COMPLEX SYSTEM MODELING AND SIMULATION
ISSN 2096-9929 03/06 pp 33–49
Volume 4, Number 1, March 2024
DOI: 10.23919/CSMS.2023.0022

Multi-Objective Rule System Based Control Model with Tunable Parameters for Swarm Robotic Control in Confined Environment

Yuan Wang, Lining Xing, Junde Wang, Tao Xie, and Lidong Chen*

Abstract: Enhancing the adaptability of Unmanned Aerial Vehicle (UAV) swarm control models to cope with different complex working scenarios is an important issue in this research field. To achieve this goal, control model with tunable parameters is a widely adopted approach. In this article, an improved UAV swarm control model with tunable parameters namely Multi-Objective O-Flocking (MO O-Flocking) is proposed. The MO O-Flocking model is a combination of a multi rule control system and a virtual-physical-law based control model with tunable parameters. To achieve multi-objective parameter tuning, a multi-objective parameter tuning method namely Improved Strength Pareto Evolutionary Algorithm 2 (ISPEA2) is designed. Simulation experiment scenarios include six target orientation scenarios with different kinds of objectives. Experimental results show that both the ISPEA2 algorithm and MO O-Flocking control model have good performance in their experiment scenarios.

Key words: swarm robotics; flocking model; parameter tuning; multi-objective optimization; heuristics

1 Introduction

In recent years, Swarm Robotics (SR) has attracted many research interests[1]. In an SR, complex tasks are executed by several automatic running robots. Though the SR concept seems to be similar to multi-robot system, there are some main differences. According to Şahin[2], the differences between multi-robot systems and SRs include:

(1) Robots must be physically embodied and situated.

(2) A minimum robot group size of 10 to 20 must be satisfied.

(3) Each robot in SR must have a simple structure and limited functions.

(4) Each robot in SR must be homogenous.

Due to Şahin's[2] SR design principles, there is no explicit leader in an SR. Each robot in an SR embodies a homogenous individual controller. Through this controller, individuals in SR achieve various behaviors (target orientation, collation avoidance, etc.). Thus, individual controller design is a crucial issue for an SR. A deliberately designed individual controller can help SR achieve its goals efficiently.

Many studies have been made on SR individual controller design. Reviews of these methods can be seen in Refs. [1, 3]. Among these methods, virtual-physical-law based design is a widely applied method for SRs that require a certain formation. The very first article that introduces virtual-physical based design method is Ref. [4]. In this article, Khatib[4] introduced a concept called artificial potential field. In works that apply virtual-physical-law based design methods, robots are subject to at least two main forces: attractive force form targets (target orientation) and repulsive force from obstacles (obstacle avoidance). According to Gazi and Passino[5], virtual-physical-law based design methods have three advantages:

(1) Robot behaviors are decided by using one

• Yuan Wang, Junde Wang, Tao Xie, and Lidong Chen are with the Beijing Institute of Advanced Studies, College of Advanced Interdisciplinary Studies, National University of Defense Technology, Beijing 100084, China. E-mail: wy1020395067@hotmail.com; wangjunde@nudt.edu.cn; xietao2016@sina.com; nudtdong11@163.com.
• Lining Xing is with the Key Laboratory of Collaborative Intelligence Systems, Ministry of Education, Xidian University, Xi'an 710071, China. E-mail: xinglining@gmail.com.
* To whom correspondence should be addressed.
※ This article was recommended by Associate Editor Xinyu Li.
Manuscript received: 2023-09-04; revised: 2023-11-19; accepted: 2023-12-04

mathematical rule instead of multiple rules or behaviors.

(2) The obtained behaviors can be combined by using vectorial operations.

(3) The robustness and stability of virtual-physical-law based controllers can be analyzed by using theoretical tools.

Studies about virtual-physical-law based design method are mostly carried out on physical rule designs. Spears et al.[6] proposed the most general virtual-physical-law based design SR control model. Fiorini and Shillert[7] proposed an important concept called Velocity Obstacle (VO). Berg et al.[8] extended VO concept by taking account the reactions between neighbor individuals. Following these articles, Santos et al.[9] proposed a new concept called Virtual Group Velocity Obstacle (VGVO). VGVO concept defines a group of robots as a virtual obstacle. This keeps robots from other groups mingling with current group of robots. La et al.[10] implemented reinforcement learning method into an artificial potential-field based controller which enables multiple robots to move together and to avoid obstacles. Woods and La[11] proposed a new drone controller for dynamic target tracking mission. In this article, they extended formal potential-field based controller with taking relative velocity into account. A real-world scenario with Parrot ARDrone 2.0 is used to test the performance of proposed controller. Ha et al.[12] and Meier et al.[13] extended virtual-physical-law based design method with velocity alignment purpose. To achieve this goal, a power law of reducing velocity difference among nearby individuals is applied. Xu et al.[14] proposed an SR formation reconfiguration model with Dynamic Reference Point (DRP). The DRP controller performs better on obstacle avoidance. Harder and Lauderbaugh[15] proposed a general model for SR formation specification. This model enables to control variable size of drones. Alfeo et al.[16] proposed an evolution-based flocking model to solve the target search problem. The control model is better in terms of swarm formation, search efficiency, and scalability. However, the emphasis on reducing the flight time results in the lack of simplicity and universality. Li and Fang[17] proposed a pigeons behavior based flocking model for free-flying scenarios with different kinds of obstacles. But this method can only deal with swarm formation control of simple formations, such as V-shape, linear-shape, and circular-shape. Zhang et al.[18] proposed an adaptive control model for Unmanned Aerial Vehicle (UAV) swarm formation control in 3D space with wind field disturbances. The wind field model is decoupled into time-varying disturbances in three directions of space. Pyke and Stark[19] proposed a UAV swarm control model called p-drones, which is a combination of Particle Swarm Optimization (PSO) and virtual-physical-law based control model. This model has better performance in navigation and collision avoidance compared with existing algorithms. Li et al.[20] introduced Light Transmission Model (LTM) into potential field control model to improve the collision avoidance ability of UAV swarm. The new control model has good advantages in collision avoidance, rapid formation reconstruction, and formation maintenance.

Though virtual-physical-law based SR controller has the advantage of easy implementation, it may lack robustness in different working scenarios. The reason is that only one mathematical rule is applied and normally a single objective fitness function is used to evaluate controller performance. On the other hand, the expansion of the UAV swarm scale and the high dynamic environment inevitably lead to the growth of the complexity of the working scenarios[21]. Hence, improving the robustness of the control model in different working scenarios is increasingly crucial.

To overcome the weakness of lacking robustness in different working scenarios, one way is to use tunable parameters in SR control model. Using tunable parameters in SR control model is an important direction of improving the performance of SR control model in different working scenarios. To our best knowledge, the very first method using SR control model with tunable parameters is Hettiarachchi[22]. In this article, Hettiarachchi[22] proposed an SR control model for obstacle avoidance based on Lennard-Jones potential function. An evolutionary algorithm is implemented to tune parameters off-line. Pugh and Martinoli[23] proposed an SR control model with an on-line parameter tuning method based on PSO. Genetic Algorithm (GA) is used as comparison. Experiment results show that SR control model with PSO parameter tuning algorithm achieves a better degree of diversity in the swarm. Folino et al.[24] proposed a self-adaptive SR controller for decentralized SR clustering. This approach avoids the sequential search of canonical clustering algorithms and permits a scalable implementation. Cetin et al.[25] implemented the fuzzy logic controller into UAV swarm control model for the

control of the altitude, speed, and heading. Yang et al.[26] proposed an SR control model with tunable parameter for agent aggregation behaviors. The model in this article is an improved Vicsek Model with a parameter that controls the strength of influence of robots in a swarm. Zhao et al.[27] proposed an improved adaptive-velocity self-organized SR control model. This model has better convergence performance under high speed. The parameter space of this model is also analyzed in this article. Very recently, Vásárhelyi et al.[28] proposed a new evolution-based SR controller. In this article, they introduced a virtual-physical-law based SR individual controller with 11 tunable parameters. To achieve model optimization, a multi-objective optimization algorithm called Covariance Matrix Adaption Evolutionary Strategy (CMA-ES) is applied. Except for the intensive computation cost (2 to 6 days for a single run on the Atlasz supercomputer cluster), this controller has good performance on scenarios of circular flying, diagonal flying, and free flying with multi obstacles.

Apart from these articles, articles that mainly focus on the UAV swarm control algorithm with tunable parameters are rare. However, in complex practical working scenarios, UAV swarm control model must simultaneously satisfy different decision preferences and be able to achieve multiple mission objectives (target orientation, obstacle avoidance, etc.). Therefore, it is necessary to introduce extensible UAV swarm control model to deal with different working conditions and multi-objective optimization mechanism to satisfy different decision preferences. In this article, we mainly focus on two objectives: first, a UAV swarm control structure which is a combination of a virtual-physical-law based control model with tunable parameters and a multi-objective parameter tuning algorithm is proposed; second, the effectiveness of different parameter tuning algorithms is evaluated. The control model is an extension version of O-Flocking, which is an expandable virtual-physical-law based UAV swarm control model proposed by Ma et al.[29] We first extend the original O-Flocking control model into a 4-rule control model to deal with different working conditions. The proposed algorithm is named Multi-Objective O-Flocking (MO O-Flocking). Then, we introduce a new multi-objective optimization algorithm namely Improved Strength Pareto Evolutionary Algorithm 2 (ISPEA2) to achieve parameter tuning of MO O-Flocking. ISPEA2 is a combination of Strength Pareto Evolutionary Algorithm 2 (SPEA2) and a dynamic diversity control strategy (Dynamic Diversity Control in Genetic Algorithm (DDCGA)). Two newly designed recombination heuristics are also implemented into ISPEA2 to improve the algorithm efficiency. At last, experiments are designed to test the performance of both the ISPPEA2 algorithm and the MO O-Flocking control model. Experiment scenarios include six target orientation scenarios with different kinds of obstacles. The ISPEA2 is compared with several widely applied multi-objective optimization algorithms including SPEA2, Non-dominated Sorting Genetic Algorithm II (NSGA II), Non-dominated Sorting Genetic Algorithm III (NSGA III), Multi-Objective Evolutionary Algorithm based on Decomposition (MOEA/D), and Imperialist Competitive Algorithm (ICA). And the MO O-Flocking control model with tuned parameters is compared with the O-Flocking control model, Reynolds model, and two subversion of Artificial Potential Field (APF) models to evaluate the target orientation and obstacle avoidance abilities.

