A Drone Formation Transformation Approach

Chenghao Jin, Bing Chen(✉), and Feng Hu

College of Computer Science and Technology,
Nanjing University of Aeronautics and Astronautics, Nanjing, China
{jinchenghao,cb_china,huf}@nuaa.edu.cn

Abstract. In the process of performing fixed-wing drone formations, it is usually necessary to perform a variety of formations according to mission requirements or environmental changes. However, performing such formation transformation during formation flight will face many technical challenges. In this paper, we first present a Six-Tuple State Coherence (STSC) model for fixed-wing drone formations, and based on this model, the definition of drone formation transformation is given. Moreover, a drone formation change algorithm (DFCA) is proposed. When a new formation is needed, the master node first adopts the centralized Hungarian algorithm to determine the location allocation scheme of the new formation, and then each node calculates and executes dubins paths distributedly to maintain the consistency of the formation yaw angle, and finally adjusts the speed of the nodes to ensure the formation of STSC. The prototype system conforming to DFCA algorithm is implemented on OMNET++ platform, and numerous simulation experiments are carried out. The experimental results show the feasibility of the DFCA algorithm and show that it can control the drone formation transformation at a lower cost.

Keywords: Drone formation · Formation transformation · Consistent state · OMNET++ platform

1 Introduction

Recently, enormous progress has been made in the field of cooperative control for multi-drone system. A formation composed of inexpensive small drones can replace expensive multi-functional large drones and be more efficient and reliable when executing missions. Formation in multi-drone system is critical to efficient execution of coordinated tasks such as surveillance [2,8], investigation [14], search and rescue [11], measuring [12], aerial photography [15]. When the number of drones is limited, the formation control can cover a larger continuous areas or shorten the time to execute the task. However, in the process of executing the above tasks, the drones often need to change the formation to adapt to the battlefield environment and mission requirements. [1] proposed a method to

X. B. Zhai et al. (Eds.): MLICOM 2019, LNICST 294, pp. 18–31, 2019.
https://doi.org/10.1007/978-3-030-32388-2_2

avoid obstacles by changing formation. [19] proposed a method of switching attack or defensive formations according to the battlefield environment.

The coordinated control of multi-drone formation has always been a hot issue for industry and academia. Master-follower is currently the most common method of formation control, but since it is a centralized method, once the master fails, the entire formation system will be paralyzed [5,10,18]. In [7], a distributed formation control method based on the master-follower model is proposed, in which each drone can respond to emergencies by switching the following, leading, and accelerating modes. [4,6] proposed a behavior-based formation control method. Each drone in formation divides its actions into collision avoidance, obstacle avoidance, target search and formation maintenance based on its perception of the external environment. The output of each drone is weighted by various behaviors. In fact, the above method must require that each drone in the formation be consistent in speed, heading, position, etc. In some studies, to simplify computation, the drone is abstracted into a freely moving particle [3,13,16,17], but this model is too idealistic, especially for fixed-wing drones. In the real world, the drone needs to move along a smooth trajectory of suitable curvature and cannot turn at any angle. Therefore, this is also a problem that must be considered when the formation changes. In [9], a novel distributed cascade robust feedback control approach is proposed for formation and reconfiguration control of a team of vertical takeoff and landing (VTOL) unmanned air vehicles (drones). But this method assumes that the drone can move and hover freely, so it is only suitable for quadrotor. But this method assumes that the drone can move and hover freely, so it is not suitable for fixed-wing drones. In [17], a B-spline-based formation control method is proposed to maintain the formation by ensuring that the spline parameters of each drone are consistent. This method supports the formation change. However this method ignore the constraints on the heading and speed of each drone in the formation. Although these drones can reach the formation at the current moment, if their headings and speeds are different, the formation will not be maintained at the next moment.

To address the above mentioned challenges, we propose a Six-Tuple State Coherence (STSC) model for the fixed-wing drone formation transformation problem, and based on the STSC model, the drone formation transformation problem is formulated as a two-process problem: the first process completes the location assignment from the old formation to the new formation; the second process is to program the route with time, speed and heading constraints for each drone during formation transformation. In addition, we propose a Drone formation change algorithm (DFCA) for this two-process problem. Firstly, Hungarian algorithm is used to solve the optimization problem of position allocation in formation transformation, which minimizes the cost of formation transformation. Then each drone independently solves the formation change route based on the dubins model. Our salient contributions are summarized as follows:

