on modified tentacle algorithm in unstructured environments

Minghuan Zhang*

School of Astronautics, Northwestern Polytechnical University, Xi'an, Shaan xi Province, People's Republic of China

* zhangmh@nwpu.edu.cn

Abstract

This paper presents a method for formation flight and collision avoidance of multiple UAVs. Due to the shortcomings such as collision avoidance caused by UAV's high-speed and unstructured environments, this paper proposes a modified tentacle algorithm to ensure the high performance of collision avoidance. Different from the conventional tentacle algorithm which uses inverse derivation, the modified tentacle algorithm rapidly matches the radius of each tentacle and the steering command, ensuring that the data calculation problem in the conventional tentacle algorithm is solved. Meanwhile, both the speed sets and tentacles in one speed set are reduced and reconstructed so as to be applied to multiple UAVs. Instead of path iterative optimization, the paper selects the best tentacle to obtain the UAV collision avoidance path quickly. The simulation results show that the method presented in the paper effectively enhances the performance of flight formation and collision avoidance for multiple high-speed UAVs in unstructured environments.

Citation: Zhang M (2017) Formation flight and collision avoidance for multiple UAVs based on modified tentacle algorithm in unstructured environments. PLoS ONE 12(8): e0182006. https://doi.org/10.1371/journal.pone.0182006

Editor: Wen-Bo Du, Beihang University, CHINA

Received: March 23, 2017

Accepted: July 11, 2017

Published: August 1, 2017

Data Availability Statement: All relevant data are within the paper and its Supporting Information file.

Funding: This work was supported by China Space Foundation (N2015KC0121).

Competing interests: The author has declared that no competing interests exist.

1. Introduction

To control the formation flight of multiple unmanned air vehicles (UAVs) is a challenge as they are widely used for military and civil purposes [1–2]. Through the cooperation among multiple UAVs, their formation flight performances in missions such as search and rescue, surveillance, mapping and deployment of troops [3]are more effectively enhanced. The flight formation technique is the building block of multiple UAVs' cooperation.

Collision avoidance is central to the UAV formation flight research [4–6]. Regarded as a complicated control problem, it faces challenges in designing a quick and robust controller which can maintain the relative position as well as safe distance in between [7]. In order to avoid collision between UAVs and obstacles or UAV pairs, it is urgently necessary to study formation switching and collision avoidance. Classical path planning methods, such as potential field method, genetic algorithm, grid-based method and geometric approach [8–19], are applied to single UAV collision avoidance. Many researchers have contributed to the development of collision avoidance algorithms for a single entity. Ref. [20] presents a modified artificial potential field (MAPF) method for a UAV to avoid collision in a 3D space. Due to the

shortcomings of the traditional artificial potential field (APF) method, the MAPF method is developed in a certain constraint reference frame to decouple the decomposed force from the MAPF method with specific physical constraints. In the constraint reference frame, the path is examined with the updated force of the MAPF method, implemented by the UAV, and corrected if the updated force disagrees with the physical constraints. Such an examination and correction loop makes sure that the planned path can practically meet the UAV's motion status and manoeuver capability. In order to enhance the estimation accuracy affected by the constantly changing path-loss factor during UAV flight, Ref. [21] proposed a UAV collision detection and decision making and path re-planning method.

In order to overcome the shortcomings of existing methods, this paper considers the manoeuver information of both UAV and aerial intruder and then presents a collision decision-making method based on the proposed regions and the interfered fluid dynamical (IFD) algorithm. Ref. [22] proposes a three-dimensional (3D) and real-time path planning method by combining the improved Lyapunov guidance vector field (LGVF) and the interfered fluid dynamical system (IFDS) with the varying receding-horizon optimization strategy based on the model predictive control (MPC).The experimental results show that the above hybrid method is applicable to various dynamic environments. Ref. [23] presents a collision avoidance algorithm for cooperative UAVs that share three-dimensional airspace. Based on the geometric optimization model, the feasible and optimal trajectory is obtained for a chosen UAV, with the local optimization scope reaching the operational level. The local optimization scope generates an optimal flight trajectory with the objective function in response to a set of restrictions that reduces the solution space. This collision avoidance manoeuver has such advantages as optimization with minimal cost, robustness that considers the global traffic condition, scalability that possesses explicit coordinates of waypoints and efficiency in implementing various tests of tuning parameters.

For multiple UAVs, it is necessary to consider their formation flight and collision avoidance simultaneously. Without taking into consideration the correlation between multiple UAVs, the collision avoidance method often works in a static structured environment and was not directly applied to the collision avoidance of formation flight. Ref. [24] proposes a guidance law for formation flight and collision avoidance. With the concept of elastic weighting factor, multiple UAVs are able to actively cope with the collision between both UAVs and static obstacles during their formation flight. Based on the sophisticated route planning, which spends time on processing environmental information, this guidance law has good collision avoidance performance. In a static structured environment, the UAV formation does not need to update environmental information frequently, and the UAVs have sufficient residual manoeuver time. However, when they fly at high speed in an unstructured environment, it is necessary to compute the information on environment and path in each time set so that the residual manoeuver time can be greatly shortened. Failure avoidance is highly possible when UAVs do not have enough time to complete their manoeuver. Its computation should be done in real time. The speed of UAV always needs to be over 100m/s to fulfil a specific mission, while the maximum speed in Ref. [24] is only 60m/s, even much less according to quadrotor in other articles. Therefore, these conventional methods have difficulties in computing in real time the collision avoidance of multiple high-speed UAVs in unstructured environments.

Felix proposed a simple but effective method for autonomous robot navigation in unstructured environments by using a set of "tentacles" that represents pre-calculated trajectories defined in the ego-centred occupancy grid [25]. To compute the collision avoidance in real time in unstructured environments, this method has the advantage of selection instead of real time computation. All the potential paths are pre-computed and stored; thus the real-time path planning becomes unnecessary. It is not necessary to create a whole environment model. This means that the cost of real-time computation will be greatly reduced. Therefore, we

propose the method for collision avoidance of multiple high-speed UAVs under unstructured environments.

However, the speed of UAV is far greater than a robot in Ref. [25] and the steering command generation suffers from the data calculation problem the application problem. This paper proposes a method for multiple UAVs' formation flight and collision avoidance based on the modified tentacle algorithm. Since the speed sets and tentacles in one speed set are reduced and reconstructed, the data calculation problem of the UAV are also solved. By modifying the ego-centred occupancy grid, we model their formation on an unstructured environment and solve the application problem. Instead of path computing, we can select the best tentacle to obtain the UAV collision avoidance path quickly. The simulation results show that our method can effectively compute in real time.

This paper is organized as follows. Section 2 proposes the formation flight controller. Section 3 gives the collision avoidance method based on the modified tentacle algorithm. Section 4 provides simulation results in which the performance of the proposed method is verified. Finally, Section 5 is devoted to the summary of the main results and the future work.

2. UAV formation flight

In order to show the dynamic and static obstacle avoidance simultaneously, this paper describe an UAV formation flight. For each single UAV, all the other UAVs are considered as a dynamic obstacle when the formation is avoiding a static obstacle.

Each UAV in its formation is considered as a mass point. We use the leader-follower formation model[26–30]: one of the UAVs in the formation is defined as the leader and the other is the follower. Therefore, the formation control problem is transformed into a tracking problem between follower and leader. This means that the follower only needs to keep an appropriate relative position and direction from the leader. This paper takes follower as origin to establish reference frame on follower[31–34] to show the relationship between leader and follower, as Fig 1 shows.

OXYZ is the inertial coordinate system, X_l,Y_l,Z_l is the coordinate of the leader in inertial system, while X_w,Y_w,Z_w is the follower's coordinate. In follower reference frame, x_w,y_w,z_w refers the distance between leader and follower. V_l,V_w refers velocity of leader and follower. ψ_l,ψ_w is the heading angle of leader and follower and θ_l,θ_w is their track angle.

The motion of UAV is controlled by autopilot, its mathematical model[35–37] as Eq (1):

$$\begin{aligned}
\dot{V}_l &= \frac{1}{\tau_{V_l}}(V_{lc} - V_l) \\
\dot{V}_w &= \frac{1}{\tau_{V_w}}(V_{wc} - V_w) \\
\dot{\psi}_l &= \frac{1}{\tau_{\psi_l}}(\psi_{lc} - \psi_l) \\
\dot{\psi}_w &= \frac{1}{\tau_{\psi_w}}(\psi_{wc} - \psi_w) \\
\dot{\theta}_l &= \frac{1}{\tau_{\theta_l}}(\theta_{lc} - \theta_l) \\
\dot{\theta}_w &= \frac{1}{\tau_{\theta_w}}(\theta_{wc} - \theta_w)
\end{aligned} \tag{1}$$