The rest of this article is organized as follows. In Section 2, we give a brief introduction of MO O-Flocking control model. In Section 3, we introduce the ISPEA2 algorithm. In Section 4, simulation experiment designs are illustrated, and experiment results are discussed. In Section 5, we give out some crucial conclusions.

2 SR Control Model

In this section, we give a brief introduction about the MO O-Flocking control model applied in this article. MO O-Flocking is a virtual-psychical-law based SR control model. This model consists of two parts: one is a basic virtual-physical-law based control model namely O-Flocking, which controls the behavior of individuals under certain circumstance; another is a rule system which helps individuals achieve multi behaviors in different working conditions. The O-Flocking control model has been approved to be efficient in target orientation and collision avoidance working scenarios. In this article, we extend the O-Flocking control model with a 4-rule control system to enhance the adaptability of O-Flocking in different circumstances. The newly designed control model, MO O-Flocking, is highly flexible because the behaviors of individuals controlled by MO O-Flocking can be simply changed by modifying the rule system of MO O-Flocking controller. The introduction of MO O-Flocking is as follows.

2.1 Basic assumption

In this article, we mainly focus on two basic SR missions: target orientation and collision avoidance. Some basic assumptions must be followed to achieve these goals while some crucial real-world constraints are satisfied:

(1) Individuals in SR must fly in a bounded arena, flying out of this arena is forbidden.

(2) Collision with other individuals in SR is forbidden.

(3) Collision with obstacles in SR is forbidden.

(4) The maximum velocity of an individual in SR is limited.

(5) Mission of the SR is to start at the beginning area, fly through the arena, and arrive at the target area.

2.2 Practical implementation

To achieve MO O-Flocking controller, we design a four-layer control architecture. Figure 1 shows the architecture of MO O-Flocking.

The MO O-Flocking controller first receives information collected by the onboard sensing hardware in sensing layer. Then, the rule-based system in decision layer decides the flying direction and velocity. The action layer receives the new direction and velocity, then the flying condition of the UAV is changed. The evolution layer collects the flying information and changes the parameters in the rule-based system to find better parameter sets. The rule-based system is updated when a better parameter set is found. In this section we introduce the rule-based system called MO O-Flocking control model. Then, the multi-objective optimization algorithm called ISPEA2 in evolution layer is introduced in Section 3.

2.3 Objective detection pattern

The objective detection pattern of MO O-Flocking includes two parts: detection pattern of other individuals in SR and detection pattern of obstacles. Figure 2 shows the detection pattern of other individuals of MO O-Flocking.

In MO O-Flocking, individuals receive three forces from other individuals, namely repulsion, alignment, and attraction. In Fig. 1, individuals detected in z_{rep} apply repulsion force to individual i. The range of z_{rep} is described as R_0. Individuals detected in z_{ali} apply alignment force to individual i. The range of z_{ali} is described as R_1. Individuals detected in z_{att} apply attraction force to individual i. The range of z_{att} is described as R_2. Figure 3 shows the obstacle detection pattern of MO O-Flocking.

In Fig. 3, obstacles detected in z_{obs} apply obstacle repulsion force to individual i. The range of z_{obs} is described as R_3.

2.4 Velocity updating rule

The MO O-Flocking model updates velocity of an individual when different objectives are detected. The velocity updating rule is defined as Eq. (1).

$$v_i(t+1) = v_i(t) + \Delta v_i,$$

$$\Delta v_i = a\Delta v_i^{\text{rep}} + b\Delta v_i^{\text{ali}} + c\Delta v_i^{\text{att}} + d\Delta v_i^{\text{tar}} + e\Delta v_i^{\text{obs}} \quad (1)$$

where $v_i(t)$ is the velocity of individual i at time t. Δv_i^{rep} is the velocity change brought by repulsion force. Δv_i^{ali} is the velocity change brought by alignment force. Δv_i^{att} is the velocity change brought by attraction force. Δv_i^{tar} is the velocity change brought by the target orientation force. Δv_i^{obs} is the velocity change brought by obstacle repulsion force. Five parameters (i.e., $\{a, b, c, d, e\}$) are used to control the importance of each force. With the help of these five forces, individuals in SR fly through the arena and avoid collision with obstacles or each other. Detailed descriptions of these five forces are as follows:

When individual i detects individual j in its z_{rep}, a velocity change Δv_i^{rep} is applied to individual i following Eq. (2).

$$\Delta v_{ij}^{\text{rep}} = (R_0 - r_{ij}) \times \frac{p_i - p_j}{r_{ij}} \quad (2)$$

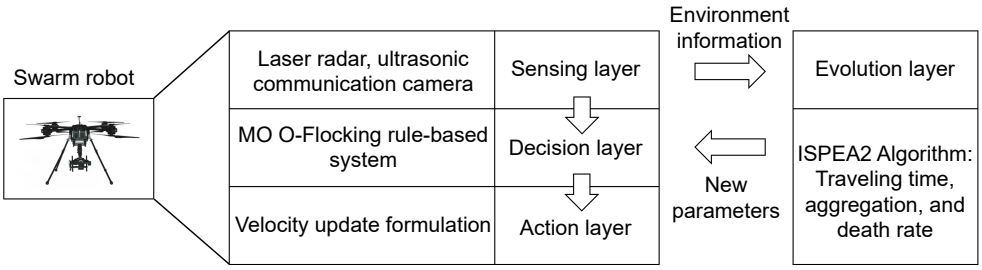

Fig. 1 MO O-Flocking practical framework.

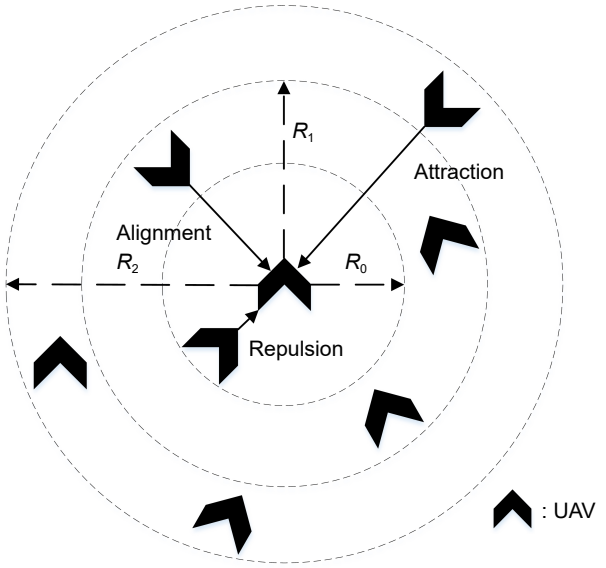

Fig. 2 Detection pattern of other individuals.

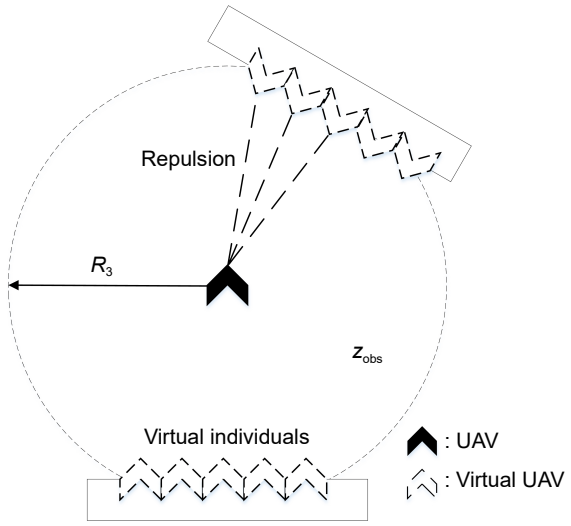

Fig. 3 Detection pattern of obstacles.

where r_{ij} is the distance between individual i and individual j. p_i and p_j are the current positions (represented as vectors) of individual i and individual j, respectively. Thus, $\Delta v_{ij}^{\text{rep}}$ can be calculated by using Eq. (3).

$$\Delta v_i^{\text{rep}} = \sum_{j=1, j\neq i}^{N} \Delta v_{ij}^{\text{rep}}, \quad \text{for } r_{ij} < R_0 \tag{3}$$

where N denotes the number of individuals in an SR. The repulsion force prevents individuals from colliding with each other.

Agent alignment force is a power that minimizes the velocity direction difference of individuals in range R_1. In MO O-Flocking model, a typical physical law introduced in previous work[29] is applied. This physical law is shown as Eq. (4).

$$\Delta v_i^{\text{ali}} = \frac{1}{n_{\text{ali}}} \sum_{j=1, j\neq i}^{N} \frac{v_j}{|v_j|}, \quad \text{for } R_0 < r_{ij} < R_1 \tag{4}$$

where v_j is the velocity vector of individual j. n_{ali} is the number of individuals in range R_1 of individual i.

The attraction force is a power that keeps individuals in SR moving in a reasonable range thus the swarm connection maintenance is achieved. The attraction force is defined by using Eq. (5).

$$\Delta v_{ij}^{\text{att}} = \frac{1}{R_2 - r_{ij}} \times \frac{p_j - p_i}{r_{ij}} \tag{5}$$

And the total attraction effect of individual i can be calculated by using Eq. (6).

$$\Delta v_i^{\text{att}} = \sum_{j=1, j\neq i}^{N} \Delta v_{ij}^{\text{att}}, \quad \text{for } R_0 < r_{ij} < R_1 \tag{6}$$

The detection of obstacles has some differences from the detection of other individuals, which is shown in Fig. 2. When an obstacle is detected by individual i, virtual agents are assigned to the surface of this obstacle. And these virtual individuals repulse individual i. Thus, when an obstacle occurs in z_{obs} of individual i, the velocity of individual i is affected by using Eq. (7).

$$\Delta v_i^{\text{obs}} = \sum_{m=1}^{M} (R_3 - r_{im}) \times \frac{p_i - p_m}{r_{im}} \tag{7}$$

where Δv_i^{obs} is the obstacle repulsion force of individual i. M is the set of virtual individuals observed by individual i.