– We propose an STSC model to define the state of each drone in the formation after the formation transformation, and then formulates the drone formation transformation as a two-process problem.
– Based on the STSC model, the DFCA algorithm is proposed. When changing to the new formation, the master drone firstly uses the Hungarian algorithm to determine the location assignment scheme of the old formation to the new formation, and then the drones distributedly calculate and execute the dubins path to make the yaw angle of each drone consistent, next, adjust the speed of each drone to ensure the STSC.
– We validate our proposed algorithm with a drone formation and evaluate its performance by extensive simulation in OMNeT++ simulation environment. Moreover, our proposed DFCA algorithm was compared with an existing deterministic programming approach. Simulation results show that our proposed algorithm performs significantly better than the existing works on both formation transformation cost and communication cost.

The rest of paper is organized as follows. We present the system model and some related solution concepts in Sect. 2. The DFCA algorithm for drone formation is detailed in Sect. 3. Section 4 reports our performance evaluation results in which we compare our approach to an existing approach. Finally, we draw a conclusion in Sect. 5.

2 System Model

2.1 STSC Model

The drone is an autonomous agent, whose state can be defined as a six-tuple, $P(t, X, Y, Z, \theta, v)$, which is located at position $[X, Y, Z]^T$ at a time t, the yaw angle is θ (the angle between the fuselage and north) and the speed is v. For a drone that flies from the starting point $P_s(t_s, X_s, Y_s, Z_s, \theta_s, v_s)$ to the end point $P_t(t_t, X_t, Y_t, Z_t, \theta_t, v_t)$, the path plan will produce one or more path r connection points P_s and P_t, which can be expressed as follows:

$$P_s(t_1, X_s, Y_s, Z_s, \theta_s, v_s) \xrightarrow{r} P_t(t_2, X_t, Y_t, Z_t, \theta_t, v_t) \tag{1}$$

Extend multiple drones into a formation consisting of N drones, which is a collection of agents. One of the drones is the master, which is the logical center of the formation, mastering and controlling the formation and route; others are slaver, which can communicate with the master, receive the instructions of master and control themselves according to the instruction and their own situation. The starting point P_{si} and the target point P_{ti} of each drone in the formation are connected by a path r_i, and the formation track is a set satisfying the form of (2):

$$\bigcup_{i \in N} P_{si}(t_1, X_{si}, Y_{si}, Z_{si}, \theta_{si}, v_{si}) \xrightarrow{r} P_{ti}(t_2, X_{ti}, Y_{ti}, Z_{ti}, \theta_{ti} i, v_{ti}) \tag{2}$$

STSC is a necessary condition for multiple drones to form a formation. It means that the six-tuple $P(t, X, Y, Z, \theta, v)$ of each drone in the formation should be consistent, that is, at the same time t, the headings are the same, the speeds v are equal, and the relative positions of each drone are unchanged.

2.2 Definition of Formation

Assume that a reference path is planned for the formation before the mission, which is denoted by $R^{ref}(t)$. The reference path $R^{ref}(t)$ is a curve in the global coordinate system that is changed with time t. $R^{ref}(t_0) = [x(t_0), y(t_0), z(t_0)]^T \in R_G^3$ represents a way point in the route at t_0. We use the tangent coordinate system of the reference path $R^{ref}(t)$ to describe the relative position between the drones and the formation of the drones [17]. In particular, we define a tangent coordinate system using coordinate axes parallel to the following vectors:

$$\begin{aligned} T(t) &= \frac{\mathrm{d}R^{ref}}{\mathrm{d}t}, \\ N(t) &= T(t) \times (-g), \\ B(t) &= N(t) \times T(t), \end{aligned} \tag{3}$$

where, $T(t)$ is the tangent to the reference trajectory, g is the acceleration due to gravity, $N(t)$ is the normal to the plane containing the tangent and the vertical direction, and (t) is the bi-normal vector. We define the orthogonal rotation matrix $M(t) \in R^{(33)}$

$$M(t) = [\frac{T(t)}{\|T(t)\|}, \frac{N(t)}{\|N(t)\|}, \frac{B(t)}{\|B(t)\|}], \tag{4}$$

So that the transformation between the local tangent frame and the global frame is given by,

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = f_t\left(\begin{bmatrix} X \\ Y \\ Z \end{bmatrix}\right) = M(t)\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + R^{ref}(t), \tag{5}$$

then the drone formation can be defined as $F_i = \{t, \theta, v_i, r_{i\_1}, r_{i\_2}, ..., r_{i\_n}\}$, where $r_{i\_j} = [x_{i\_j}, y_{i\_j}, z_{i\_j}]^T$ is the relative position of drone j in formation F_i (indicated by the tangent coordinate system F). θ_i and v_i are the formation heading and formation speed respectively, and the heading of the formation is the tangential direction of the current position, $dR^{ref}(t)/dt$.

