applied sciences

MDPI

Article

Dynamic Multi-Target Self-Organization Hunting Control of Multi-Agent Systems

Shouzhong He [1], Liangshun Wang [1,2], Mingming Liu [3], Weifeng Liu [1,*] and Zhihai Wu [4]

[1] School of Mechanical and Electrical Engineering, Hainan University, Haikou 570228, China; 21220854000008@hainanu.edu.cn (S.H.); 993576@hainanu.edu.cn (L.W.)
[2] Research Institute of Information Technology, Tsinghua University, Beijing 100101, China
[3] School of Mechanical Engineering, Beijing Institute of Technology, Beijing 100081, China; 3120220325@bit.edu.cn
[4] Engineering Research Center of Internet of Things Technology Applications of MOE, School of Internet of Things Engineering, Jiangnan University, Wuxi 214122, China; wuzhihai@jiangnan.edu.cn
* Correspondence: 993597@hainanu.edu.cn

Abstract: In this paper, we present a novel coordinated method tailored to address the dynamic multi-target hunting control problem in multi-agent systems, offering significant practical value. Our approach encompasses several key components: initially, we introduce a task allocation model that integrates a fuzzy inference system with a particle swarm optimization algorithm. This hybrid model efficiently allocates hunting tasks for scattered evading targets, effectively transforming the dynamic multi-target hunting problem into multiple dynamic single-target-hunting problems. This transformation enhances the speed and efficacy of task allocation. Subsequently, we propose an attraction/repulsive model grounded in potential field theory. This model facilitates the coordinated hunting of each target by organizing agents into subgroups. Relying solely on relative position and velocity information between agents and targets, our model simplifies computation, while maintaining effectiveness. Furthermore, the coordination of hunting activities for each target is achieved through a series of agent subgroups, guided by our proposed motion model. This systematic approach ensures a cohesive and efficient hunting strategy. Finally, we validate the effectiveness and feasibility of our proposed method through simulation results. These results provide empirical evidence of the method's efficacy and potential applicability in real-world scenarios.

Keywords: multi-agent systems; task allocation; target hunting; cooperative control

check for updates

Citation: He, S.; Wang, L.; Liu, M.; Liu, W.; Wu, Z. Dynamic Multi-Target Self-Organization Hunting Control of Multi-Agent Systems. *Appl. Sci.* **2024**, *14*, 3875. https://doi.org/10.3390/app14093875

Academic Editor: Mohammed Chadli

Received: 2 March 2024
Revised: 22 April 2024
Accepted: 30 April 2024
Published: 30 April 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

In recent years, with the rapid development of artificial intelligence and biomimetic technology, multi-agent systems (MASs) (such as drone swarms, robot swarms, multiple unmanned boats, and sensor networks) have been widely used in collaborative search and rescue, target tracking, fault detection, and other tasks due to their distributed, flexible, scalable, and low-cost characteristics. The decision making and analysis of multi-agent behavior for controlling large-scale systems have come to the forefront of current research in the field of control, as these systems rely on agent communication to achieve the overall expected behavior. With the joint efforts of numerous researchers, MASs have made significant progress in research areas such as consensus [1,2], cluster [3], flocking/swarm [4,5], and hunting control. However, with the expansion of the application scope, some challenges have also arisen for practical engineering problems. How to efficiently and reliably control MASs to perform various tasks and fully realize their potential application value has gradually become one of the current research focuses. Among these difficulties, the MAS target hunting problem is particularly challenging. It requires concise and flexible control instructions to guide agents in coordination with neighboring individuals to complete target hunting with limited information. Research in this area is important in many fields,

such as the military, transportation, and the environment. It offers multiple application prospects in many scenarios, such as search, surveillance, rescue, environmental monitoring, marine scientific research, and border law enforcement. In this context, this paper proposes a new coordination technique based on practical applications, aiming to solve the dynamic problem of multi-target hunting control.

The target-hunting problem in MASs is technically related to consistency theory and distributed control problems. Initially, Olfati-Saber [6] was the first to consider the flocking problem of MASs, providing a solution for further research regarding the hunting problem while revealing optimality and theoretically achievable issues. Furthermore, Talebi S. P. et al. [7] proposed a distributed framework for controlling the state space processes of agent networks. According to the existing literature, more researchers, such as Kou et al. [8–10], who pay more attention to the structural relationships between agents and hunting formations during the hunting process, have focused on studying individual hunting targets. Xie et al. [11] studied the formation control problem during the hunting process and applied the gray wolf tracking strategy to the single-target hunting problem. However, only centralized hunting was implemented in the paper. Guo et al. [12,13] proposed a local information control law that only utilizes the relative position information between the target and its neighbors to achieve dynamic target hunting. The results showed that multi-robot systems can coordinate and estimate the motion speed of the target. Meanwhile, Huang et al. [14] considered a more complex siege scenario, based on an autonomous underwater vehicle (AUV), and a multi-AUV collaborative hunting algorithm, based on bionic neural networks. The underwater environment was first modeled, and then an efficient capture path was planned for the AUV to surround the target. Further, Zengin et al. [15] studied the collaborative target-hunting problem in adversarial environments, and a collaborative strategy for tracking and attacking targets using UAVs was developed. Fan et al. [16] proposed a solution for collaborative capture in three-dimensional environments that can achieve both capture and obstacle navigation. This solution includes a 3-D synchronous encirclement strategy, a path planning algorithm, obstacle avoidance, and a cascaded PI (proportional integral) controller. Simulation results demonstrate that this solution can search and capture static or dynamic targets in obstacle environments. Chen et al. [17] were is no longer limited to homogeneous agents for hunting, further considering the collaborative hunting of heterogeneous underwater robots, proposing a new time-competitive mechanism to build an efficient dynamic hunting coalition. In order to improve the efficiency of target hunting for multi-AUVs, a hunting algorithm based on dynamic prediction of moving target trajectory was proposed [18] A negotiation method was used to allocate appropriate ideal hunting points for each underwater vehicle. Finally, the desired hunting points were quickly reached through the deep reinforcement learning (DRL) algorithm to achieve the hunting of moving targets. The problem of smooth switching from tracking control to bracketing control was further investigated by Yu et al. [19]. The above studies are all aimed at single-target hunting. Multi-target hunting is more complex than single-target hunting. Du et al. [20] studied cooperative tracking strategies and proposed a method based on multi-agent reinforcement learning (MARL). By introducing a parameter-sharing scheme, the proposed method achieved higher hunting rates in a shorter period of time. However, it lacks in-depth discussion on the issue of tracking task allocation, which may hinder maneuverability in the process of multi-target tracking. Regarding the collaborative multi-target pursuit of unmanned surface vehicles (USVs), Xia et al. [21] modeled the collaborative hunting problem of USV fleets as a decentralized, partially observable Markov decision process. They proposed a distributed partially observable multi-target hunting proximal strategy optimization algorithm suitable for USVs. Experiments have shown that even when certain USVs are damaged, the self-organizing ability of the entire USV fleet still maintains an advantage. The hunting of multiple targets mentioned above is achieved using machine learning methods. The advantage of reinforcement learning is that it does not require establishing a system dynamic model, as it is a model-free method. However, a drawback of machine learning methods is that

continuous learning is required to achieve good results, and the learning process consumes a significant amount of time and computing power. Reinforcement learning methods mainly address MASs at the cooperative level, and the information processing between MASs is usually centralized; therefore, it cannot effectively reflect the distributed swarm advantage among agents. Given the shortcomings of reinforcement learning methods and the excellent collaborative ability demonstrated by swarm behavior, this paper proposes a novel trapping strategy based on the research results of the aforementioned literature.

