TORCH: Strategy Evolution in Swarm Robots Using Heterogeneous-homogeneous Coevolution Method*

Meng Wu[a], Xiaomin Zhu[a,*], Li Ma[a], Ji Wang[a], Weidong Bao[a], Wenji Li[b] and Zhun Fan[b]

[a]*College of Systems Engineer, National University of Defense Technology, Changsha 410073, Hunan, P. R. China*
[b]*Guangdong Provincial Key Laboratory of Digital Signal and Image Processing, College of Engineering, Shantou University, Shantou 515063, China*

ARTICLE INFO

Keywords:
Swarm robots
strategy evolution
heterogeneous-homogeneous
coevolution
flocking

ABSTRACT

Because swarm robots have been applied widely in various fields, the evolution capability and search space of their strategy have become of primary interest; therefore, the evolution method of swarm robots' strategy has attracted attention in both industry and academia, especially for complex applications owing to their varied task scenes. Large amounts of researches have been conducted to realize strategy evolution in swarm robotics systems. However, there are few studies on the strategy evolution sufficiently examining the simultaneous improvement of evolutionary and search performance, which are two key demands of swarm robots. Besides, the strategy that evolved under the global information is difficult to fully adapt to the distributed task scenarios. To address these issues, this study presents a he<u>t</u>erogeneous-h<u>o</u>mogeneous swa<u>r</u>m <u>c</u>oevolution met<u>h</u>od known as TORCH to improve the evolution capability of swarm robots. The method uses a swarm coevolution mechanism to accelerate the evolution. For the first time, we employ a behaviour expression tree in TORCH which expands the strategy search space of the evolved strategies. TORCH makes the swarm robots' strategies evolve under local information conditions; hence, the evolutionary strategies are more adaptable to the distributed task scenarios. Extensive experiments have been conducted to verify the proposed TORCH, including a comparison with three methods based on the homogeneous swarm evolution method and parameter expression. The results demonstrate the superiority of the TORCH in terms of evolutionary efficiency improvement and strategy performance enhancement.

1. Introduction

Recently, the development of robot systems has brought significant changes to human society [53]. It is predicted that a number of robot systems will be applied in the entertainment industry [24], sports [6], and work [29] in the near future. In addition, many characteristics of robot systems inspire studies in the field of information and communication technologies [4], thereby promoting the development of industrial information integration engineering. Considering the development of science and technology, robot systems are expected to accomplish more complex tasks [5]. Whereas robot systems enjoy numerous advantages, a single robot incurs a high failure probability because of its internal complexity; it is difficult to repair after damage, expensive, etc. [21]. Consequently, swarm robotics systems (SRSs) have gradually become the research focus [19][46]. The SRSs have strong robustness [41] and can emerge intelligent behaviours [35] that a single robot system does not have because of the individuals perform tasks autonomously. Currently, SRSs have been applied to applications, including object transportation [3], region coverage [39], and fire detection [40].

A swarm robot system consists of several individual robots with their strategies. A strategy is the corresponding relationship between the state and action and determines the robot's ability to complete tasks. According to the different types of member robots, a swarm can be divided into homogeneous and heterogeneous swarms. A homogeneous swarm is defined as a robot swarm with the same hardware structure, control module, and behaviour strategy [8]. This indicates that the individual robots will make consistent decisions and behaviours in the same environment. A heterogeneous swarm is a robot swarm in which each individual has different strategies; therefore the individuals choose different behaviours in the same environment. In this study, we focus on the homogeneous swarm because it comes from the behavioural models of natural systems [7], and it is robust, invulnerable, and scalable compared to the heterogeneous swarm [1][15]. It can also promote the study on swarm systems as part of the heterogeneous swarm [9].

Considering the development of robot technology and increase in people's needs, the task scenarios of swarm robots are expected to be diversified and dynamic such as various task scenario maps [51] and dynamic task targets [32]. The robot systems for a single task will be difficult to be widely used because of their limited application scenarios. With the in-depth study, the single robot with the ability of online rule generation is designed in a mobile robot control application [20]. However, most of the current SRSs are

*This work was supported in part by the National Natural Science Foundation of China under Grants 61872378, by the Science and Technology Planning Project of Guangdong Province of China under grant (180917144960530, 2019A050519008, 2019A050520001), by the Project of Educational Commission of Guangdong Province of China under grant 2017KZDXM032, by the State Key Lab of Digital Manufacturing Equipment & Technology under grant DMETKF2019020, and by Hunan Provincial Innovation Foundation For Postgraduate under grant CX20200081.

*Corresponding author

wumeng15@nudt.edu.cn (M. Wu); xmzhu@nudt.edu.cn (X. Zhu); mali10@nudt.edu.cn (L. Ma); wangji@nudt.edu.cn (J. Wang); wdbao@nudt.edu.cn (W. Bao); liwj@stu.edu.cn (W. Li); zfan@stu.edu.cn (Z. Fan)

ORCID(s):

designed for specific scenes, whereas making the swarm robots complete tasks accurately after switching scenes is impractical in most cases. Therefore, the SRSs are expected to evolve strategies autonomously to adapt to the various task scenarios. Being a key technology, swarm robots' strategy evolution ability directly affects the function and universal applicability of an SRS in reality. In swarm robot applications, the problem of swarm autonomous strategy evolution has been widely studied nonetheless, there are still many challenges [30][55].

- In general, the efficiency of strategy evolution in an SRS is low. This is because the evaluation of the behaviour strategy of an SRS is a very time-consuming process, and it usually needs to complete a task on the simulator to obtain the evaluation results of the behaviour strategy. In addition, to achieve a better behaviour strategy, it usually needs thousands of times of behaviour strategy evaluation to obtain a satisfactory result. This leads to repeated task execution and consumes lots of computing resources.
- Many strategy evolution methods depend on the control centre and require an overall understanding of the scenarios in the evolution process [28]. However, it is often difficult for an SRS to obtain the global environment's information in advance in the actual task process. Therefore, the evolved behaviour strategies of swarm robots are difficult to apply to actual task scenarios.
- The existing design of swarm behaviour strategy usually adopts the strategy expression method based on the parametric equations. This indicates that the behaviour strategy of the swarm robots is weighted by the pre-set weights. The optimization of a behaviour strategy is only the weight-parameters' optimization. Owing to the limitation of search space, the strategies based on the parametric equations are often difficult to apply to complex task scenarios.

To address the aforementioned challenges, we propose TORCH, *i.e.*, he**t**erogeneous-h**o**mogeneous swa**r**m **c**oevolution met**h**od, which aims to coordinate the strategy evolution and distributional characteristic of behaviour strategies. TORCH includes a swarm coevolution mechanism to accelerate the evolution process and a novel strategy expression method (behaviour expression tree) to interpret the strategy. It is worth noting that the proposed method uses only the local information obtained by environment perception and neighbour communication to evolve the strategy, without the need of accessing the global information. To verify the effectiveness and feasibility of our method, the flocking task scene of swarm robots is constructed. The flocking problem is a classic problem in which the swarm robots are expected to complete the position transfer at a fast speed and high density. In this task, each robot in the swarm is required to adopt the same strategy. Based on the TORCH, the swarm can independently evolve a structured strategy to adapt to the task scenarios through the feedback of the environment in executing the tasks.

The main contributions of this study are as follows:

- A heterogeneous-homogeneous swarm coevolution mechanism is devised to improve the efficiency of strategy evolution. The proposed method can simultaneously evaluate n behaviour strategies (n is the scale of a swarm) in a single task execution. This greatly improves the evolutionary efficiency of swarm behaviour strategies.
- Based on the proposed TORCH, the swarm robots can optimize behaviour strategies by interacting with the environments. The strategy evolution of swarm robots using the TORCH method requires only the local information. Thus, the evolved strategies are more applicable to distributed actual task scenarios.
- A new flexible behaviour strategy search space for swarm robots is designed, improving the ability of swarm robots to perform complex tasks. Specifically, considering the behaviour strategy expression of the swarm robots, compared to the strategy expression method based on the parametric equations, the behaviour expression tree method has a more flexible behaviour strategy search space and can be applied to complex task scenarios.

The rest of this study is organized as follows: Section II reviews the related studies. The mechanism of heterogeneous-homogeneous coevolution is presented in Section III. Section IV introduces the structure and evolution method of the behaviour expression tree. Section V demonstrates the simulation results and conducts the performance analysis. Finally, the study is concluded with a discussion on future studies in Section VI.

2. Ralated work

Strategy evolution is an important research topic in the applications of swarm robots, and several studies have focused on enabling the swarm to evolve new strategies automatically, improving its intelligence. Two fundamental technologies are widely applied to achieve this goal: heuristic algorithm (HA) and reinforcement learning (RL) [27]. Heuristic algorithms for search problems can be considered as classical methods to solve this problem. Yu *et al.* [56] used a super hyper-heuristic algorithm to evolve the build-cleaning strategies of swarm robots. The swarm scores the behaviours in the heuristic repository through interaction and selects the most suitable behaviour for the current environmental state that has the highest score. Heuristic algorithms are also used to sort behaviours to find the best sequence of behaviours for a specific task [28]. To find the appropriate intelligent algorithms to solve specific complex problems, Tao *et al.* [48] proposed a new dynamic configuration method of intelligent algorithm. However, these methods only aim at the specific and known task scenarios and have to start learning again when the scenarios change. Zou *et al.* [58] proposed a dynamic multi-objective evolutionary algorithm to help the population adapt to the new environment by building a dynamic evolutionary environmental model. However, the algorithm relies on centralized control and is not suitable for distributed tasks. In addition, the mechanism of cell growth inspired studies in robotics [14][33]. By coding behavioural strategies into artificial chromosomes, the evolution of strategies is realized by simulating Darwinian evolution in biology. The method of generating behavioural strategies in robotics is defined as evolutionary robotics (ER) [31]. In addition to solving single-agent problems [23], evolutionary computation is also beneficial to the generation of swarm behaviour[52] such as object transport [13], aggregation [11], and swarm foraging [2]. Pugh *et al.* [36] used genetic algorithm (GA) and particle swarm optimization (PSO) to make robots learn to avoid obstacles. Takadama *et al.* [47] applied evolutionary robotics to a task scheduling problem, making the robots learn the correct behavioural order in a space truss construction task. However, these methods often require a long time for learning, and it is not easy to evolve an effective strategy within a short period of time. Therefore, this study investigates the strategy evolution method in a swarm based on the coevolution mechanism, aiming to enhance the speed and improving the efficiency of the evolution.