Since the necessary condition for the configuration of the drone formation is the STSC, the heading and speed of each drone in the formation are the same as the formation course and formation speed, and the relative position remains unchanged. Therefore, according to the formation model F_i, the six-tuple of each drone in the formation can be obtained by:

$$P_{i\_j}(t_{i\_j}, X_{i\_j}, Y_{i\_j}, Z_{i\_j}, \theta_{i\_j}, v_{i\_j}), j = 1, 2, ..., N$$

$$\begin{aligned}
t_{i\_j} &= t_i; \\
\theta_{i\_j} &= \theta_i = \frac{dR^{ref}(t_i)}{dt_i}; \\
v_{i\_j} &= v_i; \\
\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} &= f_{t\_i}\left(\begin{bmatrix} X_{i\_j} \\ Y_{i\_j} \\ Z_{i\_j} \end{bmatrix}\right)
\end{aligned} \tag{6}$$

(6) describes the conversion process, $F_i = \{P_{i\_1}, ..., P_{i\_n}\}$, from formation model to six-tuple set of all drones under STSC conditions.

2.3 Definition of Formation Transformation

During the execution of the mission, the drone formation may need to transform a variety of formations, and the STSC must be satisfied when the formation transformation is completed, as shown in Fig. 1. Assume that the transformation of the formation $F_1 \to F_2$ needs to be completed. The old formation is $F_1 = \{t_1, \theta_1, v_1, r_{1\_1}, r_{1\_2}, ..., r_{1\_n}\}$, and the new formation is $F_2 = \{t_2, \theta_2, v_2, r_{2\_1}, r_{2\_2}, ..., r_{2\_n}\}$. Through the Eq. (6), the six-tuple $f1$ from the old formation $F1$, $F1 = \{P_{1\_1}, P_{1\_2}, ..., P_{1\_n}\}$ and the six-tuple from the target formation $F2$, $F2$ $F2 = \{P_{2\_1}, P_{2\_2}, ..., P_{2\_n}\}$ can be obtained. Therefore, the formation transformation of the drone formation can be modeled as follows,

$$P_{1\_i}(t_1, X_{1\_i}, Y_{1\_i}, Z_{1\_i}, \theta_1, v_1) \overset{r_i, \phi_{ij}}{\to} P_{2\_i}(t_{2\_j}, X_{2\_j}, Y_{2\_j}, Z_{2\_j}, \theta_2, v_2) \tag{7}$$

ϕ is a position distribution matrix, $\phi_{ij} = 1$ indicates $P_{1\_i} \to P_{2\_j}$ (P_{1_i} is assigned to P_{2_j}), and r_i is a flight path from P_{1_i} to P_{2_j}. Therefore, the solution to Eq. 7 can be divided into the following two processes:

Step 1. Location assignment from old formation to new formation, i.e. solving assignment matrix ϕ;
Step 2. Calculation of flight path r_i for each drone.

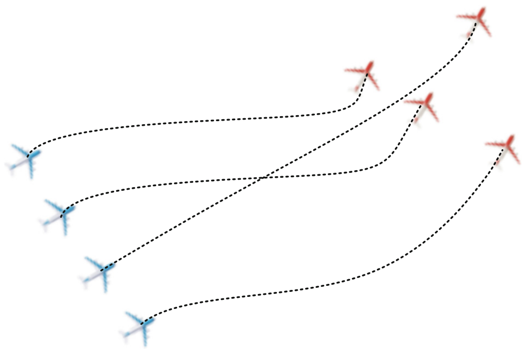

Fig. 1. An example of formation transformation.

3 Our Proposed Formation Transformation Algorithm for Drone Formation

Based on the above definition of drone formation, we propose a DFCA algorithm to calculate the distribution matrix ϕ and flight path r_i. The algorithm first solves the optimization problem to obtain the distribution matrix ϕ through the Hungarian algorithm, and then solves the route r_i of each drone that satisfies the STSC constraint based on the dubins model.