Finally, the target orientation force is defined by using Eq. (8).

$$\Delta v_i^{\text{tar}} = \frac{p_t - p_i}{r_{it}} \tag{8}$$

where Δv_i^{tar} is the target orientation force. p_t is the position of the target. r_{it} is the distance between individual i and target t.

And, to satisfy the maximum velocity limitation, a maximum velocity cutoff $v^{\max}$ is applied. Thus, the final velocity updating rule is defined as Eq. (9).

$$v_i(t+1) = \frac{v_i(t+1)}{|v_i(t+1)|} \times \min\left(|v_i(t+1)|, v^{\max}\right) \tag{9}$$

In the MO O-Flocking control model, a rule system is also applied to help achieve multi behaviors of individuals. In this article, 4 simple and efficient rules are implemented. These 4 rules are listed in Eq. (10).

$$\begin{cases}\text{Rule 1}: i=0,\ o=0;\\ \text{Rule 2}: i=0,\ o\neq 0;\\ \text{Rule 3}: i\neq 0,\ o=0;\\ \text{Rule 4}: i\neq 0,\ o\neq 0\end{cases} \tag{10}$$

In rule system (Eq. (10)), i is the number of individuals in z_{rep}, and o is the number of obstacles in z_{obs}. With these 4 rules, the final velocity updating rules are defined as Eq. (11).

$$\Delta v_i = \begin{cases} a_1\Delta v_i^{\text{rep}} + b_1\Delta v_i^{\text{ali}} + c_1\Delta v_i^{\text{att}} + d_1\Delta v_i^{\text{tar}} + e_1\Delta v_i^{\text{obs}}, \text{if Rule 1};\\ a_2\Delta v_i^{\text{rep}} + b_2\Delta v_i^{\text{ali}} + c_2\Delta v_i^{\text{att}} + d_2\Delta v_i^{\text{tar}} + e_2\Delta v_i^{\text{obs}}, \text{if Rule 2};\\ a_3\Delta v_i^{\text{rep}} + b_3\Delta v_i^{\text{ali}} + c_3\Delta v_i^{\text{att}} + d_3\Delta v_i^{\text{tar}} + e_3\Delta v_i^{\text{obs}}, \text{if Rule 3};\\ a_4\Delta v_i^{\text{rep}} + b_4\Delta v_i^{\text{ali}} + c_4\Delta v_i^{\text{att}} + d_4\Delta v_i^{\text{tar}} + e_4\Delta v_i^{\text{obs}}, \text{if Rule 4} \end{cases} \tag{11}$$

In Eq. (11), 20 parameters ($\{a_1, b_1, c_1, d_1, e_1, a_2, b_2, c_2, d_2, e_2, \ldots, a_4, b_4, c_4, d_4, e_4\}$) need to be tuned. Considering the intensiveness of computational time cost of traditional parameter tuning methods implemented in intelligent swarm parameter tuning, e.g., manual tuning or parameter sweeping, heuristic algorithm may be a more suitable way which can find feasible solutions with acceptable time costs. In this article, several population-based multi-objective heuristics are implemented to the control model. We first introduce an efficient and robust heuristic namely ISPEA2 which is a sub version of SPEA2.

3 ISPEA2 Algorithm

ISPEA2 algorithm is a subversion of the widely applied SPEA2 algorithm, which is first proposed in Zitzler et al.[30] In SPEA2, parents are selected from both solutions in the current iteration and Pareto optimal solutions found in past iterations. It is generally admitted that SPEA2 has an advantage of better solution distributions on Pareto Frontier especially when the number of objectives increases. And, to avoid ISPEA2 stuck in local optimum, a dynamic diversity control technology inspired by Chang et al.[31] is also implemented. The algorithm structure of IPSEA2 is shown in Fig. 4.

3.1 Encoding

We extract 20 parameters from our flocking model for parameter tuning. The ranges of these parameters are [0, 1]. Thus, in ISPEA2, each solution is represented as a chromosome containing 20 variables. Each variable in a chromosome is a parameter in the MO O-Flocking model described in Section 2. Figure 5 shows an example of chromosomes used in ISPEA2.

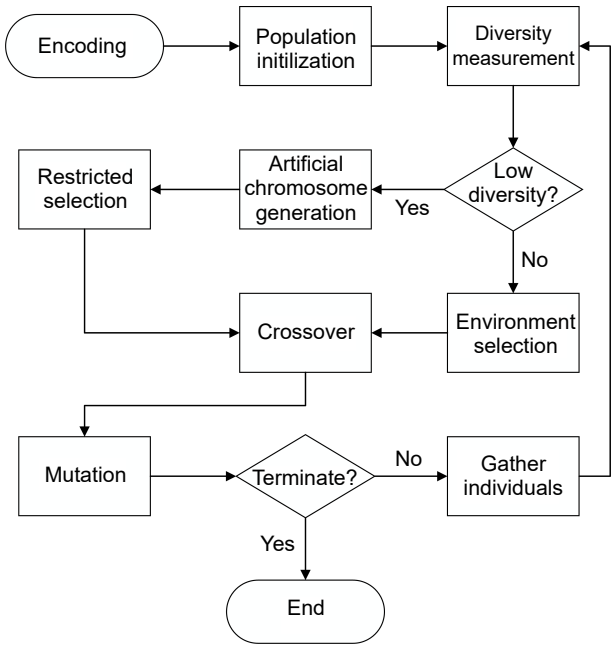

Fig. 4 Algorithm structure of ISPEA2.

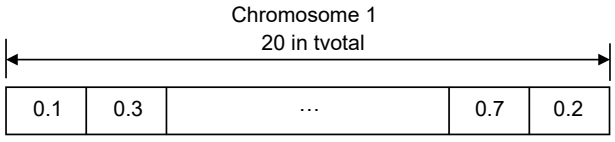

Fig. 5 Chromosome example.

3.2 Recombination

Let po_{ite} be the population of iteration ite. $\text{po}_{\text{ite}} = \{c_1, c_2, c_3, \ldots, c_p\}$, where $\{c_1, c_2, c_3, \ldots, c_p\}$ are chromosomes, and p is the maximum size of po_{ite}. To help to generate new solutions, recombination and mutation methods are applied to chromosomes in every iteration's population. Newly generated chromosomes are called off-springs and chromosomes that are used to generate off-springs are called parents. In the recombination phase, two recombination operators are implemented to generate off-springs. One operator is the random exchange operator. In the random exchange operator, up to 5 genes in Parent 1 are randomly chosen and then these genes are exchanged with genes on the same loci of Parent 2. The other is the rule exchange operator. In this operator, all 5 genes representing the same rule of the MO O-Flocking model are chosen and exchanged between 2 parents. Figure 6 shows the random exchange operator and rule exchange operator. Parents are selected by using tournament selection.

3.3 Mutation

In the mutation phase, a random mutating operator is

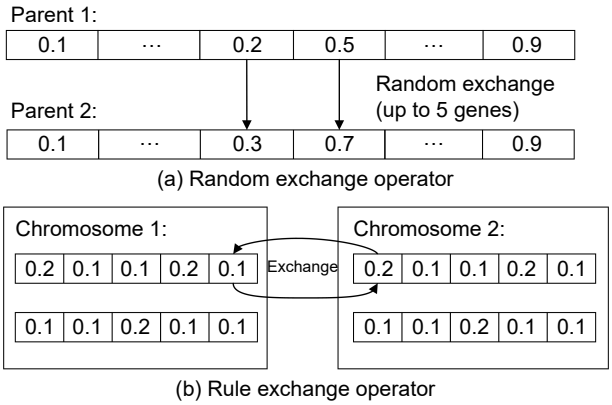

Fig. 6 Crossover operators.

implemented to generate off-springs. This operator randomly picks up to 3 genes in a chromosome. Then, these genes are assigned values randomly generated in [0, 1].

3.4 Environment selection

In ISPEA2, an external archive $\overline{\text{po}}_{\text{ite}}$ is maintained to generate $\text{po}_{\text{ite+1}}$, which is the population of the next generation. This external archive records non-dominated solutions found in past iterations. To generate $\text{po}_{\text{ite+1}}$, first, all Non-Dominated (ND) solutions in po_{ite}, and all off-springs generated in iteration ite is copied into $\text{po}_{\text{ite+1}}$. If the size of $\text{po}_{\text{ite+1}}$ exceeds a threshold $\overline{N}$, a density-based heuristic is applied to reduce the size of $\text{po}_{\text{ite+1}}$. If the size of $\text{po}_{\text{ite+1}}$ is smaller than $\overline{N}$, then fill $\text{po}_{\text{ite+1}}$ with dominated solutions in po_{ite}, $\overline{\text{po}}_{\text{ite}}$, and off-springs of iteration ite. The designs of this external archive can be seen in Zitzler et al.[30]

3.5 Dynamic diversity control

A common hypothesis is that maintaining a relatively high diversity is important for avoiding premature convergence and escaping from local optimum. In ISPEA2, another external archive $\overline{\text{ac}}_t$ is maintained to record solutions with acceptable fitness and high diversity in iteration ite. When the average diversity of po_{ite} drops down to a certain threshold, we generate artificial chromosomes from $\overline{\text{ac}}_t$ and inject these artificial chromosomes into the mating pool to increase the average diversity. In this paper, a dynamic diversity control method inspired by DDCGA[31] is implemented. This method first measures the population diversity of po_{ite} by using an entropy-based criterion. Artificial chromosomes are injected into mating pool to increase the population diversity if the population diversity reaches a threshold d_{low}. Detailed description of this method is described as follows:

Population diversity measure. In each iteration, we first measure the average diversity of po_{ite} by using Eq. (12).

$$D(P) = \frac{1}{|L|\text{ps}} \sum_{i=1}^{\text{ps}} \sqrt{\sum_{j=1}^{N} \left(s_{ij} - \overline{s}_j\right)^2} \tag{12}$$

where $|L|$ is the length of the diagonal of the solution space. ps is the population size. s_{ij} is the value of gen j of chromosome i. $\overline{s}_j$ is the average value of gen j.