Moreover, reinforcement learning (RL) has been applied to swarm robots [18][54]. Multi-agent reinforcement learning (MARL) is based on multi-agent systems and aims to investigate the algorithm design for generating adaptive agents. Reinforcement learning allows individuals to learn behaviour through repeated experiments with the environment and other factors. Many problems such as odour localization robots [16] and RoboCup Soccer [38] are solved by MARL. Iima *et al.* [17] used MARL to solve the formation problem, where agents update the Q-table value through information exchange so that all robots in a swarm can reach the target positions in the shortest time. Zhang *et al.* [57] also applied MARL to swarm confrontation environments to make agents learn to cooperate and compete with one another. The scenario-transfer training method and the self-play training method were proposed to improve the model's convergence speed and the combat capability of agents. Gebhardt *et al.* [12] used the reinforcement learning method to make the swarm complete assembly tasks automatically and applied the evolved strategy to a kilobot robot swarm, practically verifying the effectiveness of their evolved strategy. However, reinforcement learning often needs a long time of training, which usually requires offline learning and a substantial computing power. Therefore, heuristic methods were used to speed up the convergence of reinforcement learning. For instance, Shi *et al.* [44] used the pheromone mechanism of ant colony algorithm to accelerate the path selection of swarm robots. Seo *et al.* [43] used the reinforcement learning method with the support vector machine (SVM) based on structural risk minimization and distributed genetic algorithms for behaviour learning and evolution of collective autonomous mobile robots. Most of the strategies evolving from MARL are represented by neural networks, making it difficult to be interpreted and adjusted artificially. In this study, we hope that the proposed evolutionary method can effectively coordinate the evolutionary performance and strategy space for the swarm robots to evolve effective and well-structured strategies within a short period of time.

Flocking is a primary technology for many applications of SRSs. A troop of robots may be ordered to move to a specific place to perform combat operations and tasks in an obstacle task environment. During this process, the speed of the troop transfer is considered to make a surprise attack and gain an advantage in a confrontation [42]. Kumar *et al.* [22] designed a Lyapunov-based path planner that organizes vehicles by formations that ensure a safe collision-free path for the swarm to its target. However, the rules designed artificially can only adapt to one environment. Therefore, flocking was chosen to verify the feasibility and effectiveness of the proposed strategy evolution method. In swarm robots, flocking is a fundamental operation in which a troop of robots moves from a source to a destination in an aggregation. The flocking model was first studied in the pioneering work of Reynolds [37] who demonstrated the flocking behaviour from a set of simple rules, namely collision avoidance, velocity matching, and cluster centring. Based on these general rules, several models have been designed to describe the biological phenomena and cell migration [26][34]. Considering the practicability, several studies are devoted to enhancing the performance of flocking. Ma *et al.* [25] designed a control model and optimized it to make the swarm robots move smoothly. V*ásá*rhelyi *et al.* [50] established a real UAV swarm model and designed an evolutionary optimization framework to optimize the parameters. Nevertheless, the expressions of flocking behavioural strategies are primarily based on parameter expressions, and the corresponding optimization process is the adjustment process of the parameters. The parameter expressions lead to the lack of clear structures and flexible behavior control of the evolved strategies. The proposed method is expected to express the behaviour strategies of swarm robots more clearly, expand the strategy search space and is beneficial to further studies.

3. Heterogeneous-homogeneous swarm coevolution method

The swarm for strategy evolution is composed of several robots, and each robot chooses a behaviour strategy (that is, the robot's gene). Because the standard robot swarm is supposed to be homogeneous [15], the goal of the evolution is to find the best strategy for a homogeneous swarm. When all the robots in a swarm execute the strategy, the swarm will obtain the highest evaluation and complete the task as expected. Many studies have focused on strategy evolution

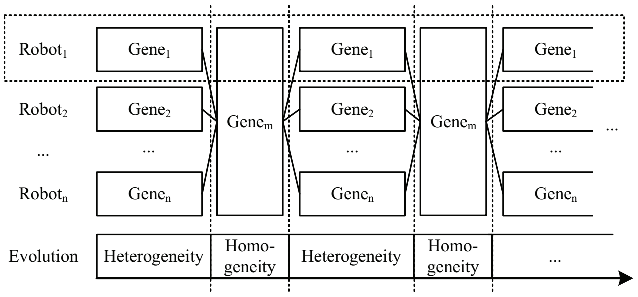


Figure 1: Gene selections of swarm robots in the heterogeneous-homogeneous swarm coevolution method.

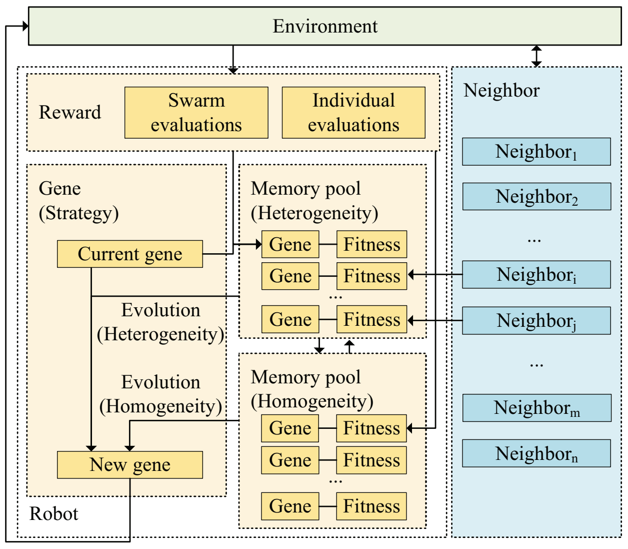


Figure 2: Strategy evolution model of an individual robot in a swarm.

while requiring multiple iterations, and usually one strategy will be evaluated in each iteration [25]. Compared to serial evolutionary algorithms, parallel methods are proven to have better performances [49]. Therefore, we designed TORCH in which the heterogeneous-homogeneous swarm coevolution mechanism could evaluate multiple strategies in parallel to improve the evolution speed and provide the possibility for online evolution.

Fig. 1 shows the gene selections of the proposed heterogeneous-homogeneous swarm coevolution mechanism. In the evolution process of the TORCH, the individuals in the swarm choose different strategies and the same strategies alternatively. Thus, this mechanism is named heterogeneous-homogeneous coevolution. At the heterogeneous stage, each individual in the swarm carries different gene sequences. This indicates that the genome carried by the swarm robots are $G = \{Gene_1, Gene_2, \dots, Gene_n\}$. Through the interaction between the robots and environment, the estimations of genes carried by the robots are given separately. At the homogeneous stage, the swarm will accurately evaluate the genes with high evaluation selected at the heterogeneous stage. Each robot in the swarm carries the same gene, indicating that the genome carried by the swarm robots are $G = \{Gene_m, Gene_m, \dots, Gene_m\}$. In each iteration, m changes from 1 to m_{max}, where m_{max} is the number of the genes with high evaluation selected at the heterogeneous stage, and it is also the number of iterations at the homogeneous stage. We evaluate the genes as swarm strategies to obtain accurate gene evaluations through the interaction between the swarm and environment. The accurately evaluated gene sequence will be used as the next heterogeneous stage's input in the robots' local memory pool to guide a new cycle of the heterogeneous-homogeneous coevolution process. Using this mechanism, swarm robots can obtain the estimated evaluations of multiple strategies by one task execution, but not only one strategy, so as to accelerate the evolution process.

This paper focuses on the strategy evolution of swarm robots. After gene selections, swarm robots need to interact with the task environment to evaluate the genes, so as to evolve the genes. Thus, it is important to study the strategy evolution process of individual robots in the swarm. Therefore, we model the swarm as $\mathbf{R} = \{R_1, R_2, ..., R_n\}$, where R_i ($i \in [1, n]$) represents different individual robots. It is assumed that the strategy of the individual robot R_i in the swarm is represented by G_i. In the process of swarm task execution, a single robot realizes strategy evolution through interaction with the environment, and the process is shown in Fig. 2.

All robots in the swarm are in the task scene, and they get rewards through their interaction with the environment. Each robot has a gene, and the gene is also the strategy used to decide the behaviour. To evaluate the strategy comprehensively, a reward value which includes swarm evaluations and individual evaluations is employed. The robot's current gene and reward value are linked and stored in the local memory pool. Meanwhile, the pairs of a neighbour's gene and reward value are obtained through communication interaction and stored in the robot's local memory pool. At different stages according to the TORCH, the pairs are stored in different memory pools, and new genes are evolved in different ways. At the heterogeneous stage, fitness is stored in the memory pool (heterogeneous), and the pool is used to support the generation of new genes. At the homogeneous stage, the high-fitness genes retained at the heterogeneous stage were added to the memory pool (homogeneous) and accurately evaluated in sequence. At the end of the homogeneous phase, these genes and their exact evaluations are returned to the memory pool (heterogeneity) as the basis for a new heterogeneous evolution. The new gene interacts with the environment to generate the new reward value and enters the next iteration. The gene evaluation, local memory pool, and communication between the robots are introduced in the following subsections.

3.1. Gene evaluation

The evaluation function of a gene usually includes fitness and reward functions. The fitness function, which is usually used in the evolutionary algorithm, gives a comprehensive evaluation of a gene by integrating the swarm and individual

Table 1
Metrics of the flocking task

Metrics	Type	Category of metrics	Name	Content	Function
f_1	Reward	Individual	Distance variation to target area	The distance variation to the target area before and after a period of time.	$d_i^{target}(t_2) - d_i^{target}(t_1)$
f_2	Reward	Individual	Obstacle distance [50]	The cumulative distance between an individual and the nearest obstacle in a period of time.	$d_i^{obstacle}(t_2)$
f_3	Reward	Individual	Speed direction smoothness [50]	The deviation between the previous and the current speed direction of a single individual.	$\lvert\theta_i(t_2) - \theta_i(t_1)\rvert$
f_4	Reward	Swarm	Aggregation degree [50]	The distance from an individual to the centre of the swarm.	$distance((x_i, y_i), \sum_{j=1}^{n_i}(x_j, y_j)/n_i)$
f_5	Reward	Swarm	Speed correlation [50]	The deviation between the individual's speed direction and the swarm's average direction.	$\lvert\theta_i(t_2) - \sum_{j=1}^{n_i}\theta_j(t_2)/n_i\rvert$
f_6	Fitness	Individual	Time to reach the target area	The time consumption of a single individual from the initial position to the target area.	T_i
f_7	Fitness	Swarm	Swarm average time	The average time taken by the swarm to reach the target area.	$\sum_j^n(T_j)/n$
f_8	Fitness	Individual	Gene complexity	The longer the behaviour expression the more complex the gene.	$Length(G_i)$

metrics at the end of the task execution. In contrast, the reward function is calculated in each step during the swarm tasks to evaluate the individual's single-step action. The reward is an important part of reinforcement learning, which guides the whole process of training. To achieve more accurate gene evaluations, we designed the reward value feedback obtained by interacting with the environment to assist the evaluation of the genes to accurately guide the direction of evolution. As a result, in our proposed strategy evolution method, the fitness and reward functions are both used to assist the evolution.