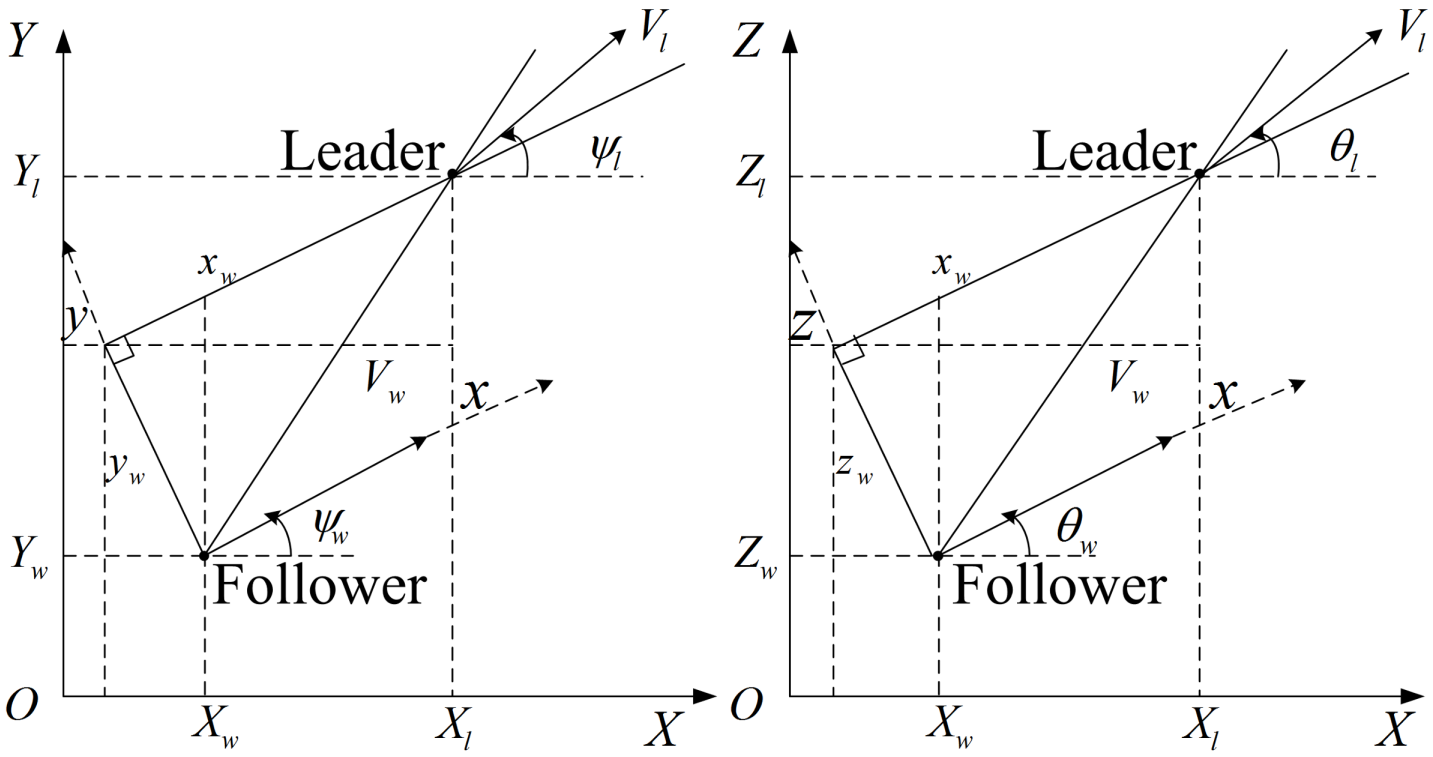


Fig 1. Inertial coordinate system and reference frame.

https://doi.org/10.1371/journal.pone.0182006.g001

In above equation, $\tau_{V_l}, \tau_{V_w}, \tau_{\psi_l}, \tau_{\psi_w}, \tau_{\theta_l}, \tau_{\theta_w}$ refers to the time constant of velocity, heading angle, track angle, while V_{lc},V_{wc},ψ_{lc},ψ_{wc},θ_{lc},θ_{wc} refers to the instruction of velocity, heading angle, track angle.

Motion equation of leader and follower in inertial coordinate:

$$\begin{aligned}
\dot{X}_l &= V_l\cos\psi_l\cos\theta_l \\
\dot{Y}_l &= V_l\sin\psi_l\cos\theta_l \\
\dot{Z}_l &= V_l\sin\theta_l \\
\dot{X}_w &= V_w\cos\psi_w\cos\theta_w \\
\dot{Y}_w &= V_w\sin\psi_w\cos\theta_w \\
\dot{Z}_w &= V_w\sin\theta_w
\end{aligned} \tag{2}$$

According to geometrical relationship in Fig 1, the coordinate in inertial frame of leader can display as follow:

$$\begin{aligned}
X_l &= X_w + x_w\cos\psi_w\cos\theta_w - y_w\sin\psi_w + z_w\cos\psi_w\sin\theta_w \\
Y_l &= Y_w + x_w\sin\psi_w\cos\theta_w + y_w\cos\psi_w + z_w\sin\psi_w\sin\theta_w \\
Z_l &= Z_w + x_w\sin\theta_w + z_w\cos\theta_w
\end{aligned} \tag{3}$$

Namely,

$$\begin{bmatrix} X_l - X_w \\ Y_l - Y_w \\ Z_l - Z_w \end{bmatrix} = A \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} \tag{4}$$

Inside,

$$A = \begin{bmatrix} \cos\psi_w \cos\theta_w & -\sin\psi_w & \cos\psi_w \sin\theta_w \\ \sin\psi_w \cos\theta_w & \cos\psi_w & \sin\psi_w \sin\theta_w \\ \sin\theta_w & 0 & \cos\theta_w \end{bmatrix} \tag{5}$$

The relative distance between two UAVs in three dimensions as:

$$\begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} = A^{-1} \begin{bmatrix} X_l - X_w \\ Y_l - Y_w \\ Z_l - Z_w \end{bmatrix} \tag{6}$$

Then a PID formation controller can be designed to keep the distance in Eq (6) by computing a group of control instruction V_{wc},ψ_{wc},θ_{wc} for the follower.

The error between follower's current position and expected position is as follows:

$$e = \begin{bmatrix} e_x \\ e_y \\ e_z \end{bmatrix} = \begin{bmatrix} X_l - \bar{X}_w \\ Y_l - \bar{Y}_w \\ Z_l - \bar{Z}_w \end{bmatrix} - \begin{bmatrix} X_l - X_w \\ Y_l - Y_w \\ Z_l - Z_w \end{bmatrix} = A \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} - \begin{bmatrix} X_l - X_w \\ Y_l - Y_w \\ Z_l - Z_w \end{bmatrix} \tag{7}$$

In above, $\bar{X}_w$, $\bar{Y}_w$, $\bar{Z}_w$ respect to follower's expected position. Follower's control instruction V_{wc},ψ_{wc},θ_{wc} can be generated by the PID algorithm:

$$\begin{aligned} V_{wc} &= K_{px} e_x + K_{ix} \int e_x dt + K_{dx} \frac{de_x}{dt} \\ \psi_{wc} &= K_{py} e_y + K_{iy} \int e_y dt + K_{dy} \frac{de_y}{dt} \\ \theta_{wc} &= K_{pz} e_z + K_{iz} \int e_z dt + K_{dz} \frac{de_z}{dt} \end{aligned} \tag{8}$$

3. Modified tentacle algorithm for collision avoidance

3.1. Basic tentacle algorithm

The primary purpose of this algorithm is to let the robot move within an unknown environment in a way similar to how a beetle crawls around and uses its antennae to avoid obstacles. Indeed, the basic idea consists of using a set of virtual antennae called tentacles that probe an ego-centred occupancy grid.

According to Ref.[25], the tentacle algorithm for an intelligent vehicle can be implemented as follows:

3.1.1 Occupancy grid. Based on the information from a sensor, the environment around an intelligent vehicle can be described through a binary image as Fig 2A.The whole environment is divided into 512×512 pixel points, each of which is a cell covering a small ground patch of 25*cm*×25*cm*, the black points showing obstacles.

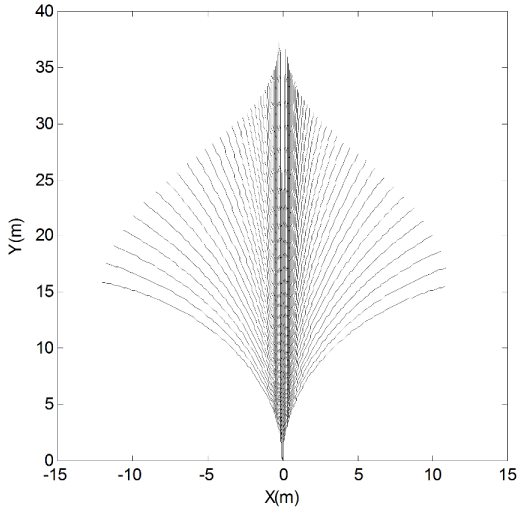
3.1.2 Tentacle structure. Each tentacle is a potential path and 16 sets of tentacles are used. Each set contains 81 tentacles corresponding to a specific velocity of the vehicle, which is divided into 16 speed sets form 0 to10m/s. All tentacles are represented in the local coordinate system of the vehicle, start at the vehicle's center of gravity and use the shape of circular arcs. The shape of 16×81 tentacles are designed to ensure an occupancy grid can be divided averagely.

The radius r_k of the $k-th$ tentacle in a set is given by:

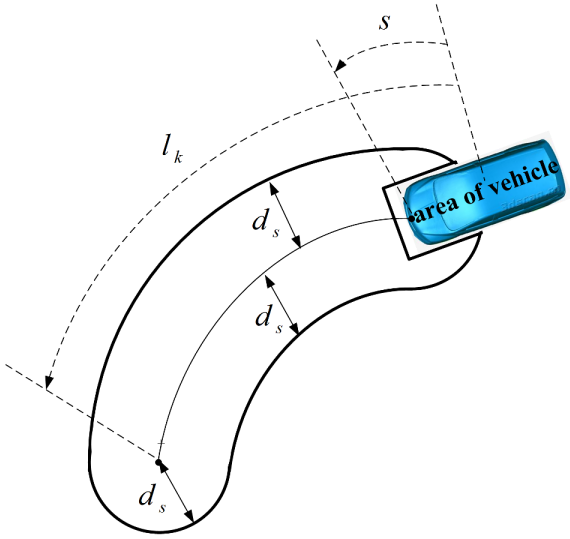
$$r_k = \begin{cases} \rho^k R_j & k = 0, \ldots, 39 \\ \infty & k = 40 \\ -\rho^{k-41} R_j & k = 41, \ldots, 80 \end{cases} \quad (9)$$

where ρ is an exponential factor and R_j is the initial radius of speed, we set $j = 0,\ldots 15$.

