Robotics and Autonomous Systems 133 (2020) 103645

Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot

An iterative optimization approach for multi-robot pattern formation in obstacle environment☆

Fangfang Zhang, Tingting Wang, Qiyan Li, Jianbin Xin *

School of Electrical Engineering, Zhengzhou University, Zhengzhou, Henan, 450001, PR China

ARTICLE INFO

Article history:
Received 3 January 2020
Received in revised form 24 August 2020
Accepted 7 September 2020
Available online 14 September 2020

Keywords:
Multi-robot systems
Pattern formation
Convex quadratic programming
Collision avoidance

ABSTRACT

Pattern formation for multi-robot systems has received increasing attention in different scenarios. However, existing methods cannot efficiently optimize pattern formation in the obstacle environment. To address this limitation, this paper proposes a new planning method that assigns the optimal goals to the robots and iteratively computes collision-free paths to reach goal positions. Firstly, according to the random initial position of the group robot and the arbitrary shape, convex quadratic programming is used to minimize the distance to obtain the optimal pattern parameters under certain constraints. Secondly, the iterative controller plans the collision-free path of each robot to the goal considering a preferred velocity. Simulation results verified the effectiveness of the proposed methodology for scenarios of letter formation, in comparison to a commonly-used method.

1. Introduction

Recently, research on control of multi-robot systems has received extensive attention due to the development of wireless communication technology, sensor technology, embedded computing [1]. For multi-robot systems, pattern formation has great research significance in various scenarios such as aerospace, military, and disaster relief. The spacecraft, UAV, or mobile robots form specialized formations for area coverage, path search, and goal search [2].

Pattern formation also can be frequently found in commercial performances, and for instance, robot performances have become a highlight in many organized important events. UAVs or mobile robots equipped with LEDs of various colors display various patterns by changing formations [3,4]. Furthermore, the ideas of pattern formation also can be applied to the chip design by modeling the droplets as robots to coordinate the droplets automatically. There, the goal formations are determined to fit within the specified regions of the chip [5,6].

Most research on pattern formation is to study path planning of swarm robots from their initial positions to the fixed target positions that are predefined [7–10]. In the literature [7], the image is converted into grayscales and then is divided into 9 grids of 3*3. The OpenCV is used to find out the centroid of each grid and combining with manual adjustment to determine the goal point. The robot moves to the goal, according to rules based on the distribution of goals in each grid. The paper [8] uses the leader-centered formation graph, the following robots align with the virtual leader through a defined static vector to form the desired pattern. And then a variable parameter is designed to scale the exclusion vector field (RVF) [11] correctly to robustly solve the problem of formation control and avoid collision under the influence of external interference.

However, in practice, the goal positions are usually not predefined, and the swarm robots need to form different formations according to specific tasks [2]. Therefore, it is of great value to obtain the goal pattern for the non-predefined target positions. In this direction, some literature has studied optimizing goal pattern generation in an obstacle-free environment [12–14]. The pattern formation on optimizing goals generation is closely related to area coverage [15,16]. In these literatures , computational geometry methods, e.g., Tyson diagrams, point clouds, can be employed to optimize goal positions. The algorithm proposed in [12] realizes the pattern formation and animation display of large-scale robots. The pattern goal is optimized using the method proposed in Centroidal Voronoi Tessellations (CVTs) [17]. Its goal generation algorithm relies on choosing an excellent initial goal position, which not only leads to faster convergence but also gets better deployment. The Optimal Reciprocal Collision Avoidance (ORCA) algorithm [18] achieves collision-free paths to reach the goal positions. The paper [13] computes the optimal matching and pattern parameters based on the initial position of the group

☆ The authors would like to acknowledge the funding received from the National Natural Science Foundation of China (61603345, 61703372, 61773351), Outstanding Foreign Scientist Support Project in Henan Province, PR China (GZS2019008), Henan Province Young Talent Lift Project, PR China (2020hytp006) to conduct this research investigation.

* Corresponding author.
E-mail addresses: zhangfangfang@zzu.edu.cn (F. Zhang), j.xin@zzu.edu.cn (J. Xin).

https://doi.org/10.1016/j.robot.2020.103645
0921-8890/

robot and makes the initial position between the robots, the position between the goals at a certain distance to avoid the collision.

Although these works of literature are helpful to pattern formation for multi-robot systems, it is observed that these methods for pattern formation cannot efficiently optimize pattern formation in the obstacle environment. To address this limitation, this paper proposes an efficient algorithm for optimizing the pattern formation in the obstacle environment. The main contributions of this paper can be summarized as follows: First of all, in order to obtain the optimal assignment and the optimal pattern parameters, we employ convex quadratic programming and establish constraints based on the obstacle environment information to minimize the distance. Then, we design an iterative obstacle avoidance controller to drive the robot to the assigned goal without collision. Simulation results demonstrate the potential of the presented methodology against one commonly-used method.

The remainder of the paper is arranged as follows: Section 2 describes the problem and an overview of the proposed algorithm. Section 3 computes the optimal matching and goal pattern parameters to obtain the goal pattern in the obstacle environment. In Section 4, the iterative obstacle avoidance controller determines collision-free paths of robots reach the goal positions. Section 5 shows and analyzes the experimental simulation results of the letter pattern. Section 6 is the conclusion of this paper.

2. Problem description

The research on optimization for multi-robot pattern formation in obstacle environment is significant. The purpose of the paper is to propose an efficient algorithm to obtain the optimal matching and pattern parameters then iteratively control the robot to reach the goal without collision for optimizing the formation in the obstacle environment. In this section, the pattern formation research problem is introduced, and the overview of the proposed algorithm for optimizing the pattern formation in the obstacle environment is given.

In two-dimensional space, let there be n disk robots with radius R. The initial position coordinates of the group robot are represented by $\mathbf{P} = [\mathbf{p}_1; ...; \mathbf{p}_n]_{n\times 2}$, where $\mathbf{p}_i = [x_i, y_i]_{1\times 2}$ represents the plane position coordinates of the robot i, $i \in \{1, \ldots, n\}$. The velocity of the robot i is $\mathbf{v}_i = \left[v_{ix}, v_{iy}\right]_{1\times 2}$. The position coordinates of m static obstacles with radius $R_{\mathbf{o}}$ are represented by $\mathbf{O} = [\mathbf{o}_1; ...; \mathbf{o}_m]_{m\times 2}$, where $\mathbf{o}_j = \left[x_{\mathbf{o}_j}, y_{\mathbf{o}_j}\right]_{1\times 2}$ represents the plane coordinates of the obstacle j, $j \in \{1, \ldots, m\}$. $\mathbf{S} = [\mathbf{s}_1; ...; \mathbf{s}_n]_{n\times 2}$ describes arbitrary given shape (letters, circles, rectangles, etc.), where $\mathbf{s}_i = \left[x_{\mathbf{s}_i}, y_{\mathbf{s}_i}\right]_{1\times 2}$ represents the plane position coordinates of the point i in given pattern. The generated goal pattern is expressed as $\mathbf{Q}^* = [\mathbf{q}_1; ...; \mathbf{q}_n]$, $\mathbf{q}_i = \left[x_{\mathbf{q}_i}, y_{\mathbf{q}_i}\right]_{1\times 2}$ represents the plane position coordinates of the generated pattern goal i. The algorithm proposed in this paper mainly includes two steps, and the algorithm process decomposition diagram is shown in Fig. 1.

First of all, according to the initial position of the group robot and the shape of the pattern, the optimal matching and goal pattern parameters are computed in the obstacle environment to obtain the goal pattern $\mathbf{Q}^*$. Fig. 2 shows the goal pattern generation of the letter O and the optimal matching. The small circles represent the robots, the big circle labeled OB represents the static obstacle, the star-shaped points constitute the given shape $\mathbf{S}$, the dots constitute the goal pattern $\mathbf{Q}^*$, and the dashed lines represent the matching of the robot to the assigned goal.

Secondly, based on the goal pattern, under the control of the iterative obstacle avoidance controller, the robots reach the assigned goal position in real-time without collision. The control process consists of three steps, as follows:

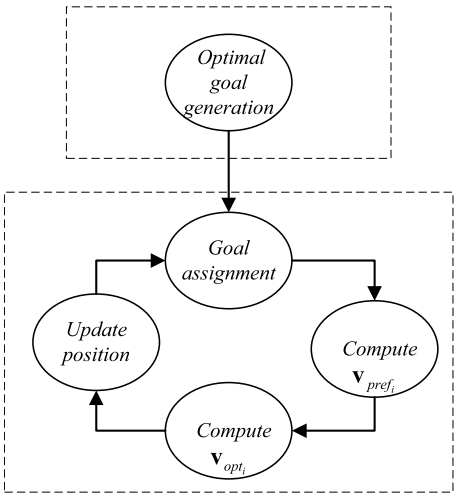


Fig. 1. Algorithm overview: optimal goal pattern generation and iterative obstacle avoidance controller.

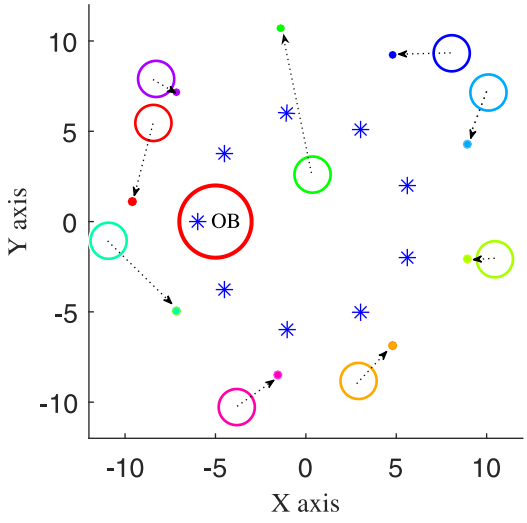


Fig. 2. The goal pattern and optimal matching example.