In order to solve the problem of hunting multiple dynamic targets that are scattered and escaping, it is necessary to not only design an appropriate hunting strategy but also to consider how to assign the tasks. The result of good task assignment plays a key role in the efficiency of multi-target hunting. Trigui et al. [22] studied the distributed allocation algorithm for multi-robots. They proposed two algorithms: the distributed market algorithm and the improved distributed market algorithm. Compared with the centralized Hungarian algorithm, these two algorithms obtained approximate optimal solutions in task allocation, which can effectively reduce the cost of task allocation. Liang et al. [23] further investigated the interactive topology and protocol of task allocation in MASs. In order to achieve interactive communication, an extended contract network protocol, based on point-to-point topology, was proposed, which improved the efficiency and quality of task allocation. Jin et al. [24,25] investigated coordination behavior strategies for task assignment using a competitive and cooperative approach. Their strategy enables distributed task assignments and ensures the fairness of the tasks. Liu et al. [26] proposed a multi-task allocation algorithm based on a self-organizing map (SOM) for unmanned surface vessels (USVs) to implement complex ocean operations. For the dynamic assignment problem of multi-robots, they used the multi-target optimization method for estimation, in order to achieve optimal task allocation [27]. Shi et al. [28,29] investigated heuristic algorithms for task assignment, and the results proved that heuristics are efficient, stable, and computationally affordable. This research has studied different task allocation problems from their respective practical application perspectives. However, for the multi-target hunting problem of scattered escape, relying solely on self-organizing task allocation based on limited speed and displacement information may result in too few or no agents around the target, thereby preventing successful capture. Therefore, this paper also considers the limited detection and communication capabilities of the agent itself, forming a multi-level, distributed task allocation method and ultimately, successfully hunting various targets.

Driven by the aforementioned issues outlined previously and based on the original work, this paper introduces a novel approach to self-organization task allocation and hunting control strategy that integrates fuzzy logic and heuristic optimization algorithms, considering the practical engineering application of multi-target hunting in multi-agent systems. By establishing appropriate target models and collaborative strategies, multi-agent systems can achieve the coordinated hunting of targets during the hunting process. The main contributions of this article are as follows:

In this paper, we first focus on the self-organizing allocation of tasks for multi-agent and multi-target hunting and propose a multi-agent multi-target task self-organization allocation algorithm for a dynamic environment. The algorithm combines fuzzy logic and heuristic optimization algorithm, determines the evaluation factor of task allocation, and then implements a globally distributed task assignment based on an improved particle swarm optimization algorithm, with practical application scenario constraints to achieve optimal global system performance. In the hunting control strategy section, an attraction/repulsive force model, based on potential field function, was designed and introduced to achieve the hunting control strategy of predicting the target's motion trajectory. Without knowing the target's motion state, only the relative position and velocity information between the agent and the target can be used for hunting control of the target. This can ensure that various agents can collaborate to reach the target area, hunt the target, and ultimately form an encirclement based on the set hunting radius for hunting.

The organizational structure of this article is as follows: Section 2 lays some preliminary groundwork; Section 3 establishes a dynamic, multi-target, self-organizing task allocation model; Section 4 introduces the design of the hunting strategy and the proof of stability; Section 5 verifies the effectiveness of the theoretical results through simulation experiments; and finally, in Section 6, a brief summary is provided.

2. Preliminary and Problem Formulation

Assuming that all agents are isomorphic—that is, each agent has the same dynamic model and functional attributes—the communication connection of multi-agent systems can be represented by a graph: $G(V, E, A)$. $V = \{V_1, V_2, \ldots, V_N\}$ represents the set composed of N agent nodes in G, and V_i represents agent i. $E \subseteq V_i \times V_j$ is the edge set, representing the node connections in graph G. $A = [a_{ij}] \in \mathbb{R}^{N \times N}$ is the adjacency matrix, and a_{ij} represents the connection weight between agent i and agent j. To ensure connectivity, assume that G is a connected graph.

Considering the limited detection distance of any actual sensor, the detection range refers to the maximum distance at which an agent can detect other agents and respond to a target, and the detection radius of the intelligent agent is set to R_{det}.

$$a_{ij} = \begin{cases} 1, d_{ij} \leq R_{det} \\ 0, d_{ij} > R_{det} \end{cases} \tag{1}$$

Here, d_{ij} is the distance between agent i and agent j. Equation (1) shows that if mutual detection is not possible, a connection cannot be established. This is in line with reality. Agents establish mutual connections based on the detection distance. This paper only considers that the topology of the multi-agent system is connected, based on the actual situation.

$d_{\min}$ is a safe distance, which is the minimum distance that prevents collisions between any two agents

$$d_{\min} - d_{ij} \leq 0 \tag{2}$$

3. Establishment of a Task Allocation Model

Assume that there are N_T targets in system. The goal of the task allocation is to obtain decision inputs that maximize the overall performance indicators under certain constraints. A target must have three agents participating in the hunting. However, relying solely on information about relative position and velocity for task allocation may result in fewer than three agents being assigned to a certain target. Therefore, this paper considers the two-step task allocation method. The first step is the initial allocation, in which the agents obtain target information through their detection and communication with neighbors. Then, task allocation evaluation factors are achieved through a fuzzy logic system according to target information, and the target with the largest evaluation factor is taken as the initial assignment target. The second step is allocation optimization. Considering the constraints of hunting, each can only choose one target, and a target requires at least three agents to participate in the hunting. The agents with the same target automatically form a subgroup, and every agent is optimally assigned to different targets through an improved distributed self-organizing particle swarm algorithm. The task assignment evaluation factor of the agents within each target subgroup should be maximized, and the optimal task allocation result is finally obtained after optimization, as follows.

3.1. Modeling of Task Allocation Evaluation Factors

The factors influencing the task assignment assessment include the position and speed of the agent and the target, as well as the regulating ability of the agent. Based on the above influencing factors, a task allocation evaluation equation is established as follows:

$$P_{ig} = f(P_v, P_l) \tag{3}$$

where P_v is the relative speed evaluation factor between the agent and the target, indicating the strength of the agent's regulating ability. P_l is the relative position evaluation factor, representing the distance between the agent and the target.

$$P_v = f_1(\boldsymbol{v}_i, \boldsymbol{v}_g, \boldsymbol{L}_{ig}) \tag{4}$$

Here, $\boldsymbol{v_i}$ is the initial velocity of agent i, $\boldsymbol{v_g}$ is the velocity of target g, and $\boldsymbol{L}_{ig}$ is the relative position between agent i and target g, $L_{ig} = |\boldsymbol{L}_{ig}|$.

$$P_l = f_2(\boldsymbol{L}_i, \boldsymbol{L}_g, \boldsymbol{L}_{ig}) \tag{5}$$

Here, $\boldsymbol{L_i}$ is the position vector of agent i, and $\boldsymbol{L_g}$ is the position vector of target g. We obtain P_v, P_l, and P_{ig}, respectively, in Sections 3.2.1 and 3.2.2.

3.2. Solving the Task Allocation Evaluation Factor Model

Analyzing the task allocation evaluation equation, we can find several variables in Equations (4) and (5), and the function relationship is difficult to solve. It is difficult to find an accurate mathematical model to solve the equation. The fuzzy logic reasoning method can solve the problem, without an accurate function relationship. In artificial intelligence, fuzzy reasoning technology is a very important technique that can understand and process fuzzy and uncertain information, thereby achieving more intelligent decision making and computation. Fuzzy reasoning is known for its qualitative analysis, which can express the laws of objective things in standardized and concise manner using qualitative analysis. Fuzzy reasoning is good at considering problems from multiple perspectives, establishing connections between things, and paying special attention to summarizing the overall characteristics of things, estimating the process of time, and reaching approximate and flexible conclusions. Therefore, we can deal with the problem more flexibly, with strong adaptability and robustness, using the fuzzy logic reasoning method. The speed efficiency factor and position efficiency factor can be solved separately, and the solution results can be used as input to solve the task allocation evaluation factor.