Theoretically, the selection and recombination operators are diversity reduction operators. However, maintaining a relatively high diversity can help population-based heuristics get better results. Thus, when the population diversity is above a certain threshold d_{low}, recombination and selection operators are implemented. When the population diversity is under d_{low}, operators that are designed to increase the population diversity are applied. These operators are described as follows:

Chromosome diversity contribution measure. When the population diversity is under d_{low}, we first measure the contribution to the diversity of every single chromosome in po_{ite}. To achieve this goal, an entropy-based diversity measurement is implemented. In this measurement, the range of each gen (i.e., [0, 1]) is distributed into 10 intervals (from [0, 0.1) to [0.9, 1]). Then, Eq. (13) is applied to calculate the probability that values signed to each gen fall into these intervals.

$$\text{pr}_{iv} = \frac{\text{na}_{iv}}{\text{ps}} \tag{13}$$

where pr_{iv} means the possibility that value on gen i falls into interval v. na_{iv} is the number that value on gen i falls into interval v. ps is the population size. Then, Eq. (14) is applied to measure the diversity of gen i.

$$H_i = -\sum_{v=1}^{V} \text{pr}_{iv} \ln(\text{pr}_{iv}) \tag{14}$$

where H_i is the diversity of gen i. V is the set of intervals. When the diversities of all gens are available, the diversity of po_{ite} is then measured by using the arithmetic average of H_i, which is shown in Eq. (15).

$$\text{PCD}_{\text{po}_{\text{ite}}} = \frac{\sum_{i=1}^{N_{\text{po}_{\text{ite}}}} H_i}{N_{\text{po}_{\text{ite}}}} \tag{15}$$

where $\text{PCD}_{\text{po}_{\text{ite}}}$ is the population diversity measured by entropy measurement. $N_{\text{po}_{\text{ite}}}$ is the problem

dimensionality. Then, Eq. (16) is applied to measure a single chromosome's contribution to the whole population.

$$\mathrm{CD}(i) = \mathrm{PCD}_{\mathrm{po}_{\mathrm{ite}}} - \mathrm{PCD}_{\mathrm{po}_{\mathrm{ite}}-i} \tag{16}$$

where CD(i) donates the contribution of chromosome i to the population. $\mathrm{PCD}_{\mathrm{po}_{\mathrm{ite}}-i}$ donates the population diversity measured by entropy measurement of population $\mathrm{po}_{\mathrm{ite}}$ without chromosome i.

Artificial chromosome generator. When the average diversity of the population drops down to d_{low}, artificial chromosomes are generated and injected into the mating pool to increase the population diversity using the Artificial Chromosome Generator (ACG). ACG uses chromosomes stored in $\overline{\mathrm{ac}}_t$. First, $\overline{\mathrm{ac}}_t$ collects all non-dominated solutions found by ISPEA2. When the number of chromosomes in $\overline{\mathrm{ac}}_t$ exceeds upper bound $\overline{A}$, chromosomes which have low diversity contribution are removed from $\overline{\mathrm{ac}}_t$ until the number of chromosomes in $\overline{\mathrm{ac}}_t$ is equal to $\overline{A}$. Then, $\overline{\mathrm{ac}}_t$ follows a probabilistic rule to generate artificial chromosome by using Eq. (17).

$$P_{ij}(t) = \frac{X_{ij}^t}{\mathrm{ps}} \tag{17}$$

where $P_{ij}(t)$ donates the probability that interval i is chosen to generate a new value on gen j at iteration t for an artificial chromosome. X_{ij}^t is the frequency that value on gen j falls into interval i in $\overline{\mathrm{ac}}_t$. When an interval is chosen, the new value is generated in this interval randomly. Then this value is assigned to gen j of a new artificial chromosome. Every time ACG works, ϑ artificial chromosomes are generated by using Eq. (17) and injected into $\mathrm{po}_{\mathrm{ite}}$. At the same time, ϑ chromosomes are picked and removed from the $\mathrm{po}_{\mathrm{ite}}$ by means of diversity contribution. Thus, the size of $\mathrm{po}_{\mathrm{ite}}$ remains unchanged.

3.6 Evaluation criteria

In this article, three SR control evaluation criteria are considered. The designs of these criteria are described as follows:

Death rate. The death rate is a criterion that evaluates the obstacle avoidance ability of an SR control model. We consider an individual in an SR as a dead individual when it hits another individual, an obstacle, or flies out of the arena. The death rate can be calculated by using Eq. (18).

$$r^{\mathrm{death}} = \frac{N^{\mathrm{death}}}{N} \tag{18}$$

where r^{death} is the death rate. N^{death} is the number of dead individuals during a simulation. It is obvious that r^{death} is a criterion that needs to be minimized.

Aggregation of individuals. Individual aggregation is a criterion that evaluates the connection maintenance ability of an SR. We use the aggregation index to measure the aggregation of individuals. To calculate the aggregation index, we first calculate the virtual centroid of the whole swarm by using Eq. (19).

$$p_c^t = \frac{\sum_{i=1}^{N} p_i^t}{N} \tag{19}$$

where p_c^t is the position of the virtual centroid of SR at simulation time t. p_i^t is the position of individual i at simulation time t. Then, we use a Euclidean distance-based criterion, which is shown in Eq. (20), to evaluate the aggregation of an SR during the whole simulation.

$$d^{\mathrm{agg}} = \frac{\sum_{t=1}^{T}\sum_{i=1}^{N} \left\| p_c^t - p_i^t \right\|_2}{NT} \tag{20}$$

where d^{agg} is the aggregation index. T is the length of the simulation experiment. Thus, d^{agg} donates the average distance between individuals to the virtual centroid of the swarm during the simulation. This is an objective that needs to be minimized.

Traveling time. At last, the average traveling time cost is calculated as

$$\overline{\mathrm{TC}} = \frac{\sum_{i=1}^{N} \mathrm{TC}_i}{N} \tag{21}$$

where $\overline{\mathrm{TC}}$ is the average traveling time cost. TC_i is the traveling time cost of individual i from starting area to the finish area. This is a criterion to evaluate the target orientation ability of an SR control model, and it needs to be minimized.

4 Simulation Experiment

4.1 Experiment design

To evaluate the performances of proposed algorithms, 6 test scenarios are designed. These scenarios include a basic target orientation scenario and target orientation scenarios with obstacles. Three typical types of obstacles are included. These obstacles are square shape obstacle, convex shape obstacle, and tunnel shape obstacle. Figure 7 shows the experiment scenarios.

The experiment configuration details include:

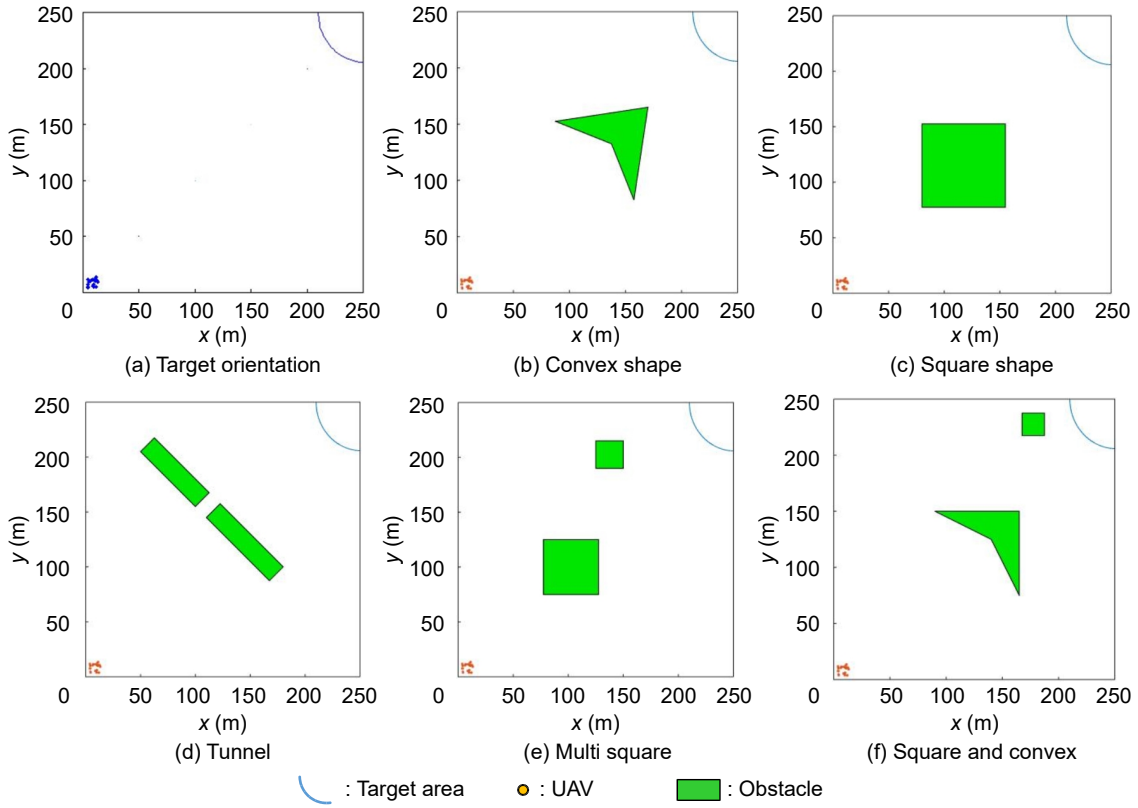

Fig. 7 Illustrations of 6 experiment scenarios.

(1) Every experiment is run 20 times, and the average performances are reported.

(2) The length of the arena is set to 250 m.

(3) The max speed is 5 m/s for real robots.

(4) The size of the swarm robotic is set to 20.

(5) The communication delay t^{del} is set to 1 s.

(6) The inaccuracy of onboard sensor r^{ios} is set to (0, 0.2) m.

(7) The maximum detection range of a single robot is set to (95, 105) m.