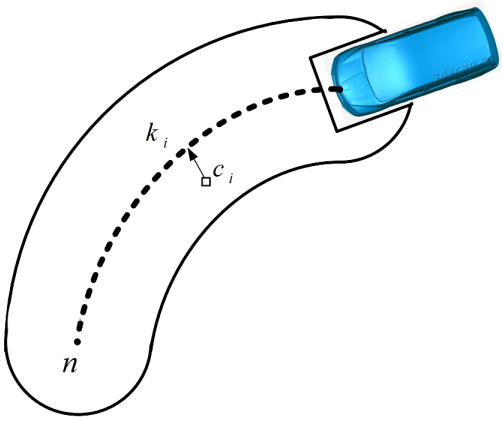
A Occupancy Grid



B 81 Tentacles in One Speed Set



C Support Area



D Obstacle Detection

Fig 2. Basic tentacle algorithm.

https://doi.org/10.1371/journal.pone.0182006.g002

The length of the $k-th$ tentacle in a set is given by:

$$l_k = \begin{cases} l + 20\sqrt{\frac{k}{40}} & k = 0, \ldots, 40 \\ l + 20\sqrt{\frac{k-40}{40}} & k = 41, \ldots, 80 \end{cases} \quad (10)$$

$$l = 8 + 33.5q^{1.2}$$

$$q = j/15$$

The velocity for speed set j is given by:

$$v_j = v_s + q^{1.2}(v_e - v_s) \quad (11)$$

where v_s is the minimum speed and v_e is the maximum speed. Fig 2B shows the 81 tentacles in one speed set.

3.1.3 Obstacle detection. The support area of a tentacle in the occupancy grid is used to determine whether a tentacle is drivable; its geometric definition is illustrated in Fig 2C. The **Rule A** of obstacle detection is as follows:

- Divide a tentacle to n sections as Fig 2D, and each point c_i in support area corresponds to a position k_i;
- Count all the black points which show the obstacles and record their positions;
- Use an array $v[n]$ to count the number of the black points in each section k_i;
- Use a sliding window to determine the position of the first obstacle. The window is initially placed at k_0 and successively slid to k_{n-1}.If the sum of binary values within this window exceeds a threshold n_o, an obstacle is detected and the position of the sliding window yields the distance l_o to the first obstacle.

3.1.4 Tentacle selection and execution. Set the crash distance l_c, which depends on the speed v, a deceleration a and a security distance l_s:

$$l_c = l_s + \frac{v^2}{2a} \quad (12)$$

In the end, one best tentacle is selected as the expected trajectory with the three functions: $v_{clearance}$,$v_{flatness}$,$v_{trajectory}$. The $v_{clearance}$ depends on the distance to the first obstacle l_o, the $v_{flatness}$ has the goal to prefer tentacles leading over smooth terrain, and the $v_{trajectory}$ pushes the vehicle towards following a given trajectory. The three function can be combined with different weight:

$$v_{combined} = a_1 v_{clearance} + a_2 v_{flatness} + a_3 v_{trajectory} \quad (13)$$

where a_1,a_2,a_3 are the weighting coefficients. The **Rule B** of tentacle selection is as follows:

- If there is no tentacle meet the condition $l_o > l_c$, the vehicle must decelerate along the tentacle with the maximum $v_{clearance}$ until the condition $l_o > l_c$ is meet again;
- If there are many tentacles meet the condition $l_o > l_c$, choose the tentacle with the minimum $v_{combined}$.

The flow chart of the basic tentacle algorithm is as Fig 3.

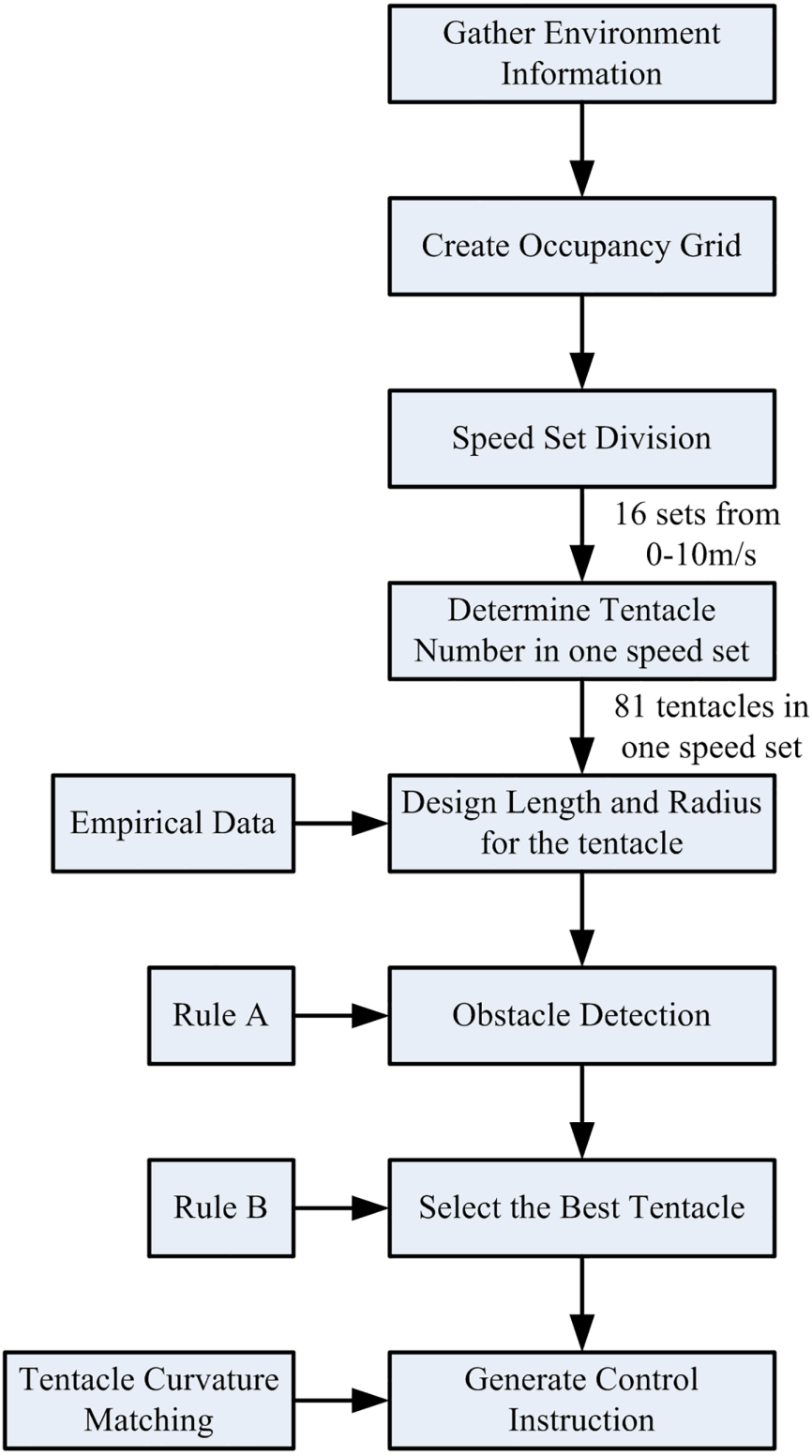


Fig 3. Basic tentacle algorithm.

https://doi.org/10.1371/journal.pone.0182006.g003

3.1.5 Problem in tentacle algorithm. **Problem 1:** It is noticed that all the parameters are related to wheel force and steering angle. This means that once a group of wheel forces and steering angles is certain, the trajectory of the vehicle may be fixed. Therefore, a group of wheel forces and steering angles refer to each tentacle. So the vehicle can be controlled to follow the selected tentacle with the two parameters.

Ref. [25] presents a method for executing the selected tentacle. This method pre-computes all the corresponding groups of wheel forces and steering angles through creating a steady state (as shown in Fig 4) according to each tentacle by curvature matching, and then stores