(1) In each iteration, the robot does not always move along the line to the assigned goal position in order to avoid collision with obstacles or other robots, which may cause them to move away from the initially assigned goal positions, thus reassign is needed in each iteration.
(2) Based on the assigned goal, we compute the preferred velocity $\mathbf{v}_{pref_i}$ [12] of the robot i toward the goal without considering the obstacles and other robots.
(3) We adopt the Reciprocal Velocity Obstacles (*RVO*) to find the optimal obstacle avoidance velocity $\mathbf{v}_{opt_i}$ outside the velocity obstacles and closest to the $\mathbf{v}_{pref_i}$ or select the velocity with the minimal penalty as $\mathbf{v}_{opt_i}$ [19], then update the robot position. Until all the robots reach the ε (is a small positive scalar close to 0) neighborhood of goal, the algorithm ends.

3. Optimal goal generation

The purpose of this section is to minimize the objective function (the sum of squared distance between the robots and the goal positions) to obtain the optimal matching and pattern parameters, that is to optimize the path length of the multi-robot pattern formation. Then, the goal pattern $\mathbf{Q}^*$ is obtained.

3.1. System model

The model for optimal assignment and goal pattern parameters is as follows:

$$\min C(\alpha, \mathbf{d}, \sigma) = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} \tag{1}$$

$$s.t. \begin{cases} \sum_{i=1}^{n} x_{ij} = 1 & j = 1, \ldots, n \\ \sum_{j=1}^{n} x_{ij} = 1 & i = 1, \ldots, n \end{cases} \tag{2}$$

where

$$x_{ij} = \begin{cases} 1, & \text{the robot } i \text{ assigned to the goal } j \\ 0, & \text{the robot } i\text{no assigned to the goal } j \end{cases}$$

$$c_{ij} = \|\mathbf{p}_i - \mathbf{q}_j\|^2 \tag{3}$$

$$\mathbf{q}_j = \alpha \mathbf{s}_j + \mathbf{d} \tag{4}$$

$C(\alpha, \mathbf{d}, \sigma) = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij}$ is the objective function; c_{ij} is the squared distance between the initial position $\mathbf{p}_i$ of the robot i and the position $\mathbf{q}_j$ of the goal j; $\sigma = (x_{ij})_{n\times n}$ is the assignment matrix, the mapping between the goal and the robot; $\mathbf{s}_j = [x_{\mathbf{s}_j}, y_{\mathbf{s}_j}]_{1\times 2}$ is the plane position coordinates of the point j in the given pattern; α and $\mathbf{d} = [\mathrm{d}_1, \mathrm{d}_2]_{1\times 2}$ are the goal pattern parameters, α is the scale parameter and $\alpha \in (0, \infty)$, $\mathbf{d}$ is the translation parameter.

Lemma 1. *According to the formula (2), the double sum formula $\sum_{i=1}^{n}\sum_{j=1}^{n} a_i x_{ij}$, where a_i is a constant only related to index i, can be written as*

$$\sum_{i=1}^{n}\sum_{j=1}^{n} a_i x_{ij} = \sum_{i=1}^{n} a_i \tag{5}$$

Proof. Since a_i is only related to i, we can obtain

$$\sum_{i=1}^{n}\sum_{j=1}^{n} a_i x_{ij} = \sum_{i=1}^{n}\left(a_i \sum_{j=1}^{n} x_{ij}\right) \tag{6}$$

where $\sum_{j=1}^{n} x_{ij} = 1, i = 1, \ldots, n$ can be obtained based on formula (2), then the formula $\sum_{i=1}^{n}\sum_{j=1}^{n} a_i x_{ij} = \sum_{i=1}^{n} a_i$. □

Lemma 2. *According to the formula (2), the double sum formula $\sum_{i=1}^{n}\sum_{j=1}^{n} b_j x_{ij}$, where b_j is a constant only related to index j, can be written as*

$$\sum_{i=1}^{n}\sum_{j=1}^{n} b_j x_{ij} = \sum_{j=1}^{n} b_j \tag{7}$$

Proof. Since a finite term double sum can swap the position of the sum symbol, we can get

$$\sum_{i=1}^{n}\sum_{j=1}^{n} b_j x_{ij} = \sum_{j=1}^{n}\sum_{i=1}^{n} b_j x_{ij} \tag{8}$$

Similar to Lemma 1, further

$$\begin{aligned} \sum_{i=1}^{n}\sum_{j=1}^{n} b_j x_{ij} &= \sum_{j=1}^{n}\sum_{i=1}^{n} b_j x_{ij} \\ &= \sum_{j=1}^{n}\left(b_j \sum_{i=1}^{n} x_{ij}\right) \\ &= \sum_{j=1}^{n} b_j \quad \square \end{aligned} \tag{9}$$

Lemma 3. *The objective function of variables α, $\mathbf{d}$ and σ, can be written as*

$$\begin{aligned} C(\alpha, \mathbf{d}, \sigma) =& M\alpha^2 - 2\alpha \sum_{i=1}^{n}\sum_{j=1}^{n} \mathbf{p}_i \mathbf{s}_j^T x_{ij} \\ &+ 2\alpha \mathbf{s}\mathbf{d}^T - 2\mathbf{p}\mathbf{d}^T + n\mathbf{d}\mathbf{d}^T + N \end{aligned} \tag{10}$$

where M, N, $\mathbf{s}$, $\mathbf{p}$ are independent of α, $\mathbf{d}$, σ.

Proof. Substituting formula (4) into formula (3), we can get

$$\begin{aligned} c_{ij} =& x_i^2 + y_i^2 + \alpha^2(x_{\mathbf{s}_j}^2 + y_{\mathbf{s}_j}^2) - 2\alpha(x_i x_{\mathbf{s}_j} + y_i y_{\mathbf{s}_j}) \\ &+ 2\alpha(x_{\mathbf{s}_j} d_1 + y_{\mathbf{s}_j} d_2) - 2(x_i d_1 + y_i d_2) + d_1^2 + d_2^2 \\ =& \alpha^2 \mathbf{s}_j \mathbf{s}_j^T - 2\alpha \mathbf{p}_i \mathbf{s}_j^T + 2\alpha \mathbf{s}_j \mathbf{d}^T - 2\mathbf{p}_i \mathbf{d}^T + \mathbf{d}\mathbf{d}^T + \mathbf{p}_i \mathbf{p}_i^T \end{aligned} \tag{11}$$

Taking formula (11) into the objective function $C(\alpha, \mathbf{d}, \sigma) = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij}$ and combining Lemmas 1, 2 we have

$$\begin{aligned} C(\alpha, \mathbf{d}, \sigma) =& \sum_{i=1}^{n}\sum_{j=1}^{n} \left(\alpha^2 \mathbf{s}_j \mathbf{s}_j^T x_{ij} - 2\alpha \mathbf{p}_i \mathbf{s}_j^T x_{ij} + 2\alpha \mathbf{s}_j \mathbf{d}^T x_{ij}\right. \\ &\left. - 2\mathbf{p}_i \mathbf{d}^T x_{ij} + \mathbf{d}\mathbf{d}^T x_{ij} + \mathbf{p}_i \mathbf{p}_i^T x_{ij}\right) \\ =& \alpha^2 \sum_{i=1}^{n}\sum_{j=1}^{n} \mathbf{s}_j \mathbf{s}_j^T x_{ij} - 2\alpha \sum_{i=1}^{n}\sum_{j=1}^{n} \mathbf{p}_i \mathbf{s}_j^T x_{ij} \\ &+ 2\alpha \sum_{i=1}^{n}\sum_{j=1}^{n} \mathbf{s}_j \mathbf{d}^T x_{ij} - 2\sum_{i=1}^{n}\sum_{j=1}^{n} \mathbf{p}_i \mathbf{d}^T x_{ij} \\ &+ \sum_{i=1}^{n}\sum_{j=1}^{n} \mathbf{d}\mathbf{d}^T x_{ij} + \sum_{i=1}^{n}\sum_{j=1}^{n} \mathbf{p}_i \mathbf{p}_i^T x_{ij} \\ =& \alpha^2 \sum_{j=1}^{n} \mathbf{s}_j \mathbf{s}_j^T - 2\alpha \sum_{i=1}^{n}\sum_{j=1}^{n} \mathbf{p}_i \mathbf{s}_j^T x_{ij} \\ &+ 2\alpha \sum_{j=1}^{n} \mathbf{s}_j \mathbf{d}^T - 2\sum_{i=1}^{n} \mathbf{p}_i \mathbf{d}^T \\ &+ \sum_{i=1}^{n} \mathbf{d}\mathbf{d}^T + \sum_{i=1}^{n} \mathbf{p}_i \mathbf{p}_i^T \\ =& M\alpha^2 - 2\alpha \sum_{i=1}^{n}\sum_{j=1}^{n} \mathbf{p}_i \mathbf{s}_j^T x_{ij} + 2\alpha \mathbf{s}\mathbf{d}^T \\ &- 2\mathbf{p}\mathbf{d}^T + n\mathbf{d}\mathbf{d}^T + N \end{aligned} \tag{12}$$

where $M = \sum_{j=1}^{n} \mathbf{s}_j \mathbf{s}_j^T$, $N = \sum_{i=1}^{n} \mathbf{p}_i \mathbf{p}_i^T$, $\mathbf{s} = \sum_{j=1}^{n} \mathbf{s}_j$, $\mathbf{p} = \sum_{i=1}^{n} \mathbf{p}_i$ are independent of α, $\mathbf{d}$, σ. □

Lemma 4. *Suppose the optimal assignment $\sigma^* = \mathrm{argmin}_\sigma \mathrm{C}(\alpha, \mathbf{d}, \sigma)$ at some values of $\alpha \in (0, \infty)$, $\mathbf{d}$, then it is the optimal assignment at any $\alpha \in (0, \infty)$, $\mathbf{d}$.*

Proof. Suppose σ^* is the optimal assignment at some values of $\alpha \in (0, \infty)$, $\mathbf{d}$, we have the inequality subjecting to $\mathrm{C}(\alpha, \mathbf{d}, \sigma^*) \leq$