(8) R_3 is equal to the maximum detection range of a single robot. R_0=1/3R_3, R_1=2/3R_3, and R_2=R_3.

(9) The simulation length is set to 200 s. In a single simulation run, 1 step is set to be equal to 1 s.

The simulation experiment is divided into two parts. In the first part, algorithm performances of 6 multi-objective optimization heuristics (including ISPEA2, SPEA2, NSGA II[32], NSGA III[33], MOEA/D[34], and ICA[35]) are tested and analyzed. In the second part, the MO O-Flocking model with tuned parameters is compared with some widely applied SR control models. In ISPEA2, SPEA2, NSGA II, NSGA III, and MOEA/D, crossover and mutation operators are exactly the same. The maximum iteration of each algorithm is set to 200. The population size $\overline{N}$ is set to 30. The mutation rate is set to 30%. $\overline{A}$ is set to 30. d_{low} is set to 0.2. ϑ is set to 10.

All tests were run on a computer with an Intel CoreTM i7-6700HQ central processing unit (4 cores, 2.6 GHz) with 16 GB of memory. The code of the experiment platform, test scenarios, and all tested algorithms have been uploaded to https://github.com/Downloadmarktown/Flocking-experiment-platform.

4.2 Parameter tunning algorithm performance analysis

In this part, the performances of all parameter tunning algorithms are reported. The experiment results are listed in Table 1.

In Table 1, “TO” means “target orientation”, which means scenario with mission of target orientation and no obstacle. MS means “multi square”. SC means “square and convex”. “DR” means “death rate”. “AI” means “aggregation index”. “TC” means “travelling cost”. In Table 1, the best result of each objective is shown with bold font. The best and worst analysis shows that in six tested scenarios, ISPEA2 algorithm is better at finding parameter sets with the lowest death rate and the best SR aggregation performance. The NSGA II algorithms has better performance on finding

Table 1 Best/worst analysis results.

Scenario	Index	ISPEA2			SPEA2			NSGA II			NSGA III			ICA			MOEA/D		
		DR	AI	TC	DR	AI	TC	DR	AI	TC	DR	AI	TC	DR	AI	TC	DR	AI	TC
Square	Best	**0.00**	**0.22**	0.34	**0.00**	0.23	0.34	**0.00**	0.29	0.34	**0.00**	0.25	0.34	**0.00**	0.36	0.34	**0.00**	0.29	**0.33**
	Worst	0.25	0.66	0.51	0.25	0.69	0.51	0.25	0.67	0.51	0.25	0.53	0.51	0.00	0.40	0.35	0.20	0.66	0.48
	Average	0.08	0.29	0.39	0.08	0.32	0.40	0.05	0.38	0.38	0.07	0.35	0.39	0.00	0.41	0.35	0.06	0.36	0.39
Convex	Best	**0.00**	**0.27**	0.35	**0.00**	**0.27**	0.37	**0.00**	0.33	**0.34**	**0.00**	0.30	0.36	**0.00**	0.38	0.39	**0.00**	0.32	0.37
	Worst	0.05	0.73	0.42	0.10	0.73	0.47	0.15	0.70	0.49	0.10	0.74	0.45	0.00	0.55	0.40	0.25	0.75	0.54
	Average	0.01	0.44	0.39	0.01	0.46	0.40	0.01	0.54	0.39	0.02	0.48	0.39	0.00	0.43	0.39	0.03	0.41	0.41
TO	Best	**0.00**	0.33	**0.31**	**0.00**	0.34	0.32	**0.00**	0.37	**0.31**	**0.00**	**0.32**	**0.31**	**0.00**	0.38	**0.31**	**0.00**	0.40	**0.31**
	Worst	0.00	0.33	0.31	0.00	0.34	0.32	0.00	0.34	0.31	0.00	0.32	0.31	0.00	0.41	0.40	0.00	0.40	0.31
	Average	0.00	0.35	0.31	0.00	0.38	0.32	0.00	0.37	0.31	0.00	0.33	0.31	0.00	0.43	0.35	0.00	0.43	0.31
Tunnel	Best	**0.00**	**0.26**	**0.33**	**0.00**	0.28	0.34	**0.00**	0.30	**0.33**	**0.00**	0.27	0.34	**0.00**	0.35	0.34	**0.00**	0.29	0.34
	Worst	0.15	0.67	0.45	0.05	0.75	0.38	0.00	0.71	0.35	0.05	0.56	0.38	0.00	0.51	0.35	0.20	0.67	0.48
	Average	0.01	0.42	0.35	0.01	0.45	0.35	0.01	0.49	0.35	0.02	0.43	0.36	0.00	0.40	0.34	0.06	0.37	0.38
MS	Best	**0.00**	**0.21**	0.33	**0.00**	0.23	0.33	**0.00**	0.26	0.33	**0.00**	0.33	**0.31**	**0.00**	0.37	0.33	**0.00**	0.26	0.33
	Worst	0.25	0.29	0.50	0.25	0.31	0.50	0.25	0.38	0.50	0.00	0.33	0.31	0.00	0.47	0.33	0.25	0.34	0.50
	Average	0.10	0.27	0.40	0.08	0.29	0.39	0.06	0.35	0.37	0.00	0.34	0.32	0.00	0.41	0.33	0.09	0.34	0.39
SC	Best	**0.00**	**0.26**	0.38	**0.00**	**0.26**	0.36	**0.00**	0.34	0.34	**0.00**	0.32	**0.31**	**0.00**	0.40	0.38	**0.00**	0.33	0.38
	Worst	0.25	0.57	1.00	0.20	0.68	1.01	0.25	0.69	0.55	0.00	0.32	0.31	0.00	0.52	0.40	0.15	0.67	0.48
	Average	0.02	0.35	0.41	0.04	0.42	0.42	0.01	0.47	0.41	0.00	0.33	0.31	0.00	0.44	0.40	0.02	0.42	0.41

parameter sets with the lowest travelling time.

In this paper, we mainly focus on the multi-objective optimization ability of each algorithm. Thus, we applied Hyper Volume (HV) analysis and C-metric analysis.

HV analysis is an important method to measure the coverage ability of a multi-objective optimization algorithm in the whole solution space. In this paper, we applied a Monte Carlo method based HV indicator since the exact Pareto Frontier of each scenario is unknown. This indicator can be obtained from https://ww2.mathworks.cn/matlabcentral/fileexchange/30785-hypervolume-computation. Figure 8 shows a graphic explanation of HV analysis.

In this paper, the upper bound of each scenario is set to be {1, 1, 1, 1}. The lower bound is set to be equal to the best results of each objective found by all algorithms, which is shown in Table 2.

Table 3 shows the HV analysis results.

The HV analysis results show that the ISPEA2 algorithm has the best performances in 5 scenarios (square, convex, tunnel, multi square, and square and convex). IPSEA2 is followed by SPEA2 and NSGA III. In target orientation scenario, NSGA III algorithm has the best performance (an average HV value of 0.9898), and it is followed by ISPEA2 (an average HV value of 0.9647). The gap of HV value is around 2.5%. The

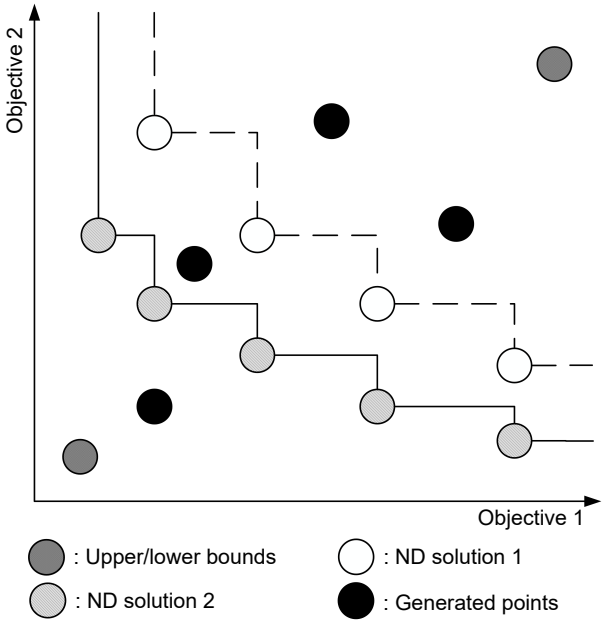

Fig. 8 Example of HV analysis.

Table 2 Lower bound of each scenario.

Scenario	DR	AI	TC
Square	0	0.216	0.334
Convex	0	0.268	0.341
Target orientation	0	0.324	0.314
Tunnel	0	0.259	0.330
Multi square	0	0.214	0.314
Square and convex	0	0.256	0.314

Table 3 HV analysis results.

Scenario	Index	HV value					
		ISPEA2	SPEA2	NSGA II	NSGA III	ICA	MOEA/D
Square	Best	**0.9676**	0.9516	0.8743	0.9298	0.8051	0.8928
	Worst	**0.9254**	0.8632	0.7872	0.6556	0.7296	0.8012
	Average	**0.9509**	0.9198	0.8526	0.8704	0.7541	0.8442
	Coefficient of variants	0.0128	0.0201	0.0249	0.0658	0.0258	0.0248
Convex	Best	**0.9244**	0.9042	0.8459	0.8765	0.7754	0.8496
	Worst	**0.8486**	0.8098	0.7006	0.4479	0.7092	0.7616
	Average	**0.8937**	0.8560	0.7891	0.7673	0.7372	0.8013
	Coefficient of variants	0.0216	0.0288	0.0497	0.1439	0.0226	0.0301
TO	Best	0.9892	0.9648	0.9832	**1.0000**	0.8807	0.8879
	Worst	**0.9072**	0.8428	0.8657	0.8817	0.8253	0.7816
	Average	0.9647	0.9047	0.9284	**0.9898**	0.8560	0.8515
	Coefficient of variants	0.0237	0.0367	0.0442	0.0261	0.0170	0.0380
Tunnel	Best	**0.9842**	0.9574	0.9384	0.9696	0.8554	0.9304
	Worst	**0.9361**	0.9055	0.8216	0.5850	0.7979	0.8274
	Average	**0.9689**	0.9303	0.8899	0.8534	0.8214	0.8868
	Coefficient of variants	0.0125	0.0169	0.0333	0.0988	0.0196	0.0332
MS	Best	**0.9683**	0.9410	0.9032	0.9551	0.7862	0.9032
	Worst	**0.9031**	0.8886	0.7823	0.8336	0.7230	0.7163
	Average	**0.9437**	0.9130	0.8386	0.8865	0.7521	0.8314
	Coefficient of variants	0.0162	0.0159	0.0389	0.0419	0.0246	0.0630
SC	Best	**0.8713**	0.8523	0.7907	0.7975	0.7000	0.7678
	Worst	**0.7886**	0.7318	0.6123	0.4647	0.6506	0.5753
	Average	**0.8302**	0.7794	0.7191	0.7478	0.6802	0.6906
	Coefficient of variants	0.0296	0.0354	0.0673	0.0950	0.0225	0.0671

coefficient of variants of ISPEA2 in all tested scenarios are between 0.01 to 0.03, this means that in all tested scenarios ISPEA2 has good robust performance.