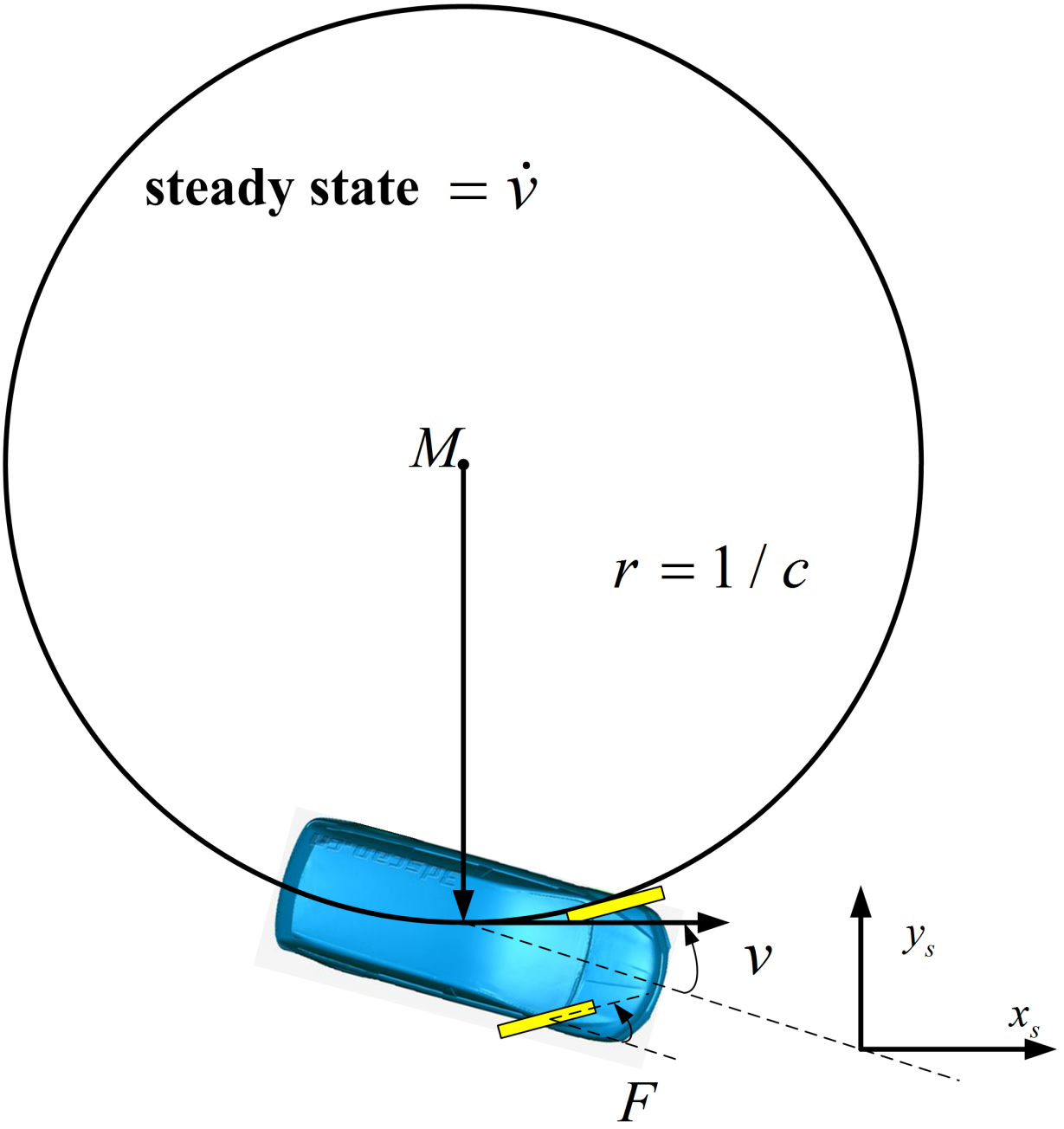


Fig 4. Steady state in tentacle algorithm.

https://doi.org/10.1371/journal.pone.0182006.g004

them so that once a tentacle is selected, the corresponding wheel force and steering angle can be acquired at the same time.

Although this method is correct in theory, some problems may cause the tentacle algorithm invalid because when the radius of a tentacle is too large, the matching between this tentacle and the corresponding steady state costs plenty of time and still worst a steady state cannot be achieved.

Problem 2: The basic tentacle algorithm in Ref. [25] is applied to an autonomous robot although this algorithm needs to be modified on a UAV platform. The velocity of the UAV is much greater than the autonomous robot. So the radius of each tentacle should be greater to match the velocity. Meanwhile, this change results that most tentacles gather in a small area, so that many neighbouring tentacles will provide the same curvature. Thus, the number of tentacles in one speed set should be reduced so as to enhance the ability of real time computation for UAVs.

3.2. Modified tentacle algorithm

Although the UAV formation is in 3D, its collision avoidance can be considered as two 2D problems. Similar as Fig 1, we can project all the UAVs and obstacles to *XOY* and *XOZ* in inertial coordinate system, and design our modified tentacle algorithm in the two planes separately. We use the *XOY* plane here to show how the modify tentacle algorithm solves the two problems above.

The tentacle algorithm for UAV is modified as follows:

3.2.1 Occupancy grid. Since the area of flight airspace is much greater than the ground, the whole environment is divided into 1000×1400 pixel points, each pixel point is a cell which covers a ground patch of $1m \times 1m$, the red area refers to the obstacles, as Fig 5A shows:

3.2.2 Tentacle structure. In the basic tentacle algorithm, once the radius of each tentacle is fixed, the corresponding control instruction only can be obtained by matching a steady state with the same curvature, but it is hard to match when the tentacle has large radius. For solving this problem, different from determining the radius firstly for each tentacle, we consider using an inverse derivation method, firstly determine the manoeuver capacity for an UAV.

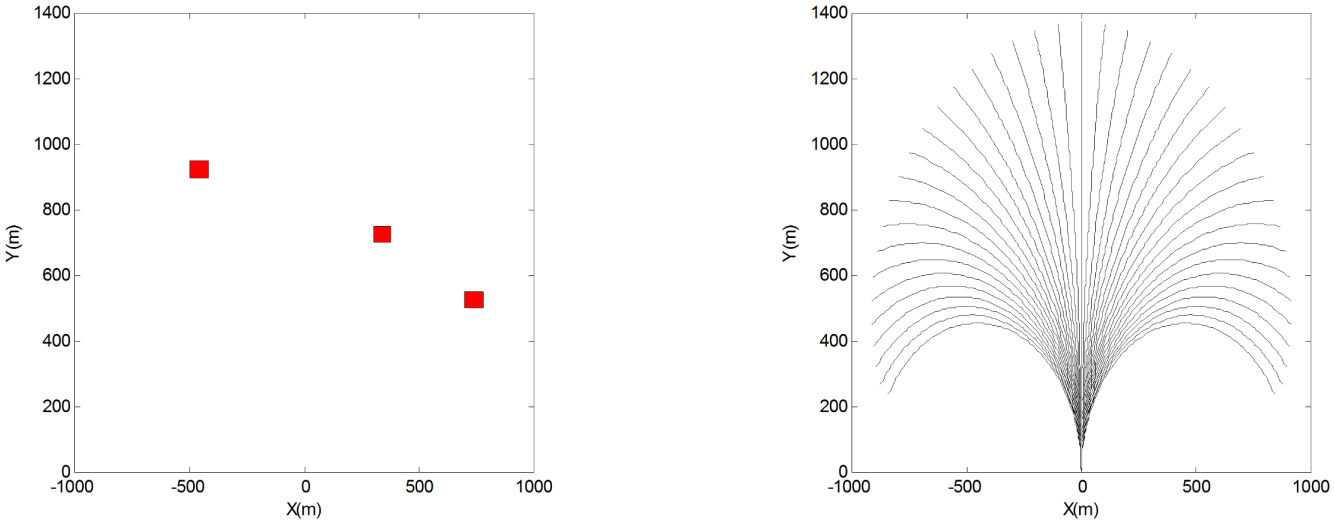


Fig 5. Modified occupancy grid and tentacles.

https://doi.org/10.1371/journal.pone.0182006.g005

According to the manoeuver capacity of an UAV, the velocity of UAV from $50m/s$ to $150m/s$ is divided into 10 sets, and in one speed set, the horizontal overload capacity $-2g$ to $2g$ is divided into 41 sets. Therefore there are 10×41 tentacles totally, we consider this division ensure an occupancy grid can be divided averagely.

According to Newton's second law, the radius r of a tentacle meets the following condition:

$$n_y = \frac{v_y^2}{r} \tag{14}$$

where n_y is the horizontal overload, v_y is the flight velocity in XOY plane. So the radius of each tentacle r_k in one speed set j can be computed by:

$$\begin{aligned} r_k &= \frac{v_j^2}{n_{yk}}, k = 0, \ldots, 40 \\ n_{yk} &= n_{y\min} + k\frac{n_{y\max} - n_{y\min}}{40} \\ v_j &= v_{\min} + j\frac{v_{\max} - v_{\min}}{10}, j = 0, \ldots, 9 \end{aligned} \tag{15}$$

It should be noted that, both of the control instruction and the radius of tentacle are transferred to the functions with tentacle number k. It means once a tentacle is selected, instead of curvature matching in basic algorithm, the corresponding control instruction can be computed directly through reading the variable k. Thus, the modification can avoid the curvature matching in basic tentacle algorithm, and the Problem 1 can be solved in theory.

Similarly, the length l_k can be given by:

$$l_k = \begin{cases} 400\frac{j}{9} + 200\sqrt{\frac{k}{20}} & k = 0, \ldots, 20 \\ 400\frac{j}{9} + 200\sqrt{\frac{40-k}{20}} & k = 21, \ldots, 40 \end{cases} \tag{16}$$

Fig 5B shows the 41 tentacles in one speed set. It can be seen that each tentacle divide the occupancy grid averagely. It proves the Problem 2 is solved in theory.

3.2.3 Obstacle detection. The obstacle detection is similar to the basic tentacle algorithm, which, therefore, can be used for the UAV. The obstacle can be detected through **Rule A**.

It should be noted that, for an UAV in its formation, all the other UAVs are considered to be the dynamic obstacles in the occupancy grid, so this detection method may eliminate impact on the manoeuver of multiple UAVs.

3.2.4 Tentacle selection. The crash distance l_c also is given by Eq (12), and the three function $v_{clearance}$,$v_{flatness}$,$v_{trajectory}$ in Eq (13) are given by the following equations respectively.

$v_{clearance}$ follows a normal distribution:

$$\begin{aligned} v_{clearance}(l_o) &= \begin{cases} 0 & \textit{if there is no obstacle} \\ e^{-c_1 l_o^2} & \textit{otherwise} \end{cases} \\ c_1 &= -\frac{\ln\frac{1}{2}}{l_{0.5}^2} \\ l_{0.5} &= 300 \end{aligned} \tag{17}$$

where $v_{clearance} \in [0,1]$ has been normalization and meets the following conditions:

$$\begin{aligned} v_{clearance}(0) &= 1 \\ v_{clearance}(l_{0.5}) &= 0.5 \\ v_{clearance}(\infty) &= 0 \end{aligned} \tag{18}$$

This means that the smaller $v_{clearance}$ shows the corresponding tentacle with a large distance from the next obstacle, so that the tentacles with small $v_{clearance}$ are preferable.

Because the UAV motion is projected to *XOY* plane, the effect of $v_{flatness}$ should be ignored:

$$v_{flatness} = 0 \tag{19}$$

Similar to $v_{clearance}$, $v_{trajectory}$ also follow a normal distribution:

$$\begin{aligned} v_{trajectory}(k) &= e^{-c_3(k-k_0)^2} - 1 \\ c_3 &= -\frac{\ln 2}{1600} \end{aligned} \tag{20}$$

where $v_{trajectory} \in [0,1]$ has been normalization and meet the following conditions:

$$\begin{aligned} v_{trajectory}(k_0 + 40) &= 1 \\ v_{trajectory}(0) &= 0 \end{aligned} \tag{21}$$

k_0 can be obtained through matching the current overload instruction.