The swarm obtains some attribute metrics in the process of task execution, and these attributes can be used to evaluate the gene from various aspects. The attributes of metrics are divided into two categories: swarm metrics and individual metrics. Considering the flocking task, the attribute metrics we selected are shown in Table 1. Regarding the task execution, the robot will calculate the step-by-step reward to calculate the fitness function of its gene.

3.1.1. Step-by-step reward

Limited by hardware and software conditions, the interactions between individuals and environment are usually discretized. Regarding the reinforcement learning algorithm, the individuals obtain rewards from the environment at each step in the process of interacting with the environment. Similarly, the step-by-step reward is set in the process of strategy evolution. An individual calculates the reward by combining the changes of its internal state and the environment (including the states of the neighbours) each time it interacts with the environment. The rewards are recorded within the individuals as the basis for evaluating their genes.

Considering the flocking task, the attributes of metrics we selected are shown in Table 1, where the metrics f_1-f_5 are step-by-step rewards. Regarding robot R_i, the functions of the step-by-step reward from t_1 to t_2 are listed, where t_2 represents the current time and t_1 is the time of the last step. d^{target} is the distance to the target area, $d^{obstacle}$ is the distance to the nearest obstacle, and θ is the velocity direction. (x_i, y_i) is the current position of robot R_i in the coordinate system, and n_i represents the number of neighbours of robot R_i within its communication range. (x_j, y_j) and θ_j are the current position and velocity direction of the neighbour R_j, respectively.

It is worth noting that owing to the genetic inconsistency of individuals in the process of heterogeneous evolution, the distribution of rewards should reduce the interference of neighbours as much as possible. Therefore, the weight of the swarm metrics should be reduced. In addition, since the goal of the evolution is to make the robot learn to avoid obstacles, the position of the robot will be judged in each step. If the robot is inside the obstacle, a huge negative reward will be given to the robot.

3.1.2. Fitness function

After an iteration of a task, the individuals will comprehensively evaluate the task execution and calculate the fitness value of the genes. This evaluation represents the fitness of the gene to the current task scenario. Regarding the flocking task, the time consumption to complete the task is added to the fitness as shown as f_6 and f_7 in Table 1, where T_i is the time taken by robot R_i to reach the target area, and n is number of robots in the swarm.

Furthermore, the genetic complexity of the robot is set as f_8 because the evolved strategy is expected to be simple. The gene complexity is simply calculated using the length of the behaviour expression.

Individual and swarm metrics have different weights, namely α and β. In the process of heterogeneous evolution, β is a small value to reduce the influence of the swarm metrics. The fitness f is designed as Eq. (1).

$$f = \left[\alpha(f_1^* + f_2^* + f_3^*) + \beta(f_4^* + f_5^*)\right]\times(\alpha f_6^* + \beta f_7^*)\times\alpha f_8, \tag{1}$$

where * represents the normalization of the metric.

3.2. Local memory pool

Because there is no centralized control in the strategy evolution process of a swarm, an individual's local memory pool is an important part of the realization of swarm cooperation, which plays the same role as the population in the traditional genetic algorithms. An individual's local memory pool stores its gene and gene fitness, as well as the pairs of genes and gene fitness obtained from neighbours through communication. Each pair of gene and gene fitness is stored as a memory in the local memory pool. Owing to the weak storage capacity of swarm robots, the individual local memory pool is utilized. The mechanism of increasing and deleting memory is explained as follows:

3.2.1. Increase of memory

When a new pair of gene and gene fitness is obtained, it needs to be added to the local memory pool. It is first determined whether the gene has been stored in the memory pool. If it already exists, the gene fitness is replaced by the average of the old and the new gene fitness. If it does not exist, it is added to the memory pool as a new memory.

3.2.2. Deletion of memory

At the end of an iteration of the task execution, the memory in the individual's local memory pool is arranged in a descending order according to the gene fitness. The maximum storage capacity is specified, and the memory that does not exceed the maximum storage capacity is retained. The rest is deleted.

3.2.3. Transfer of memory

Robots have different memory pools at different stages in the TORCH. At the heterogeneous stage, the robot widely acquires the pairs of a neighbour's genes and fitness and stores them in the local memory pool (heterogeneous). This memory pool stores the estimated fitness of the genes, and the new genes evolve based on this memory pool. At the homogeneous stage, the high-fitness genes in the memory pool (heterogeneous) are transferred to the memory pool (homogeneous). As the evolution iterates, accurate evaluations of the genes in this pool are obtained.

In the transition of homogeneous and heterogeneous stages, the memories in the robot memory pool is transferred. When a robot enters the homogeneous stage from the heterogeneous stage, the memories in the memory pool (heterogeneous) is sorted according to the gene fitness. Thereafter, the selected m_{max} high-quality memories are copied to the memory pool (homogeneous). At the end of the homogeneous stage, the memories in the memory pool (homogeneous) with accurate fitness are returned to the memory pool (heterogeneous) to update the gene's evaluations.

3.3. Gene generation

Before entering a new iteration, the individual robots in the swarm generate new genes from their current genes. There are different ways by which the robots generate new genes at the heterogeneous and homogeneous stages.

3.3.1. Heterogeneous stage

The current gene is defined as the basic gene, the memory pool (heterogeneous) is considered as the "population", and the new genes are generated through the evolutionary operations. Different basic genes and random evolutionary operations lead to gene diversity, and similar memory pools (heterogeneity) ensure the convergence of the evolution.

3.3.2. Homogenous stage

The genes in the memory pool (homogenous) are selected in order. After obtaining the accurate evaluation value of the selected gene, the evaluation of this gene is replaced, and the next gene in the memory pool (homogenous) is selected as the current gene.

3.4. Communication

Owing to no centralized control in the process of strategy evolution, communication is a necessary method to achieve the cooperation of individuals in the swarm. The main goal of communication in this method is to reach a consensus of strategy in the swarm. This method requires the swarm to obtain the genes and gene fitness from neighbours as much as possible.

Therefore, the local communication of individuals is ensured by setting the maximum communication range C^{max} and calculating the neighbour communication proximity matrix. The content of the communication in the proposed TORCH is the pair of gene and gene fitness. It is assumed

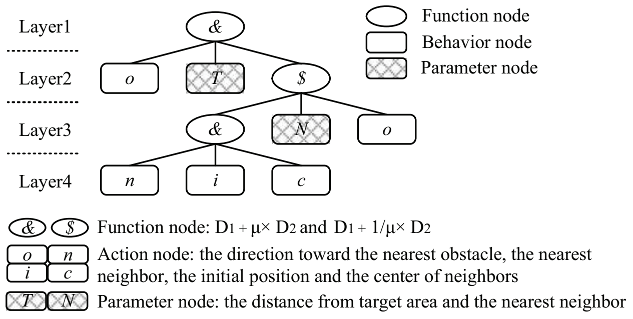

Layer1
Layer2
Layer3
Layer4
Function node
Behavior node
Parameter node
Function node: D1 + μ× D2 and D1 + 1/μ× D2
Action node: the direction toward the nearest obstacle, the nearest neighbor, the initial position and the center of neighbors
Parameter node: the distance from target area and the nearest neighbor

Figure 3: Example of the structure of a behaviour expression tree.

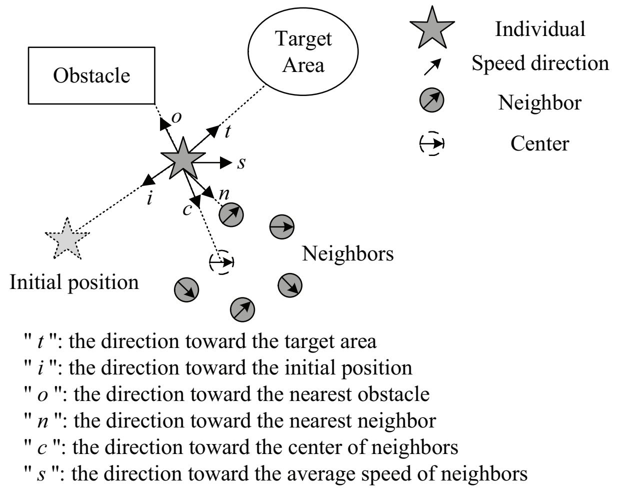

Obstacle
Target Area
Individual
Speed direction
Neighbor
Center
Initial position
Neighbors
" t ": the direction toward the target area
" i ": the direction toward the initial position
" o ": the direction toward the nearest obstacle
" n ": the direction toward the nearest neighbor
" c ": the direction toward the center of neighbors
" s ": the direction toward the average speed of neighbors

Figure 4: Specific directions of meta-actions of a robot in the flocking task.

that the communication bandwidth of the individual robot is enough for the transmission of the content.

4. Robot control structure: behaviour expression tree

In the strategy evolution of swarm robots, each robot corresponds to a gene which is evolved to adapt to the environment. This gene is also the behavioural strategy for the robot which is evolvable. Inspired by the expression tree, we proposed the behaviour expression tree as the control structure to express the strategies of the robots in a swarm in the TORCH. The behaviour expression tree is a kind of hierarchical and structured expression of robot control using a tree. The advantages of the behaviour expression tree include: 1) the behaviour expression tree has a wide range of applications which can be applied to a variety of tasks by adjusting the content of the nodes in the tree; 2) a behaviour expression tree can be encoded through a specific coding method which reduces the communication complexity in the robot evolution; 3) the behaviour expression tree has a clear structure and changes on a larger strategy space. An example of the structure of a behaviour expression tree is shown in Fig. 3.

Fig. 3 shows a complete behaviour expression tree that is composed of nodes and lines, in which the nodes include leaf and middle nodes. The leaf nodes are either the behaviours that the robot can perform or the parameters affiliated to the function nodes and are represented by squares. The middle nodes are the functions that combine these leaf nodes and are represented by circles. The number of child nodes to the function nodes is set to three. Starting from the root node, the final action of the robot can be obtained by recursively traversing all the nodes of the tree. The structure, encoding, decoding, and evolution of a behaviour expression tree are detailed in the following subsections.

4.1. Nodes

Nodes are important parts of a behaviour expression tree and are divided into function, parameter, and behaviour nodes, where the function nodes are intermediate nodes, and the others are leaf nodes. A behaviour expression tree is a structured combination of behaviour, function, and parameter nodes.

4.1.1. Behaviour nodes

The behaviour nodes are the terminal nodes of a behaviour expression tree, and the degree of all the behaviour nodes is zero. The behaviour nodes contain the meta-actions that a single robot can perform in a task scene.

Considering the flocking task of swarm robots, we set the meta actions that a single robot can perform using the actuator as directions of movement, including "*t*"- the direction toward the target area, "*i*"- the direction toward the initial position, "*o*"- the direction toward the nearest obstacle, "*n*"- the direction toward the nearest neighbour, "*c*"- the direction toward the centre of the neighbours, and "*s*"- the direction toward the average speed of the neighbours. The specific directions of the meta-actions of a robot are shown in Fig. 4.