3.1 Location Assignment

According to the definition of the formation transformation, we need to solve the distribution matrix ϕ and the route r_i of each drone. Suppose the formation needs to be transformed from the formation $F1 = \{P_{1\_1}, P_{1\_2}, ..., P_{1\_n}\}$ to the formation $F2 = \{P_{2\_1}, P_{2\_2}, ..., P_{2\_n}\}$. Then, the first step of the formation transformation is to assign a position in $F2$ for each drone in $F1$, that is, to solve the distribution matrix ϕ. Before we solve the distribution matrix ϕ, a cost function is constructed.

$$C\left(P_i, P_j\right) = w_d \left\| [X_i, Y_i, Z_i]^T - [X_j, Y_j, Z_j]^T \right\|^2 + w_h \left\|\theta_i - \theta_j\right\|^2 + w_v \left\|v_i - v_j\right\|^2 \tag{8}$$

$C\left(P_i, P_j\right)$ represents the cost of the transformation of the six-tuple from P_i to P_j, which is the weighted sum of the squared differences, and w_d, w_h, w_v are the weights of the items. Our goal is to minimize the total cost of building an initial formation, which is an optimization issue as follows,

$$\min_{\phi} \sum_{i=1}^{n} \sum_{j=1}^{n} \phi_{ij} C\left(P_i, P_{init\_j}\right) \tag{9}$$

This optimization problem is solved using the Hungarian algorithm. The Hungarian algorithm employs a reduction process on the distance matrix, $D_{ij} = C\left(P_{1\_i}, P_{2\_j}\right)$. This reduction process involves minimizing each element of the distance matrix, D_{ij}, through row and column operations. These row and column operations involve adding and subtracting minimum row and column elements, resulting in entries where $D_{ij} = 1$. When this occurs in non-conflicting rows/columns (i.e. unique j for each i), the algorithm terminates, and the assignment matrix, ϕ_{ij}, can be determined. This is done by searching the reduced distance matrix for $D_{ij} = 0$, such that for each i there is a unique j. When this is the case, $\phi_{ij} = 1$, otherwise, $\phi_{ij} = 0$. It is a centralized algorithm that needs to be calculated on the master. The detailed solution process for ϕ is illustrated in Algorithm 1.

3.2 Path Programming

This section describes how to solve the flight path r_i for each drone formation transformation. This method is a distributed algorithm, and each drone is

Algorithm 1. Location Assignment

```
Input: Old formation F1, new formation F2.
Output: $\phi_{ij}$.
1: Get the Six-tuple of each drone in old formation $F1$:
2: for $P_{1\_i} \in F1$ do
3:     $\theta_{1\_i} = \theta_1$,
4:     $v_{1\_i} = v_1$,
5:     $[X_{1\_i}, Y_{1\_i}, Z_{1\_i}] = M(t_1)[X_{1\_j}, Y_{1\_j}, Z_{1\_j}] + R^{ref}(t_1)$,
6:     $P_{1\_i} \leftarrow [t_1, X_{1\_i}, Y_{1\_i}, Z_{1\_i}, \theta_{1\_i}, v_{1\_i}]$.
7: end for
8: Get the Six-tuple of each drone in new formation $F2$:
9: for $P_{2\_i} \in F2$ do
10:     $\theta_{2\_i} = \theta_2$,
11:     $v_{2\_i} = v_2$,
12:     $[X_{2\_i}, Y_{2\_i}, Z_{2\_i}] = M(t_2)[X_{2\_j}, Y_{2\_j}, Z_{2\_j}] + R^{ref}(t_2)$,
13:     $P_{2\_i} \leftarrow [t_2, X_{2\_i}, Y_{2\_i}, Z_{2\_i}, \theta_{2\_i}, v_{2\_i}]$.
14: end for
15: for $P_{1\_i} \in F1$ do
16:     for $P_{2\_i} \in F2$ do
17:         Get $C(P_{1\_i}, P_{2\_j})$ by (9).
18:     end for
19: end for
20: $\phi^* = arg\min_\phi \sum_{i=1}^{n} \sum_{j=1}^{n} \phi_{ij} C(P_{1\_i}, P_{2\_j})$.
21: return $\phi^*$.
```