3.2.1. Solution of the Speed Evaluation Factor

At a certain moment, agent i and target g are, respectively, in the positions shown in Figure 1. The black triangle represents agent i, and the red circle represents target g. The speed of agent i is v_i, $v_i = |\boldsymbol{v_i}|$, and the angle between the moving directions of agent i and target g is α_{ig}. When agent i selects target g as the target, it will ultimately generate a speed that tends to be consistent with the direction of target g. Therefore, the speed that agent i requires to change is as follows:

$$v_{\boldsymbol{q}} = v_i \sin \alpha_{ig} \tag{6}$$

$$v_{\boldsymbol{p}} = \begin{cases} v_i \cos \alpha_{ig}, & 0 \le \alpha_{ig} \le 90^\circ \\ -v_i \cos \alpha_{ig}, & 90^\circ \le \alpha_{ig} \le 180^\circ \end{cases} \tag{7}$$

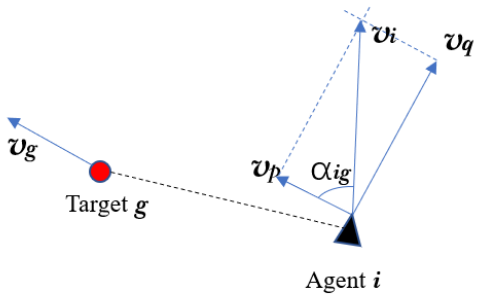

Figure 1. Schematic diagram of the position of the agent and target at a certain moment.

We can design a fuzzy logic inference system using v_q and v_p as the input variables for solving the speed efficiency factor, and the speed efficiency factor P_v as the output variable. The domain of each variable is set as follows:

$$v_q \in \left\{0, \frac{1}{4}v_i, \frac{1}{2}v_i, \frac{3}{4}v_i, v_i\right\}$$
$$v_p \in \left\{-v_i, -\frac{3}{4}v_i, -\frac{1}{2}v_i, -\frac{1}{4}v_i, 0, \frac{1}{4}v_i, \frac{1}{2}v_i, \frac{3}{4}v_i, v_i\right\}$$
$$P_v \in \left\{-v_i, -\frac{3}{4}v_i, -\frac{1}{2}v_i, -\frac{1}{4}v_i, 0, \frac{1}{4}v_i, \frac{1}{2}v_i, \frac{3}{4}v_i, v_i\right\}$$

We can define fuzzy set language variables and select appropriate fuzzy word sets for each variable, as show in Table 1.

Table 1. Fuzzy inference language variable definition table.

Linguistic Variable	**nb**	**nm**	**ns**	**z0**	**ps**	**pm**	**pb**	**pbb**	**pbbb**
Definition	Negative big	Negative median	Negative small	Zero	Positive small	Positive median	Positive big	Positive bigger	Positive biggest

The fuzzy language variables corresponding to v_q include z0, ps, pm, and pb. The fuzzy language variables corresponding to v_p include nb, nm, ns, z0, ps, pm, and pb. The fuzzy language variables corresponding to P_v include nb, nm, ns, z0, ps, pm, and pb.

Next, we can determine the fuzzy rules of inference. The following fuzzy rules of inference are designed according to the constraints of the agents and other actual conditions, combined with existing experience. They are listed in Table 2.

Table 2. Fuzzy rule of inference setting table of P_v.

P_v		v_q			
		z0	**ps**	**pm**	**pb**
v_p	nb	nb	nm	nm	nm
	nm	nm	nm	nm	nm
	ns	ns	ns	ns	ns
	z0	z0	ns	ns	ns
	ps	ps	ps	ps	ps
	pm	pm	pm	pm	pm
	pb	pb	pb	pm	pm

3.2.2. Solving the Path Evaluation Factor

The agents need to communicate with neighboring agents during the hunting process. If the agents are all clustered toward the system center, the communication loss will be reduced, and the communication stability among the agents will be improved, which is conducive to improving hunting efficiency. The design of the position evaluation factor also needs to take into account the movement state of the agent. Hence, the agent i clusters toward the center, and the position evaluation factor is positive. The position evaluation factor is negative when agent i moves in the direction opposite to the center.

The position of agent i relative to the target during the process of target hunting can be divided into two cases. If the distance between agent i and the center of the MAS is longer than that between target g and the center of the MAS, then agent i moves toward the center of the task area, and the position evaluation factor is positive. If the distance from agent i to the center of the MAS is shorter than the distance of target g to the center of

the MAS, then agent i will move away from the center of the task area, and the position evaluation factor is negative.

$$P_l = \begin{cases} |\boldsymbol{L}_{ig}|, |ON_i| \geq |OT_g| \\ -|\boldsymbol{L}_{ig}|, |ON_i| \leq |OT_g| \end{cases} \tag{8}$$

Here, $|ON_i|$ represents the distance from agent i to the center of the MAS, and $|OT_g|$ represents the distance from target g to the center of the MAS, $i \in N, g \in N_T$.

Based on P_v and P_l, P_{ig} is solved using the fuzzy logic reasoning method as the output. The theoretical domain of each variable is set as follows:

$$\begin{aligned} P_v &\in \left\{-v_i, -\tfrac{3}{4}v_i, -\tfrac{1}{2}v_i, -\tfrac{1}{4}v_i, 0, \tfrac{1}{4}v_i, \tfrac{1}{2}v_i, \tfrac{3}{4}v_i, v_i\right\} \\ P_l &\in \left\{-L_{ig}, -\tfrac{3}{4}L_{ig}, -\tfrac{1}{2}L_{ig}, -\tfrac{1}{4}L_{ig}, 0, \tfrac{1}{4}L_{ig}, \tfrac{1}{2}L_{ig}, \tfrac{3}{4}L_{ig}, L_{ig}\right\} \\ P_{ig} &\in \{0, 0.2, 0.4, 0.6, 0.8, 1\} \end{aligned}$$

We can set the fuzzy set language variable of each variable separately:

The fuzzy language variables corresponding to P_v include nb, nm, ns, z0, ps, pm, and pb. The fuzzy language variables corresponding to P_l include nb, nm, ns, z0, ps, pm, and pb. The fuzzy language variables corresponding to P_{ig} include z0, ps, pm, pb pbb, and pbbb.

Then, we can determine the fuzzy rule of inference. According to P_v and P_l, the fuzzy rules of inference are shown in Table 3.

Table 3. Fuzzy rule of inference setting table of P_{ig}.

P_{ig}		P_v						
		nb	**nm**	**ns**	**z0**	**ps**	**pm**	**pb**
P_l	nb	z0	z0	ps	ps	pm	pb	pb
	nm	z0	ps	ps	pm	pb	pb	pbb
	ns	ps	ps	pm	pb	pb	pbb	pbb
	z0	ps	pm	pb	pbb	pbb	pbbb	pbbb
	ps	ps	ps	pm	pb	pb	pbb	pbb
	pm	z0	ps	ps	pm	pbb	pb	pbb
	pb	z0	z0	ps	ps	pm	pb	pb

3.3. Self-Organizing Distributed Collaborative Task Allocation Optimization Model

Firstly, the agent obtains the task allocation evaluation factor of the target through self-detection and communication with neighbor agents. This factor quantifies the probability of task completion. Then, each agent self-organizes to form a task assignment subgroup according to the task assignment evaluation factor, and every subgroup must make the final assignment result of every subgroup member optimal to satisfy the constraints. The distributed task allocation method adopts the improved particle swarm optimization algorithm, with good search ability and strong robustness, to achieve self-organized task allocation optimization. Particle swarm optimization is an optimization algorithm based on swarm intelligence. It has strong global search capabilities. The algorithm adopts the idea of swarm intelligence and can efficiently search for the global optimal solution by utilizing group collaboration and information sharing. It has a fast convergence speed. The algorithm can quickly converge during the search process, and compared to some traditional optimization algorithms, such as the genetic algorithm or ant colony algorithm, it can find the approximate optimal solution faster. It is not sensitive to initial values. The algorithm can also achieve similar optimization results under different initial values, which

makes it more robust. The proposed method is flexible and can be applied to various scenarios with different constraints.

$$J_T = \arg\max \sum_{i=1}^{Na} \sum_{g=1}^{N_T} c_{ig} P_{ig} \tag{9}$$

Here, N_a is the number of the neighbor agents of target g, and c_{ig} is the hunting factor, indicating that agent i hunts target g.

$$\mathrm{c}_{ig} = \begin{cases} 1, & \text{if agent i hunts target } g \\ 0, & \text{if agent i does not hunts target } g \end{cases} \tag{10}$$

The constraints of the multi-agent hunting multi-target task allocation model are as follows:

$\sum_{g=1}^{N_T} c_{ig} = 1, \forall i \in \{1, \cdots, N_a\}$ indicates that each agent i hunts only one target.