4.1.2. Function nodes

The function nodes are the middle nodes of a behaviour expression tree. These are used to combine the actions of leaf nodes, and the degree of a function node cannot be zero. Each function node is represented by a symbol and performs different functions. Considering the flocking task, the function nodes are defined as the vector sum of the direction vector. The weights of the vectors are also set as leaf nodes. In this task, we define several functions to operate the unit direction vector. The function nodes are shown in Table 2. For a function node, the number of its child nodes is three; nonetheless, the number of inputs is two or three. This is because when there is no parameter node in the child nodes of a function node, the parameter μ is set to the default value of one. In this situation, the behaviour nodes are the first two child nodes orderly selected, and the third child node is not calculated.

Table 2
Function nodes in the flocking task

Symbol	Number of Inputs	Inputs	Function Operation
"&"	3	Direction D_1, D_2, Parameter μ	$D_1 + D_2 \times \mu$
"#"	3	Direction D_1, D_2, Parameter μ	$D_1 - D_2 \times \mu$
"$"	3	Direction D_1, D_2, Parameter μ	$D_1 + D_2/\mu$
"@"	3	Direction D_1, D_2, Parameter μ	$D_1 - D_2/\mu$

4.1.3. *Parameter nodes*

The parameter nodes are affiliated to the function, which should be able to be adjusted according to the task scenario. Thus, three distances are set as the parameter nodes in the flocking task, including "*T*"- the distance from the target area, "*O*"- the nearest distance from the obstacles, and "*N*"- the distance from the nearest neighbour.

4.2. Encoding and decoding of the behaviour expression tree

The genetic material of an organism is stored in chromosomes. A chromosome is composed of gene fragments which determine the genetic information carried by the chromosome. Considering the biological evolution theory, each individual of a population carries a chromosome, and the evolution of the population is realized through the crossover and mutation of gene fragments in the chromosome. To benefit the variation and evolution of swarm robots, the behaviour expression tree can be encoded into a chromosome to simulate the genetic evolution process of an organism. The transformation between the behaviour expression tree and chromosome is realized by encoding and decoding.

4.2.1. *Encoding*

The behaviour expression tree is encoded in a hierarchical order. There is one root node in the first layer of a behaviour expression tree. This is encoded as the first symbol of the behaviour expression tree. Thereafter, the behaviour expression tree is traversed according to the breadth-first principle, and the traversal sequence of the nodes is the symbol sequence of the chromosome.

As shown in Fig. 5, the root node of the behaviour expression tree is "&", which is the first node of behaviour expression. The first node of the second level is "*o*"; therefore, the behaviour expression tree can be coded as "*&oT$&Nonic*" chronologically.

4.2.2. *Decoding*

The purpose of coding the behaviour expression trees into chromosomes is to simplify the communication and facilitate evolution. When the robots need to choose actions based on genes, the chromosome needs to be decoded to the behaviour expression tree structure.

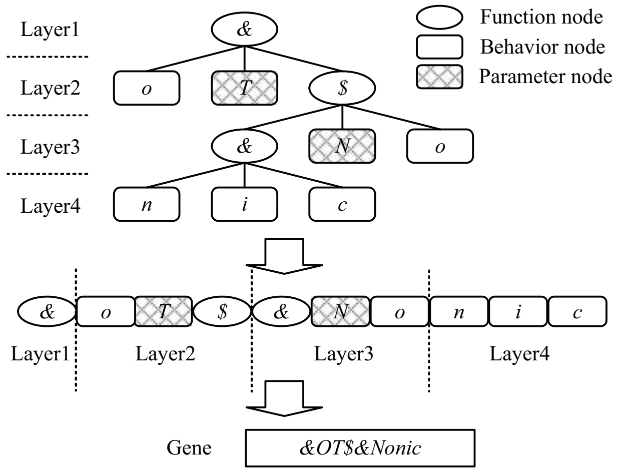


Figure 5: Process of encoding a behaviour expression tree.

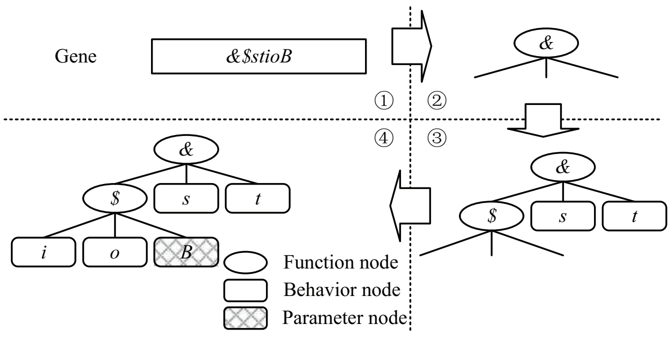


Figure 6: Process of decoding a behaviour expression tree.

In the designed behaviour expression tree, the number of input variables of function nodes is two. According to the steps in Fig. 6, the chromosome is decoded into the behaviour expression tree.

4.3. Legality inspection

Variation can occur anywhere within the genes. However, the structural organization of genes must remain legal. Since the mutation may convert a leaf node into a function node, there may be no enough leaf nodes to ensure that the behavior expression tree is complete. Therefore, after the mutation operation, gene legalization is required to ensure that the generated new gene is complete. Due to the limitation of communication bandwidth, we expect to simplify the behavior expression of a robot. Therefore, after the legalization of a gene, a gene simplification operation is applied, so that the shortest gene used to generate a complete behavior expression tree is retained. Finally, a new legal gene with shortest length is generated.

A behavioural expression is considered as a gene which consists of a head and a tail. The head consists of middle and leaf nodes, whereas the tail contains only leaf nodes. For each problem, the length *h* of the gene's head is determined by the length of the expression sequence before the last middle node. To limit the complexity of the problem, we need to limit the maximum length h_{max} of the head. After deciding on the length h of the head, the minimum length of the tail t_{min} required to build a complete tree is a function of h and k, where k is the number of inputs (also known as the maximum number of operands) of the function with the largest number of variables. The size of t_{min} is obtained using the following equation:

$$t_{min} = h * (k - 1) + 1. \tag{2}$$

Therefore, after the behaviour expression chromosome is evolved, the generated new behaviour expression chromosome should meet the following requirements:

$$h < h_{max}, \tag{3}$$

$$t > h * (k - 1) + 1. \tag{4}$$

If the length h of the gene's head does not satisfy Eq. (3), the redundant fragments of the head after h_{max} will be removed. If the tail's length t of the gene does not satisfy Eq. (4), the gene will be supplemented with random gene fragments to meet the requirement of the length.

In addition, because each function node requires a fixed number of inputs, its leaf nodes must meet this requirement. The gene whose root node is a leaf node is also illegal because it indicates that the robot only performs a meta-action, making it difficult to produce effective behaviours. When a gene is detected as illegal, it needs to re-evolve and generate a new gene.

4.4. Evolution of a behaviour expression

In processing a task, each robot has its own action strategy as the gene, which is interpreted by the behaviour expression tree. Therefore, a robot's action strategy can evolve through the evolution of the behaviour expression tree.

In the genetic evolution stage of swarm robots, individuals select the genes with the highest fitness values as the basic genes by combining the gene information obtained from the local memory pools, and make evolutionary operations to generate new genes. In this process, the gene evolutionary operations include replication, single-point mutation, two-point mutation, single-point recombination, fragment recombination, single-point insertion, and fragment insertion [10]. The probabilities of these evolutionary operations are set as P_r, P_{sm}, P_{tm}, P_{sr}, P_{fr}, P_{si}, and P_{fi}, respectively. Among them, the single-point and two-point mutations only occur in a gene, and there is no need to select another gene in the memory pool to assist the evolution process. On the contrary, insertion and recombination operations need to select another gene to assist the evolutionary operations.

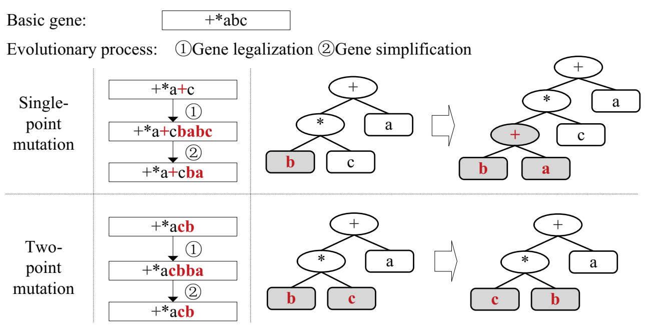


Figure 7: Processes of single-point and two-point mutations.

4.4.1. *Replication*

Individuals have a probability to directly copy the basic gene into the new one without the evolutionary operation to avoid the displacement of high-quality genes as much as possible.

4.4.2. *Mutation*

The single-point mutation is one of the most efficient operators, considering its modification ability which makes a point in a gene to be mutated into any symbol in the symbol dictionary. Similarly, a mutation operation in which two points mutate simultaneously is known as a two-point mutation.

Fig. 7 illustrates the behaviour expression tree of a basic gene before and after mutation under single-point and two-point mutations. Considering the single-point mutation, we note that there is only one-point mutation in the process of basic gene replication, that is, the "b" of Position 4 mutates to "+". Regarding the process of the two-point mutation, there are two-point mutations in the gene: the "b" of Position 4 mutates to "c", and the "c" of Position 5 mutates to "b". The effect of the mutation may be trivial in some particular cases. For instance, a two-point mutation only changes the order of two-leaf nodes as shown in Fig. 7. Considering other instances, although the mutation may only change a point, it can cause obvious changes in the structure of the behaviour expression tree. For example, the single-point mutation shown in Fig. 7 leads to the addition of one more layer to the structure of the behaviour expression tree. The mutation operation in the coding sequence mostly dramatically changes the shape of a behaviour expression tree, which is essential for the evolutionary ability.

4.4.3. *Recombination*

Recombination refers to the pairing of two randomly selected parent chromosomes and the exchange of some fragments. Here, two kinds of recombination are mainly considered: single-point and two-point recombinations. Recombination always involves two parent genes and the generation of two new genes. Therefore, in the recombination operation of a behaviour expression tree, the parent genes are designated as a basic gene and a reference gene.

Single-point recombination: the parent genes pair with each other and exchange the parts after the recombination

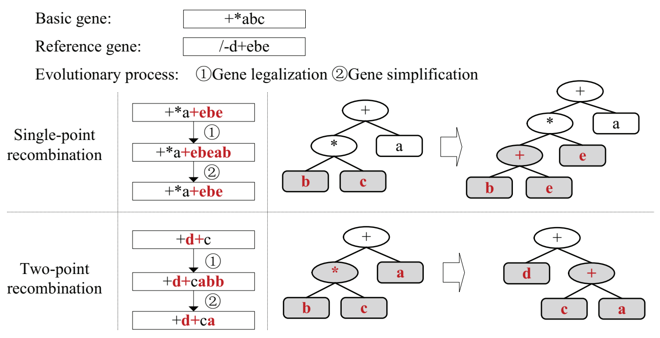


Figure 8: Processes of single-point and two-point recombinations.