Using the previous three new functions, the evaluation index $v_{combined}$ is calculated, then the best tentacle can be selected through **Rule B**, and we can compute the control instruction $[V_{yc}, \psi_c]$ by Eq (15) with the number of the best tentacle.

Similarly in the *XOZ* plane, another control instruction $[V_{zc}, \theta_c]$ can be obtained. So the 3D control instruction for an UAV is $[V_c, \psi_c, \theta_c]$, where $V_c = \sqrt{V_{yc}^2 + V_{zc}^2}$.

The flow chart of the modified tentacle algorithm is as Fig 6.

3.2.5 Problem solution in modified tentacle algorithm. **Problem 1:** Different from computing the steering command according to the radius of each tentacle with the basic tentacle algorithm, by using the inverse derivation, the modified tentacle algorithm divides the control instruction into some uniform sections, and then computes the radius of each tentacle with Newton's second law. It ensures each tentacle corresponds with only one overload instruction, so the curvature matching in basic algorithm become unnecessary. Thus, the data calculation problem(Problem 1) can be solved.

Problem 2: Because the speed sets and tentacles in one speed set are reduced and reconstructed, the influence of computation load in real time computation become insignificant. It ensures the modified tentacle algorithm meet the requirement of the real time computation for UAVs(Problem 2).

4. Experimental verification

4.1. Simulation environment

In order to verify the performance of the modified algorithm, a complex scenario with 3 obstacles and 5 UAVs is created in this paper.

The leader-follower formation model contains one leader and four followers. The initial states of each UAV are given in Table 1, where $\tau_v, \tau_\psi, \tau_\theta$ represent the time constants of the autopilot.

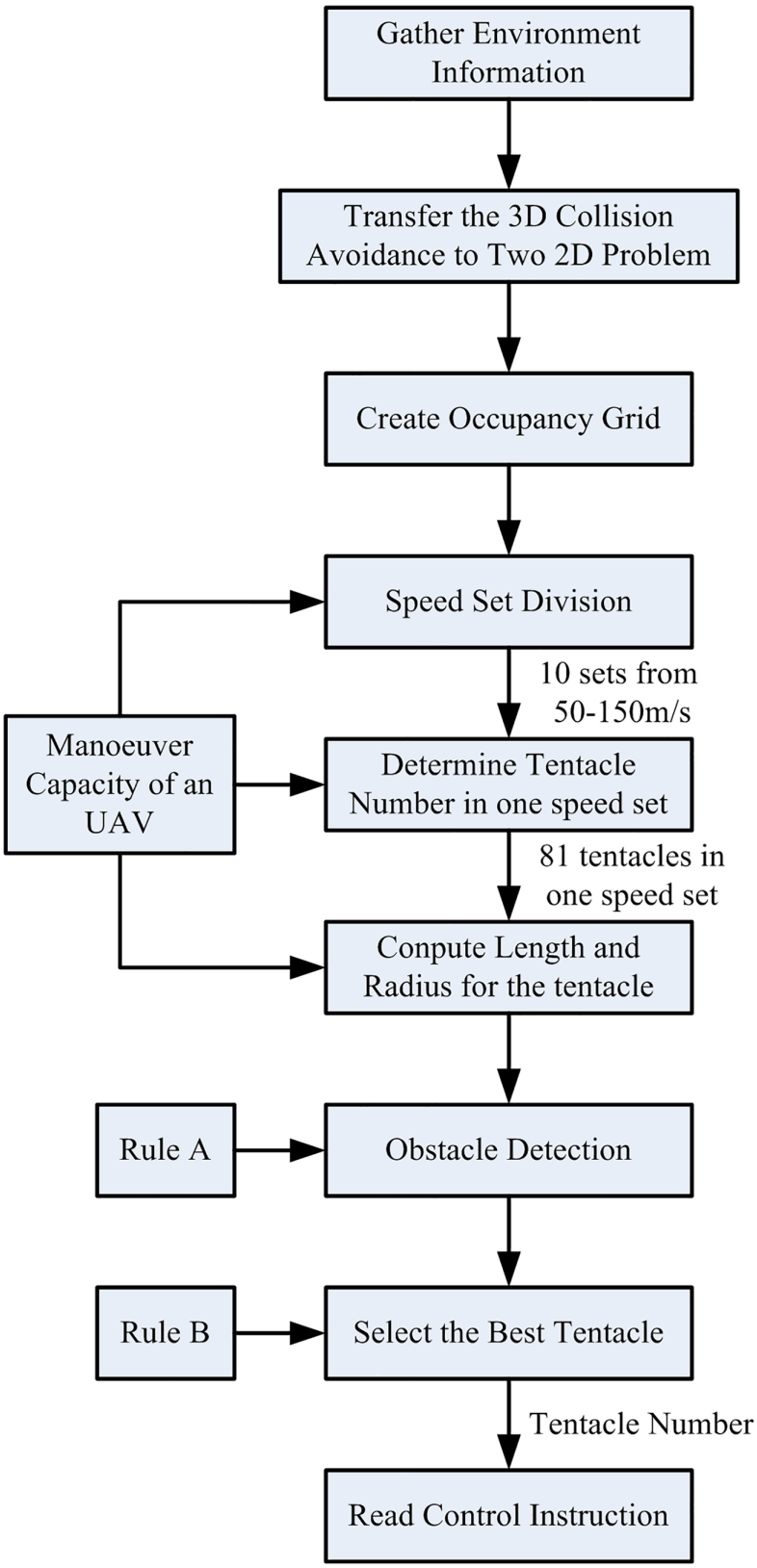


Fig 6. Modified tentacle algorithm.

https://doi.org/10.1371/journal.pone.0182006.g006

Table 1. UAV's initial states.

Parameters	Position(m)	Speed(m/s)	Heading angle(Degree)	Track angle(Degree)	τ_v	τ_ψ	τ_θ
Leader	(500,200,190)	100	0	0	5	1	1
Follower#1	(0,0,100)	100	0	0	5	1	1
Follower#2	(0,300,300)	100	0	0	5	1	1
Follower#3	(0,-200,50)	100	0	0	5	1	1
Follower#4	(0,700,350)	100	0	0	5	1	1

https://doi.org/10.1371/journal.pone.0182006.t001

Table 2. Formation flight requirements.

Parameters	Relative Distance in the X Direction (m)	Relative Distance in the Y Direction (m)	Relative Distance in the Z Direction (m)
Leader	0	0	0
Follower#1	400	150	100
Follower#2	400	-150	-100
Follower#3	400	300	120
Follower#4	400	-300	-130

https://doi.org/10.1371/journal.pone.0182006.t002

Table 3. Obstacle parameters.

Parameters	Position(m)	Radius(m)	Safe distance d_s in tentacle(m)
Obstacle#1	(2000,100,100)	80	30
Obstacle#2	(4000,200,250)	80	30
Obstacle#3	(2000,500,330)	80	30

https://doi.org/10.1371/journal.pone.0182006.t003

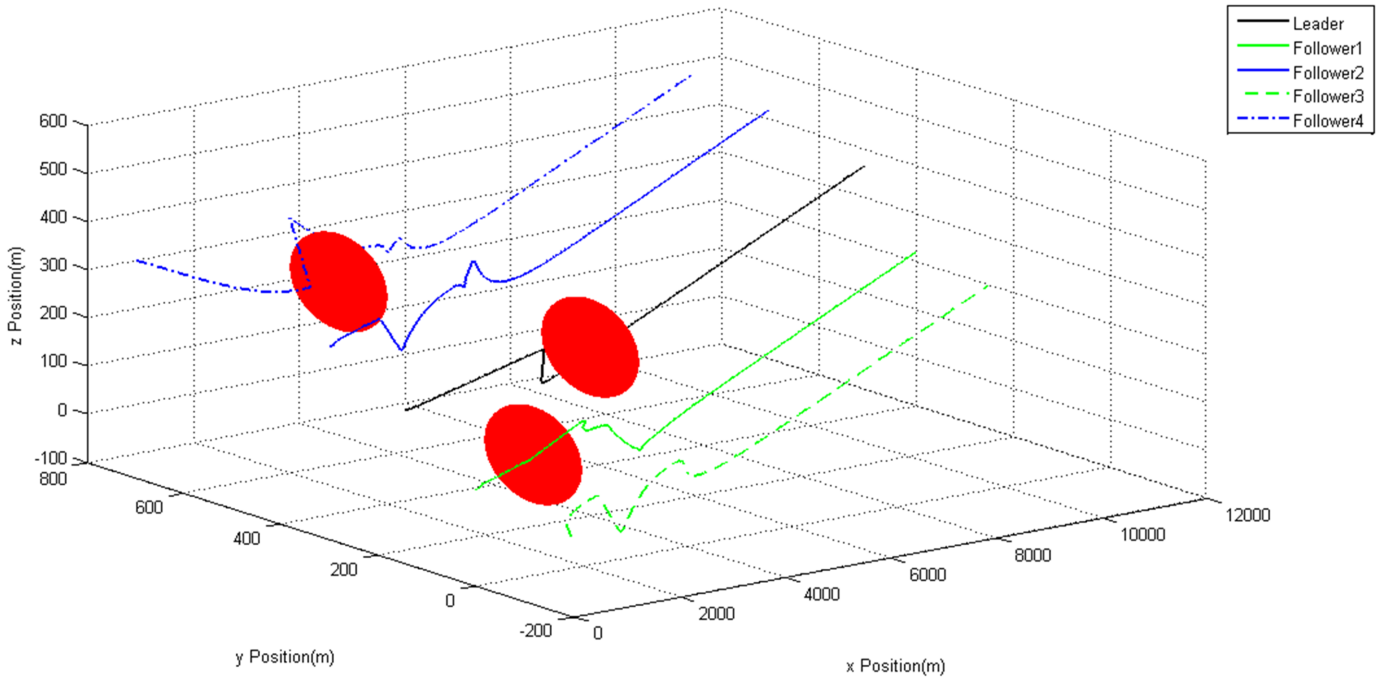


Fig 7. 3D Trajectories of UAVs.