$C(\alpha, \mathbf{d}, \sigma)$ (σ is arbitrary matching matrix at the same $\alpha \in (0, \infty)$, $\mathbf{d}$). According to formula (10), we can get

$$\begin{aligned} &M\alpha^2 - 2\alpha \sum_{i=1}^{n}\sum_{j=1}^{n} \mathbf{p}_i \mathbf{s}_j^T x_{ij}{}^* + 2\alpha \mathbf{s}\mathbf{d}^T - 2\mathbf{p}\mathbf{d}^T + n\mathbf{d}\mathbf{d}^T + N \\ \le &M\alpha^2 - 2\alpha \sum_{i=1}^{n}\sum_{j=1}^{n} \mathbf{p}_i \mathbf{s}_j^T x_{ij} + 2\alpha \mathbf{s}\mathbf{d}^T - 2\mathbf{p}\mathbf{d}^T + n\mathbf{d}\mathbf{d}^T + N \end{aligned} \tag{13}$$

Due to $\alpha \in (0, \infty)$, it can be obtained by formula (13):

$$\sum_{i=1}^{n}\sum_{j=1}^{n} -\mathbf{p}_i \mathbf{s}_j^T (x_{ij})^* \le \sum_{i=1}^{n}\sum_{j=1}^{n} -\mathbf{p}_i \mathbf{s}_j^T (x_{ij}) \tag{14}$$

The inequality (14) is independent of α, $\mathbf{d}$. It implies that if the inequality $C(\alpha, \mathbf{d}, \sigma^*) \le C(\alpha, \mathbf{d}, \sigma)$ at some value of $\alpha \in (0, \infty)$, $\mathbf{d}$ is established, that is, σ^* is the optimal matching at any value of $\alpha \in (0, \infty)$, $\mathbf{d}$. □

Further, by formula (14), let a pseudo cost $k_{ij} = -\mathbf{p}_i \mathbf{s}_j^T$ and the new model can be established:

$$\begin{aligned} \min K(\sigma) &= \sum_{i=1}^{n}\sum_{j=1}^{n} k_{ij} x_{ij} \\ &= \sum_{i=1}^{n}\sum_{j=1}^{n} \left(-\mathbf{p}_i \mathbf{s}_j^T\right) x_{ij} \end{aligned} \tag{15}$$

and the constraint is still (2).

To solve the above model (15), the Hungarian algorithm [20] can be used to obtain the optimal assignment σ^* and the new objective function minimum value K^*. σ^* is also the optimal assignment of the original objective function C.

Conclusion 1. *At the optimal assignment σ^*, the original objective function:*

$$C(\alpha, \mathbf{d}, \sigma^*) = M\alpha^2 + 2\alpha K^* + 2\alpha \mathbf{s}\mathbf{d}^T - 2\mathbf{p}\mathbf{d}^T + n\mathbf{d}\mathbf{d}^T + N \tag{16}$$

(16) is a quadratic strict convex function of the variables α, $\mathbf{d}$.

Proof. The Hessian matrix of the formula (16) is:

$$H = \begin{bmatrix} M & X_{\mathbf{s}} & Y_{\mathbf{s}} \\ X_{\mathbf{s}} & n & 0 \\ Y_{\mathbf{s}} & 0 & n \end{bmatrix} \tag{17}$$

where $X_{\mathbf{s}} = \sum_{j=1}^{n} x_{\mathbf{s}_j}$, $Y_{\mathbf{s}} = \sum_{j=1}^{n} y_{\mathbf{s}_j}$. The sequential principal minors of the matrix are:

$$\begin{aligned} D_1 &= M \\ D_2 &= nM - X_s^2 \\ &= \sum_{i=1}^{n}\sum_{j=i+1}^{n} \left(x_{\mathbf{s}_i} - x_{\mathbf{s}_j}\right)^2 + n\sum_{j=1}^{n} y_{\mathbf{s}_j}^2 \\ D_3 &= n^2 M - nX_s^2 - nY_s^2 \\ &= n\left[\sum_{i=1}^{n}\sum_{j=i+1}^{n} \left(x_{\mathbf{s}_i} - x_{\mathbf{s}_j}\right)^2 + \sum_{i=1}^{n}\sum_{j=i+1}^{n} \left(y_{\mathbf{s}_i} - y_{\mathbf{s}_j}\right)^2\right] \\ &= n\sum_{i=1}^{n}\sum_{j=i+1}^{n} \left\|\mathbf{s}_i - \mathbf{s}_j\right\|^2 \end{aligned} \tag{18}$$

The values of the sequential principal minors are positive, so the Hessian matrix (17) is a positive definite matrix, and (16) is a quadratic strict convex function. □

The problem of computing the optimal parameters α, $\mathbf{d}$, is transformed to solve the nonlinear programming problem. The formula (16) is the objective function of the nonlinear programming, $[\alpha, \mathbf{d}]$ are independent variables.

3.2. System constraints

We consider system constraints based on the obstacle environment information. In this paper, the constraints of the system need to meet three requirements for reaching the goal without collision:

(1) The generated goal pattern positions should be in the application area;
(2) The generated goal pattern positions should not be in the obstacles;
(3) The generated goal pattern positions should ensure that the distance between them maintains at least two robot radius.

The requirements (2) and (3) are to ensure that the robot can reach the goal positions without collision and complete pattern formation. The mathematical model of the constraint is established according to the above three requirements.

Consider formula (4), the goal $\mathbf{q}_j = \left[x_{\mathbf{q}_j}, y_{\mathbf{q}_j}\right]$, where

$$\begin{cases} x_{\mathbf{q}_j} = \alpha x_{\mathbf{s}_j} + d_1 \\ y_{\mathbf{q}_j} = \alpha y_{\mathbf{s}_j} + d_2 \end{cases} \quad j = 1, \ldots, n \tag{19}$$

Subject to condition (1), we have

$$\begin{aligned} &\begin{cases} X_{\min} + R \le x_{\mathbf{q}_j} \le X_{\max} - R \\ Y_{\min} + R \le y_{\mathbf{q}_j} \le Y_{\max} - R \end{cases} \\ \Rightarrow &\begin{cases} X_{\min} + R \le \alpha x_{\mathbf{s}_j} + d_1 \le X_{\max} - R \\ Y_{\min} + R \le \alpha y_{\mathbf{s}_j} + d_2 \le Y_{\max} - R \end{cases} \end{aligned} \tag{20}$$

where X_{min}, X_{max} are the minimum and maximum boundary of the application area on the X axis, respectively; Y_{min}, Y_{max} are the minimum and maximum boundary of the application area on the Y axis, respectively.

Since $\alpha \in (0, \infty)$, the boundary points in the given pattern $\mathbf{S}_{n\times 2}$ still are the boundary points after scale and translation, so the constraint (20) can be equivalent to:

$$\begin{cases} \alpha \max\left(x_{\mathbf{s}_j}\right) + d_1 \le \mathrm{X}_{\max} - R \\ \alpha \min\left(x_{\mathbf{s}_j}\right) + d_1 \ge \mathrm{X}_{\min} + R \\ \alpha \max\left(y_{\mathbf{s}_j}\right) + d_2 \le \mathrm{Y}_{\max} - R \\ \alpha \min\left(y_{\mathbf{s}_j}\right) + d_2 \ge \mathrm{Y}_{\min} + R \end{cases} \tag{21}$$

Compared with the formula (20), the formula (21) can reduce the number of constraints and accelerate the convergence of the algorithm, especially for large-scale robots.

Subject to condition (2), combining with the formula (19), we have

$$\begin{aligned} &\|\mathbf{q}_j - \mathbf{o}_i\| \ge \mathrm{R}_{\mathbf{o}} + \mathrm{R} \Rightarrow \\ &\sqrt{\left[\left(\alpha x_{\mathbf{s}_j} + d_1\right) - x_{\mathbf{o}_i}\right]^2 + \left[\left(\alpha y_{\mathbf{s}_j} + d_2\right) - y_{\mathbf{o}_i}\right]^2} \ge \mathrm{R}_{\mathbf{o}} + \mathrm{R} \end{aligned} \tag{22}$$

where $\mathbf{q}_j$ is the position coordinates of the goal j; $\mathbf{o}_i$ is the position coordinates of the static obstacle i; $R_{\mathbf{o}}$, R are the radius of the obstacles and the robot, respectively. $j \in \{1, \ldots, n\}$, $i \in \{1, \ldots, m\}$.

Formula (22) is a nonlinear inequality constraint and is a non-convex set, and then the above optimization problem belongs to non-convex optimization. The non-convex optimization problem has the complexity of solving and the possibility of no global optimal solution. Therefore, it is necessary to scale nonlinear constraints to linear constraints.

We approximate the circle obstacle into a circumscribed square, as shown in Fig. 3. In this way, the constraint (22) about the Euclidean distance between the goal point and the center of the static obstacle is scaled to the constraint (23) about the

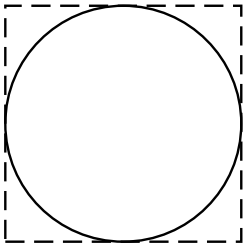


Fig. 3. An approximate model of a disk obstacle.

absolute value of the distance difference between the goal point and the center of the static obstacle on the X and Y axes.

$$\begin{cases} \left|\alpha x_{\mathbf{s}_j} + d_1 - x_{\mathbf{o}_i}\right| \geq R_{\mathbf{o}} + R \\ \left|\alpha y_{\mathbf{s}_j} + d_2 - y_{\mathbf{o}_i}\right| \geq R_{\mathbf{o}} + R \end{cases} \tag{23}$$

Then, we introduce binary variables to transform the nonlinear constraints (23) into linear constraints [21]:

$$\begin{cases} \alpha x_{\mathbf{s}_j} + d_1 - x_{\mathbf{o}_i} \geq R_{\mathbf{o}} + R - FB_{ij}\,(1) \\ x_{\mathbf{o}i} - \left(\alpha x_{\mathbf{s}_j} + d_1\right) \geq R_{\mathbf{o}} + R - FB_{ij}\,(2) \\ \alpha y_{\mathbf{s}_j} + d_2 - y_{\mathbf{o}_i} \geq R_{\mathbf{o}} + R - FB_{ij}\,(3) \\ y_{\mathbf{o}_i} - \left(\alpha y_{\mathbf{s}_j} + d_2\right) \geq R_{\mathbf{o}} + R - FB_{ij}\,(4) \\ \sum_{w=1}^{4} B_{ij}(w) \leq 3 \end{cases} \tag{24}$$

$j \in \{1, \ldots, n\}$, $i \in \{1, \ldots, m\}$. $B_{ij}(w)$ is the introduced binary variable and has a value of 1 or 0, $w \in \{1, 2, 3, 4\}$; F is a positive value and is much larger than $R_{\mathbf{o}} + R$. When $B_{ij}(w) = 0$, the wth constraint of the formula (24) satisfies the corresponding position coordinate difference not less than $R_{\mathbf{o}} + R$. When $B_{ij}(w) = 1$, the wth constraint of the formula (24) is relaxed. $\sum_{w=1}^{4} B_{ij}(w) \leq 3$ ensures that the coordinate difference constraint is not relaxed beyond three positions, at least one direction must satisfy the coordinate difference constraint.

