Proceedings of the 39th Chinese Control Conference
July 27-29, 2020, Shenyang, China

Research on multi-robots self-organizing cooperative pursuit algorithm based on Voronoi graph

Yan Wang[1], Guohao He[1], Yuedong Ma[2], Guojie Kong[1, 3], Jianwei Gong[1*]

1. Intelligent Vehicle Research Center, Beijing Institute of Technology, Beijing 100081, China
E-mail: gongjianwei@bit.edu.cn

2. North Automatic Control Technology Institute, Taiyuan 030006, China
E-mail: mayuedong@cqu.edu.cn

3. Beijing Special Vehicle Academy, Beijing 100072, China
E-mail: daxuewuhee@163.com

Abstract: For the problem that the traditional distributed algorithm based on Voronoi graph has a low efficiency and a large standard deviation in capturing random targets, an improved Voronoi graph self-organizing cooperative hunting algorithm is proposed. Through selecting the nearest midpoint of common Voronoi graph boundary, pursuers find their own targets, and become the pursuers alliance for the same target. Each pursuer in the group uses target Voronoi graph area minimization strategy to capture. Simulation results show that the algorithm proposed in this paper can significantly improve the average time and standard deviation of target capture when the targets adopts different escape strategies.

Key Words: Multi-robots, Self-organizing, Cooperative hunting, Voronoi graph

1 Introduction

Multi-robots cooperative hunting is the study of how to make multiple robots cooperate to hunt another group of robots. Through the cooperative cooperation among multiple robots, they can capture the evader more efficiently and quickly than multiple robots alone.

The early research on the problem of multi-robots cooperative pursuit mainly focused on the pursuit of a single evader by a single pursuer in a continuous bounded space. The problem is modeled as a differential model, and the optimal solution has been found to the problem, that is, the saddle point at the distance between the pursuer and the evader[1]. On this basis, [2,3] puts forward a specific solution to the specific problem and a general solution is proposed in [4,5].

The research on multiple pursuers and multiple evaders is becoming popular. Compared with single pursuer and single evader, the difficulty is that as the number of pursuers and evaders increases, the computational burden becomes heavier and heavier. Therefore, how to realize the coordination among multiple pursuers without substantially increasing the computational amount is an important research field. The usual solution is to allow computation only in the state space of a single pursuer, rather than the joint state space of all pursuers, so that the optimal results can be computed in real time.

Wang [6] proposed a hunting alliance algorithm based on greedy optimal returns to solve the real-time computing difficulty of large-scale robot hunting tasks, so as to reduce computing time and improve hunting efficiency. Sun [7] proposed a neural network algorithm based on self-organizing mapping, which integrated task assignment into the network training process, and learned the competitive function values through reinforcement learning to shorten the hunting time. Zhou [8] improved the traditional contract network protocol and improved the performance of the algorithm by introducing the auxiliary decision matrix to improve alliance decision-making and the method of narrowing the scope of task bidding based on case-based reasoning. Hollinger [9] proposed a heuristic algorithm based on cost entropy to coordinate the robot's pursuit strategy and reduce the expected capture time by discretizing the environment in a complex indoor environment. Chung [10] modeled the chase problem as a random control problem, and decomposed the global objective function into a local objective function through priority weight to realize the chase target. Malafeyev[11] used a game model to solve the problem of optimal interception time of an underwater submarine for an unknown moving target under the condition of incomplete information. Zheng[12] proposed a distributed probability algorithm to express the uncertain state information of the pursuer and target through probability to minimize the expected time.

We put forward a method for multiple pursuers pursuing multiple evaders. We assume that multiple pursuers and evaders are in the convex bounded area, while pursuers just know the location information of evaders and don't know the speed and heading information. Even if the pursuers and evaders velocity are the same, we can still finish the capture tasks. Also our method performs better than traditional algorithm in the average time and variance in the random simulation experiment.

2 Multi-targets pursuit model

Consider that in the bounded convex environment $\boldsymbol{D}$ in $\mathbb{R}^2$, there are n objects, the position vector of the object is $\boldsymbol{e}_i$, where $\boldsymbol{e}_i \in \mathbb{R}^2$, $i \in \{1, 2, 3, \ldots, \mathrm{n}\}$;There are m pursuers, and the position vector of the pursuer is $\boldsymbol{p}_j$, where $\boldsymbol{p}_j \in \mathbb{R}^2$,

This work is supported by China Advanced Research Project under Grant 41412050101 and 301060701.

$j \in \{1,2,3..., m\}$. First, define the ultimate distance of the target:

$$d_{min}(t) \triangleq \min_j \|\boldsymbol{p}_j(t) - \boldsymbol{e}_i(t)\| \quad (1)$$

We set the criteria for successful capture as, when t_0, $d_{\min}(t) \leq r_c$, where r_c is a predefined capture distance, which is a constant, indicating that when the limit distance is less than r_c, the target is successfully captured, and when $t > t_0$, we assume that the target is always captured.