$\sum_{i=1}^{N_a} c_{ig} \geq 3, \forall g \in \{1, \cdots, N_T\}$ indicates that a target requires at least three agents for hunting.

Agent i cannot hunt target g with $P_{ig} = 0$.

Using the above organization, a multi-agent system can globally distribute task allocation and achieve the global optimal allocation result, while meeting the constraints.

4. Hunting Strategy

The continuous time motion model of the system is as follows:

$$\dot{\boldsymbol{x}}_i = \boldsymbol{v}_i, i = 1, \cdots, N_a \tag{11}$$

$$\dot{\boldsymbol{v}}_i = \boldsymbol{u}_i, i = 1, \cdots, N_a \tag{12}$$

$$\boldsymbol{u}_i = \sum_{j=1, j\neq i}^{N_a} (\boldsymbol{x}_j - \boldsymbol{x}_i)\phi(\|\boldsymbol{x}_j - \boldsymbol{x}_i\|) + \sum_{j=1, j\neq i}^{N_a} k_0(\boldsymbol{v}_j - \boldsymbol{v}_i) + k_1(\boldsymbol{x}_i - \boldsymbol{x}_g) + k_2(\boldsymbol{v}_i - \boldsymbol{v}_g), i = 1, \ldots, N_a \tag{13}$$

where $\boldsymbol{x}_i$ denotes the position of agent i, $\boldsymbol{v}_i$ denotes the velocity of agent i, and $\boldsymbol{u}_i$ denotes the control input of agent i. $\boldsymbol{x}_g$ denotes the position of target g, and $\boldsymbol{v}_g$ denotes the velocity of target g. $k_0 > 0$ is the weight of the velocity error between the agents, $k_1 > 0$ is the weight of the position error between agent i and target g, and $k_2 > 0$ is the weight of the velocity error between agent i and target g.

$$\phi(\boldsymbol{x}_j - \boldsymbol{x}_i) = \frac{1}{2}[(a+b)\sigma(\|\boldsymbol{x}_j - \boldsymbol{x}_i\| - d_{\min} + \frac{b-a}{\sqrt{4ab}}) + a - b] \cdot (\|\boldsymbol{x}_j - \boldsymbol{x}_i\| - d_{\min}) \tag{14}$$

Here, $\sigma(z) = z/\sqrt{1+z^2}$, and a and b are constant and $0 < a \leq b$. Equation (13) gives the control strategy for the multi-agent system. The first summation in Equation (13) is a control term that ensures the convergence of all the agents in the system, the second summation term maintains the velocity consensus of all the agent, the third makes certain that the agent is close to the target, and the fourth term guarantees that the agent can maintain the velocity consensus with the target. Thus, using Equation (13), we may suppose that the subgroup of multi-agent systems can finally hunt its target.

Under the consideration of the control inputs $\boldsymbol{u}_i$, the system can achieve stability. We can use the Lyapunov function method for proof.

Firstly, we can define the position error of agent i: $e_{x_i} = \boldsymbol{x}_i - \boldsymbol{x}_g$, and the velocity error is: $e_{v_i} = \boldsymbol{v}_i - \boldsymbol{v}_g$.

Then, the Lyapunov function is chosen as:

$$\begin{aligned} l &= \frac{1}{2}\sum_{i=1}^{N_a}\left(\|e_{x_i}\|^2 + \|e_{v_i}\|^2 + \phi(\|x_j - x_i\|)\right) \\ &= \frac{1}{2}\sum_{i=1}^{N_a}\left((x_i - x_g)^T(x_i - x_g) + (v_i - v_g)^T(v_i - v_g) + \phi(\|x_j - x_i\|)\right) \end{aligned} \tag{15}$$

From Wu et al. [30], we know that the positive semidefinite matrix $L(t)$ and $H(t)$ are in existence for the derivation of the Lyapunov function,

$$\dot{l} = -(v_i - v_g)^T[(L(t) + H(t)) \otimes I_n](v_i - v_g) \tag{16}$$

$\dot{l} \leq 0$ because $L(t) + H(t)$ is a positive semidefinite matrix. Then, we may obtain $l(t) \leq l(0)$, $t > 0$. Thus, $\|x_i - x_g\| \leq \sqrt{2l(t)} \leq \sqrt{2l(0)}$. This inequality tells us that agent i can catch up with the target. Meanwhile, according to the LaSalle invariance principle, for each hunting subgroup, $v_1 = v_2 = \cdots = v_{N_a} = v_g$.

The above hunting control model shows that the entire hunting process is achieved by obtaining the position error and velocity error of the target through the agent. That is to say, throughout the entire hunting process, the agent does not need to know the target's motion state to hunt.

5. Simulation Analysis

This section provides the multi-target hunting simulation. In the simulation, the targets scatter in all directions, and the graph composed of all agents is not completely connected, but is connected at first. More importantly, the number of agents near some targets is less three, so we must initially finish the task allocation work before hunting begins. We can randomly generate three scattered escape targets and 12 agents in the area of X = [−1000 m,1000 m] and Y = [−1000 m,1000 m] to satisfy the above conditions. The detection distance is $R_{\text{det}} = 400$ m, and the collision avoidance distance is $d_{\text{min}} = 10$ m.

A schematic diagram of the 12 agents and three targets at the initial moment is shown in Figure 2.

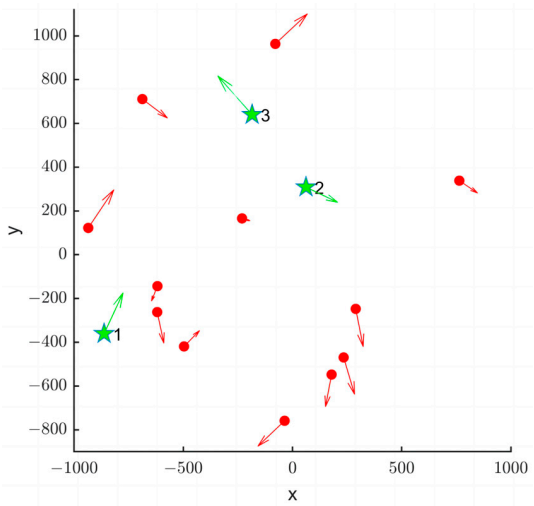

Figure 2. Initial distribution map of the agent and target.

In Figure 2, the red dot represents the agent, the green pentagram represents the target, the arrow direction represents the velocity direction, and the length of the arrow represents the speed. We can set the target to move at a constant speed of 3 m/s, according to the initial speed direction. The initial speed of the agent is 5 m/s, and the agents must change the speed in real-time, according to the assigned task situation, to ensure fast and efficient target hunting.

The adjacency matrix is composed of 12 agents:

$$a_{ij} = \begin{bmatrix} 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \end{bmatrix}$$

where 0 indicates no connection between the corresponding agents, and 1 indicates a connection relationship between the corresponding agents. The connection between the agents is shown in Figure 3. Figure 3 shows that the initial topology graph of the multi-agent systems is connected.

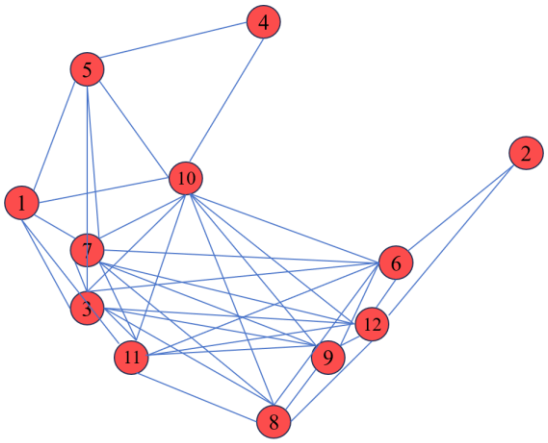

Figure 3. The initial connection topology diagram, composed of 12 agents.