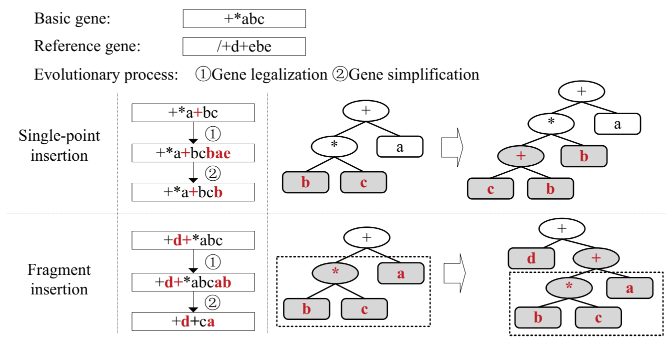


Figure 9: Process of single-point and fragment insertions.

points. To demonstrate the working mechanism of the single-point recombination, the recombination of a basic gene is shown in Fig. 8 as an illustration. In the process of single-point recombination, the basic and reference genes exchange the whole gene fragments after the recombination point. In this example, the recombination point of the basic gene is after "a", and the recombination point of the reference gene is after "d". Therefore, the gene fragment "bc" of the basic gene is replaced by fragment "$+ebe$" after the recombination point of the reference gene, resulting in a new gene.

Two-point recombination: The parent genes pair with each other, and two recombination points are randomly selected from the genes to cut off the genes. The two genes exchange the parts between the two recombination points and form two new child genes. Fig. 8 shows the working process of two-point recombination. The fragment "$* \ ab$" of the basic gene is replaced by the segment "$d+$" between the recombination points of the reference gene. The recombination results in the addition of new function nodes. In the process of gene legalization, new leaf nodes are randomly generated, and effective leaf nodes are retained in the process of gene simplification. Finally, a new gene "$+d + ca$" is generated.

4.4.4. *Insertion*

Insertion refers to randomly selecting a single gene point or fragment in one parent gene and inserting it into the selected positions of the other parent gene. Similar to recombination, insertion involves two parent genes and produces two new children. The parent genes are designated as a basic gene and a reference gene.

Single-point insertion: This involves selecting a gene point from one of the parent genes, selecting an insertion position from the other gene, and inserting a single gene unit into the position. Fragment insertion: This also includes selecting a gene fragment from one of the parent genes, selecting an insertion position from the other gene, and inserting the gene fragment into the selected position. Fig. 9 illustrates the insertion process. In the single-point insertion, the gene point "+" of the reference gene is inserted between the gene points "a" and "b" of the basic gene. In the fragment insertion, the gene point "$d+$" of the reference gene is inserted between the gene points "+" and "$*$" of the basic gene to obtain a new gene.

5. Experimental results

In this section, four experiments are designed to verify the proposed TORCH. First, to verify that the proposed method is feasible, the TORCH effectively evolves a strategy for the flocking task. Thereafter, we apply the evolved strategy in a changed task scenario to verify the method's adaptability to different task scenarios. Finally, comparative experiments are designed to verify the evolutionary efficiency improvement of the proposed method and performance enhancement of the evolved behaviour model.

Considering the flocking task, swarm robots are expected to move from the initial position to the target area in an environment with obstacles. The task area size is 50m×50m. In this scene, swarm robots are expected to reach the target position, that is, within the radius of 10m with [50,50] as the centre). Swarm robots have no centralized control, and each robot decides its own subsequent action according to its genes and the current scenario. The speed of the robots in the swarm is fixed at 1m/s. The maximum simulation step given to the task is 200, *i.e.*, even if the swarm does not reach the task area within 200 steps, the task will stop.

Other parameters are set as follows: the maximum communication range is C^{max} = 5m, the weight $\alpha = 1$, $\beta = 1$ is in the homogeneous evolution stage, and the weight $\alpha = 1$, $\beta = 0.1$ is in the heterogeneous evolution stage. The evolution probabilities of behaviour expression chromosomes are set as $P_r = 0.1$, $P_{sm} = 0.1$, $P_{tm} = 0.2$, $P_{sr} = 0.2$, $P_{fr} = 0.1$, $P_{si} = 0.1$, and $P_{fi} = 0.1$. In the case where a reference gene is required, the probability is set as $P_r = 0.1$, $P_{sm} = 0.5$, $P_{tm} = 0.4$, $P_{sr} = 0$, $P_{fr} = 0$, $P_{si} = 0$, and $P_{fi} = 0$ with an empty local memory pool.

5.1. Correlation experiments

In this experiment, we explored the correlation between the fitness function values of the homogeneous and heterogeneous genomes in different swarms.

The experiment was conducted in swarms with size $N = 10, 20, 30$, and 40. First, a genome $\mathbf{G} = \{G_1, G_2, \dots, G_{n_r}\}$ is randomly generated, and the number of genes in the

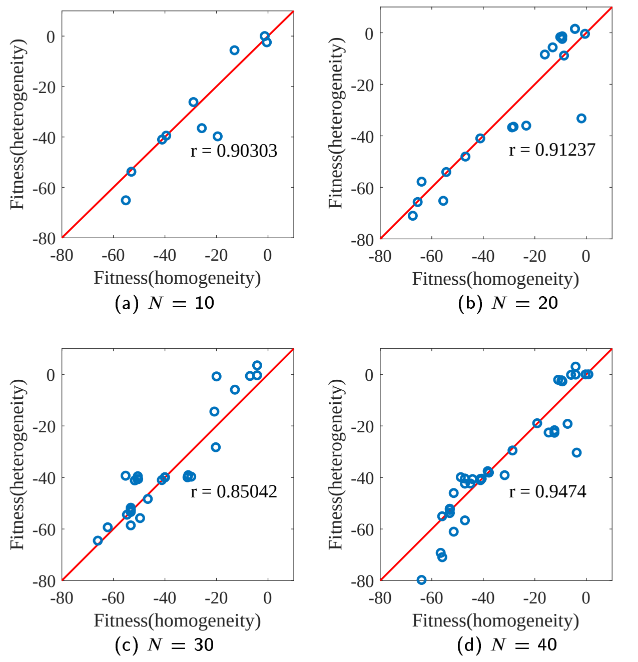


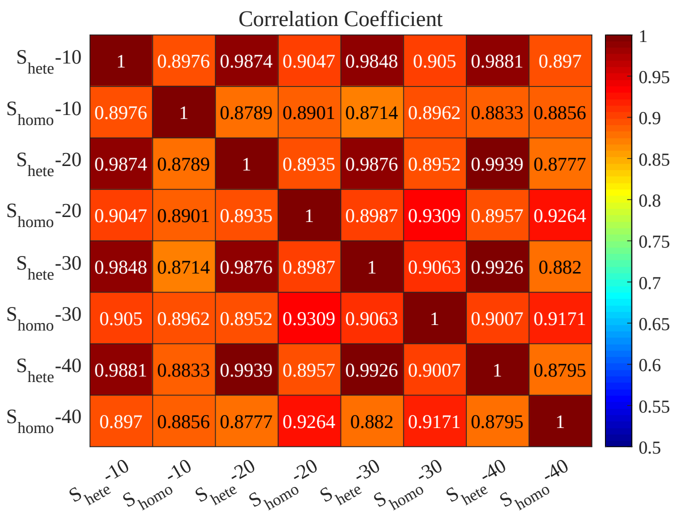
Figure 10: Correlation between the estimated and accurate fitness of homogeneous and heterogeneous swarm robots with different scales.

genome is equal to the swarm size N, that is, $n_r = N$. Thereafter, these genes are assigned to the homogeneous and the heterogeneous swarm robots, respectively. Regarding the homogeneous swarm robots, each robot selects the same gene, and the swarm evaluates a gene accurately through one task execution to evaluate the n_r random genes using n_r flocking task executions. Considering the heterogeneous swarm robots, each robot selects a gene in the genome, and the swarm estimates all the genes in the genome through one task execution.

The correlation between the estimated and accurate fitness of the homogeneous and heterogeneous swarm robots under different scales is shown in Fig. 10. The horizontal coordinate is the estimated evaluation value of the genes obtained by a heterogeneous swarm, whereas the vertical coordinate is the accurate evaluation value of genes obtained by a homogeneous swarm. The similarity between the fitness is measured using the Spearman correlation coefficient [45]. The correlation between them is calculated as r ($r \in [-1, 1]$), according to Eq. (5). The output range of r is -1 to +1, 0 represents no correlation; a negative value indicates a negative correlation, whereas a positive value represents a positive correlation. The line with a correlation of 1 is shown. The closer the points are to the line, the greater the correlation is. This indicates that the estimated evaluation is approximately equal to the accurate evaluation.

$$r = 1 - \frac{6\sum d^2}{n\left(n^2 - 1\right)}, \tag{5}$$

where d is the difference between the ranks of the two



	S_hete-10	S_homo-10	S_hete-20	S_homo-20	S_hete-30	S_homo-30	S_hete-40	S_homo-40
S_hete-10	1	0.8976	0.9874	0.9047	0.9848	0.905	0.9881	0.897
S_homo-10	0.8976	1	0.8789	0.8901	0.8714	0.8962	0.8833	0.8856
S_hete-20	0.9874	0.8789	1	0.8935	0.9876	0.8952	0.9939	0.8777
S_homo-20	0.9047	0.8901	0.8935	1	0.8987	0.9309	0.8957	0.9264
S_hete-30	0.9848	0.8714	0.9876	0.8987	1	0.9063	0.9926	0.882
S_homo-30	0.905	0.8962	0.8952	0.9309	0.9063	1	0.9007	0.9171
S_hete-40	0.9881	0.8833	0.9939	0.8957	0.9926	0.9007	1	0.8795
S_homo-40	0.897	0.8856	0.8777	0.9264	0.882	0.9171	0.8795	1

Figure 11: Fitness correlation coefficient of 120 genes calculated using the heterogeneous and homogeneous swarms of different sizes. The evaluated fitness of the heterogeneous and homogeneous swarms are indicated as S_{hete} and S_{homo}, respectively. The swarm sizes are 10, 20, 30, and 40. S_{hete}-10 is the fitness of a heterogeneous swarm, including ten robots.

columns of fitness and n is the length of each column.

Considering Fig. 10 the swarms of different sizes have the values of correlation r as 0.90303, 0.91237, 0.85042, and 0.9474, respectively. The values of correlation are in the range [0.8,1], indicating that there is a strong correlation between the estimated evaluation of the heterogeneous swarm and the accurate evaluation of the homogeneous swarm. Therefore, we can evaluate the fitness of genes with the help of the estimated evaluation of the heterogeneous swarm.