Subject to condition (3), we have

$$\begin{aligned} &\|\mathbf{q}_i - \mathbf{q}_j\| \geq 2R \\ &\Rightarrow [(\alpha x_{\mathbf{s}_i} - \alpha x_{\mathbf{s}_j})^2 + (\alpha y_{\mathbf{s}_i} - \alpha y_{\mathbf{s}_j})^2] \geq 4R^2 \\ &\Rightarrow \alpha^2 \left[\left(x_{\mathbf{s}_i} - x_{\mathbf{s}_j}\right)^2 + \left(y_{\mathbf{s}_i} - y_{\mathbf{s}_j}\right)^2\right] \geq 4R^2 \\ &\Rightarrow \alpha \min \left\|\mathbf{s}_i - \mathbf{s}_j\right\| \geq 2R, \qquad i \neq j, i, j \in \{1, \ldots, n\} \end{aligned} \tag{25}$$

Conclusion 2. *Based on the linear constraints (21), (24), (25), and the objective convex quadratic function (16), the computing optimal goal pattern parameters problem is transformed into solving a convex quadratic programming problem under convex constraint.*

Since the convex quadratic programming problem has a global optimal solution [22], we can use the CPLEX optimization software [23] to find the global optimal goal pattern parameters α^*, $\mathbf{d}^*$, and then obtain the optimal goal pattern $\mathbf{Q}^*$.

Next, we will design an iterative obstacle avoidance control method to drive the robot to the goal without collision.

4. Collision-free path planning

This section details how the iterative controller regulates the robots to reach their goals without collision. Following the assignment obtained from the prevision section, the preferred velocity of each robot is first computed, and the optimal collision-free velocity is then obtained.

The iterative obstacle avoidance planning process is Algorithm 1. In the kth iteration ($k \in \mathbb{N}$, the time-step index of the controller), the controller first determines if there are any robots that have not reached the ε neighborhood of goal. If so, the optimal assignment σ_k^* is calculated. Then, for robot i that has not reached the assigned goal ($\left\|\mathbf{q}_{\sigma_k^*(i)} - \mathbf{p}_i^{k-1}\right\| > \varepsilon$), its preferred velocity $\mathbf{v}_{pref_i}^k$ and optimal obstacle avoidance velocity $\mathbf{v}_{opt_i}^k$ are computed. Based on the $\mathbf{v}_{opt_i}^k$, the robot i updates its position. The above steps are repeated until all the robots reach the ε neighborhood of goal. The algorithm ends. The primary process of the algorithm is divided into the following three steps.

Algorithm 1 The Iterative Obstacle Avoidance Controller

```
In the k-th iteration;
while exist robot no reach the ε neighborhood of goal. do
  Compute σ_k^*;
  for each i ∈ [1, n] do
    if ‖q_{σ_k^*(i)} − p_i^{k−1}‖ ≤ ε then
      continue;
    else
      Compute v_{pref_i}^k ;
      Compute v_{opt_i}^k ;
      Update p_i^k;
    end if
  end for
  k = k + 1;
end while
```

4.1. Goal assignment

Based on the initial positions of the group robot, the optimal assignment σ^* has been obtained in Section 3, but the robot does not always move straight to the assigned goal position. It may be away from the assigned goal in order to avoid collision with obstacles or other robots. Thus, reassign is needed. Based on the goal pattern $\mathbf{Q}^*$, we can make a new pseudo cost $k_{ij} = -\mathbf{p}_i^{k-1}\mathbf{q}_j^T$, where $\mathbf{p}_i^{k-1}$ is the position coordinates of the robot i in the $(k-1)$th iteration, $i \in \{1, \ldots, n\}$, $j \in \{1, \ldots, n\}$. Based on the model (15) and the constraint (2), the optimal assignment σ_k^* is obtained according to the Hungarian algorithm.

4.2. Compute the preferred velocity

Based on the optimal assignment σ_k^*, we compute the preferred velocity $\mathbf{v}_{pref_i}^k$ toward the goal without considering the obstacles and other robots.

$$\mathbf{v}_{pref_i}^k = V_p \min(1, \frac{\|\mathbf{q}_{\sigma_k^*(i)} - \mathbf{p}_i^{k-1}\|}{K_a}) \frac{\mathbf{q}_{\sigma_k^*(i)} - \mathbf{p}_i^{k-1}}{\|\mathbf{q}_{\sigma_k^*(i)} - \mathbf{p}_i^{k-1}\|} \tag{26}$$

where V_p is the maximum speed of the robot; $\mathbf{q}_{\sigma_k^*(i)}$ is the goal position coordinates assigned to the robot i; $K_a > 0$ ensures convergence, and it needs to satisfy $K_a \geqslant V_p\tau$ [12], where τ is the time step.

4.3. Compute the optimal obstacle avoidance velocity and update position

The *RVO* algorithm proposed in Ref. [19] realizes that the robot reaches the goal position without collision.

Given the current position coordinates $\mathbf{p}_i$ and the current velocity $\mathbf{v}_i$ of the disk robot i. Defining the velocity obstacles

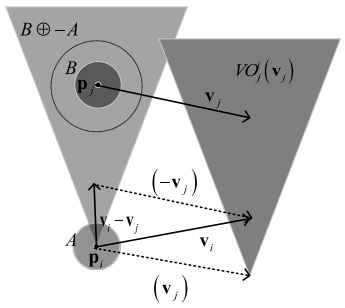


Fig. 4. The velocity obstacles $VO_j^i(\mathbf{v}_j)$ of disk robot i relative to disk robot j.

$VO_j^i(\mathbf{v}_j)$ is the velocity set consisting of all those velocities $\mathbf{v}_i$ for robot i that will cause a collision with the robot j moving with velocity $\mathbf{v}_j$ at some time in the future. The geometric definition of the velocity obstacles as follows.

Definition 1. Velocity Obstacles

$$VO_j^i(\mathbf{v}_j) = \{\mathbf{v}_i | \lambda(\mathbf{p}_i, \mathbf{v}_i - \mathbf{v}_j) \cap \mathrm{B} \oplus -\mathrm{A} \neq \emptyset\} \tag{27}$$

where A represents a set of circular regions with $\mathbf{p}_i$ as the center and R as the radius; B represents a set of circular regions with $\mathbf{p}_j$ as the center and R as the radius; $A \oplus B$ represents the Minkowski sum of A and B; $-A$ represents A reflected in its reference point, i.e.,

$$A \oplus B = \{a + b | a \in A, b \in B\} \tag{28}$$

$$-A = \{-a | a \in A\} \tag{29}$$

$\lambda(\mathbf{p}, \mathbf{v})$ is the ray with $\mathbf{p}$ as the starting point and $\mathbf{v}$ as the direction:

$$\lambda(\mathbf{p}, \mathbf{v}) = \{\mathbf{p} + t\mathbf{v} | t \geqslant 0\} \tag{30}$$

The geometric representation is shown in Fig. 4. If the ray with the starting point $\mathbf{p}_i$ and the direction $\mathbf{v}_i - \mathbf{v}_j$ intersects the $B \oplus -A$ region centered at $\mathbf{p}_j$, then $\mathbf{v}_i$ is in the velocity obstacles region relative to the robot j.

When $\mathbf{v}_i \in VO_j^i(\mathbf{v}_j)$, the robot i and the robot j will collide at some time later if the robot i moves with the velocity $\mathbf{v}_i$, the robot j moves with the velocity $\mathbf{v}_j$. So in order to avoid collision between robot i and j, the velocity $\mathbf{v}_i$ should be selected outside the velocity obstacles, $\mathbf{v}_i \notin VO_j^i(\mathbf{v}_j)$.

When the robot i selects the candidate velocity $\mathbf{v}_i'$ outside $VO_j^i(\mathbf{v}_j)$ and the robot j selects the candidate velocity $\mathbf{v}_j'$ outside $VO_i^j(\mathbf{v}_i)$ to avoid collisions with each other, the undesirable oscillatory motions will occur. Therefore we use RVO.

$$RVO_j^i(\mathbf{v}_j, \mathbf{v}_i) = \{\mathbf{v}_i' | 2\mathbf{v}_i' - \mathbf{v}_i \in VO_j^i(\mathbf{v}_j)\} \tag{31}$$

In multi-robot and static obstacle environments, the RVO_i for the robot i:

$$RVO_i = \bigcup_{j \neq i} RVO_j^i(\mathbf{v}_j, \mathbf{v}_i) \cup \bigcup_{\mathbf{o} \in \mathbf{O}} VO_{\mathbf{o}}^i(\mathbf{v}_{\mathbf{o}}) \tag{32}$$

where $\mathbf{O}$ is the static obstacle coordinate set; $\mathbf{v}_{\mathbf{o}}$ is the velocity of the static obstacle, $\mathbf{v}_{\mathbf{o}} = 0$.

The robot in this paper is the holonomic robot that performs the continuous cycle of sensing and acting [18,19]. We give the kinematic model in two-dimensional plane:

$$\begin{cases} x_i^{k+1} = x_i^k + v_{ix}^k \tau \\ y_i^{k+1} = y_i^k + v_{iy}^k \tau \\ \mathbf{v}_i^{k+1} = \mathbf{v}_i^k + \boldsymbol{a}_i^k \tau \end{cases} \tag{33}$$

where x_i, y_i is the plane position coordinates of the center of the disk robot; v_{ix}, v_{iy} is the velocity component; $\boldsymbol{a}_i$ is the acceleration; τ is the time step.