C-metric is another tool that helps compare the multi-objective optimization performances of different multi-objective optimization algorithms. In C-metric analysis, the non-dominated solutions found by all tested algorithms are compared with each other. C-metric analysis helps to find the algorithm which finds the best non-dominated solution set. Figure 9 shows a graphic explanation of C-metric analysis.

Table 4 shows the results of C-metric analysis.

The C-metric analysis shows that except target orientation scenario, the non-dominate solution sets found by ISPEA2 algorithm are better than solution sets found by other algorithms. In target orientation scenario, the best solution set is found by NSGA III, and it is followed by ISPEA2. In convex scenario and tunnel scenario, 17% of solutions in the non-dominate solution set found by ISPEA2 are dominated by other algorithms (NSGA II and NSGA III). However, in

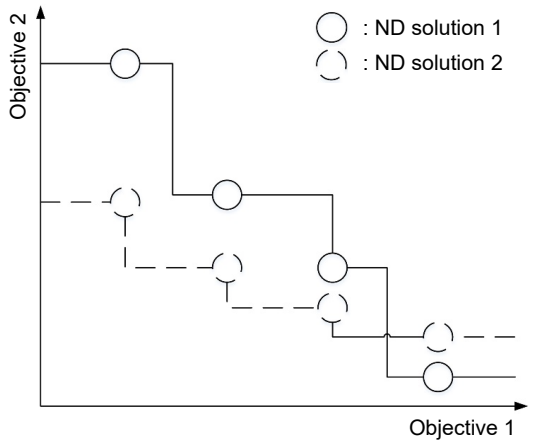

Fig. 9 Example of C-metric analysis.

convex scenario, 86% solutions found by NSGA II are dominated by solutions found by ISPEA2. And in tunnel scenario, 84% solutions found by NSGA III are dominated by solutions found by ISPEA2. In square scenario, only 12% solutions found by ISPEA2 are dominated by solutions found by MOEA/D. But 89% solutions found by MOEA/D are dominated by

Table 4 C-metric analysis results.

Scenario	Algorithm	ISPEA2	SPEA2	NSGA II	NSGA III	ICA	MOEA/D
Square	ISPEA2	—	0.97	1.00	1.00	1.00	0.89
	SPEA2	0.04	—	0.31	0.57	0.00	0.26
	NSGA II	0.00	0.37	—	0.00	1.00	0.26
	NSGA III	0.00	0.37	1.00	—	1.00	0.74
	ICA	0.00	0.17	0.00	0.00	—	0.00
	MOEA/D	0.12	0.37	0.62	0.00	1.00	—
Convex	ISPEA2	—	0.90	0.86	0.78	1.00	1.00
	SPEA2	0.00	—	0.81	0.13	1.00	0.55
	NSGA II	0.17	0.33	—	0.22	1.00	0.25
	NSGA III	0.07	0.57	0.86	—	1.00	0.90
	ICA	0.00	0.00	0.00	0.00	—	0.00
	MOEA/D	0.00	0.19	0.81	0.03	1.00	—
TO	ISPEA2	—	1.00	1.00	0.00	1.00	1.00
	SPEA2	0.00	—	0.00	0.00	0.00	0.00
	NSGA II	0.00	1.00	—	0.00	1.00	1.00
	NSGA III	1.00	1.00	1.00	—	1.00	1.00
	ICA	0.00	0.00	0.00	0.00	—	0.00
	MOEA/D	0.00	0.00	0.00	0.00	1.00	—
Tunnel	ISPEA2	—	1.00	1.00	0.84	1.00	1.00
	SPEA2	0.00	—	0.00	0.00	0.17	0.26
	NSGA II	0.00	0.86	—	0.00	0.83	0.47
	NSGA III	0.17	0.79	0.67	—	1.00	1.00
	ICA	0.00	0.21	0.00	0.00	—	0.00
	MOEA/D	0.00	0.64	0.27	0.00	1.00	—
MS	ISPEA2	—	1.00	1.00	1.00	1.00	1.00
	SPEA2	0.00	—	0.56	0.13	0.33	0.60
	NSGA II	0.00	0.06	—	0.00	1.00	0.67
	NSGA III	0.00	0.82	1.00	—	1.00	1.00
	ICA	0.00	0.00	0.00	0.00	—	0.00
	MOEA/D	0.00	0.00	0.56	0.00	1.00	—
SC	ISPEA2	—	0.47	0.81	0.68	1.00	1.00
	SPEA2	0.07	—	0.75	0.32	0.86	0.86
	NSGA II	0.00	0.00	—	0.00	0.57	0.29
	NSGA III	0.00	0.58	0.88	—	0.86	1.00
	ICA	0.00	0.00	0.06	0.09	—	0.29
	MOEA/D	0.00	0.00	0.25	0.00	0.29	—

solutions found by ISPEA2. Another thing that should be noticed is that in square and convex scenario, only 47% solutions found by SPEA2, and 68% solutions found by NSGA III are dominated by solutions found by ISPEA2. This means that the convergence performance of ISPEA2 can be further improved in complex scenarios. From all these tests above, we can tell that the ISPEA2 algorithm has the best performance in all tested algorithms.

4.3 Controller performance analysis

In this section, MO O-Flocking with tuned parameters is compared with the original O-Flocking and three other controllers (the classic Reynolds model[36] and two APF based model (Kala[37] and Yang et al.[38])). To choose a certain control parameter set from the ND solutions found by ISPEA2, we use a technique called linear weighted aggregation, which means that we evaluate all solutions in non-dominated solution sets by using Eq. (22):

$$F(s) = w_1 D\left(r_s^{\text{death}}\right) + w_2 D\left(d_s^{\text{agg}}\right) + w_3 D\left(\overline{\text{TC}}_s\right) \tag{22}$$

where $D()$ is the normalization function. w_1, w_2, and w_3 are factors that indicate the importance of each objective. w_1, w_2, and w_3 are all equal to 1/3, which means that we consider three objectives as equally important. Equation (23) is the objective normalization

function.

$$D(o)=\frac{o-o_{\min}}{o_{\max}-o_{\min}} \tag{23}$$

Table 5 shows the simulation results.

Controller performance analysis shows that in all six tested scenarios, the tuned MO O-Flocking control model has better performance in both swarm aggregation and travelling time cost. Compared with the basic Reynolds model and O-Flocking control model, the tuned MO O-flocking model has improvements in all three objectives. Compared with two APF-based control models, the tuned MO O-Flocking control model has better performance on travelling time cost and SR aggregation. The two APF-based control models have lower death rate compared with Reynolds model[36], but the travelling time cost is higher. One thing must be noticed is that the APF by Yang et al.[38] seems to have good performance on SR aggregation in 4 scenarios (square, target orientation, tunnel, and multi square), but it has poor performance in scenarios with convex obstacles. Figures 10−15 show the simulation results.

5 Conclusion

In this article, a virtual-physical-law based swarm robotic control model with 4 control rules and tunable parameters namely MO O-Flocking is proposed. And a multi-objective optimization algorithm namely ISPEA2 for parameter tuning of MO O-Flocking control model is proposed. Two recombination heuristics are also designed to improve the algorithm efficiency. These two heuristics can also be implemented as local search operators by other parameter tuning methods.

Table 5 Controller performance analysis results.

Scenario	Tuned MO O-Flocking			Tuned O-Flocking[29]			Reynolds model[36]			APF by Kala[37]			APF by Yang et al.[38]		
	DR	AI	TC	DR	AI	TC	DR	AI	TC	DR	AI	TC	DR	AI	TC
Square	**0.000**	**0.251**	**0.345**	**0.000**	0.394	0.347	0.200	0.790	0.479	0.100	4.748	0.415	**0.000**	0.550	0.727
Convex	**0.000**	**0.294**	**0.392**	**0.000**	0.329	0.394	0.050	0.969	0.425	**0.000**	3.187	1.343	**0.000**	5.709	0.752
TO	**0.000**	**0.332**	**0.314**	**0.000**	0.396	0.315	**0.000**	0.733	**0.314**	**0.000**	1.637	0.337	**0.000**	0.444	0.546
Tunnel	**0.000**	**0.270**	**0.345**	**0.000**	0.357	0.355	0.100	0.765	0.411	**0.000**	4.077	1.359	**0.000**	0.294	0.933
MS	**0.000**	**0.252**	**0.334**	**0.000**	0.361	0.337	0.200	0.824	0.476	0.200	0.888	0.523	**0.000**	0.325	0.761
SC	**0.000**	**0.290**	**0.391**	**0.000**	0.407	0.399	0.150	1.041	0.496	**0.000**	3.355	1.397	**0.000**	6.431	0.781

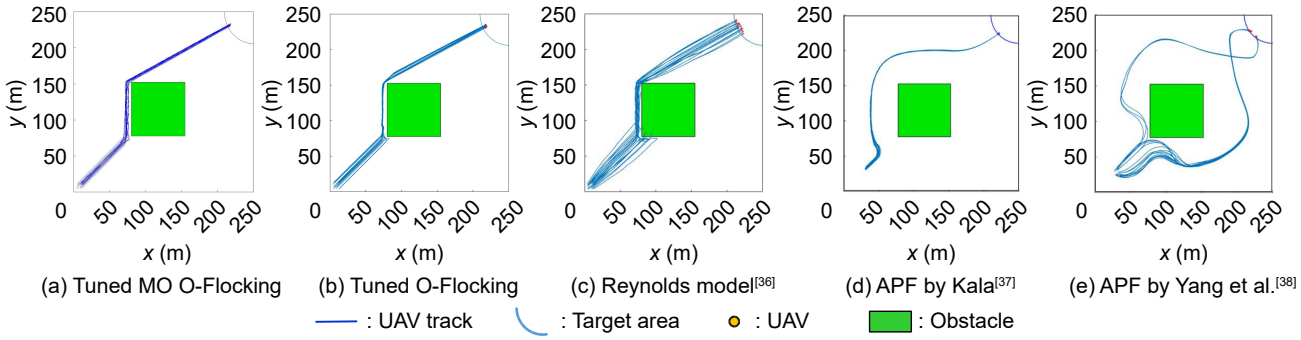

Fig. 10 Results of square scenario.