We evaluated a genome with 120 genes in different swarms of $N = 10, 20, 30,$, and 40 and calculated their estimated and accurate fitness. The correlation coefficient is shown in Fig. 11. The fitness correlation of all the swarms fluctuates at approximately 0.9. This demonstrates that there is a little difference in the calculation results of the gene fitness among different swarms. Therefore, the heterogeneous swarm evolution can be used to speed up gene evaluation. Among them, the correlation of S_{hete} is close to one in different sizes of the heterogeneous swarm, indicating that the size of the heterogeneous swarm has a little effect on the gene's fitness evaluation. Therefore, we can flexibly set the size of the heterogeneous swarms. Considering the computational efficiency, in the following experiments, the swarm scale N is fixed at ten.

5.2. Effectiveness experiments

Based on the calculated correlation in the previous experiments, the feasibility of the proposed TORCH is verified. To verify the effectiveness of the proposed method, it is applied to evolve the strategy of a swarm of ten robots to perform the flocking task. The optimal fitness and its corresponding gene changes during the evolution are shown in Fig. 12.

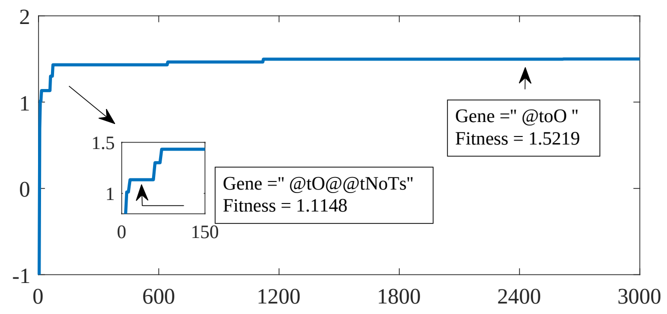


Figure 12: Optimal fitness and its gene changes during the evolution of a flocking task.

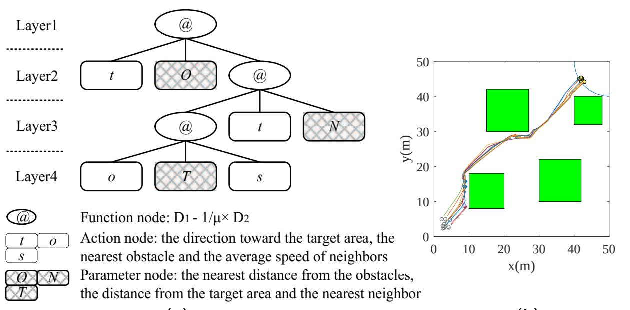

Layer1
Layer2
Layer3
Layer4
Function node: D1 - 1/μ× D2
Action node: the direction toward the target area, the nearest obstacle and the average speed of neighbors
Parameter node: the nearest distance from the obstacles, the distance from the target area and the nearest neighbor
(a)
(b)

Figure 13: Behaviour expression tree of the gene "@$tO@@tNoTs$" obtained in the evolution process and the trajectories of the swarm robots under this gene: (a) behaviour expression tree (b) trajectories of the swarm robots.

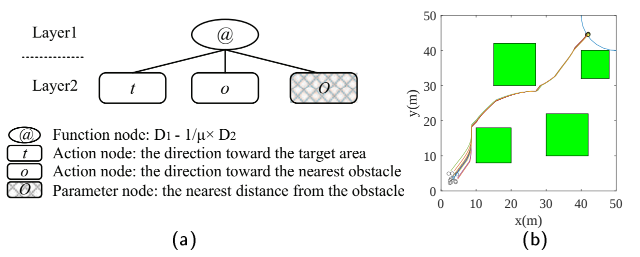

Layer1
Layer2
Function node: D1 - 1/μ× D2
Action node: the direction toward the target area
Action node: the direction toward the nearest obstacle
Parameter node: the nearest distance from the obstacle
(a)
(b)

Figure 14: Behaviour expression tree of the gene "@toO" when fitness = 1.5219 and the trajectories of the swarm robots under this gene: (a) behaviour expression tree (b)trajectories of swarm robots.

With the increase of iterations in the evolutionary process, the fitness of the optimal gene is gradually improved. When the evolutionary generation reaches 1200, the process of the strategy evolution of the swarm has almost converged. We focus on the two genes obtained in the evolution process, which are also the strategies for swarm robots to perform the flocking task.

At the early stage of the evolution process, the behaviour expression of the optimal gene is "@$tO@@tNoTs$", and the fitness is calculated as 1.1148. The specific structure after decoding it into a behaviour expression tree and the trajectories of the swarm robots under this gene are shown in Fig. 13. By decoding the behaviour expression tree, the behaviour strategy of the swarm robots (i.e., the next movement direction) is shown as Eq. (6).

$$\theta_i = t - 1/O \times ((o - s/T) - t/N), \tag{6}$$

where, θ_i is the next movement direction of robot R_i. O, N, and T are the parameters, which are the distance toward the nearest obstacle, nearest neighbour, and target area, respectively. t, o, and s are the vectors pointing to the target area, nearest obstacle, and average velocity of the neighbours.

Under this evolved behaviour model, the swarm robots reach the terminal area stably. However, the swarm does not achieve smooth obstacle avoidance. In addition, the aggregation degree and speed smoothness of the swarm need to be improved. When the evolutionary generation reaches 2600, the optimal strategy evolves to "@toO" and 1.5219 fitness. The specific structure after decoding into the behaviour expression tree and trajectories of the swarm robots are shown in Fig. 14. The behaviour strategy expression of the swarm robots is shown as Eq. (7).

$$\theta_i = t - 1/O \times o, \tag{7}$$

where θ_i is the next movement direction of robot R_i. O is the distance toward the nearest obstacle. t and o are the vectors pointing to the target area and nearest obstacle, respectively.

The expression clearly shows the main factors that need to be considered most in the swarm robots flocking task: the direction toward the target area, the direction toward the nearest obstacle, and the nearest distance from the obstacles. The function $D_1 - 1/\mu \times D_2$ is used to connect these factors. A clear strategy expression will contribute to a systematic and more structured analysis and further research of the task. The evolved strategy shows that the target area is the main direction, and the closer the robot is to the obstacle, the larger deflection is needed to keep it away from the nearest obstacle. Fig. 14(b) shows the trajectories of the swarm robots. The evolved gene guides the swarm robots to reach the target area more quickly with more intensive format, while the obstacle avoidance is smoother and the evolved behavior expression is simpler.

5.3. Adaptability experiments

The goal of strategy evolution is to evolve a strategy that can be applied to a class of scenario rather than a specific task scenario. Therefore, in order to verify the adaptability of our evolved strategy to different task scenarios, we adjust the shape and position of obstacles in the task scenario to generate new complex task scenarios. We make the swarm to perform the flocking task in the new complex task scenarios with the evolved gene "@toO".

The trajectories of the swarm under the evolved gene are shown in Fig. 15. In different scenes, the swarms can avoid obstacles smoothly. The fitness in Fig. 15(b) is less than Fig. 15(a) because the time to reach the target area is increased with changed obstacles.

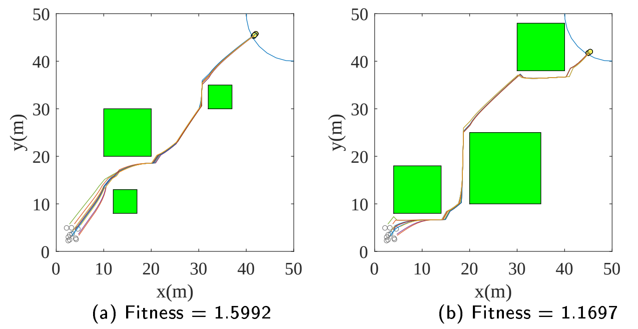

(a) Fitness = 1.5992 (b) Fitness = 1.1697

Figure 15: Trajectories of swarm robots in the newly set task scenes.

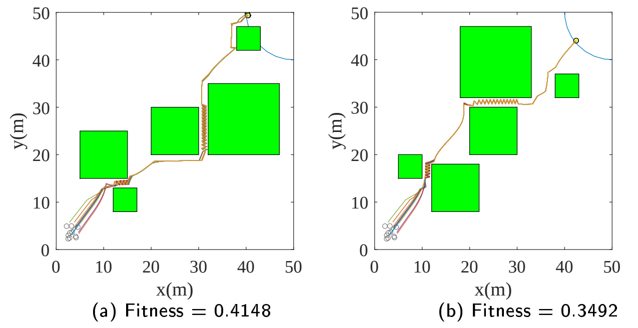

(a) Fitness = 0.4148 (b) Fitness = 0.3492

Figure 16: Trajectories of swarm robots in task scenes with narrow channels.

To further increase the complexity of the scene, we use the combination of two obstacles to set up a narrow channel in the task scene. In the complex scene, the swarm is requested to pass through the narrow channel to reach the target area in time. Fig. 16 shows the trajectories of swarm robots in the task scenes with narrow channels. Under the evolved strategy, swarm robots can still pass through the narrow channels and reach the target area in a short period of time despite the fluctuation in the motion. The fitness of the gene reduced owing to the movement of the swarm robots in the narrow channels.

5.4. Superiority experiments

In this section, we quantitatively compare the TORCH to the traditional homogeneous swarm evolution method to verify the performance improvement of the TORCH. The proposed TORCH uses the behaviour expression tree as the expression of strategy which has a larger strategy search space than the traditional parameter expression. Therefore, we compare the behaviour expression tree to the parameter expression. Details of the two methods in the comparison are given as follows:

- Homogeneous swarm evolution method: All the individuals of the swarm follow the same behavioural model. The swarm obtains a gene evaluation in each iteration, and the whole genome evaluations are obtained through multiple iterations. After calculating the evaluation values of the genes, a new genome is evolved and iterates for several iterations to obtain the evaluation value of the new genome.

- Parameter expression: When an individual chooses an action, all the possible motion directions are calculated, and the next motion direction in the current state is obtained using the weighted average of these motion directions. Only the parameters of these weights need to be optimized in the evolution process.

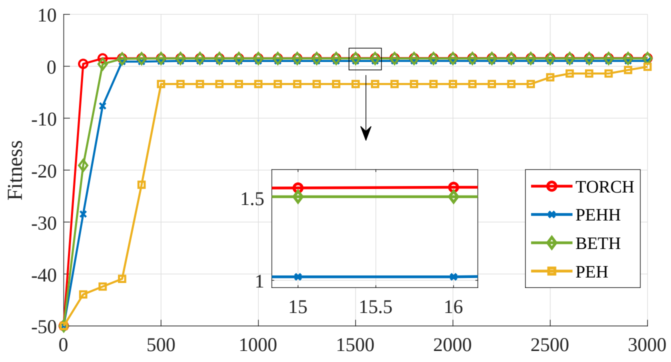


Figure 17: Fitness of the four methods under 3000 iterations.