Each robot i is subject to kinematic constraints. These constraints limit the choice of feasible velocities [19]. We set the maximum speed of the robot not to exceed V_p and the maximum acceleration not to exceed a, then the feasible velocities set is AV_i:

$$AV_i = \{\mathbf{v}_i' \,|\, \|\mathbf{v}_i'\| < V_p \wedge \|\mathbf{v}_i' - \mathbf{v}_i\| < a\tau\} \tag{34}$$

Ideally, in the kth iteration, robot i selects the feasible velocity outside the RVO_i^k and closest to $\mathbf{v}_{pref_i}^k$ as the optimal obstacle avoidance velocity $\mathbf{v}_{opt_i}^k$:

$$\mathbf{v}_{opt_i}^k = \underset{\mathbf{v}_i \notin RVO_i^k, \mathbf{v}_i \in AV_i}{\operatorname{argmin}} \|\mathbf{v}_i - \mathbf{v}_{pref_i}^k\| \tag{35}$$

However, the environment is so dense that the feasible velocities set $AV_i \subset RVO_i$, and $\mathbf{v}_{opt_i}^k$ cannot be calculated by formula (35). To solve this problem, we select a velocity inside RVO_i, but the velocity is penalized by this choice. Refer to literature [19], the penalty of the velocity $\mathbf{v}_i$ is given:

$$penalty_i^k(\mathbf{v}_i) = \omega_i^k \frac{1}{tc_i^k(\mathbf{v}_i)} + \|\mathbf{v}_{pref_i}^k - \mathbf{v}_i\| \tag{36}$$

where ω_i^k can vary among the agents to reflect differences in aggressiveness and sluggishness; $tc_i^k(\mathbf{v}_i)$ is the expected time to collision. For detailed description and calculation of parameters, see [19]. We select the velocity with the minimal penalty in the AV_i as the optimal collision avoidance velocity:

$$\mathbf{v}_{opt_i}^k = \underset{\mathbf{v}_i \in AV_i}{\operatorname{argmin}} penalty_i^k(\mathbf{v}_i) \tag{37}$$

If $AV_i \not\subset RVO_i$, we obtain $\mathbf{v}_{opt_i}^k$ according to formula (35), otherwise according to formula (37), which can avoid the robot from sticking in equilibrium point.

Then robot i updates its position:

$$\mathbf{p}_i^k = \mathbf{p}_i^{k-1} + \mathbf{v}_{opt_i}^k \tau \tag{38}$$

τ is the time step.

Conclusion 3. *(1) Based on the position $\mathbf{P}_{n\times 2}^{k-1}$ of the robots in the $(k-1)$th iteration and the goal $\mathbf{Q}^*$, the optimal assignment σ_k^* in the kth iteration is obtained according to the Hungarian algorithm; (2) based on σ_k^*, we compute the preferred velocity $\mathbf{v}_{pref_i}^k$ toward the assigned goal without considering obstacles and other robots by formula (26); (3) according to the RVO algorithm, we compute the optimal obstacle avoidance velocity $\mathbf{v}_{opt_i}^k$ closest to the preferred velocity $\mathbf{v}_{pref_i}^k$ by formula (35) or select the velocity with the minimal penalty by formula (37); (4) above steps are repeated until all the robots reach the ε neighborhood of goals. The algorithm ends.*

Theorem 1. *The convergence of the robot reaching to the assigned goal is guaranteed.*

Proof. In the kth iteration, the robot i selects the collision avoidance velocity that is close to the preferred velocity $\mathbf{v}_{pref_i}^k$ calculated by formula (26). According to the formula (26), we can obtain the $\mathbf{v}_{pref_i}^k \tau$:

$$\mathbf{v}_{pref_i}^k \tau = \min\left(\frac{V_p \tau}{\left\|\mathbf{q}_{\sigma_k^*(i)} - \mathbf{p}_i^{k-1}\right\|}, \frac{V_p \tau}{K_a}\right)\left(\mathbf{q}_{\sigma_k^*(i)} - \mathbf{p}_i^{k-1}\right) \tag{39}$$

where $K_a \geq V_p \tau$, further, we can get the following formula:

$$K_a \geq V_p \tau \Rightarrow \frac{V_p \tau}{K_a} \leq 1 \tag{40}$$

There are two cases ① and ② in the process of robot movement, and we comprehensively discuss these two cases.

① When $\left\|\mathbf{q}_{\sigma_k^*(i)}-\mathbf{p}_i^{k-1}\right\| \geq V_p\tau$, we can get :

$$\frac{V_p\tau}{\left\|\mathbf{q}_{\sigma_k^*(i)}-\mathbf{p}_i^{k-1}\right\|} \leq 1 \tag{41}$$

Combining the formula (40), the formula (39) is equal to:

$$\mathbf{v}_{pref_i}^k\tau = K_c\left(\mathbf{q}_{\sigma_k^*(i)}-\mathbf{p}_i^{k-1}\right) \tag{42}$$

where $K_c \leq 1$.

② When $\left\|\mathbf{q}_{\sigma_k^*(i)}-\mathbf{p}_i^{k-1}\right\| \leq V_p\tau$, we can get:

$$\frac{V_p\tau}{\left\|\mathbf{q}_{\sigma_k^*(i)}-\mathbf{p}_i^{k-1}\right\|} \geq 1 \tag{43}$$

Combining the formula (40) and (43), we can obtain $\frac{V_p\tau}{\left\|\mathbf{q}_{\sigma_k^*(i)}-\mathbf{p}_i^{k-1}\right\|} \geq \frac{V_p\tau}{K_a}$, so the formula (39) is equal to:

$$\begin{aligned}\mathbf{v}_{pref_i}^k\tau &= \frac{V_p\tau}{K_a}\left(\mathbf{q}_{\sigma_k^*(i)}-\mathbf{p}_i^{k-1}\right)\\ &= K_c\left(\mathbf{q}_{\sigma_k^*(i)}-\mathbf{p}_i^{k-1}\right)\end{aligned} \tag{44}$$

where $K_c \leq 1$.

Synthesizing the derivation of ① and ②, we can obtain:

$$\mathbf{v}_{pref_i}^k\tau = K_c\left(\mathbf{q}_{\sigma_k^*(i)}-\mathbf{p}_i^{k-1}\right) \tag{45}$$

where $K_c \leq 1$.

The robot i updates its position:

$$\begin{aligned}\mathbf{p}_i^k &= \mathbf{p}_i^{k-1}+\mathbf{v}_{pref_i}^k\tau\\ &= \mathbf{p}_i^{k-1}+K_c\left(\mathbf{q}_{\sigma_k^*(i)}-\mathbf{p}_i^{k-1}\right)\end{aligned} \tag{46}$$

where $K_c \leq 1$. The formula (46) implies $\left(\mathbf{q}_{\sigma_k^*(i)}-\mathbf{p}_i^k\right)$ and $\left(\mathbf{q}_{\sigma_k^*(i)}-\mathbf{p}_i^{k-1}\right)$ are collinear vectors, $\left\|\mathbf{q}_{\sigma_k^*(i)}-\mathbf{p}_i^k\right\| \leq \left\|\mathbf{q}_{\sigma_k^*(i)}-\mathbf{p}_i^{k-1}\right\|$ and $\left\|\mathbf{q}_{\sigma_k^*(i)}-\mathbf{p}_i^k\right\| \geq 0$. Therefore $\left\|\mathbf{q}_{\sigma_k^*(i)}-\mathbf{p}_i^k\right\|$ is monotonically decreasing and has lower bound, the convergence of the robot reaching to the assigned goal is guaranteed. □

Corollary 1. *The robot reaching the ε neighborhood of the assigned goal in finite time can be guaranteed.*

Proof. When the robot i approaches the assigned goal $\mathbf{q}_{\sigma_k^*(i)}$, $\left\|\mathbf{q}_{\sigma_k^*(i)}-\mathbf{p}_i^{k-1}\right\| \leq V_p\tau$, and $K_a \geq V_p\tau$, we can get:

$$\left\|\mathbf{q}_{\sigma_k^*(i)}-\mathbf{p}_i^{k-1}\right\| \leq K_a \Rightarrow \frac{\left\|\mathbf{q}_{\sigma_k^*(i)}-\mathbf{p}_i^{k-1}\right\|}{K_a} \leq 1 \tag{47}$$

According to the formula (47), the formula (26) can be written as:

$$\begin{aligned}\mathbf{v}_{pref_i}^k &= V_p\min(1, \frac{\|\mathbf{q}_{\sigma_k^*(i)}-\mathbf{p}_i^{k-1}\|}{K_a})\frac{\mathbf{q}_{\sigma_k^*(i)}-\mathbf{p}_i^{k-1}}{\|\mathbf{q}_{\sigma_k^*(i)}-\mathbf{p}_i^{k-1}\|}\\ &= V_p\frac{\|\mathbf{q}_{\sigma_k^*(i)}-\mathbf{p}_i^{k-1}\|}{K_a}\frac{\mathbf{q}_{\sigma_k^*(i)}-\mathbf{p}_i^{k-1}}{\|\mathbf{q}_{\sigma_k^*(i)}-\mathbf{p}_i^{k-1}\|}\\ &= \frac{V_p}{K_a}\left(\mathbf{q}_{\sigma_k^*(i)}-\mathbf{p}_i^{k-1}\right)\end{aligned} \tag{48}$$

where V_p and K_a are positive constants. Let $K = \frac{V_p}{K_a} > 0$, further:

$$\left\|\mathbf{v}_{pref_i}^k\right\| = K\left\|\mathbf{q}_{\sigma_k^*(i)}-\mathbf{p}_i^{k-1}\right\| \tag{49}$$

The formula (49) indicates that $\left\|\mathbf{v}_{pref_i}^k\right\|$ is proportional to the distance toward assigned goal.