Based on the information detected by the agent and the information obtained through communication with neighboring agents, the task assignment evaluation factor matrix is obtained through a fuzzy logic system:

$$P_{ig} = \begin{bmatrix} 0.81 & 0 & 0.41 & 0.09 & 0.29 & 0.16 & 0.51 & 0.2 & 0.26 & 0.49 & 0.83 & 0.22 \\ 0.38 & 0.8 & 0.75 & 0.8 & 0.77 & 0.81 & 0.43 & 0.47 & 0.57 & 0.73 & 0.52 & 0.78 \\ 0.62 & 0.32 & 0.35 & 0.57 & 0.3 & 0.2 & 0.39 & 0.06 & 0.1 & 0.59 & 0.27 & 0.13 \end{bmatrix}$$

The parameters of the improved particle swarm optimization algorithm used in the distributed task allocation method are set as follows: the number of particles is 50, the number of iterations is 1000, the inertia weight is 0.8, and the learning factors are c1 = 2.0, c2 = 2.0.

The optimal task allocation results are as follows:

$$\text{optimal\_}allocation = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 \\ 1 & 2 & 3 & 3 & 2 & 2 & 1 & 2 & 2 & 3 & 1 & 2 \end{bmatrix}$$

where the first row indicates each of the 12 agents, and the second row indicates the serial number of the target assigned to each agent.

The evaluation factor value for optimal fitness/optimal task allocation is 7.87. Figure 4 schematically shows the change curves of the number of iterations and optimal fitness/optimal task assignment evaluation factors.

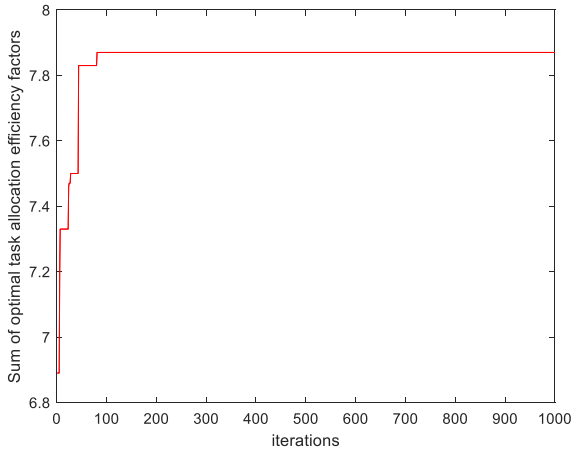

Figure 4. Schematic diagram of the variation curve of iteration number and optimal task allocation evaluation factor.

The relevant parameters for the hunting phase are set as follows: the step period is 0.25 s, and the number of iterations is 2000; meanwhile, $K_0 = 0.8$, $K_1 = 0.7$, and $K_2 = 0.75$.

Figures 5 and 6 show the position change process of the multi-agents. To intuitively understand the entire process of multi-agent hunting targets, Figure 5 shows the 2-D array diagrams of the changes in the positions of the agents and targets, respectively. The figures also display the real-time coordinate values of the targets, which further helps us observe the motion process. Figure 6 adds a time dimension and displays a 3-D array of changes in the positions of the agent and the target.

From the position change graph, it can be seen that in Figure 5(1) and Figure 6(1), starting from the initial position, each agent has gradually moved toward the target direction of task allocation after 35 iteration steps. In Figure 5(2) and Figure 6(2), after 112 iteration steps, agents are already reaching the vicinity of targets 1 and 3. In Figure 5(3) and Figure 6(3), after 180 iteration steps, targets 1 and 3 are no longer visible, which indicates that the agent has reached a position close to targets 1 and 3. Figure 5(4) and Figure 6(4) shows that after 253 iteration steps, target 1 has been hunted by three agents. In Figure 5(5) and Figure 6(5), after iteration to 451 iteration steps, the three targets represented by the green pentagram can no longer be seen. In Figure 5(7) and Figure 6(7), after 1528 iteration steps, the agent formed an encirclement cluster that hunts the target. In Figure 5(8) and Figure 6(8), after 1528 until 2000 iteration steps, the agent always follows the formed bounding encirclement and moves forward with the target, without losing or leaving the target. What requires further explanation here is that the agent does not know the motion state of the target. However, the agent can adjust, step by step, its position, velocity, and acceleration through its position error, velocity error, and acceleration error, with the target under the control of the hunting control strategy, always rounding up the target at a certain safe distance.

Figure 7 shows the complete trajectory of the multi-agent encirclement of dynamic multiple targets. All three targets move at a uniform speed in their respective directions. Figure 7(1) shows the hunting process of multiple dynamic targets in a 2-D environment of the multi-agent system. Compared to Figure 7(1),(2) has added a time coordinate axis. Figures 8–10 show the process and results of multi-agent subgroups surrounding targets 1, 2, and 3, respectively. In the trajectory distribution diagram in Figures 7–10, the black line shows the motion trajectory of the agent, and the green line indicates the motion trajectory of the target.