Consider the following robot equation of state:

$$\dot{\mathbf{e}}_i = \boldsymbol{u}_e^i, i = 1,2,3,\dots,n$$
$$\dot{\mathbf{p}}_j = \boldsymbol{u}_p^j, j = 1,2,3,\dots,m$$

Where $\boldsymbol{u}_e^i$ is the control input of target *i*, $\boldsymbol{u}_p^j$ is the control input of pursuer *j*, and $\boldsymbol{u}_e^i$ and $\boldsymbol{u}_p^j$ meet the following speed constraints:

$$\|\boldsymbol{u}_e^i\| \leq v_{max}^e, \|\boldsymbol{u}_p^j\| \leq v_{max}^p, \forall t \geq 0$$

We proposes a hypothesis without loss of generality:

$$v_{max}^e = v_{max}^p = 1$$

3 Self-organizing cooperative pursuit algorithm

3.1 Single-target chase algorithm

Firstly, the safe region of target *i* is defined. The safe region of target *i* refers to all points $\boldsymbol{x}$ in *D* that target *i* can reach before any other pursuer *j*. This safe region is equivalent to the Voronoi partition of target *i*[13]. The region is defined as:

$$V_i = \{\boldsymbol{x} \in D \mid \|\boldsymbol{x} - \boldsymbol{e}_i\| \leq \|\boldsymbol{x} - \boldsymbol{p}_j\| \quad \forall j = 1,2,3\dots,m\} \quad (2)$$

It can be seen that from the above equation that the safe area of target *i* is related to the location of the target and the position of the pursuer around it. When the safe area of target *i* is small enough, that is, the radius of the inner circle of the safe area is less than r_c, the target is successfully captured. Therefore, the pursuer adopts the pursuit idea to minimize the safe area where the target is located[13].

Define the area of security zone V_i:

$$S_i = \int_{V_i} dx \quad (3)$$

Substitute V_i into the above equation and take the derivative. According to the chain derivation rule, it can be known that:

$$\dot{S}_i = \frac{\partial S_i}{\partial \boldsymbol{e}_i}\dot{\boldsymbol{e}}_i + \sum_{j=1}^{m}\frac{\partial S_i}{\partial \boldsymbol{p}_j}\dot{\boldsymbol{p}}_j \quad (4)$$

The pursuer's goal is to minimize the region S and analyze the derivative components of S. Only the partial derivative of the area to the pursuer's state and the pursuer's speed can be controlled. The control quantity is the pursuer's speed. The control strategy chosen by each pursuer is

$$\boldsymbol{u}_p^j = \dot{\boldsymbol{p}}_j = -\frac{\frac{\partial S_i}{\partial \boldsymbol{p}_j}}{\left\|\frac{\partial S_i}{\partial \boldsymbol{p}_j}\right\|} \quad (5)$$

Then the area S will decrease along the gradient descent direction[14], in $\mathbb{R}^2$

$$\frac{\partial S_i}{\partial \boldsymbol{p}_j} = \frac{L_j}{\|\boldsymbol{p}_j - \boldsymbol{e}_i\|}\left(p_j - C_{b_j}\right) \quad (6)$$

Where, b_j is the Voronoi boundary between target *i* and pursuer *j*, L_j is the length of Voronoi boundary, and C_{b_j} is the midpoint of Voronoi boundary. Substitute (6) into (5) to obtain the optimal control strategy of pursuer *j*

$$\boldsymbol{u}_p^{j^*} = \frac{C_{b_j} - \boldsymbol{p}_j}{\left\|C_{b_j} - \boldsymbol{p}_j\right\|} \quad (7)$$

Our goal is to minimize the Voronoi diagram partition area of the area where target *i* is located. Under this pursuit strategy, the area of target I will not increase, that is, $\dot{S}_i \leq 0$, and currently $\dot{S}_i = 0$ only when the target adopts the following control strategy.

$$\boldsymbol{u}_e^{i^*} = \frac{C_b - \boldsymbol{e}}{\|C_b - \boldsymbol{e}\|} \quad (8)$$

Proof: in the pursuit task of a single pursuer j for a single target I, substitute (6) into (4) to obtain:

$$\dot{S}_i = \frac{L_j}{\|\boldsymbol{p}_j - \boldsymbol{e}_i\|}\left[\left(\boldsymbol{p}_j - \boldsymbol{C}_b\right)^T\dot{\boldsymbol{p}}_j - (\boldsymbol{e}_i - \boldsymbol{C}_b)^T\dot{\boldsymbol{e}}_j\right] \quad (9)$$

The optimal control strategy (7) of the pursuer is substituted into formula (9) to obtain

$$\dot{S}_i = \frac{L_j}{\|\boldsymbol{p}_j - \boldsymbol{e}_i\|}\left[-\|\boldsymbol{p}_j - \boldsymbol{C}_b\| - (\boldsymbol{e}_i - \boldsymbol{C}_b)^T\dot{\boldsymbol{e}}_i\right]$$

So, we can conclude that $\dot{S}_i \leq 0$, and $\dot{S}_i = 0$ only if $\dot{\boldsymbol{e}}_i = \boldsymbol{u}_e^{i^*} = \frac{C_b - e_i}{\|C_b - e_i\|}$.