We convert the discrete system (49) into a continuous system for analysis. Suppose the path function of the robot i concerning time t is $s_i(t)$, and the total distance to the assigned goal is d (a positive constant). We can get:

$$s_i(t) = \int_{t_0}^{t}\left\|\mathbf{v}_{pref_i}^t\right\| dt \tag{50}$$

According to the formula (49), further:

$$s_i(t) = \int_{t_0}^{t} K\left[d-s_i(t)\right]dt \Rightarrow \dot{s}_i(t) = Kd-Ks_i(t) \tag{51}$$

Solving the linear differential equation (51), we can get:

$$\begin{aligned}&s_i(t) = d-de^{-Kt}\\ \Rightarrow &e_i(t) = d-s_i(t) = de^{-Kt}\end{aligned} \tag{52}$$

where $e_i(t)$ is the distance to the assigned goal. Easy to get:

$$\lim_{t\to\infty} e_i(t) = 0 \tag{53}$$

From the formula (53), asymptotic convergence is guaranteed.

For the discrete control system, the sampling period is τ. $\exists M \in \mathbb{N}$, according to the formula (49) and (52), The speed at time M is:

$$\left\|\mathbf{v}_{pref_i}^M\right\| = Kde^{-KM\tau} \tag{54}$$

From M to $M+1$, the path length of the robot i is:

$$\left\|\mathbf{v}_{pref_i}^M\right\|\tau = Kde^{-KM\tau}\tau \tag{55}$$

Always $\exists\varepsilon$ such that $\left\|\mathbf{v}_{pref_i}^M\right\|\tau = Kde^{-KM\tau}\tau > \varepsilon$ and $M\left\|\mathbf{v}_{pref_i}^M\right\|\tau > d-\varepsilon$.

Therefore the robot reaching the ε neighborhood of the assigned goal in finite time can be guaranteed. □

Algorithm 2 expresses the complete process of the optimization of multi-robot pattern formation. First, the optimal matching σ^* and the goal pattern parameters α^*, $\mathbf{d}^*$ are computed in the obstacle environment to obtain the goal pattern $\mathbf{Q}^*$. Second, under the control of the iterative obstacle avoidance controller, the robots reach the assigned goal position in real-time without collision. The algorithm ends, and the pattern is formed until all the robots reach the ε neighborhood of goal.

Below we will design experiments to verify the feasibility and the effectiveness of the proposed algorithm in the obstacle environment.

5. Experimental results and analysis

In this section, simulation experiments and results show the performance of the proposed algorithm. The algorithm proposed in this paper is to optimize the goal pattern parameters, then iteratively assign the goals and control multi-robots to reach the assigned goal without collision, under the presence of obstacles. The experiment of generating reachable goals illustrates the algorithm optimizes the goal in the obstacle environment. The experimental results of forming a single letter show the effectiveness and convergence of the algorithm in achieving pattern formation. The performance of the optimal goal pattern

Algorithm 2 The Optimization of Multi-Robot Pattern Formation Algorithm

// Optimal Goal Generation
Compute σ^*;
Compute α^*, $\mathbf{d}^*$;
Obtain $\mathbf{Q}^*$;
// The iterative obstacle avoidance controller
In the k-th iteration;
while exist robot no reach the ε neighborhood of goal. **do**
  Compute σ_k^*;
  for each $i \in [1, n]$ **do**
    if $\left\| \mathbf{q}_{\sigma_k^*(i)} - \mathbf{p}_i^{k-1} \right\| \leq \varepsilon$ **then**
      continue;
    else
      Compute $\mathbf{v}_{pref_i}^k$;
      Compute $\mathbf{v}_{opt_i}^k$;
      Update $\mathbf{p}_i^k$;
    end if
  end for
  $k = k + 1$;
end while

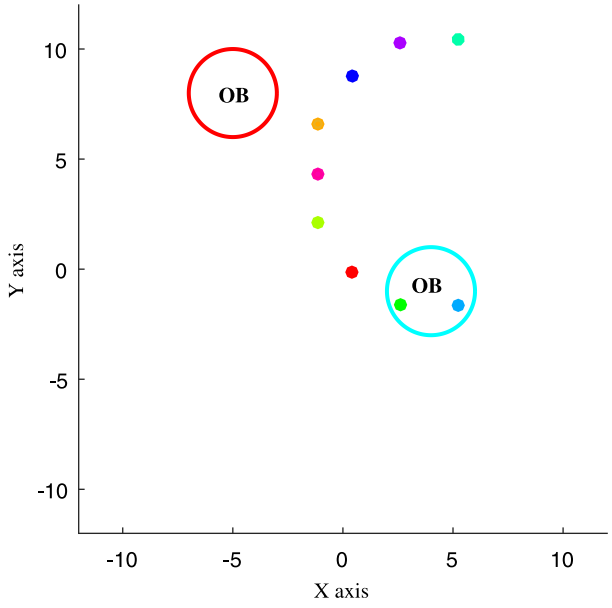


Fig. 5. The unreachable goal generation. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

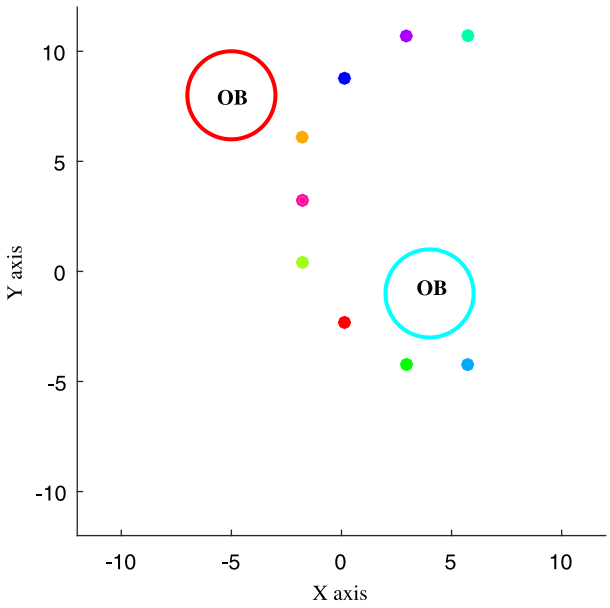


Fig. 6. The reachable goal generation. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

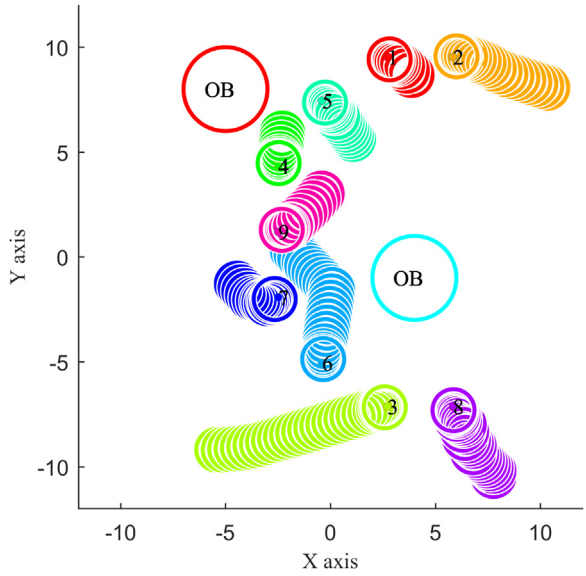


Fig. 7. The trajectory of nine robots forming the letter C pattern.

parameters obtained by the proposed algorithm and the performance of other parameters are compared and analyzed, which verifies the optimization of parameters by the algorithm. The comparison of experimental results between iterative match and traditional one-time match verifies the optimization of the algorithm on match. Finally, the algorithm is extended to the pattern formation of multiple letters experiments, and the changes in algorithm performance with the changes in the number of robots are analyzed.

We designed several experiments using Matlab 2016a with CPLEX optimization software on the computer (Windows 10, Intel(R) Core(TM) i7-6700, CPU @ 3.40 GHz with 16.0 GB RAM). We recorded and analyzed the experimental data. T_{opt}: solve the optimal goal pattern time; K: the total number of iterations of all robots reaching the goal positions; T: total running time of all robots reaching the goal positions; L: total path length of all robots.

5.1. Reachable goal generation

A goal generation model built under no obstacle environment includes obstacle positions in the feasible region [13], which may result in the goal position is generated in the obstacle zone. As shown in Fig. 5, the red circle and blue circle marked with *OB* indicate static obstacles. Nine different colored dots labeled 1 to 9 represent the generated goal positions. The blue and green goals positions are in the obstacle, which will cause the robot cannot reach the goals. However, based on the algorithm proposed in this

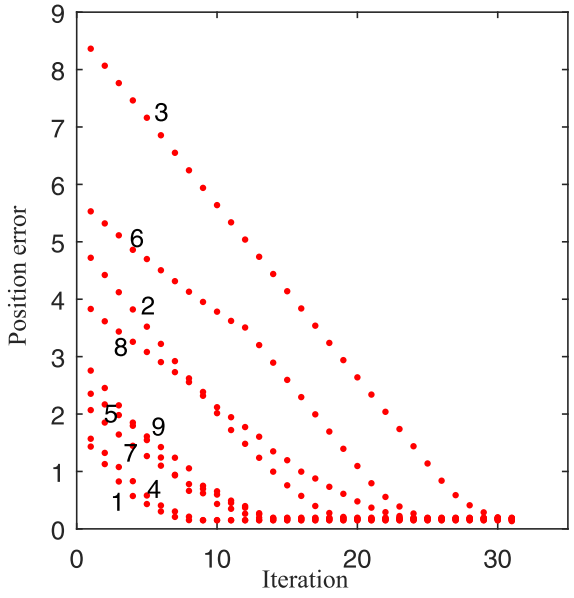
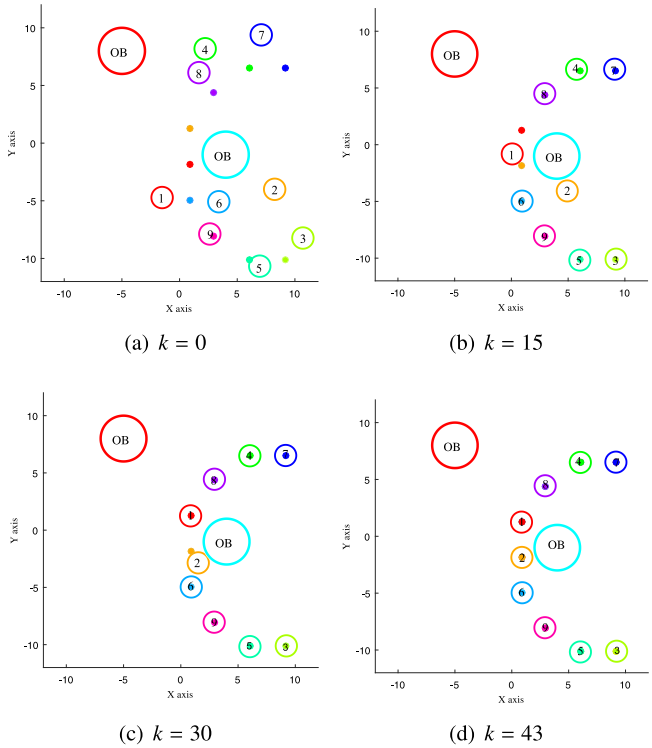


Fig. 8. Position error.