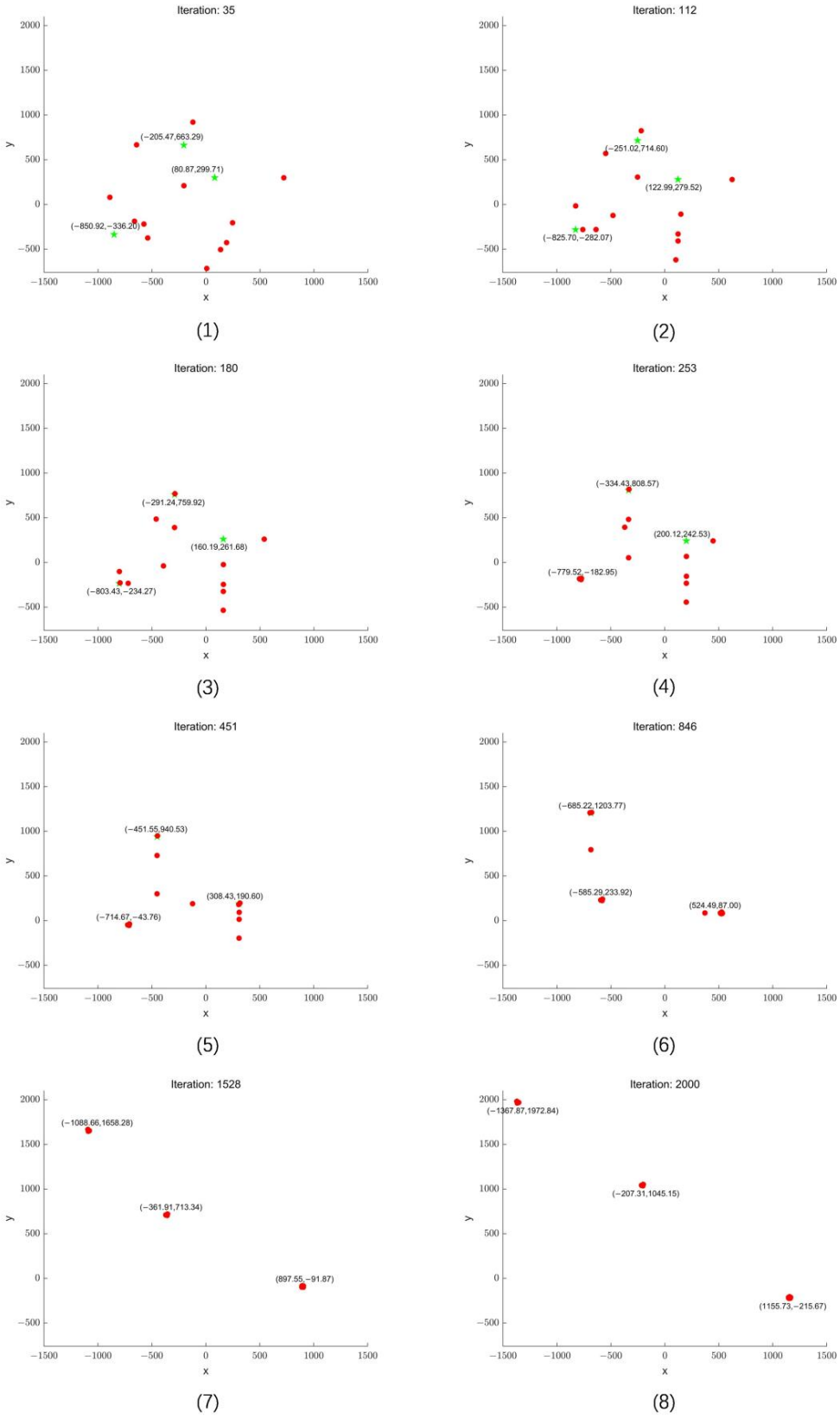

Figure 5. Iteration step position distribution map. (**1**) Iterative 35 step position distribution map; (**2**) Iterative 112 step position distribution map; (**3**) Iterative 180 step position distribution map; (**4**) Iterative 253 step position distribution map; (**5**) Iterative 451 step position distribution map; (**6**) Iterative 846 step position distribution map; (**7**) Iterative 1528 step position distribution map; (**8**) Iterative 2000 step position distribution map.

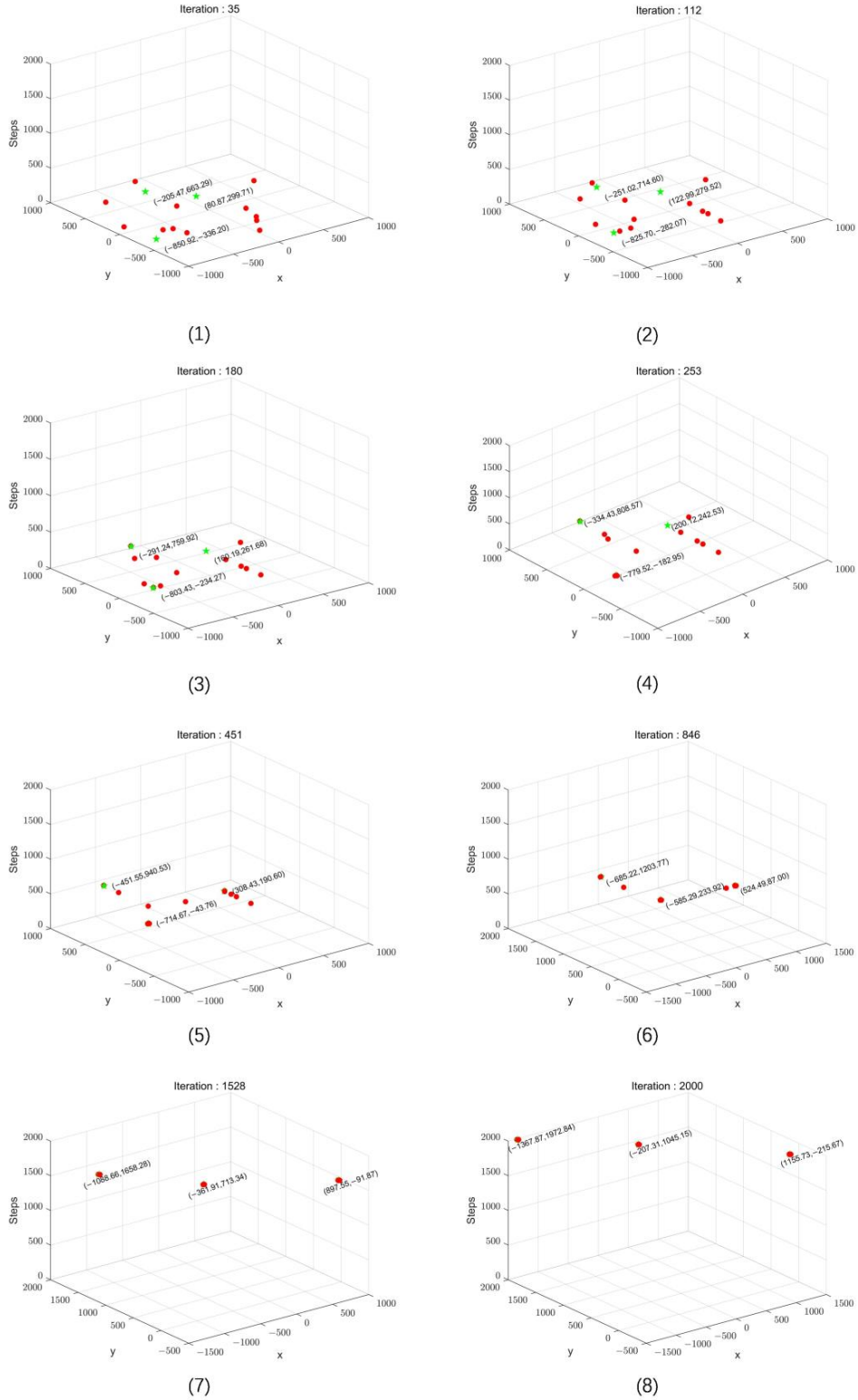

Figure 6. The 3-D position change map. (**1**) Iterative 35 step position change map; (**2**) Iterative 112 step position change map; (**3**) Iterative 180 step position change map; (**4**) Iterative 253 step position change map; (**5**) Iterative 451 step position change map; (**6**) Iterative 846 step position change map; (**7**) Iterative 1528 step position change map; (**8**) Iterative 2000 step position change map.

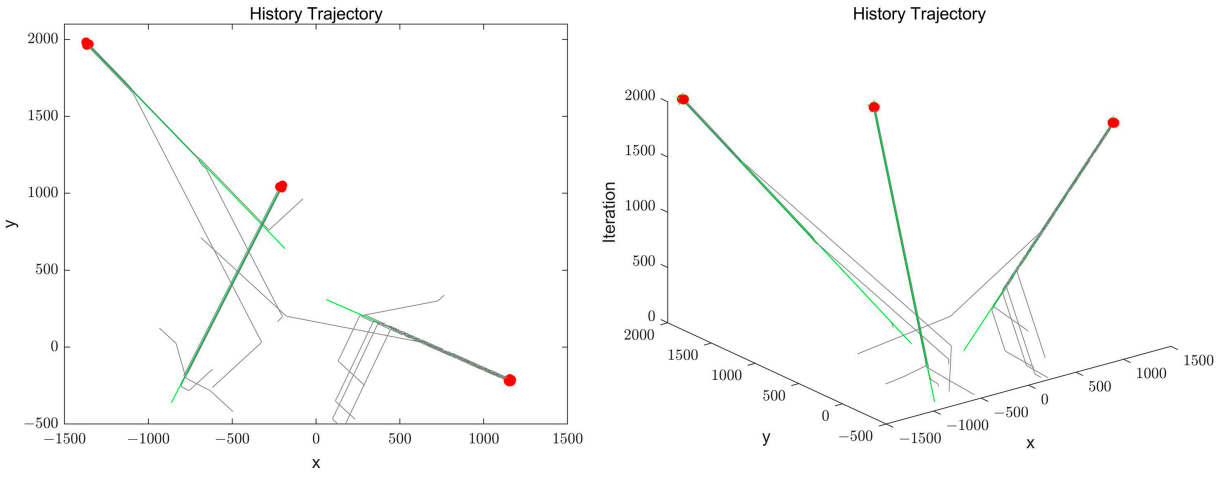

Figure 7. Distribution of trajectories obtained by iterating 2000 steps: (**1**) Trajectory distribution obtained through 2000 iterations in 2-D coordinates; (**2**) Trajectory distribution obtained through 2000 iterations in 3-D coordinates.