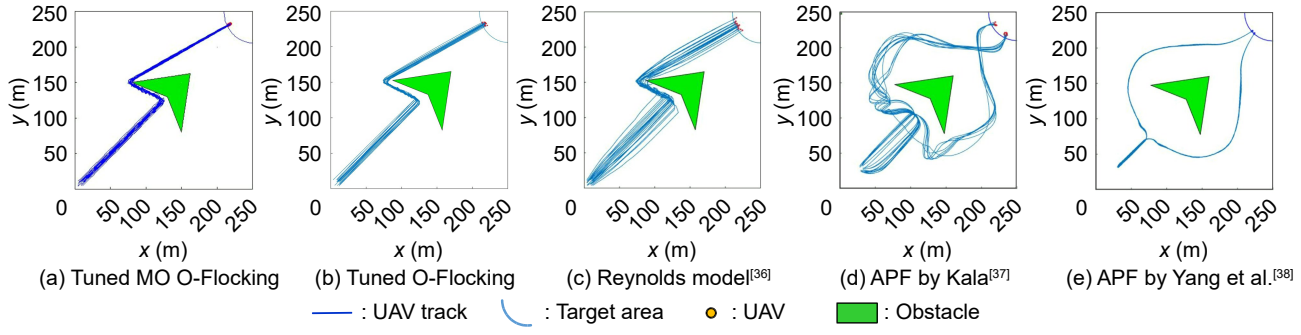

Fig. 11 Results of convex scenario.

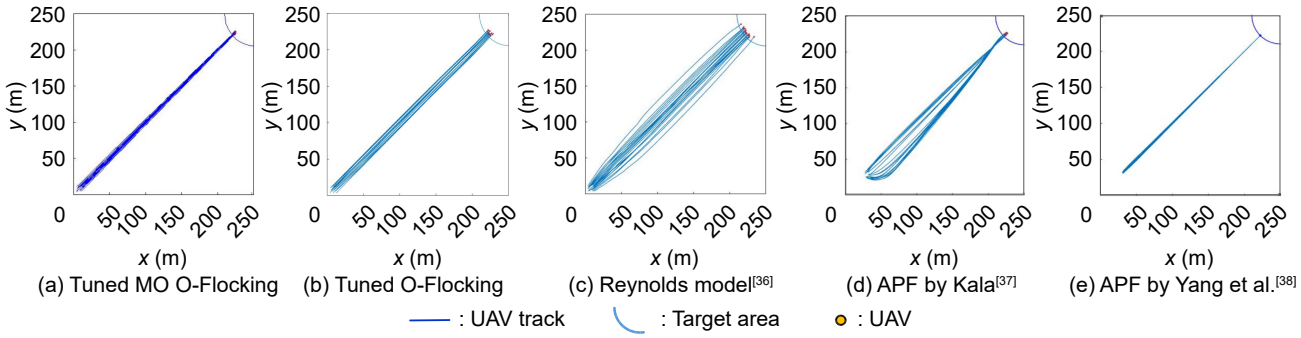

Fig. 12 Results of target orientation scenario.

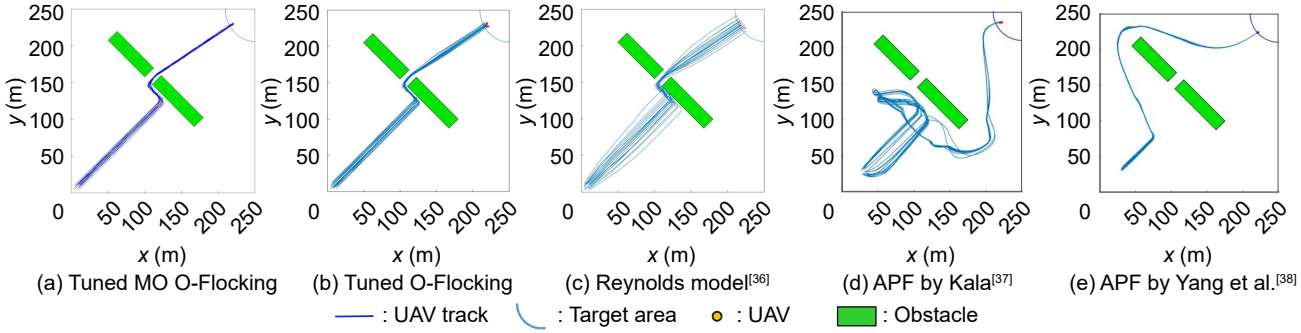

Fig. 13 Results of tunnel scenario.

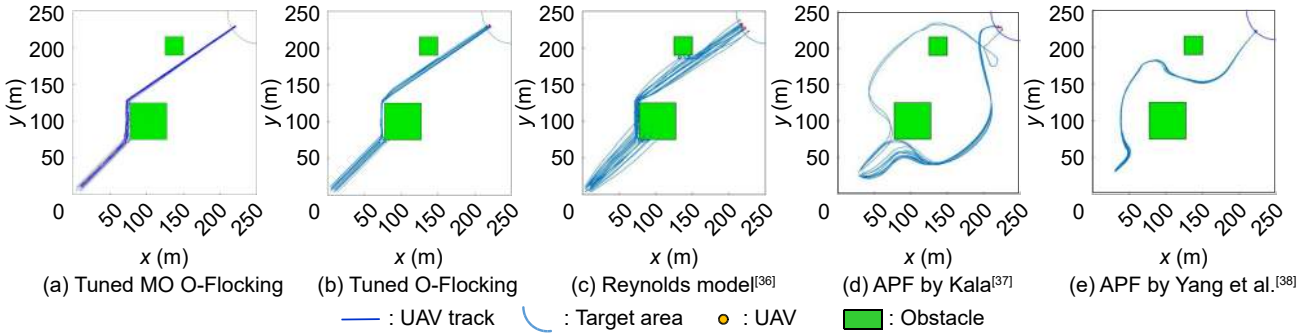

Fig. 14 Results of multi square scenario.

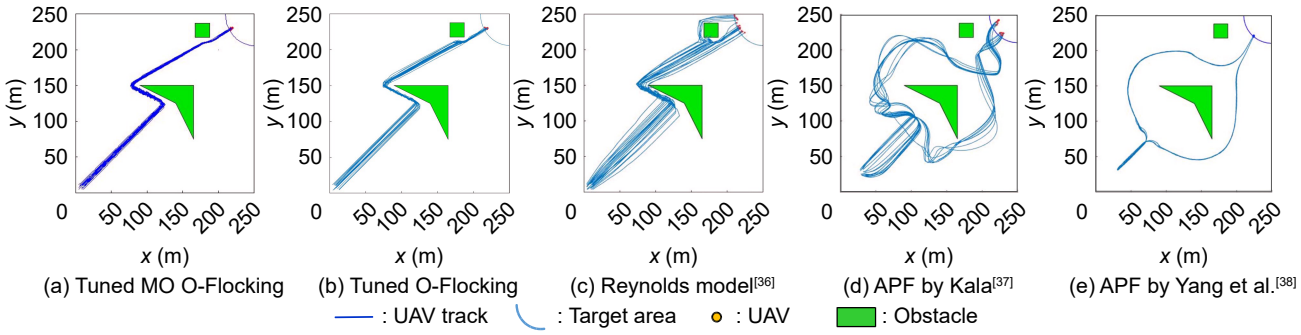

Fig. 15 Results of convex and square scenario.

Experiments are designed to test the performance of both the ISPEA2 algorithm and MO O-Flocking control model. Experiment results show that the ISPEA2 algorithm has better performance on both accuracy and diversity respects, and MO O-Flocking with tuned parameters performs better on target orientation and collision avoidance in all tested scenarios. From the simulation experiment results, we can tell that the MO O-Flocking control model can be applied to target orientation missions in complex

environments, such as battlefield area search and rescue of victims. The application of MO O-Flocking control model can effectively improve the collision avoidance and formation maintenance abilities of UAV swarm in complex environments. In addition, the ISPEA2 can also be applied separately to multi-objective parameter tuning of UAV swarm control models with multiple tunable parameters. Compared with current state-of-the-art algorithms, the ISPEA2 has significant advantages in terms of solution diversity performance, which means that ISPEA2 can simultaneously provide a feasible solution set that satisfies multiple decision preferences.

Further research can be carried out on following directions: (1) More real-world experiments can be included; (2) Since ISPEA2 has been proved to be efficient on MO O-Flocking, it can be applied on other control models; (3) More complex missions (e.g., target orientation with moving obstacles and target orientation with multi obstacles of complicated shapes) can be tested.

Acknowledgment

This work was supported by the Hunan Provincial Natural Science Foundation of China (No. 2023JJ40686).

References

[1] M. Brambilla, E. Ferrante, M. Birattari, and M. Dorigo, Swarm robotics: A review from the swarm engineering perspective, *Swarm Intell.*, vol. 7, no. 1, pp. 1–41, 2013.

[2] E. Şahin, Swarm robotics: From sources of inspiration to domains of application, in *Proc. International Workshop on Swarm Robotics*, Santa Monica, CA, USA, 2005, pp. 10–20.