https://doi.org/10.1371/journal.pone.0182006.g007

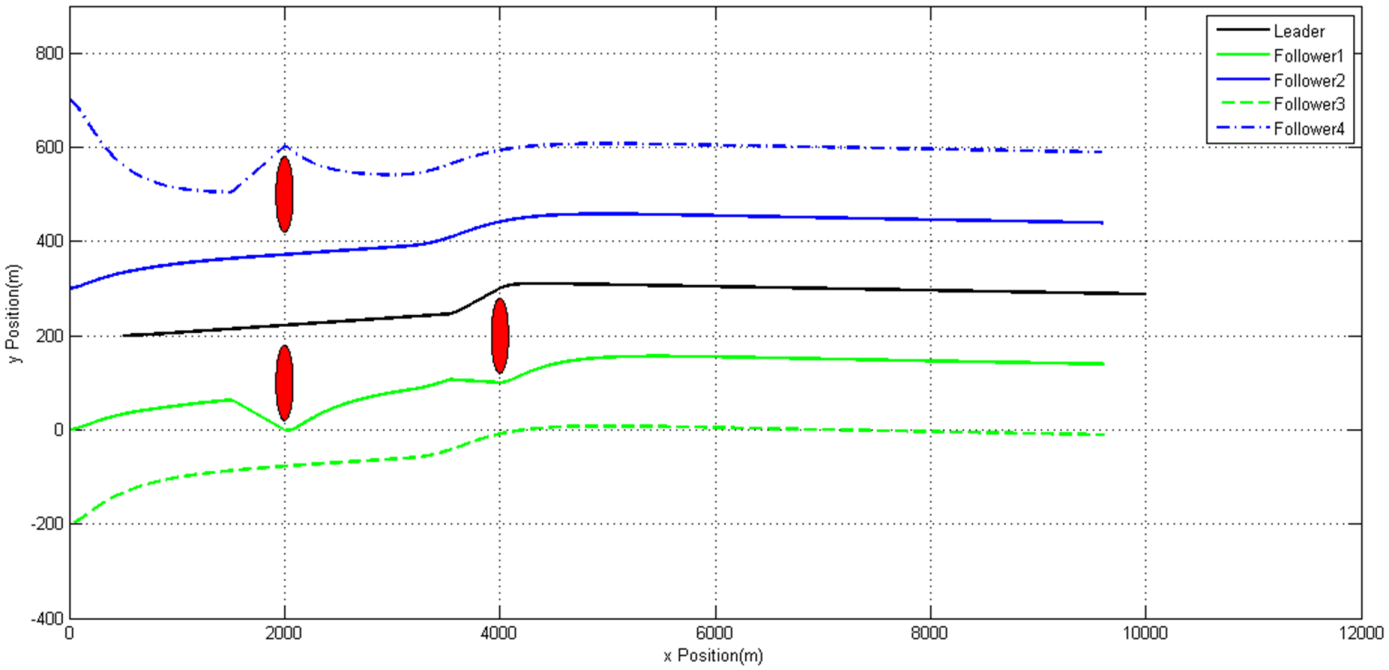


Fig 8. Trajectories of UAVs in *xoy* plane.

https://doi.org/10.1371/journal.pone.0182006.g008

Table 2 shows the relative distance in three dimensions between each follower and leader:

Table 3 gives the information on obstacles during the simulation, where d_s represents the requirements for safe distances in each tentacle.

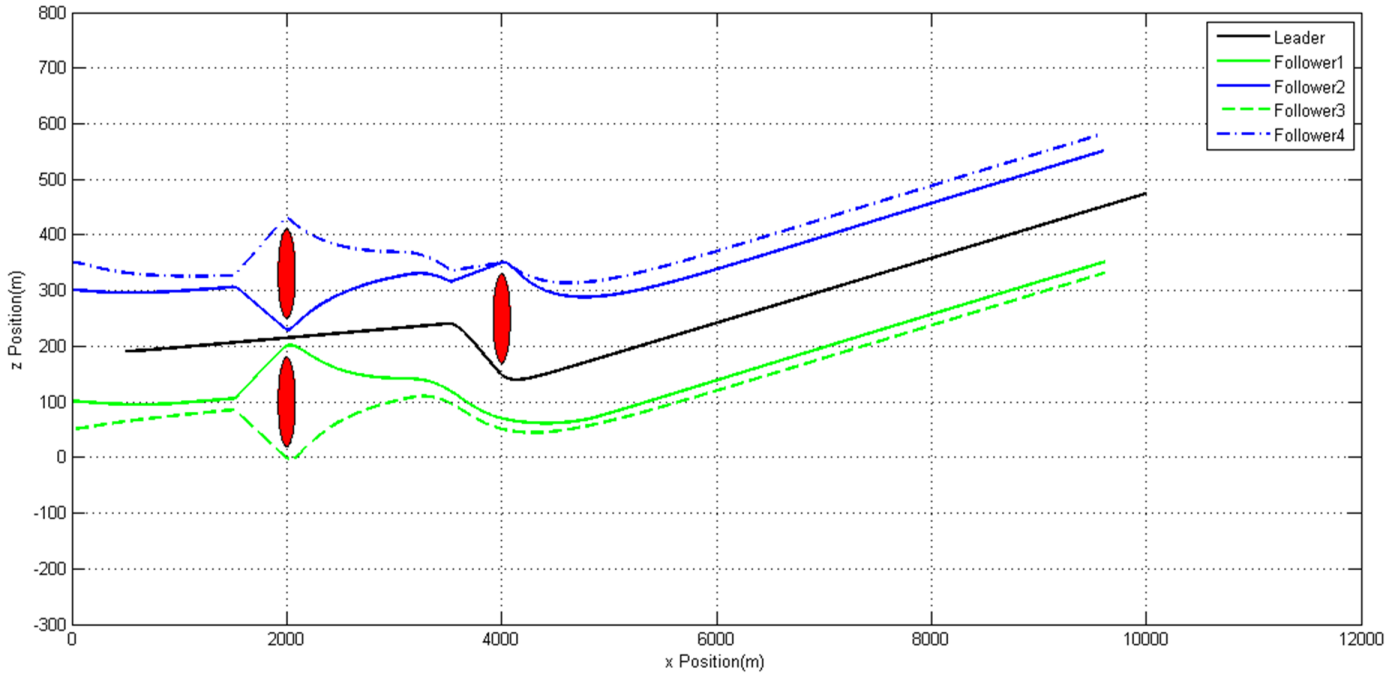


Fig 9. Trajectories of UAVs in *xoz* plane.

https://doi.org/10.1371/journal.pone.0182006.g009

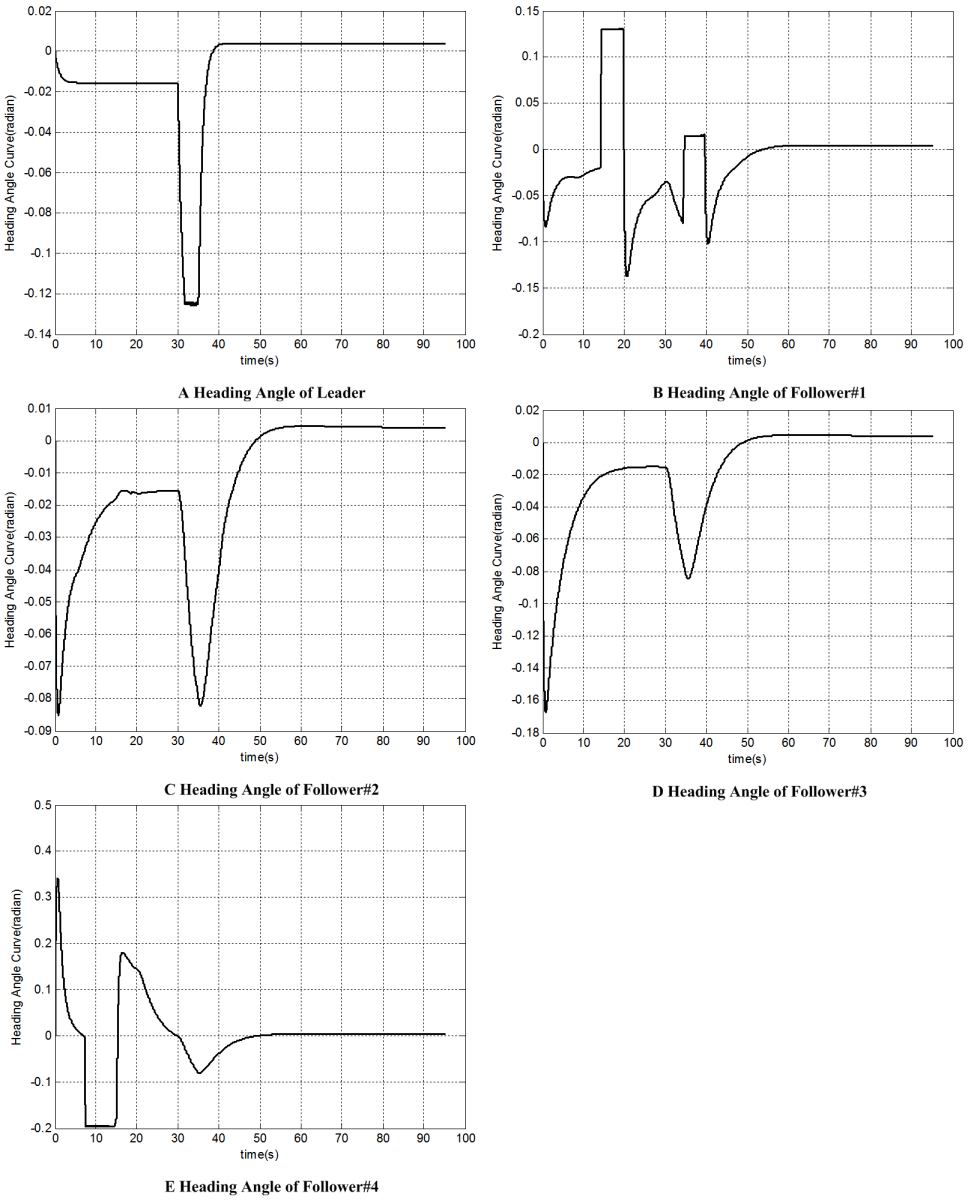


Fig 10. Heading angle histories of each UAV.

https://doi.org/10.1371/journal.pone.0182006.g010

The simulation will be terminated when the leader UAV reach 10000m in x direction. The simulation step is 0.001s.