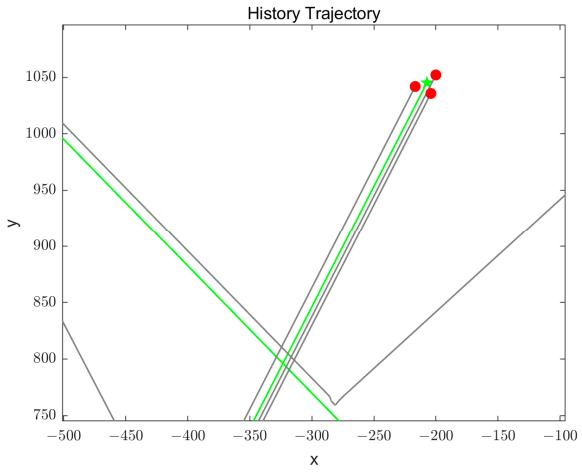

Figure 8. Target 1 movement process trajectory map.

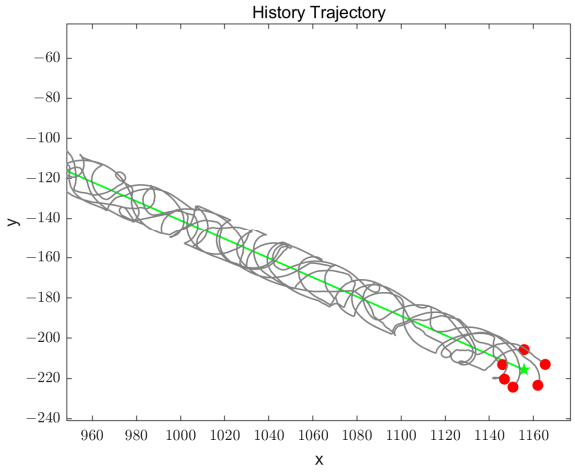

Figure 9. Target 2 movement process trajectory map.

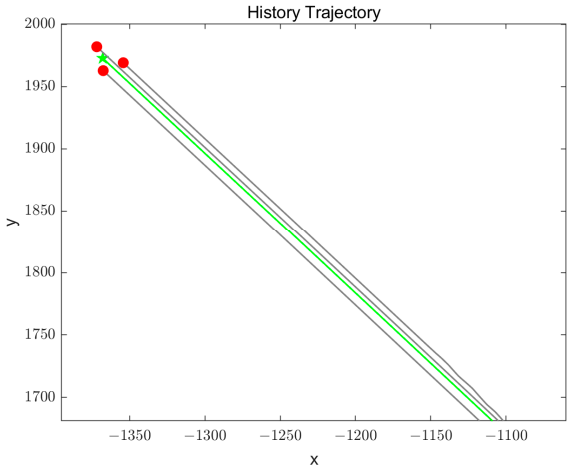

Figure 10. Target 3 movement process trajectory map.

Figure 11 shows the variation in the relative positions of each agent relative to their hunting target, which indicates that each agent can approach and gather around the target it has chosen.

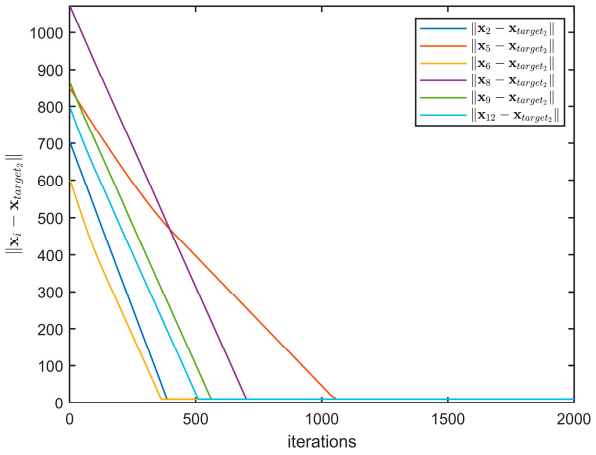

Figure 11. The relative distance between the agent and target 2.

Figures 8–10 show that after successful hunting of targets 1 and 3, the comparative position of the agent relative to its target remains unchanged. In contrast, after successfully hunting target 2, the agent rotates relative to its target. The reason for this is that the number of agents around target 1 and target 3 is both three, which is an odd number. For any agent, the other two agents are symmetrically distributed on both sides of the line connecting the agent and the target, and the forces can cancel each other out, without any force that drives its rotation. Target 2 is not such a case. The number of hunting agents is six, which is an even number, and the forces cannot cancel each other out, so the agents rotate around the target. Figure 12 shows the variation process of the distance between any two agents of the six agents around target 2. Figure 12(1) clearly shows that the six agents involved in hunting target 2 gradually approach each other over time and eventually maintain a close distance range. The locally enlarged image in Figure 12(2) shows that the distance between the six agents was ultimately maintained within a range of 20 m.

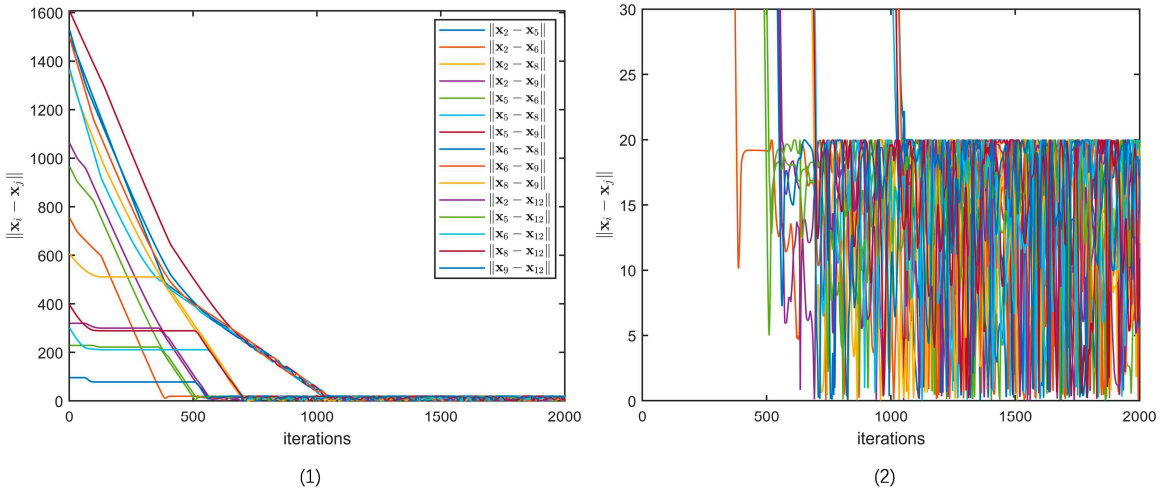

Figure 12. Distance map between various agents for hunting target 2. (**1**) Global display diagram. (**2**) Enlarged map of local layout.

Through the above simulation experiments, we can see that the hunting task can be successfully accomplished. It should be further emphasized that our simulation experiments consider the case in which the number of agents is small, and even though the number of agents is only three, the roundup can be successfully achieved. If the number of agents is greater, hunting will be easier to achieve, and the effect will also be better. This further illustrates the superiority of our designed hunting system.

6. Conclusions

This paper proposed a hunting method based on task allocation for the dynamic multi-target hunting problem of multi-agent systems with dispersed escape, which offers outstanding advantages in application value and practical guidance significance. We introduced fuzzy logic systems and heuristic optimization algorithms for task allocation, transforming the dynamic, multi-target hunting problem into dynamic, single-target hunting, which greatly simplifies the problem of multi-target hunting, ultimately achieving a dynamic, multi-target hunting. In designing a hunting controller, an attraction/repulsive force model based on potential field function was adopted and introduced to support the hunting control strategy of predicting the target's motion trajectory. Hunting was achieved through the tracking error and velocity error between the agent and the target, ensuring that all agents can cooperate to reach the target's surroundings and encircle the target. Finally, we provided a proof of the stability of the hunting system using the Lyapunov stability theorem, and the effectiveness of the designed hunting method was verified using simulation results. Compared to a single agent, MASs exhibit advantages such as high efficiency, robustness, and scalability. The research on multi-target collaborative hunting methods has practical significance and can be applied to tasks such as military target strikes and disaster site rescues. Due to the dynamic avoidance of targets, multi-target collaborative hunting tasks are complex. In multi-target collaborative hunting tasks, there are key sub-tasks, such as task allocation, collaborative strategies of multiple agents, and surrounding targets. The results required for task allocation directly affect the success rate and efficiency of hunting tasks.