(a) $k = 0$ (b) $k = 15$

(c) $k = 30$ (d) $k = 43$

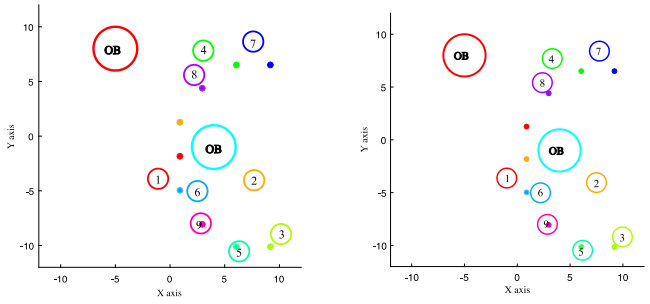
Fig. 9. Pattern C formation.

paper, the optimized goal generation model removes obstacle position information out of the feasible region to generate reachable goals, as shown in Fig. 6. The potential case where a goal position is in an obstacle zone will not appear in the next experiments.

5.2. Pattern of a single letter

5.2.1. Experiment 1

Suppose there are two static disk obstacles with a radius of 2 in a square area with an application region of $[-12, 12; -12, 12]$



(a) $k = 4$ (b) $k = 5$

Fig. 10. The goal match changes.

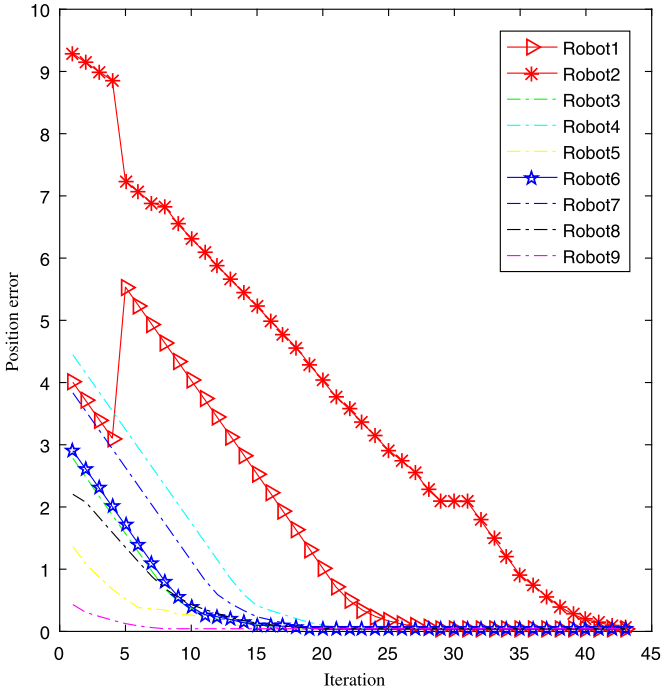


Fig. 11. Position error.

and nine disk robots with a radius of 1 form the letter pattern C. As shown in Fig. 7, the robots represented by nine different colors circles labeled 1 to 9 move from the random initial position to the assigned goal of the corresponding color without collision, and then the robots form the letter C. The time of solving the optimal goal pattern $T_{opt} = 2.98$ s, the total number of iterations $K = 31$, the total running time $T = 7.954$ s, the total path length $L = 35.11$. Then we analyzed the position error between the nine robots and the assigned goal positions. As shown in Fig. 8, the nine dashed lines labeled 1 to 9 indicate the change in position errors of the corresponding robots. After 31 iterations, the position errors of the nine robots converge to ε neighborhood of 0.

5.2.2. Experiment 2

In order to show that the path and time of our design method are the shortest, three groups of forming single letter experiments based on different random positions are designed. The initial conditions (initial positions of the robots, positions of obstacles,

Table 1
The performance comparison of the optimal pattern parameters with any other pattern parameters.

		K	T	L
1	$\alpha^* = 1.1389$, $\mathbf{d}^* = [1.8177, 0.3099]$	**42**	**10.81**	**38.09**
	$\alpha = 1$, $\mathbf{d} = [2, 0.3]$	48	12.76	39.87
	$\alpha = 1.3$, $\mathbf{d} = [2, 0.3]$	48	12.16	40.68
	$\alpha = 1.2$, $\mathbf{d} = [3, 0.3]$	43	11.03	38.63
		K	T	L
2	$\alpha^* = 1.0088$, $\mathbf{d}^* = [3.0148, -1.7201]$	**36**	**10.69**	**43.49**
	$\alpha = 1$, $\mathbf{d} = [3, 0]$	40	11.53	48.87
	$\alpha = 1.1$, $\mathbf{d} = [3, -1]$	38	10.85	44.54
	$\alpha = 1.1$, $\mathbf{d} = [3.5, -0.5]$	41	12.17	47.31
		K	T	L
3	$\alpha^* = 0.9626$, $\mathbf{d}^* = [0.2864, -3.0989]$	**36**	**8.47**	**29.05**
	$\alpha = 1$, $\mathbf{d} = [2, 0.3]$	33	8.78	35.33
	$\alpha = 1.3$, $\mathbf{d} = [2, 0.3]$	31	8.65	31.37
	$\alpha = 1.2$, $\mathbf{d} = [3, 0.3]$	29	8.54	29.12

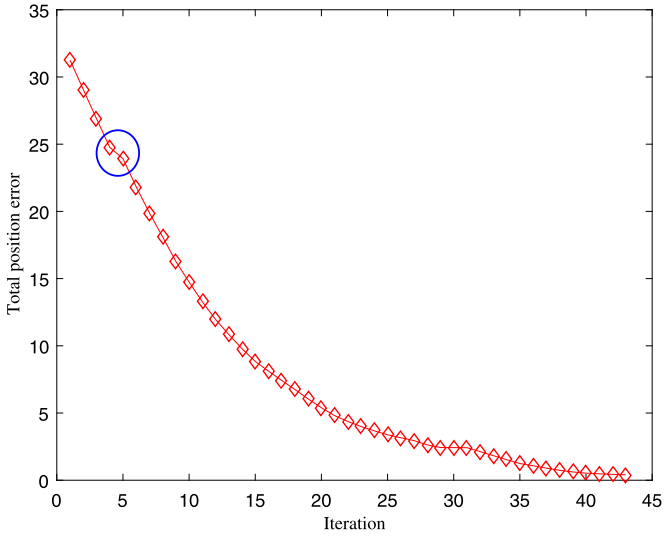


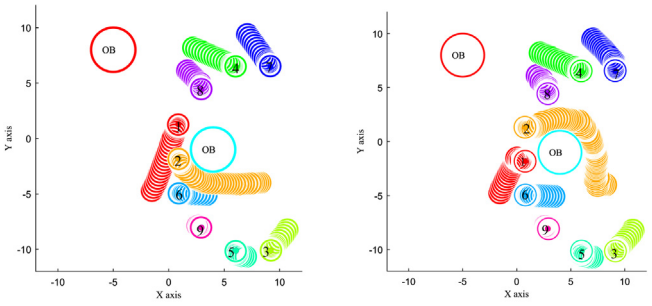
Fig. 12. Total position error.

region size, desired pattern, etc.) in the same set of experiments are the same. The initial conditions in the different set of experiments are different. In each set of experiments, we obtained the optimal goal pattern parameters α^*, $\mathbf{d}^*$ and recorded the corresponding K, L, T to compare with K, L, T obtained from any other parameters that can generate reachable goal. As shown in Table 1, regardless of the initial position, in each set of experiments, the performance of the optimal parameters is better than the performance of other parameters, especially in total path length.

5.2.3. Experiment 3

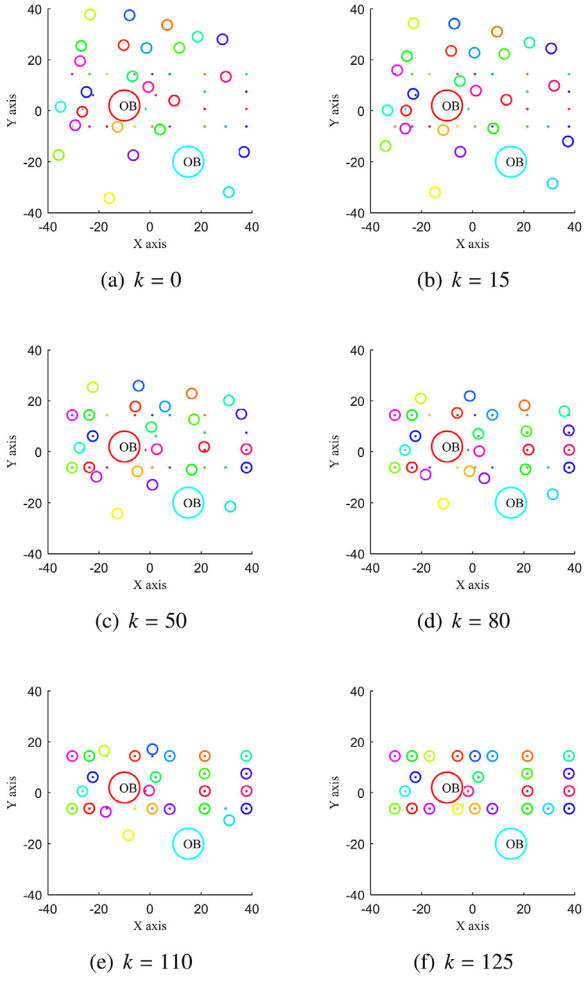
In order to show that the iterative match used in this paper can find the shortest path compared to the traditional one-time match [1,13], we did experiments of forming a single letter for the two match methods, respectively. Fig. 9 presents the snapshots of position update and goal assignment of multiple robots, based on the iterative match. Fig. 10 displays the snapshots of the goal match changes. At $k = 4$, as the existence of obstacles, robot 2 needs to avoid collision with the obstacle and stays away from the assigned goal. Therefore, at $k = 5$, the goal match changes after reassignment, robot 1 and robot 2 exchange goal.