4.2. Simulation results and analysis

Figs 7–9 show the trajectories of each UAV. It can be observed that all the static obstacles are successfully avoided. Though sometimes there is no static obstacle around an UAV, this UAV will manoeuver in order to avoid other UAVs which getting close to it. These prove our modified algorithm can avoid static and dynamic obstacles effectively.

Figs 10 and 11 give the heading angle and track angle of each UAV; Fig 12 shows the distance from each follower to its formation position. Both of them show that each follower can recover its original formation position after the avoidance.

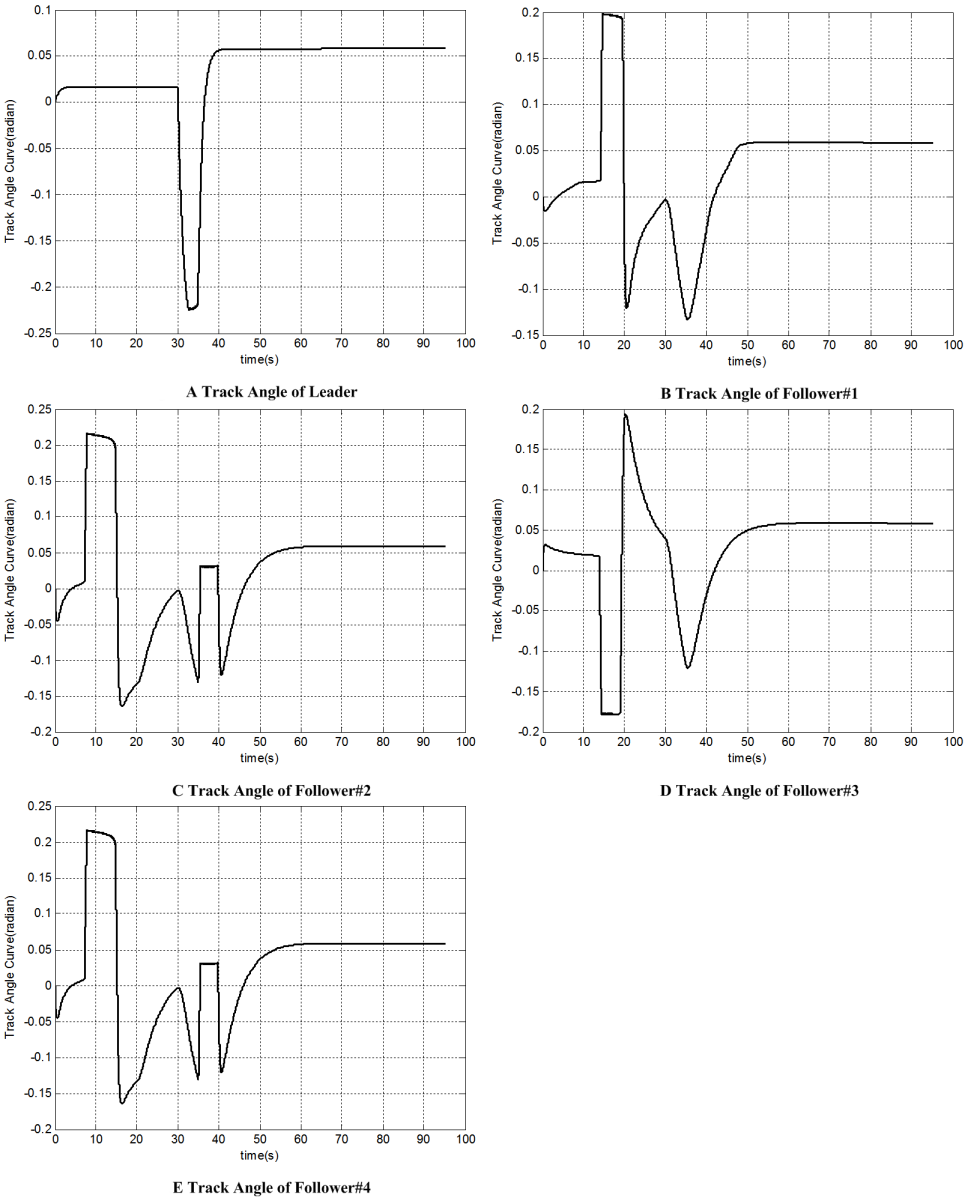


Fig 11. Track angle histories of each UAV.

https://doi.org/10.1371/journal.pone.0182006.g011

Meanwhile, each wave crest or wave trough in Figs 10 and 11 shows a large-radius manoeuver. The successful multiple large-radius manoeuver of 5 UAVs prove our inverse derivation solution to the data calculation problem(Problem 1) is credible.

Tables 4 and 5 show the minimum distances among all UAVs and the minimum distances from each UAV to obstacles. The results show that all UAVs can maintain their relative distances farther than the safe distances from other followers and obstacles. These prove that the collision avoidance method proposed in this paper has a high performance.

Meanwhile, the high-performance collision avoidance method proves that our reduction and reconstruction solution to the application problem (Problem 2) can be solved, and the modified tentacle algorithm can be successfully applied into the collision avoidance of multiple high-speed UAVs.

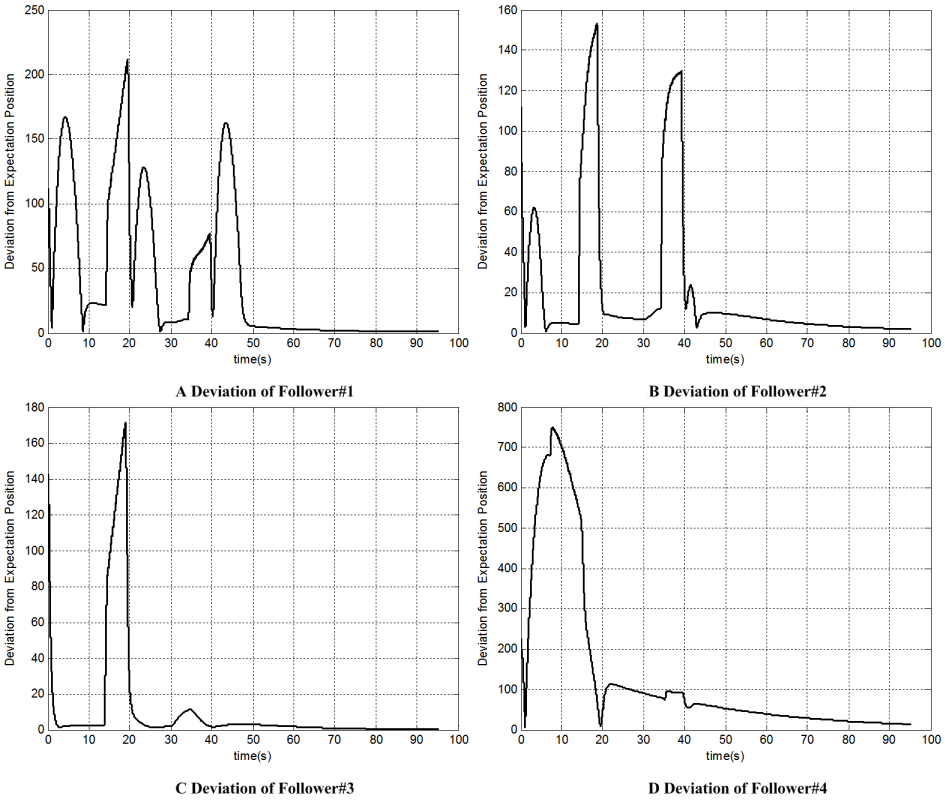


Fig 12. Deviation from expectation position of each UAV.

https://doi.org/10.1371/journal.pone.0182006.g012

Table 4. The minimum distances among all UAVs.

Parameters	Leader	Follower#1	Follower#2	Follower#3	Follower#4
Leader	0.0m	297.6m	364.4m	512.0m	348.8m
Follower#1	297.6m	0.0m	357.5m	111.4m	505.8m
Follower#2	364.4m	357.5m	0.0m	500.0m	153.5m
Follower#3	512.0m	111.4m	500.0m	0.0m	650.3m
Follower#4	348.8m	505.8m	153.5m	650.3m	0.0m

https://doi.org/10.1371/journal.pone.0182006.t004

Table 5. The minimum distances from each UAV to obstacles.

Parameters	Leader	Follower#1	Follower#2	Follower#3	Follower#4
Obstacle#1	87.4m	59.7m	221.0m	123.9m	499.1m
Obstacle#2	221.0m	435.5m	82.7m	584.1m	58.7m
Obstacle#3	59.5m	125.5m	180.9m	208.2m	325.5m

https://doi.org/10.1371/journal.pone.0182006.t005

With the modified tentacle algorithm and other classical collision avoidance methods, Table 6 compares the computational load of path planning and environment modelling in 5 continuous time sets. The comparison results show that the computational load of our algorithm is much less than others, reducing about 50%, thereby proving that our algorithm can effectively compute in real time.

Table 6. The computation load in 5 continuous time sets.

Parameters	Set1	Set2	Set3	Set4	Set5
Our algorithm	0.023755s	0.023211s	0.021982s	0.021497s	0.023328s
Artificial potential field method	0.046742s	0.046635s	0.042276s	0.042587s	0.044569s
Grid method	0.050287s	0.049872s	0.049247s	0.048973s	0.051846s

https://doi.org/10.1371/journal.pone.0182006.t006

The modified tentacle algorithm are validated in two other scenarios, and the corresponding simulation results are shown in S1 File.