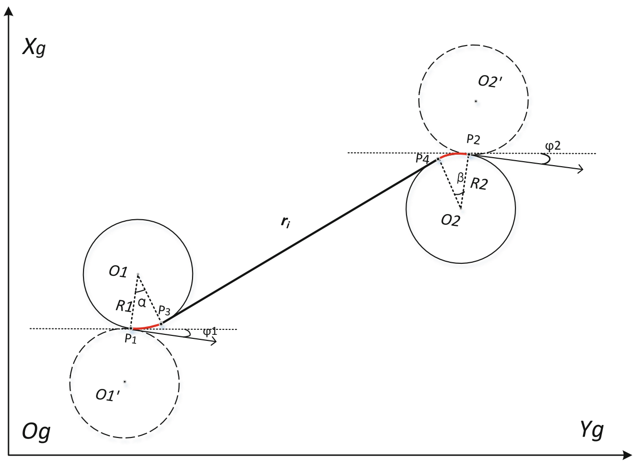

Fig. 2. An example of formation transformation.

executed independently. Assuming $\phi_{ij} = 1$, we will calculate the smoothing of a connection P_{1_i} and P_{2_j} for the drone using the dubins model as shown in Fig. 2. In order to ensure that the route is the shortest to reduce the cost of flight, we make the radius of the two turning circles $R_1 = R_2 = r_{min}$. r_{min} is the minimum turning radius of the drone, which is related to the flying speed and rolling angle of the drone:

$$r_{min} = \frac{v^2}{g * \tan \gamma_{max}} \tag{10}$$

where v is the flight speed of the drone, and γ_{max} is the maximum roll angle of the drone.

Since the transformation of $P_{1\_i} \rightarrow P_{2\_j}$ needs to satisfy the STSC constraint, the dubins model can only satisfy the transformation constraints of the four-tuple of position and velocity (ie, X, Y, Z, θ). Therefore, it is necessary to apply a speed control method based on the dubins model to ensure that each drone can reach the target state at the same time and at the same speed. In order to ensure the same speed before and after formation transformation of drone, that is $v_1 = v_2 = v$. The velocity of each drone needs to converge to V at the same time after the formation transformation process is completed along r_i. Since the formation transformation time $T = t_2 - t_1$, the drone completes the entire formation conversion process at a fixed speed v, and the time required is:

$$T_i = \frac{L_{r_i}}{v} \tag{11}$$

where L_{r_i} is the length of the path r_i. If $T_i < T$, the drone must decelerate to consume redundant time, and if $T_i > T$, the drone must accelerate the flight to compensate for this time difference, besides $T_i = T$ then the drone continues to maintain a uniform speed with v. Figure 3 shows the speed adjustment process of $T_i > T$, that is, the drone needs to accelerate the flight. Assuming that the acceleration during the acceleration of the drone is α_1, and the deceleration is α_2, then the relationship of the speed change is as follows.

$$\begin{aligned} &\frac{1}{2}\left(t^{''} - t^{'} + T\right) \cdot (v_{max} - v) + vT = L_{r_i}, \\ &v_{max} = v + \alpha_1 \cdot t^{'}, \\ &v = v_{max} + \alpha_2 \cdot \left(T - t^{''}\right) \end{aligned} \tag{12}$$

Through (12), the acceleration phase $\left[t_1, t_1 + t^{'}\right]$ and the deceleration phase $\left[t_1 + t^{'}, t_2\right]$ can be solved. The case of $T_i < T$ is similar here. The detailed process of solving the route r_i is in Algorithm 2.

4 Simulation

To verify the feasibility of our proposed DFCA algorithm, we first carry out the simulation experiment design, then carry out the simulation experiment, and finally analyze the experimental results.

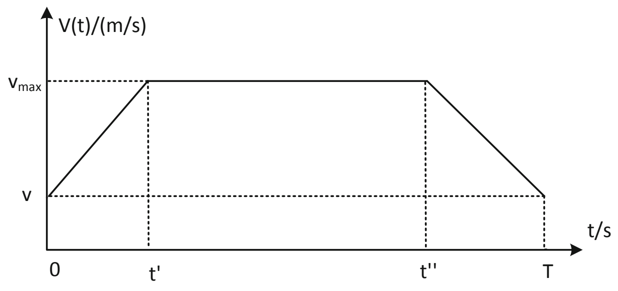

Fig. 3. An example of drone speed adjustment.