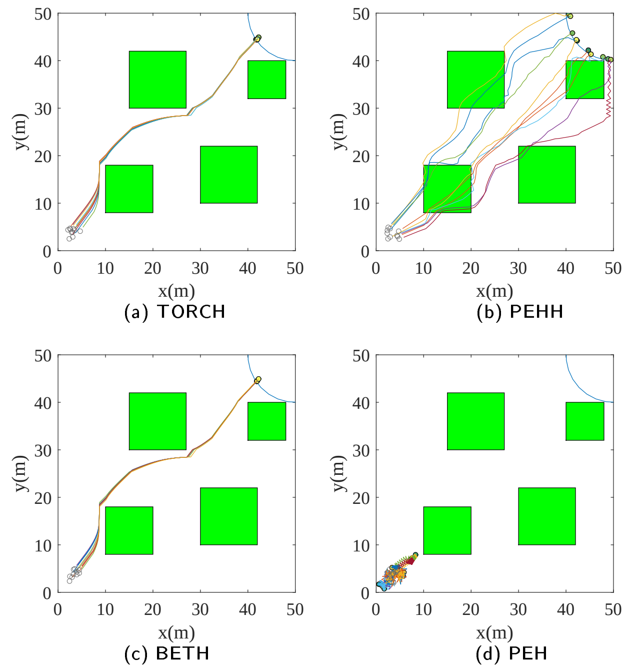

(a) TORCH (b) PEHH (c) BETH (d) PEH

Figure 18: Paths of the swarm flocking under the optimal strategies of the four methods.

We quantitatively compare the four methods to verify the advantages of the proposed method. The three compared methods are briefly described as follows: Parameter Expression based Heterogeneous-Homogeneous swarm coevolution method (PEHH), Behaviour Expression Tree based Homogeneous swarm coevolution method (BETH), and Parameter Expression based Homogeneous swarm coevolution method (PEH).

The experiments were conducted with the same initial task scene and the setting of the evolution parameters. Fig. 17 shows the optimal gene fitness of the four methods under 3000 iterations. The TORCH improves the fitness at the fastest speed and keeps it at a high value. The search speed of the BETH is second to the TORCH, and the optimal gene fitness is higher than the other methods. This indicates that the behaviour expression tree method can not only expand the strategy search space but also improve the flocking performance of the evolved strategies. The TORCH is superior to other methods in terms of performance and search speed.

After 3000 iterations, the paths of the swarm flocking under the optimal strategies of each method are shown in Fig. 18. It is shown that the approximate optimal strategies can be found using the method of the behaviour expression tree. In Fig. 18(a) and (c), the trajectories of swarm robots are similar; nonetheless, because of the long strategy of the robots, the calculated fitness is lower in Fig. 18(c). However, the evolved strategies cannot achieve obstacle avoidance using the parameter expression. This is because the strategy search space of the parametric strategy expression method is small, and it is unable to find a better strategy to complete the flocking task. In Fig. 18(b), the swarm reaches the target area quickly and reduces the time inside the obstacles; nevertheless, it does not avoid obstacles reliably. Moreover, in Fig. 18(d), the swarms move around the initial area because they do not find a way to reach the target area without obstacles. Because of the negative reward for entering the obstacles, hovering around the initial area is an effective way to avoid a low reward value.

6. Conclusion and future work

In this study, we propose an effective strategy evolution method, TORCH. The TORCH uses the heterogeneous-homogeneous swarm coevolution mechanism to improve the performance and convergence speed of the strategy evolution method. Furthermore, TORCH is based on a novel strategy expression method known as the behaviour expression tree which is an extension of the conventional expression tree to enhance the performance of the evolved strategy. The proposed TORCH only uses local information in the evolution process; thus, the evolved strategies are more suitable for distributed task scenarios. The TORCH is a method that can be applied to solve a variety of problems of swarm robots. Considering the designed experiments, the TORCH successfully addresses the issue of autonomous obstacle avoidance in the flocking task of swarm robots. It can also achieve super performance and efficient strategy search. In addition, the strategy expression method based on the behaviour expression tree results in a larger strategy search space. Extensive experiments demonstrate that TORCH can improve the efficiency of strategy evolution and enhance the performance of the evolved strategy effectively.

The following issues will be addressed in our future studies. First, we will take the communication instability of swarm robots' tasks into consideration to improve the robustness of the model to consider more challenging task scenes. Second, we will extend our heterogeneous-homogeneous swarm coevolution method to enable online strategy evolution of the swarm. Third, we plan to implement the TORCH in a real SRS to further test its performance. Finally, it is expected that the evolved strategies can inspire an artificial strategy design, and better artificial strategies design can inspire the evolutionary process to obtain strategies beyond expectation.

References

[1] Bjerknes, J.D., Winfield, A.F.T., Melhuish, C., 2007. An analysis of emergent taxis in a wireless connected swarm of mobile robots, in: 2007 IEEE Swarm Intelligence Symposium, SIS 2007, Honolulu, Hawaii, USA, April 1-5, 2007, IEEE. pp. 45–52. URL: https://doi.org/10.1109/SIS.2007.368025.

[2] Cai, Y., Yang, S., Mittal, G., 2013. A pso-based approach to cooperative foraging multi-robots in unknown environments, in: IEEE 6th International Conference on Robotics, Automation and Mechatronics, RAM 2013, Manila, Philippines, November 12-15, 2013, pp. 67–72. URL: https://doi.org/10.1007/s10015-020-00642-2.

[3] Chen, J., Gauci, M., Li, W., Kolling, A., Groß, R., 2015. Occlusion-based cooperative transport with a swarm of miniature mobile robots. IEEE Trans. Robotics 31, 307–321. URL: https://doi.org/10.1109/TRO.2015.2400731.

[4] Chen, Y., 2016. Industrial information integration-a literature review 2006-2015. Journal of Industrial Information Integration 2, 30–64. URL: https://doi.org/10.1016/j.jii.2016.04.004.

[5] Chen, Y., 2020. A survey on industrial information integration 2016-2019. Journal of Industrial Integration and Management 5, 33–163. URL: https://doi.org/10.1142/S2424862219500167.

[6] Ding, I.J., Chang, Y.J., 2016. On the use of kinect sensors to design a sport instructor robot for rehabilitation and exercise training of the elderly, in: IEEE International Conference on Applied System Innovation, ICASI 2016, Okinawa, Japan, May 26-30, 2016, pp. 463–476.

[7] Dorigo, M., Floreano, D., Gambardella, L.M., Mondada, F., Nolfi, S., Baaboura, T., Birattari, M., Bonani, M., Brambilla, M., Brutschy, A., Burnier, D., Campo, A., Christensen, A.L., Decugniere, A., Caro, G.D., Ducatelle, F., Ferrante, E., Förster, A., Gonzales, J.M., Guzzi, J., Longchamp, V., Magnenat, S., Mathews, N., de Oca, M.A.M., O'Grady, R., Pinciroli, C., Pini, G., Rétornaz, P., Roberts, J.F., Sperati, V., Stirling, T.S., Stranieri, A., Stützle, T., Trianni, V., Tuci, E., Turgut, A.E., Vaussard, F., 2013. Swarmanoid: A novel concept for the study of heterogeneous robotic swarms. IEEE Robotics Autom. Mag. 20, 60–71. URL: https://doi.org/10.1109/MRA.2013.2252996.

[8] Dorigo, M., Tuci, E., Groß, R., Trianni, V., Labella, T.H., Nouyan, S., Ampatzis, C., Deneubourg, J., Baldassarre, G., Nolfi, S., Mondada, F., Floreano, D., Gambardella, L.M., 2004. The SWARM-BOTS project, in: Swarm Robotics, SAB 2004 International Workshop, Santa Monica, CA, USA, July 17, 2004, Springer. pp. 31–44. URL: https://doi.org/10.1007/978-3-540-30552-1_4.

[9] Ducatelle, F., Caro, G.A.D., Gambardella, L.M., 2010. Cooperative stigmergic navigation in a heterogeneous robotic swarm, in: 11th International Conference on Simulation of Adaptive Behavior, SAB 2010, Paris, France, August 25-28, 2010, Springer. pp. 607–617. URL: https://doi.org/10.1007/978-3-642-15193-4_57.

[10] Ferreira, C., 2001. Gene expression programming: a new adaptive algorithm for solving problems. Complex Syst. 13, 87–129. URL: http://www.complex-systems.com/abstracts/v13_i02_a01.html.

[11] Gauci, M., Chen, J., Li, W., Dodd, T., Groß, R., 2014. Clustering objects with robots that do not compute, pp. 421–428. URL: http://dl.acm.org/citation.cfm?id=2615800.

[12] Gebhardt, G., Daun, K., Schnaubelt, M., Neumann, G., 2018.

Learning robust policies for object manipulation with robot swarms, in: IEEE International Conference on Robotics and Automation, ICRA 2018, Brisbane, Australia, May 21-25, 2018, pp. 7688–7695. URL: https://doi.org/10.1109/ICRA.2018.8463215.

[13] Groß, R., Dorigo, M., 2009. Towards group transport by swarms of robots. International Journal of Bio-Inspired Computation 1, 1–13. URL: https://doi.org/10.1504/IJBIC.2009.022770.

[14] Guo, H., Meng, Y., Jin, Y., 2009. A cellular mechanism for multi-robot construction via evolutionary multi-objective optimization of a gene regulatory network. Biosyst. 98, 193–203. URL: https://doi.org/10.1016/j.biosystems.2009.05.003.

[15] Hamann, H., 2018. Swarm Robotics - A Formal Approach. Springer. URL: https://doi.org/10.1007/978-3-319-74528-2.

[16] Hayes, A.T., Martinoli, A., Goodman, R.M., 2003. Swarm robotic odor localization: Off-line optimization and validation with real robots. Robotica 21, 427–441. URL: https://doi.org/10.1017/S0263574703004946.

[17] Iima, H., Kuroe, Y., 2013. Swarm reinforcement learning method for a multi-robot formation problem, in: IEEE International Conference on Systems, Man, and Cybernetics, Manchester, SMC 2013, United Kingdom, October 13-16, 2013, pp. 2298–2303. URL: https://doi.org/10.1109/SMC.2013.393.

[18] Jin, B., Liang, Y., Han, Z., Ohkura, K., 2020. Generating collective foraging behavior for robotic swarm using deep reinforcement learning. Artif. Life Robotics 25, 588–595. URL: https://doi.org/10.1007/s10015-020-00642-2.

[19] Jin, Y., 2011. Surrogate-assisted evolutionary computation: Recent advances and future challenges. Swarm Evol. Comput. 1, 61–70. URL: https://doi.org/10.1016/j.swevo.2011.05.001.

[20] Juang, C., Jeng, T., Chang, Y., 2016. An interpretable fuzzy system learned through online rule generation and multiobjective ACO with a mobile robot control application. IEEE Trans. Cybern. 46, 2706–2718. URL: https://doi.org/10.1109/TCYB.2015.2486779.