We analyzed the position error of each robot and the sum of the position errors of all robots. As shown in Fig. 11, at $k = 5$, as



(a) Under iterative match (b) Under one-time match

Fig. 13. The trajectory of nine robots forming the letter C pattern.



(a) $k = 0$ (b) $k = 15$ (c) $k = 50$ (d) $k = 80$ (e) $k = 110$ (f) $k = 125$

Fig. 14. Pattern ZZU formation.

the change of the target of the robot 1, 2, the position error of the robot 1 suddenly increases, and the position error of the robot 2 suddenly decreases. Although the increasing amplitude of robot 1 is greater than the decreasing amplitude of robot 2, the sum of position errors is decreased, as shown in Fig. 12.

Fig. 13 presents the trajectories of robot forming letter C under iterative match and traditional one-time match, respectively. We recorded the $L = 33.75$, $K = 43$ and $T = 10.55$ corresponding to the iterative match and the $L = 49.43$, $K = 75$ and $T =$

20.64 corresponding to the one-time match, which illustrates that the iterative match has better performance than the traditional one-time match.

5.3. Pattern of multiple letters

5.3.1. Experiment 4

Suppose there are two static disk obstacles with a radius of 6 in a square area with an application region of $[-40, 40; -40, 40]$ and 25 disk robots with a radius of 2 to form the pattern ZZU, as shown in Fig. 14. After 125 iterations, the pattern is formed. The pattern solution time $T_{opt} = 3.36$ s, the total running time $T = 158$ s, the total number of iterations $K = 125$, and the total path length $L = 454.54$. Compared with the experiments of 9 robots, the solution goal time T_{opt} of the 25 robots increases less, but the total running time T increases significantly, which is because the iterative obstacle avoidance controller adopts the Hungarian algorithm its complexity is $O(n^3)$ that the amount of calculation increases exponentially as the number of robots increases.

6. Conclusions and future research

This paper proposes an iterative optimization approach for multi-robot pattern formation in an obstacle environment. The proposed approach obtains the optimal pattern parameters, then iteratively assigns the goals and plans a collision-free path for each robot to reach the goal position. We carry out simulation experiments for pattern formations of a single letter and multiple letters. The study concludes that the proposed approach efficiently addresses the optimization for pattern formation problem multi-robot systems in the obstacle environment, compared to a commonly-used method.

In the future, we will consider developing distributed algorithms for the scenario of large-scale robots to improve overall computational efficiency. The algorithm proposed in this paper is limited to the static obstacle environment, and an improved algorithm will be applied to the dynamic environment in the future.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] J.C. Derenick, J.R. Spletzer, Convex optimization strategies for coordinating large-scale robot formations, IEEE Trans. Robot. 23 (6) (2007) 1252–1259.

[2] M. Turpin, N. Michael, V. Kumar, Decentralized formation control with variable shapes for aerial robots, in: 2012 IEEE International Conference on Robotics and Automation, IEEE, 2012, pp. 23–30.

[3] Q. Li, 2019 world drone conference shenzhen held thousands of drones unveiled, 2019, http://sz.cnr.cn/szfwgb/szyw/20190621/t20190621_524658879.shtml accessed June 21, 2019.

[4] H. So, Drone + unmanned vessel + unmanned vehicle, 2018 CCTV spring festival evening stunning technology fan, 2018, http://www.sohu.com/a/223445997_204321 accessed February 22, 2018.

[5] Z. Ma, S. Akella, Coordination of droplets on light-actuated digital microfluidic systems, in: 2012 IEEE International Conference on Robotics and Automation, IEEE, 2012, pp. 2510–2516.

[6] V. Shekar, M. Campbell, S. Akella, Towards automated optoelectrowetting on dielectric devices for multi-axis droplet manipulation, in: 2013 IEEE International Conference on Robotics and Automation, IEEE, 2013, pp. 1439–1445.

[7] J. Vaishnav, A.B. Uday, T. Poulose, Pattern formation in swarm robotic systems, in: 2018 2nd International Conference on Trends in Electronics and Informatics (ICOEI), IEEE, 2018, pp. 1466–1469, http://dx.doi.org/10.1109/ICOEI.2018.8553906.

[8] J. González-Sierra, A. Dzul, H. Ríos, Robust sliding-mode formation control and collision avoidance via repulsive vector fields for a group of Quad-Rotors, Internat. J. Systems Sci. 50 (7) (2019) 1483–1500.

[9] H.-a. Yang, S. Cao, L. Bai, Z. Zhang, J. Kong, A distributed and parallel self-assembly approach for swarm robotics, Robot. Auton. Syst. 118 (2019) 80–92, http://dx.doi.org/10.1016/j.robot.2019.04.011.

[10] M. Rubenstein, A. Cornejo, R. Nagpal, Programmable self-assembly in a thousand-robot swarm, Science 345 (6198) (2014) 795–799, http://dx.doi.org/10.1126/science.1254295.

[11] E.G. Hernández-Martínez, E. Aranda-Bricaire, F. Alkhateeb, E. Maghayreh, I. Doush, Convergence and Collision Avoidance in Formation Control: A Survey of the Artificial Potential Functions Approach, INTECH Open Access Publisher Rijeka, Croatia, 2011.

[12] J. Alonso-Mora, A. Breitenmoser, M. Rufli, R. Siegwart, P. Beardsley, Image and animation display with multiple mobile robots, Int. J. Robot. Res. 31 (6) (2012) 753–773, http://dx.doi.org/10.1177/0278364912442095.

[13] S. Agarwal, S. Akella, Simultaneous optimization of assignments and goal formations for multiple robots, in: 2018 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2018, pp. 6708–6715, http://dx.doi.org/10.1109/ICRA.2018.8460542.

[14] S. Akella, Assignment algorithms for variable robot formations, in: 12th International Workshop on the Algorithmic Foundations of Robotics (WAFR 2016), 2016.

[15] M. Minelli, J. Panerati, M. Kaufmann, C. Ghedini, G. Beltrame, L. Sabattini, Self-optimization of resilient topologies for fallible multi-robots, Robot. Auton. Syst. 124 (2020) 103384, http://dx.doi.org/10.1016/j.robot.2019.103384.

[16] G. Li, L. Dong, H. Xu, Y. Lin, Research on region coverage approach with swarm robots, Jiqiren/robot 39 (5) (2017) 670–679.

[17] J. Cortes, S. Martinez, T. Karatas, F. Bullo, Coverage control for mobile sensing networks, IEEE Trans. Robot. Autom. 20 (2) (2004) 243–255, http://dx.doi.org/10.1109/TRA.2004.824698.

[18] J. Van den Berg, S.J. Guy, M. Lin, D. Manocha, Reciprocal n-body collision avoidance, in: Robotics Research, Springer, 2011, pp. 3–19, http://dx.doi.org/10.1007/978-3-642-19457-3_1.

[19] J. Van den Berg, M. Lin, D. Manocha, Reciprocal velocity obstacles for real-time multi-agent navigation, in: 2008 IEEE International Conference on Robotics and Automation, IEEE, 2008, pp. 1928–1935, http://dx.doi.org/10.1109/ROBOT.2008.4543489.

[20] H.W. Kuhn, The hungarian method for the assignment problem, Naval Res. Logist. Q. 2 (1–2) (1955) 83–97, http://dx.doi.org/10.1002/nav.20053.

[21] A. Richards, J.P. How, Aircraft trajectory planning with collision avoidance using mixed integer linear programming, in: Proceedings of the 2002 American Control Conference (IEEE Cat. No. CH37301), IEEE, 2002, pp. 1936–1941, http://dx.doi.org/10.1109/ACC.2002.1023918.

[22] Y. Hu, Operations Research Course, Tsinghua University Press, 2012.

[23] I. Ilog, Cplex user's manual, 2002.

Fangfang Zhang received the B.E. and M.E. degrees in applied mathematics from Shandong University, Jinan, in 2008 and 2011, respectively. He received the Ph.D. degree in control science and engineering from Shandong University, in 2015. He is currently associate professor at Zhengzhou University, Zhengzhou, Henan, China. His research interests include optimal control of multi-agent systems, multi-robot formation, machine vision.

Tingting Wang received the B.S. degree from the North University of China, China, in 2017, where she is currently pursing the M.S. degree with the School of Electrical Engineering, Zhengzhou University, China. Her research interests include multi-agent, pattern formation and nonlinear programming.

Qiyan Li received the B.S. degree from the North China University of Water Resources and Electric Power, China, in 2018, where he is currently pursing the M.S. degree with the School of Electrical Engineering, Zhengzhou University, China. His research interests include visual SLAM, motion detection, image processing and deep learning.

Jianbin Xin obtained M.Sc. degree in Control Science and Engineering from Xi'an Jiaotong University, China in 2010. In 2015, he received Ph.D. degree specialized in operational control of automated container terminals from Delft University of Technology The Netherlands Currently, he is an associate professor in School of Electrical Engineering at Zhengzhou University, China. His research interests include modeling and control of smart logistics systems and hybrid systems control.