Algorithm 2. Path Programming

```
Input: Six-tuple $P_{1\_i}\left(t_1, X_{1\_i}, Y_{1\_i}, Z_{1\_i}, \theta_1, v_1\right)$, $P_{2\_i}\left(t_{2\_j}, X_{2\_j}, Y_{2\_j}, Z_{2\_j}, \theta_2, v_2\right)$, initialize time $T$.
Output: Dubins path $r_i$, two time for adjusting velocity $t^{'}$ and $t^{''}$.
1: Get the dubins route $r_i$ and $L_{r_i}$:
2: $r_i$=dubins.shortest_path($P_{1\_i}, P_{2\_j}$),
3: $L_{r_i} = r_i.length\left(\right)$.
4: Velocity adjustment:
5: $T_i = L_{r_i}/v_i$.
6: Slow down to consume redundant time:
7: if $T_i < T$ then
8:     Get $t^{'}$ and $t^{''}$ by solving the formula.
9: end if
10: Acceleration to compensate time:
11: if $T_i > T$ then
12:     Get $t^{'}$ and $t^{''}$ by solving the formula.
13: end if
14: No need to adjust the speed:
15: if $T_i = T$ then
16:     $t^{'} = 0$, $t^{''} = T$.
17: end if
18: return $r_i$, $t^{'}$ and $t^{''}$.
```

4.1 Simulation Platform

In the process of drone formation flight and formation transformation, the drones need to exchange information continuously, so the communication process between drones must be incorporated into the simulation platform to make the simulation more realistic. In view of the fact that there is no mature drone simulation platform, we choose OMNET++ as the basic environment of the simulation platform. OMNET++ is an open source discrete event simulator with modular, component-based C++ simulation library and framework. In view of the fact that there is no mature drone group simulation platform, we choose OMNET++ as the basic environment of the simulation platform. OMNET++

is an open source discrete event simulator with modular, component-based C++ simulation library and framework. In particular, the link layer model of the INET framework includes PPP, the Internet, and 802.11, as well as wireless and mobile emulation. At the same time, various types of mobile models are integrated in the INET framework, including deterministic movement models and random movement models. Users can build their own mobile models by expanding these mobile models. Based on the INET framework, we built the drone simulation platform shown in Fig. 4. The drone simulation platform includes three modules: communication module, processor module, and mobile module, in which the communication module utilizes the INET framework, and the physical layer mainly adopts the Radio model and the Medium model, the MAC layer uses the Ad hoc-based 802.11 model, the network layer uses the OLSR model, and the transport layer uses the UDP model. The processor module is abstracted as an embedded computer, which mainly runs the DFCA algorithm. Since the Assignment algorithm is a centralized algorithm, it runs only on the master, while the route planning algorithm is a distributed algorithm that runs on all drones. The mobile module is an extension and redesign module based on INET mobile model, which can simulate drone's point flight mode. The interaction between modules during the formation transformation includes:

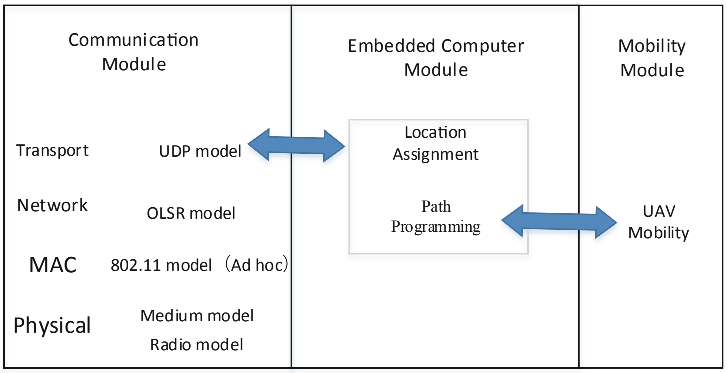

Fig. 4. Drone simulation platform.

Step 1. The processor module of the master sends the location allocation result obtained by the Assignment algorithm to each of the drones in the form of UDP datagram through communication module;

Step 2. The processor module of follower obtains the location allocation result sent by master through communication module, runs Route planning algorithm according to the result, and finally transmits the output result of route planning algorithm to mobile module to control drone movement.

Table 1. Formation parameters.

Parameter	Formation	
	Abreast	Diamond
t	0	35
θ	$\pi/3$	0
v	$20\,\text{m/s}$	0
r_1	$(0, 25)$	$(0, 25)$
r_2	$(0, 75)$	$(0, -25)$
r_3	$(0, -25)$	$(-40, 0)$
r_4	$(0, -75)$	$(40, 0)$

4.2 Simulation Results and Analysis

We designed a simulation experiment to verify the feasibility and performance of the DFCA algorithm. We designed a simulation experiment to verify the feasibility and performance of the DFCA algorithm. The formation consisting of four drones was transformed from the formation 'Abreast' to the formation 'Diamond' (See Fig. 5), and the detailed parameters of each formation are in the Table 1, t, θ, v are time, heading, and speed, respectively, besides r_i is the relative position of each drone.