Author Contributions: Material preparation, data collection, and analyses were performed by S.H. The first draft of the manuscript was written by S.H. The review and editing of the manuscript were performed by L.W. The supervision of the project was performed by W.L. and Z.W. Project administration was performed by L.W., W.L. and M.L. All authors commented on previous versions of the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This project was supported by the Hainan Provincial Natural Science Foundation of China (Grant No. 621RC512), the National Natural Science Foundation of China (Grant No. 61876073), and the Fundamental Research Funds for the Central Universities of China (Grant No. JUSRP21920).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare that they have no conflicts of interest.

References

1. Zhu, F.; Tan, C. Consensus Control of Linear Parameter-Varying Multi-Agent Systems with Unknown Inputs. *Sensors* **2023**, *23*, 5125. [CrossRef]
2. Jin, S.; Zhang, G. Adaptive Consensus of the Stochastic Leader-Following Multi-Agent System with Time Delay. *Mathematics* **2023**, *11*, 3517. [CrossRef]
3. Ma, L.; Wang, Y.L.; Han, Q.L. H_∞ Cluster Formation Control of Networked Multiagent Systems With Stochastic Sampling. *IEEE Trans. Cybern.* **2021**, *51*, 5761–5772. [CrossRef] [PubMed]
4. Yu, D.; Long, J.; Chen, C.L.P.; Wang, Z. Adaptive Swarm Control Within Saturated Input Based on Nonlinear Coupling Degree. *IEEE Trans. Syst. Man Cybern. Syst.* **2022**, *52*, 4900–4911. [CrossRef]
5. Qiu, Y.; Jin, Y.; Yu, L.; Wang, J.; Wang, Y.; Zhang, X. Improving Sample Efficiency of Multiagent Reinforcement Learning With Nonexpert Policy for Flocking Control. *IEEE Internet Things J.* **2023**, *10*, 14014–14027. [CrossRef]
6. Olfati-Saber, R. Flocking for multi-agent dynamic systems: Algorithms and theory. *IEEE Trans. Autom. Control* **2006**, *51*, 401–420. [CrossRef]
7. Talebi, S.P.; Werner, S. Distributed Kalman Filtering and Control Through Embedded Average Consensus Information Fusion. *IEEE Trans. Autom. Control* **2019**, *64*, 4396–4403. [CrossRef]
8. Kou, L.; Chen, Z.; Xiang, J. Cooperative Fencing Control of Multiple Vehicles for a Moving Target With an Unknown Velocity. *IEEE Trans. Autom. Control* **2022**, *67*, 1008–1015. [CrossRef]
9. Xu, B.; Zhang, H.T.; Meng, H.; Hu, B.; Chen, D.; Chen, G. Moving Target Surrounding Control of Linear Multiagent Systems With Input Saturation. *IEEE Trans. Syst. Man Cybern. Syst.* **2022**, *52*, 1705–1715. [CrossRef]
10. Hu, B.B.; Chen, Z.; Zhang, H.T. Distributed Moving Target Fencing in a Regular Polygon Formation. *IEEE Trans. Control Netw. Syst.* **2022**, *9*, 210–218. [CrossRef]
11. Xie, Y.; Han, L.; Dong, X.; Li, Q.; Ren, Z. Bio-inspired adaptive formation tracking control for swarm systems with application to UAV swarm systems. *Neurocomputing* **2021**, *453*, 272–285. [CrossRef]
12. Guo, J.; Yan, G.; Lin, Z. Local control strategy for moving-target-enclosing under dynamically changing network topology. *Syst. Control Lett.* **2010**, *59*, 654–661. [CrossRef]
13. Chen, Z. A cooperative target-fencing protocol of multiple vehicles. *Automatica* **2019**, *107*, 591–594. [CrossRef]
14. Huang, Z.; Zhu, D.; Sun, B. A multi-AUV cooperative hunting method in 3-D underwater environment with obstacle. *Eng. Appl. Artif. Intell.* **2016**, *50*, 192–200. [CrossRef]
15. Zengin, U.; Dogan, A. Cooperative target pursuit by multiple UAVs in an adversarial environment. *Rob. Auton. Syst.* **2011**, *59*, 1049–1059. [CrossRef]
16. Fan, S.; Liu, H.H.T. Multi-UAV Cooperative Hunting in Cluttered Environments Considering Downwash Effects. *Guid. Navig. Control.* **2023**, *3*, 2350004. [CrossRef]
17. Chen, M.; Zhu, D. A Novel Cooperative Hunting Algorithm for Inhomogeneous Multiple Autonomous Underwater Vehicles. *IEEE Access* **2018**, *6*, 7818–7828. [CrossRef]
18. Cao, X.; Xu, X. Hunting Algorithm for Multi-AUV Based on Dynamic Prediction of Target Trajectory in 3D Underwater Environment. *IEEE Access* **2020**, *8*, 138529–138538. [CrossRef]
19. Yu, D.; Long, J.; Philip Chen, C.L.; Wang, Z. Bionic tracking-containment control based on smooth transition in communication. *Inf. Sci.* **2022**, *587*, 393–407. [CrossRef]
20. Du, W.; Guo, T.; Chen, J.; Li, B.; Zhu, G.; Cao, X. Cooperative pursuit of unauthorized UAVs in urban airspace via Multi-agent reinforcement learning. *Transp. Res. Part C Emerg. Technol.* **2021**, *128*, 103122. [CrossRef]
21. Xia, J.; Luo, Y.; Liu, Z.; Zhang, Y.; Shi, H.; Liu, Z. Cooperative multi-target hunting by unmanned surface vehicles based on multi-agent reinforcement learning. *Def. Technol.* **2022**, *29*, 80–94. [CrossRef]
22. Trigui, S.; Koubaa, A.; Cheikhrouhou, O.; Youssef, H.; Bennaceur, H.; Sriti, M.-F.; Javed, Y. A Distributed Market-based Algorithm for the Multi-robot Assignment Problem. *Procedia Comput. Sci.* **2014**, *32*, 1108–1114. [CrossRef]
23. Tao, L.H.; Ju, K.F. Distributed task allocation modeling based on agent topology and protocol for collaborative system. *Optik* **2016**, *127*, 7776–7781. [CrossRef]
24. Jin, L.; Li, S.; La, H.M.; Zhang, X.; Hu, B. Dynamic task allocation in multi-robot coordination for moving target tracking: A distributed approach. *Automatica* **2019**, *100*, 75–81. [CrossRef]

25. Baert, Q.; Caron, A.C.; Morge, M.; Routier, J.C. Fair multi-agent task allocation for large datasets analysis. *Knowl. Inf. Syst.* **2018**, *54*, 591–615. [CrossRef]
26. Liu, Y.; Bucknall, R. Efficient multi-task allocation and path planning for unmanned surface vehicle in support of ocean operations. *Neurocomputing* **2018**, *275*, 1550–1566. [CrossRef]
27. Tolmidis, A.T.; Petrou, L. Multi-objective optimization for dynamic task allocation in a multi-robot system. *Eng. Appl. Artif. Intell.* **2013**, *26*, 1458–1468. [CrossRef]
28. Shi, J.; Tan, L.; Lian, X.; Xu, T.; Zhang, H.; Zhang, Y. A multi- unmanned aerial vehicle dynamic task assignment method based on bionic algorithms. *Comput. Electr. Eng.* **2022**, *99*, 107820. [CrossRef]
29. Wang, L.; Wang, Z.; Hu, S.; Liu, L. Ant Colony Optimization for task allocation in Multi-Agent Systems. *China Commun.* **2013**, *10*, 125–132. [CrossRef]
30. Wu, Z.H.; Xie, L.B. Fault-tolerant finite-time dynamical consensus of double-integrator multi-agent systems with partial agents subject to synchronous self-sensing function failure. *Chin. Phys. B* **2022**, *31*, 128902. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.