[3] B. Xin, J. Zhang, J. Chen, Q. Wang, and Y. Qu, Overview of research on transformation of multi-AUV formations, *Complex System Modeling and Simulation*, vol. 1, no. 1, pp. 1–14, 2021.

[4] O. Khatib, Real-time obstacle avoidance for manipulators and mobile robots, in *Autonomous Robot Vehicles*, I. J. Cox and G. T. Wilfong, eds. New York, NY, USA: Springer, 1986, pp. 396–404.

[5] V. Gazi and K. M. Passino, Stability analysis of social foraging swarms: Combined effects of attractant/repellent profiles, in *Proc. 41st IEEE Conf. Decision and Control*, Las Vegas, NV, USA, 2003, pp. 2848–2853.

[6] W. M. Spears, D. F. Spears, J. C. Hamann, and R. Heil, Distributed, physics-based control of swarms of vehicles, *Auton. Rob.*, vol. 17, nos. 2&3, pp. 137–162, 2004.

[7] P. Fiorini and Z. Shiller, Motion planning in dynamic environments using velocity obstacles, *Int. J. Robot. Res.*, vol. 17, no. 7, pp. 760–772, 1998.

[8] J. van den Berg, M. Lin, and D. Manocha, Reciprocal velocity obstacles for real-time multi-agent navigation, in *Proc. 2008 IEEE Int. Conf. Robotics and Automation*, Pasadena, CA, USA, 2008, pp. 1928–1935.

[9] V. G. Santos, M. F. M. Campos, and L. Chaimowicz, On segregative behaviors using flocking and velocity obstacles, in *Distributed Autonomous Robotic Systems*, M. A. Hsieh and G. Chirikjian, eds. Berlin, Germany: Springer, 2014, pp. 121–133.

[10] H. M. La, R. S. Lim, W. Sheng, and J. Chen, Cooperative flocking and learning in multi-robot systems for predator avoidance, in *Proc. 2013 IEEE Int. Conf. Cyber Technology in Automation, Control and Intelligent Systems*, Nanjing, China, 2013, pp. 337–342.

[11] A. C. Woods and H. M. La, Dynamic target tracking and obstacle avoidance using a drone, in *Proc. 2015 Int. Symposium on Visual Computing: Advances in Visual Computing*, Las Vegas, NV, USA, 2015, pp. 857–866.

[12] S. Y. Ha, T. Ha, and J. H. Kim, Emergent behavior of a cucker-smale type particle model with nonlinear velocity couplings, *IEEE Trans. Autom. Contr.*, vol. 55, no. 7, pp. 1679–1683, 2010.

[13] L. Meier, P. Tanskanen, F. Fraundorfer, and M. Pollefeys, PIXHAWK: A system for autonomous flight using onboard computer vision, in *Proc. 2011 IEEE Int. Conf. Robotics and Automation*, Shanghai, China, 2011, pp. 2992–2997.

[14] G. Xu, D. Zhao, J. Wang, and H. Zhang, Distributed control of UAV formation reconfiguration in terms of dynamic reference point, *Int. J. Contr. Autom.*, vol. 10, no. 1, pp. 155–166, 2017.

[15] S. A. Harder and L. K. Lauderbaugh, Formation specification for control of active agents using artificial potential fields, *J. Intell. Rob. Syst.*, vol. 95, no. 2, pp. 279–290, 2019.

[16] A. L. Alfeo, M. G. C. A. Cimino, and G. Vaglini, Enhancing biologically inspired swarm behavior: Metaheuristics to foster the optimization of UAVs coordination in target search, *Comput. Oper. Res.*, vol. 110, pp. 34–47, 2019.

[17] S. Li and X. Fang, A modified adaptive formation of UAV swarm by pigeon flock behavior within local visual field, *Aerosp. Sci. Technol.*, vol. 114, p. 106736, 2021.

[18] J. Zhang, P. Zhang, and J. Yan, Distributed adaptive finite-time compensation control for UAV swarm with uncertain disturbances, *IEEE Trans. Circuits Syst. I Regul. Pap.*, vol. 68, no. 2, pp. 829–841, 2021.

[19] L. M. Pyke and C. R. Stark, Dynamic pathfinding for a swarm intelligence based UAV control model using particle swarm optimisation, *Front. Appl. Math. Stat.*, vol. 7, p. 744955, 2021.

[20] J. Li, Y. Fang, H. Cheng, Z. Wang, Z. Wu, and M. Zeng, Large-scale fixed-wing UAV swarm system control with collision avoidance and formation maneuver, *IEEE Syst. J.*, vol. 17, no. 1, pp. 744–755, 2023.

[21] Q. Peng, H. Wu, and R. Xue, Review of dynamic task allocation methods for UAV swarms oriented to ground targets, *Complex System Modeling and Simulation*, vol. 1, no. 3, pp. 163–175, 2021.

[22] S. D. Hettiarachchi, Distributed evolution for swarm robotics, PhD dissertation, Computer Science Department, University of Wyoming, Laramie, WY, USA, 2007.
[23] J. Pugh and A. Martinoli, Parallel learning in heterogeneous multi-robot swarms, in *Proc. 2007 IEEE Congress on Evolutionary Computation*, Singapore, 2007, pp. 3839–3846.
[24] G. Folino, A. Forestiero, and G. Spezzano, An adaptive flocking algorithm for performing approximate clustering, *Inf. Sci.*, vol. 179, no. 18, pp. 3059–3078, 2009.
[25] O. Cetin, S. Kurnaz, O. Kaynak, and H. Temeltas, Potential field-based navigation task for autonomous flight control of unmanned aerial vehicles, *Int. J. Autom. Contr.*, vol. 5, no. 1, pp. 1–21, 2011.
[26] H. X. Yang, T. Zhou, and L. Huang, Promoting collective motion of self-propelled agents by distance-based influence, *Phys. Rev. E Stat. Nonlin. Soft Matter Phys.*, vol. 89, no. 3, p. 032813, 2014.
[27] M. Zhao, H. Su, M. Wang, L. Wang, and M. Z. Q. Chen, A weighted adaptive-velocity self-organizing model and its high-speed performance, *Neurocomputing*, vol. 216, pp. 402–408, 2016.
[28] G. Vásárhelyi, C. Virágh, G. Somorjai, T. Nepusz, A. E. Eiben, and T. Vicsek, Optimized flocking of autonomous drones in confined environments, *Sci. Robot.*, vol. 3, no. 20, p. eaat3536, 2018.
[29] L. Ma, W. Bao, X. Zhu, M. Wu, Y. Wang, Y. Ling, and W. Zhou, O-flocking: Optimized flocking model on autonomous navigation for robotic swarm, in *Proc. Advances in Swarm Intelligence: 11th Int. Conference, ICSI 2020*, Belgrade, Serbia, 2020, pp. 628–639.
[30] E. Zitzler, M. Laumanns, and L. Thiele, SPEA2: Improving the strength pareto evolutionary algorithm for multiobjective optimization, in *Proc. Evol. Meth. Design, Optim. Control Appl. Industr. Probl.*, Athens, Greece, 2001, pp. 95–100.
[31] P. C. Chang, W. H. Huang, and C. J. Ting, Dynamic diversity control in genetic algorithm for mining unsearched solution space in TSP problems, *Expert Syst. Appl.*, vol. 37, no. 3, pp. 1863–1878, 2010.
[32] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, 2002.
[33] K. Deb and H. Jain, An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: Solving problems with box constraints, *IEEE Trans. Evol. Comput.*, vol. 18, no. 4, pp. 577–601, 2014.
[34] Q. Zhang and H. Li, MOEA/D: A multiobjective evolutionary algorithm based on decomposition, *IEEE Trans. Evol. Comput.*, vol. 11, no. 6, pp. 712–731, 2007.
[35] E. Atashpaz-Gargari and C. Lucas, Imperialist competitive algorithm: An algorithm for optimization inspired by imperialistic competition, in *Proc. 2007 IEEE Congress on Evolutionary Computation*, Singapore, 2007, pp. 4661–4667.
[36] C. W. Reynolds, Flocks, herds and schools: A distributed behavioral model, *SIGGRAPH Comput. Graph.*, vol. 21, no. 4, pp. 25–34, 1987.
[37] R. Kala, Code for robot path planning using artificial potential fields, Indian Institute of Information Technology Allahabad, http://rkala.in/codes.html, 2014.
[38] Y. Yang, M. Liu, H. Cui, J. Xiang, J. Guo, and J. Ding, Research on formation behavior of flock with visual guidance algorithm, in *Proc. 2018 IEEE Int. Conf. Robotics and Biomimetics* (*ROBIO*), Kuala Lumpur, Malaysia, 2018, pp. 2360–2365.

Yuan Wang received the BS and PhD degrees from National University of Defense Technology, Changsha, China in 2016 and 2021, respectively. He is a lecturer at the College of Advanced Interdisciplinary Studies, National University of Defense Technology, China. His research interests include swarm intelligence, intelligent multi-objective optimization, and deep learning.

Lining Xing received the BS degree in economics and science from Xi'an Jiaotong University, Xi'an, China in 2002, and the PhD degree in management science from National University of Defense Technology, Changsha, China, in 2009. He visited the School of Computer, University of Birmingham, UK, from November 2007 to November 2008. He is a professor at Xidian University, China. His main research directions are intelligent optimization methods and resource scheduling.

Junde Wang received the BS and PhD degrees from National University of Defense Technology, Changsha, China in 2014 and 2018, respectively. He is an assistant researcher at the College of Advanced Interdisciplinary Studies, National University of Defense Technology, China. His research interests include big data mining and analysis, and military optimization.

Tao Xie received the BS and PhD degrees from National University of Defense Technology, Changsha, China in 2015 and 2018, respectively. He is an assistant researcher at the College of Advanced Interdisciplinary Studies, National University of Defense Technology, China. His research interests include big data mining and analytics, and modern communication technology.

Lidong Chen received the BS and PhD degrees from National University of Defense Technology, Changsha, China in 2005 and 2012, respectively. He is an associate professor at the College of Advanced Interdisciplinary Studies, National University of Defense Technology, China. His research interests include big data mining and analytics, and intelligent optimization.