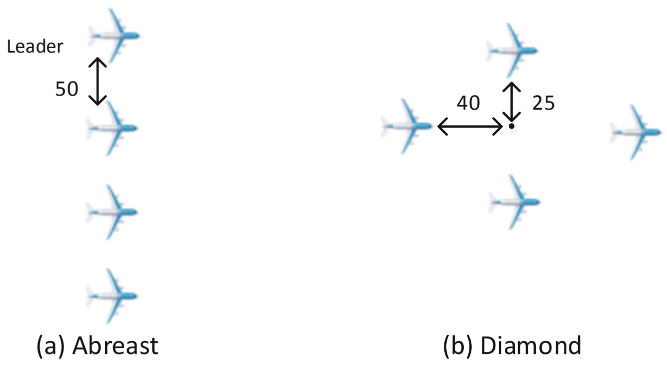

Fig. 5. Formation definitions of drone position offsets.

In the process of the drone formation changing from Abreast formation to Diamond formation, a path point is collected every 1 s, which forms the path map of drone formation transformation shown in Fig. 6. It can be seen from Fig. 6 that the final drone has reached the target formation and the heading has been adjusted to be consistent. Compared with the method in [17], the DFCA algorithm can complete the formation transformation in a shorter distance to

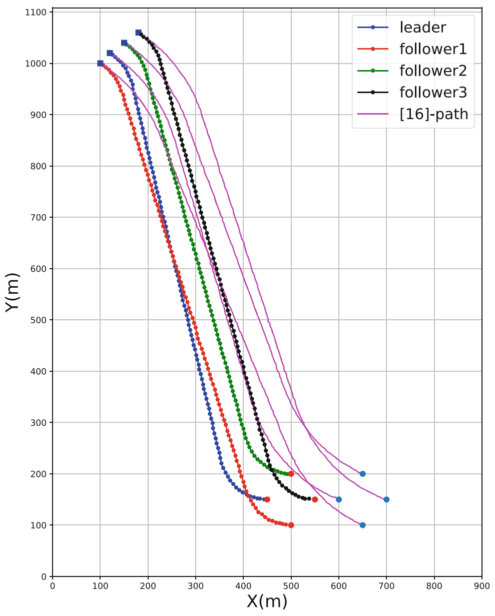

Fig. 6. Path map of drone formation.

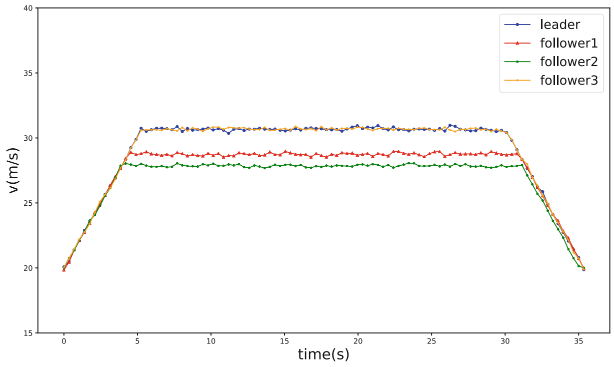

Fig. 7. Speed changing of drone formation.

save resources. Figure 7 shows the speed change diagram of the formation transformation process. In order to ensure that the speed of the formation before and after the formation transformation is constant, each drone has a process of first accelerating and then decelerating. It can be seen from Fig. 7 that the final drone speed converges to 20 m/s. The velocity transformation process can also be seen in Fig. 6, where the points are denser at the initial and end stages and sparse at the intermediate process points because the flight speeds are slower and the intermediate process speeds are faster in the initial and final phases. It can be concluded that the DFCA algorithm can satisfy the STSC constraint when each drone formation transformation is completed. Table 2 shows the time consumed to run DFCA algorithm in the process of drone formation transformation. All our DFCA algorithms are implemented based on C/C++, so they have high efficiency. Assignment algorithm runs only on master, and Path planning algorithm runs on all drones. It can be seen from the table that the running time of each algorithm is in milliseconds and will not affect the normal flight of drone.

Table 2. Running time of DFCA algorithm.

ID	Algorithm	
	Assignment algorithm	Path planning algorithm
Master	0.00098 s	0.0044 s
Follower 1	NULL	0.0033 s
Follower 2	NULL	0.0035 s
Follower 3	NULL	0.0036 s

5 Conclusion

In the process of performing missions, a drone formation usually change from one formation to another according to mission requirements or environmental changes. Aiming at this problem, we firstly presents a STSC model of fixed-wing drone formation, and gives the definition of formation transformation. Then a DFCA algorithm for STSC model is proposed. Finally, the simulation platform is presented. The simulation results show that the DFCA algorithm can complete the formation transformation of the drone formation at a lower cost.