5. Conclusion

This paper proposes the modified tentacle algorithm for the formation flight and collision avoidance of multiple UAVs. It concludes that the two problems for applying the conventional tentacle algorithm to the high-speed UAV in unstructured environments: (1) the data calculation problem and (2) the application problem. To solve Problem 1, it modified the tentacle algorithm to rapidly match the radius of each tentacle and the steering command by using the inverse derivation. To solve Problem2, it reduced and reconstructed the speed sets and tentacles in one speed set. By converting global path optimization into local searching, the best tentacle is selected to quickly obtain the UAV collision avoidance path. Consequently, the overall formation flight and collision avoidance mission are guaranteed simultaneously, so that the high-speed UAV formation can avoid collision in unstructured environments through applying our collision avoidance method. The simulation results in two scenarios do indeed confirm the feasibility and effectiveness of the method. The results also show that the multiple UAVs can cope with other collision threats coming from other UAVs in the formation and unknown obstacles. Multiple-UAV formation flight and collision avoidance can be achieved by data calculation at every moment. Therefore, the collision avoidance method can effectively compute in real time the collision avoidance of multiple high-speed UAVs under unstructured environments.

Supporting information

S1 File. Supporting information.
(PDF)

Acknowledgments

Thanks are due to Zhengxi Song for valuable discussion.

Author Contributions

Conceptualization: MHZ.

Data curation: MHZ.

Formal analysis: MHZ.

Funding acquisition: MHZ.

Investigation: MHZ.

Methodology: MHZ.

Project administration: MHZ.

Resources: MHZ.

Software: MHZ.

Supervision: MHZ.

Validation: MHZ.

Visualization: MHZ.

Writing – original draft: MHZ.

Writing – review & editing: MHZ.

References

1. Dong W., Farrell J., A.: 'Formation control of multiple under actuated surface vessels', IET Control Theory Appl.,2008, 2, (12), pp. 1077–1085
2. Shan J.: 'Six-degree-of-freedom synchronised adaptive learning control for spacecraft formation flying', IET Control Theory Appl., 2008, 2, (10), pp. 930–949
3. Qiu H., Duan H., B.: 'Receding horizon control for multiple UAV formation flight based on modified brain storm optimization', Nonlinear Dyn, 2014, 78, pp. 1973–1988
4. Xu J., Andrew N., S., Bower G., Kroo I.: 'Aircraft route optimization for formation flight', Journal of Aircraft, 2014, 51, (2), pp. 490–501
5. Brodecki M., Subbarao K.: 'Autonomous Formation Flight Control System Using In-Flight Sweet-Spot Estimation', Journal of Guidance, Control, and Dynamics, 2014, 38, (6), pp. 1083–1096
6. Seo J., Kim Y., Kim S., Tsourdos A.: 'Consensus-based reconfigurable controller design for unmanned aerial vehicle formation flight', Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering, 2012, 226, (7), pp. 817–829
7. Duan H., B., Liu S., Q.: 'Non-linear dual-mode receding horizon control for multiple unmanned air vehicles formation flight based on chaotic particle swarm optimisation', IET Control Theory Appl., 2010, 4, (11), pp. 2565–2578
8. Zhang L., Zhou Z., Zhang F,.: 'Mixed Integer Linear Programming for UAV Trajectory Planning Problem', ENGINEERING AND MANUFACTURING TECHNOLOGIES, 2014, 541–542, pp. 1473–1477
9. Ruchti, J., Senkbeil, R., Carroll, J., Dickinson, J., Holt, S.: 'UAV Collision Avoidance Using Artificial Potential Fields Technical Report #CSSE11-03', Technical Report, Auburn University, Auburn, AL, July, 2011
10. Yang X., Alvarez L. M., Bruggemann T.: 'A 3D Collision Avoidance Strategy for UAVs in a Non-Cooperative Environment', JOURNAL OF INTELLIGENT & ROBOTIC SYSTEMS, 2013, 70, (1–4), pp.315–327
11. Falkowski M., Jedrzejek C.: 'Formation control and obstacles avoidance for multi-agent systems based on position estimation', Chinese Control Conference, 2014, 5333, pp.1150–1155
12. Li M., W., Peng H., Zhong W. 'Optimal control of loose spacecraft formations near libration points with collision avoidance', Nonlinear Dynamics, 2015, pp. 1–21
13. Cao Z., Q., Tan M., Wang S., Fan Y.: 'The Optimization Research of Formation Control for Multiple Mobile Robots', Intelligent Control and Automation, 2002, 10, (14), pp. 1270–1274
14. Xia Y., Q., Na X., T., Sun Z., Q., Chen J.: 'Formation control and collision avoidance for multi-agent systems based on position estimation', ISA Transactions, 2016, 61, pp. 287–296 https://doi.org/10.1016/j.isatra.2015.12.010 PMID: 26786907
15. Zhen Z., Gao C., Zheng F., Jiang J.: 'Cooperative path replanning method for multiple unmanned aerial vehicles with obstacle collision avoidance under timing constraints', Proceedings of the Institution of Mechanical Engineers Part G Journal of Aerospace Engineering, 2015, 229, (10), pp. 1813–1823
16. Alonso-Mora J., Naegeli T., Siegwart R., Beardsley P.: 'Collision avoidance for aerial vehicles in multi-agent scenarios', Autonomous Robots, 2015, 39, (1), pp. 1–21
17. Shanmugavel M., Tsourdos A., White B. A. 'Collision avoidance and path planning of multiple UAVs using flyable paths in 3D', I2010 15TH INTERNATIONAL CONFERENCE ON METHODS AND MODELS IN AUTOMATION AND ROBOTICS, 2010,pp. 218–222
18. Gageik N., Benz P., Montenegro S.: 'Obstacle Detection and Collision Avoidance for a UAV With Complementary Low-Cost Sensors', Access IEEE 3, 2015, 2015

19. Paul T., Krogstad T. R., Gravdahl J. T.: 'Modelling of UAV formation flight using 3D potential field', SIMULATION MODELLING PRACTICE AND THEORY, 2008, 16, (9), pp. 1453–1462
20. Zhu L., Cheng X., Yuan F., G.: 'A 3D collision avoidance strategy for UAV with physical constraints', Measurement, 2016, 77, pp. 40–49
21. Liu C., Wang H., Yao P.: 'UAV Autonomous Collision Avoidance Method Based on Three-dimensional Dynamic Collision Region Model and Interfered Fluid Dynamical System', INTERNATIONAL CONFERENCE ON CONTROL AND AUTOMATION, 2016, pp. 263–269
22. Yao P., Wang H., Su Z.: 'Real-time path planning of unmanned aerial vehicle for target tracking and obstacle avoidance in complex dynamic environment', AEROSPACE SCIENCE AND TECHNOLOGY, 2015, 47, pp. 269–279
23. Tang J., Fan L., Lao S.: 'Collision Avoidance for Multi-UAV Based on Geometric Optimization Model in 3D Airspace', Arabian Journal for Science and Engineering, 2014, 39, (11), pp. 8409–8416
24. Seunghoon K., Hyunjin C., Youdan K.: 'Formation Flight and Collision Avoidance for Multiple UAVs usingConcept of Elastic Weighting Factor', International Journal of Aeronautical and Space Sciences, 2013, 14, (1), pp. 75–84
25. Hundelshausen F., Himmelsbach M., Hecker F., Muller A., Wuensche H., J.: 'Driving with Tentacles—integral structures of sensing and motion', International Journal of Field Robotics Research, 2008, pp. 393–440
26. Pachter M., D'Azzo J., J., Dargan J., L.: 'Automatic Formation Flight Control', Journal of Guidance, Control, and Dynamics, 1994, 17, (6), pp. 1380–1384
27. Cho H., Han C.: 'Effect of sideslip angle on the aerodynamic characteristics of a following aircraft in close formation flight', Journal of Mechanical Science and Technology, 2015, 29, (9), pp. 3691–3698
28. Rinaldi F., Chiesa S., Quagliotti F.: 'Linear quadratic control for quadrotors UAVs dynamics and formation flight', Journal of Intelligent & Robotic Systems, 2013, 70, (1–4), pp. 203–220
29. Ru C., Wei R., Wang Y., Che J.: 'Multi model predictive control approach for UAV formation flight', Mathematical Problems in Engineering, 2014, 2014, Article ID 835301, 11 pages
30. Duan H., B., Luo Q., N., Yu Y., X.: 'Trophallaxis network control approach to formation flight of multiple unmanned aerial vehicles', Science China Technological Sciences, 2013, 56, (5), pp. 1066–1074
31. Wei C., Guo J., Cui N.: 'Research on the Missile Formation Keeping Optimal Control forCooperative Engagement', Journal of Astronautics, 2010, 31, (4), pp. 1043–1050
32. Ambroziak L., Gosiewski Z.: 'Two stage switching control for autonomous formation flight of unmanned aerial vehicles', Aerospace Science and Technology, 2015, 46, pp. 221–226
33. Chao Z., Zhou S., L., Ming L., Zhang W., G.: 'UAV formation flight based on nonlinear model predictive control', Mathematical Problems in Engineering, 2012, 2012, Article ID 261367, 15 pages
34. Deng W., Wang X., M., Wang X., Y., Xiao Y., H.: 'Controller Design of UAVs Formation Keep and Change', Computer Simulation, 2011, 28, (10), pp. 73–77
35. Li W., Zhang Y.: 'PIDA Plus Inversion Flight Controller for Multiple Unmanned Aerial Vehicle Formation', Journal of System Simulation, 2009, 21, (19),pp.6221–6224
36. Wan J., Ai J., L.: 'Design and Simulation of Fuzzy Control System of UAV Formation Flight', Journal of System Simulation, 2009, 21, (13), pp. 4183–4185, 4189
37. Ma P., B., Ji J.: 'Three-dimensional Multi-missile Formation Control', ACTA Aeronautica et Astronautica Sinica, 2010, 31, (8), pp. 1660–1666