[21] Khan, A., Rinner, B., Cavallaro, A., 2018. Cooperative robots to observe moving targets: Review. IEEE Trans. Cybern. 48, 187–198. URL: https://doi.org/10.1109/TCYB.2016.2628161.

[22] Kumar, S., Vanualailai, J., Sharma, B., Prasad, A., 2021. Velocity controllers for a swarm of unmanned aerial vehicles. Journal of Industrial Information Integration 22, 100198. URL: https://doi.org/10.1016/j.jii.2020.100198.

[23] Kwok, N.M., Liu, D., Dissanayake, G., 2006. Evolutionary computing based mobile robot localization. Eng. Appl. Artif. Intell. 19, 857–868. URL: https://doi.org/10.1016/j.engappai.2006.01.020.

[24] Leottau, D., Ruiz-del Solar, J., Macalpine, P., Stone, P., 2015. A study of layered learning strategies applied to individual behaviors in robot soccer, in: 19th Annual RoboCup International Symposium, RoboCup 2015, Hefei, China, July 23, 2015, pp. 290–302. URL: https://doi.org/10.1007/978-3-319-29339-4_24.

[25] Ma, L., Bao, W., Zhu, X., Wu, M., Wang, Y., Ling, Y., Zhou, W., 2020. O-Flocking: Optimized Flocking Model on Autonomous Navigation for Robotic Swarm. pp. 628–639. URL: https://doi.org/10.1007/978-3-030-53956-6_58.

[26] Mehes, E., Vicsek, T., 2014. Collective motion of cells: From experiments to models. Integrative biology: quantitative biosciences from nano to macro 6, 831–854. doi:10.1039/c4ib00115j.

[27] Mukhlish, F., Page, J., Bain, M., 2018. Evolutionary-learning framework: Improving automatic swarm robotics design. International journal of intelligent unmanned systems 6, 197–215. URL: https://doi.org/10.1108/IJIUS-06-2018-0016.

[28] Nagavalli, S., Chakraborty, N., Sycara, K., 2017. Automated sequencing of swarm behaviors for supervisory control of robotic swarms, in: IEEE International Conference on Robotics and Automation, ICRA 2017, Singapore, Singapore, May 29-June 3, 2017, pp. 2674–2681. URL: https://doi.org/10.1109/ICRA.2017.7989312.

[29] Naidoo, N., Bright, G., Stopforth, R., 2019. A distributed framework for programming the artificial intelligence of mobile robots in smart manufacturing systems, in: Southern African Universities Power Engineering Conference/Robotics and Mechatronics/Pattern Recognition Association of South Africa, SAUPEC/RobMech/PRASA 2019, Bloemfontein, South Africa, January 28-30, 2019, pp. 34–41. doi:10.1109/RoboMech.2019.8704788.

[30] Nedjah, N., de Macedo Mourelle, L., 2016. Distributed learning algorithms for swarm robotics. Neurocomputing 172, 290–291. URL: https://doi.org/10.1016/j.neucom.2015.06.085.

[31] Nolfi, S., Bongard, J., Husbands, P., Floreano, D., 2016. Evolutionary robotics , 2035–2068.URL: https://doi.org/10.1007/978-3-319-32552-1_76.

[32] Oh, H., Jin, Y., 2014. Evolving hierarchical gene regulatory networks for morphogenetic pattern formation of swarm robots, in: IEEE Congress on Evolutionary Computation, CEC 2014, Beijing, China, July 6-11, 2014, pp. 776–783. URL: https://doi.org/10.1109/CEC.2014.6900365.

[33] Oh, H., Shirazi, A.R., Sun, C., Jin, Y., 2017. Bio-inspired self-organising multi-robot pattern formation: A review. Robotics Auton. Syst. 91, 83–100. URL: https://doi.org/10.1016/j.robot.2016.12.006.

[34] Olfati-Saber, R., 2006. Flocking for multi-agent dynamic systems: algorithms and theory. IEEE Trans. Autom. Control. 51, 401–420. URL: https://doi.org/10.1109/TAC.2005.864190.

[35] Parker, L., Touzet, C., 2000. Multi-robot learning in a cooperative observation task. Distributed Autonomous Robotic Systems 4, 391–402. URL: https://doi.org/10.1007/978-4-431-67919-6_37.

[36] Pugh, J., Martinoli, A., 2007. Parallel learning in heterogeneous multi-robot swarms, in: IEEE Congress on Evolutionary Computation, CEC 2007, Singapore, Singapore, September 25-28, 2007, pp. 3839–3846. URL: https://doi.org/10.1109/CEC.2007.4424971.

[37] Reynolds, C., 1987. Flocks, herds and schools: A distributed behavioral model. Computer Graphics 21, 25–34. URL: https://doi.org/10.1145/37401.37406.

[38] Riedmiller, M., Gabel, T., Hafner, R., Lange, S., 2009. Reinforcement learning for robot soccer. Auton. Robots 27, 55–73. URL: https://doi.org/10.1007/s10514-009-9120-4.

[39] Rubenstein, M., Cornejo, A., Nagpal, R., 2014. Robotics. programmable self-assembly in a thousand-robot swarm. Science (New York, N.Y.) 345, 795–799. doi:10.1126/science.1254295.

[40] Sadeghi-Esfahlani, S., 2018. Mixed reality and remote sensing application of unmanned aerial vehicle in fire and smoke detection. Journal of Industrial Information Integration 15, 42–49. URL: https://doi.org/10.1016/j.jii.2019.04.006.

[41] Sayama, H., 2010. Robust morphogenesis of robotic swarms [application notes]. IEEE Comput. Intell. Mag. 5, 43–49. URL: https://doi.org/10.1109/MCI.2010.937323.

[42] Sedjelmaci, H., Senouci, S.M., Ansari, N., 2017. A hierarchical detection and response system to enhance security against lethal cyber-attacks in uav networks. IEEE Trans. Syst. Man Cybern. Syst. 48, 1594–1606. URL: https://doi.org/10.1109/TSMC.2017.2681698.

[43] Seo, S.W., Ko, K., Yang, H.C., Sim, K.B., 2007. Behavior learning and evolution of swarm robot system using support vector machine. Journal of Korean institute of intelligent systems 18, 1238–1242. doi:10.1109/ICCAS.2007.4406524.

[44] Shi, Z., Tu, J., Li, Y., Wang, Z., 2013. Adaptive reinforcement q-learning algorithm for swarm-robot system using pheromone mechanism, in: IEEE International Conference on Robotics and Biomimetics, ROBIO 2013, Shenzhen, China, December 12-14, 2013, pp. 952–957. URL: https://doi.org/10.1109/ROBIO.2013.6739586.

[45] Spearman, C., 1904. The proof and measurement of association between two things. The American Journal of Psychology 15, 72–101. URL: https://www.jstor.org/stable/1412159.

[46] Tahir, A., BÃűling, J., Haghbayan, M.H., Toivonen, H., Plosila, J., 2019. Swarms of unmanned aerial vehicles - a survey. Journal of Industrial Information Integration 16, 100106. URL: https://doi.org/10.1016/j.jii.2019.100106.

[47] Takadama, K., Hajiri, K., Nomura, T., Shimohara, K., Okada, M.,

Nakasuka, S., 1998. Learning model for adaptive behaviors as an organized group of swarm robots. Artif. Life Robotics 2, 123–128. URL: https://doi.org/10.1007/bf02471168.

[48] Tao, F., Laili, Y., Liu, Y., Feng, Y., Wang, Q., Zhang, L., Xu, L., 2014. Concept, principle and application of dynamic configuration for intelligent algorithms. IEEE Syst. J. 8, 28–42. URL: https://doi.org/10.1109/JSYST.2013.2275619.

[49] Tao, F., Laili, Y., Xu, L., Zhang, L., 2013. FC-PACO-RM: A parallel method for service composition optimal-selection in cloud manufacturing system. IEEE Trans. Ind. Informatics 9, 2023–2033. URL: https://doi.org/10.1109/TII.2012.2232936.

[50] V*ásá*rhelyi, G., Vir*á*gh, C., Somorjai, G., Nepusz, T., Eiben, A., Vicsek, T., 2018. Optimized flocking of autonomous drones in confined environments. Science Robotics. 3, eaat3536. URL: https://doi.org/10.1126/scirobotics.aat3536.

[51] Wang, Q., Mao, X., Yang, S., Chen, Y., Liu, X., 2018. Grouping-based adaptive spatial formation of swarm robots in a dynamic environment. International Journal of Advanced Robotic Systems 15, 1–14. doi:10.1177/1729881418782359.

[52] Wang, T., Peng, X., Wu, Y., Gao, J., 2019. A GP based two-layer framework for data-driven modeling of swarm self-organizing rules, in: IEEE Congress on Evolutionary Computation, CEC 2019, Wellington, New Zealand, June 10-13, 2019, IEEE. pp. 174–181. URL: https://doi.org/10.1109/CEC.2019.8790126.

[53] Yang, G.Z., Bellingham, J., Dupont, P., Fischer, P., Floridi, L., Full, R., Jacobstein, N., Kumar, V., McNutt, M., Merrifield, R., Nelson, B., Scassellati, B., Taddeo, M., Taylor, R., Veloso, M., Wang, Z., Wood, R., 2018. The grand challenges of science robotics. Science Robotics 3, eaar7650. URL: https://doi.org/10.1126/scirobotics.aar7650.

[54] Yasuda, T., Ohkura, K., 2019. Sharing experience for behavior generation of real swarm robot systems using deep reinforcement learning. Journal of Robotics and Mechatronics 31, 520–525. URL: https://doi.org/10.20965/jrm.2019.p0520.

[55] Yasuda, T., Wada, N., Ohkura, K., Matsumura, Y., 2013. Analyzing collective behavior in evolutionary swarm robotic systems based on an ethological approach, in: IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning, ADPRL 2013, Singapore, Singapore, April 16-19, 2013, IEEE. pp. 148–155. URL: https://doi.org/10.1109/ADPRL.2013.6615001.

[56] Yu, S., Aleti, A., Barca, J., Song, A., 2018. Hyper-heuristic Online Learning for Self-assembling Swarm Robots. pp. 167–180. URL: https://doi.org/10.1007/978-3-319-93698-7_13.

[57] Zhang, G., Li, Y., Xu, X., Dai, H., 2019. Multiagent reinforcement learning for swarm confrontation environments, in: 12th Intelligent Robotics and Applications, ICIRA 2019, Shenyang, China, August 8-11, 2019, Proceedings, Part III, pp. 533–543. URL: https://doi.org/10.1007/978-3-030-27535-8_48.

[58] Zou, J., Li, Q., Yang, S., Zheng, J., Peng, Z., Pei, T., 2019. A dynamic multiobjective evolutionary algorithm based on a dynamic evolutionary environment model. Swarm Evol. Comput. 44, 247–259. URL: https://doi.org/10.1016/j.swevo.2018.03.010.