Acknowledgments. This work was supported in part by the National Key Research and Development Program of China, under Grant 2017YFB0802303, in part by the National Natural Science Foundation of China, under Grant 61672283.

References

1. Alonso-Mora, J., Baker, S., Rus, D.: Multi-robot formation control and object transport in dynamic environments via constrained optimization. Int. J. Rob. Res. **36**(9), 1000–1021 (2017)
2. Beard, R.W., Lawton, J., Hadaegh, F.Y.: A coordination architecture for spacecraft formation control. IEEE Trans. Control Syst. Technol. **9**(6), 777–790 (2001)
3. Bogdanowicz, Z.R.: Flying swarm of drones over circulant digraph. IEEE Trans. Aerosp. Electron. Syst. **53**(6), 2662–2670 (2017)
4. Chen, J., Gan, M., Huang, J., Dou, L., Fang, H.: Formation control of multiple euler-lagrange systems via null-space-based behavioral control. Sci. China Inf. Sci. **59**(1), 1–11 (2016)

5. Defoort, M., Polyakov, A., Demesure, G., Djemai, M., Veluvolu, K.: Leader-follower fixed-time consensus for multi-agent systems with unknown non-linear inherent dynamics. IET Control Theory Appl. **9**(14), 2165–2170 (2015)
6. Dong, X., Yu, B., Shi, Z., Zhong, Y.: Time-varying formation control for unmanned aerial vehicles: theories and applications. IEEE Trans. Control Syst. Technol. **23**(1), 340–348 (2015)
7. Duan, H., Qiu, H.: Unmanned aerial vehicle distributed formation rotation control inspired by leader-follower reciprocation of migrant birds. IEEE Access **6**, 23431–23443 (2018)
8. Jaimes, A., Kota, S., Gomez, J.: An approach to surveillance an area using swarm of fixed wing and quad-rotor unmanned aerial vehicles UAV(s). In: Proceedings of IEEE International Conference on System of Systems Engineering, pp. 1–6, June 2008
9. Liao, F., Teo, R., Wang, J.L., Dong, X., Lin, F., Peng, K.: Distributed formation and reconfiguration control of VTOL UAVs. IEEE Trans. Control Syst. Technol. **25**(1), 270–277 (2017)
10. Loria, A., Dasdemir, J., Alvarez Jarquin, N.: Leader-follower formation and tracking control of mobile robots along straight paths. IEEE Trans. Control Syst. Technol. **24**(2), 727–732 (2016)
11. Mcgee, T.G.: Autonomous search and surveillance with small fixed wing aircraft. Ph.D. thesis, Berkeley, CA, USA (2006)
12. Paull, L., Thibault, C., Nagaty, A., Seto, M., Li, H.: Sensor-driven area coverage for an autonomous fixed-wing unmanned aerial vehicle. IEEE Trans. Cybern. **44**(9), 1605–1618 (2014)
13. Pinciroli, C., Beltrame, G.: Swarm-oriented programming of distributed robot networks. Computer **49**(12), 32–41 (2016)
14. Rafi, F., Khan, S., Shafiq, K., Shah, M.: Autonomous target following by unmanned aerial vehicles. In: Proceedings of SPIE 6230, Unmanned Systems Technology VIII, vol. 6230, May 2006
15. Tokekar, P., Vander Hook, J., Mulla, D., Isler, V.: Sensor planning for a symbiotic UAV and UGV system for precision agriculture. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 5321–5326, November 2013
16. Wang, Y., Sun, T., Rao, G., Li, D.: Formation tracking in sparse airborne networks. IEEE J. Sel. Areas Commun. **36**(9), 2000–2014 (2018)
17. Whitzer, M., et al.: In-flight formation control for a team of fixed-wing aerial vehicles. In: Proceedings of International Conference on Unmanned Aircraft Systems (ICUAS), pp. 372–380, June 2016
18. Xianfu, Z., Liu, L., Feng, G.: Leader–follower consensus of time-varying nonlinear multi-agent systems. Automatica **52**, 8–14 (2014)
19. Yu, D., Chen, C.L.P.: Automatic leader-follower persistent formation generation with minimum agent-movement in various switching topologies. IEEE Trans. Cybern. 1–13 (2018)