3.2 Multi-targets self-organizing cooperative hunting algorithm

In a convex bounded environment, for a single target, a single pursuer must be able to complete the chase task in a limited time. Multiple pursuers pursuing multiple targets is a subset of the above situation, which can ensure the completion of the pursuit task within a limited time[15].

In the process of the task, a number of pursuers through the organization of mutual cooperation, to improve the efficiency of the chase. For multi-targets chase task, we propose a self-organizing cooperative chase algorithm based on Voronoi diagram.

Let S_i be the partition area of Voronoi graph where the target *i*, and S be the sum of the partition area of Voronoi graph of all targets.

$$S = \sum_{i=1}^{n} S_i$$

The derivative of the chain is

$$\dot{S} = \sum_{i=1}^{n}\frac{\partial S_i}{\partial \boldsymbol{e}_i}\dot{\boldsymbol{e}}_i + \sum_{i=1}^{n}\sum_{j=1}^{m}\frac{\partial S_i}{\partial \boldsymbol{p}_j}\dot{\boldsymbol{p}}_j$$

The core idea of self-organizing cooperative algorithm is divided into two parts: self-organizing combination and cooperative pursuit calculation.

Self-organizing: first, $\boldsymbol{e}_i(i = 1,2,3\dots,n)$ and $\boldsymbol{p}_j(j = 1,2,3\dots,m)$ construct Voronoi diagram, the common edge of Voronoi partition composed of target and pursuer is *C*, and then calculate the point $\boldsymbol{C}_b^{ij}$ in the common boundary of each pursuer and target.

The pursuer chases a_{ij}, the closest distance to the target at the point in their common boundary

$$a_{ij} = \|\mathbf{p}_j - \boldsymbol{C}_b^{ij}\|$$

Through calculating $\min_i a_{ij}$ to get the pursuers j pursuing the goal of i, when there are multiple pursuers calculation of pursuing goals at the same time, the min is calculated for the pursuers $\min_i a_{ij}$ is same. The pursuers self-organization form collection pursuers $P_{set} = \{\dots, P_j, \dots\}$, thus forming multiple pursuers to organize to pursue the same target.

Collaboration: after multiple pursuers form $P_{set_1}, P_{set_2}, \dots$, they start to hunt targets. At this point, the pursuers gather to temporarily ignore other targets and only hunt for the current target i.

This problem is changed to a single target and can be handled in the pursuers sub problems, through the method based on Voronoi region least solving the pursuers for the same optimal control of the single target volume.

According to single objective pursuit conclusion, choosing the optimal control for a single target i $\boldsymbol{u}_p^{j^*} = \frac{C_{b_j} - \boldsymbol{p}_j}{\|C_{b_j} - \boldsymbol{p}_j\|}$, $j \in P_{set}$, pursue the optimal performance.

In the pursuit of multiple targets in the field, the pursuer conducts the pursuit by means of self-organization and cooperation. If the pursuer finds that the distance to the points in the common boundary of Voronoi diagram from the new target is smaller, namely $a_{ij} < a_{kj}$, where k is the target with sequence number k, the target of the pursuer j switches to k to recalculate the control quantity.

The specific algorithm steps of self-organizing cooperative pursuit are as follows.

Algorithm 1: self-organizing cooperative hunting

INPUT: $\boldsymbol{e}_1, \boldsymbol{e}_2, \dots, \boldsymbol{e}_n, \mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_m$

1: Initializes the location of the pursuer and target, $\boldsymbol{e}_1, \boldsymbol{e}_2, \dots, \boldsymbol{e}_n, \mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_m$

2: Calculate the global Voronoi diagram for all pursuers and all targets.

3: Each pursuer calculates the minimum distance d_{mid} of C_b in the common boundary according to the global Voronoi diagram, and determines the set of chasing target I and pursuer according to the minimum distance.

4: Calculate the Voronoi diagram of the pursuer set and the target when the pursuer set is chasing the target.

5: The optimal control of the collection of pursuers $u_p^{j^*} = \frac{C_{b_j} - p_j}{\|C_{b_j} - p_j\|}$, including C_{b_j} for local Voronoi diagram of pursuers public the halfway point of the boundary and the target.

6. If all targets are successfully captured, complete the task and exit the algorithm; otherwise, skip to step 2 to continue the algorithm.

4 Simulation and results analysis

4.1 Two escape strategies for the targets

In the simulation experiment, we use two escape strategies to verify the performance of our algorithm. Simulation results show that under different escape strategies, the self-organizing co-operation algorithm proposed in this paper has significantly improved both the mean capture time and the standard deviation of the capture time compared with the traditional algorithm based on Voronoi diagram.

The first escape strategy is based on the idea that the total area of the Voronoi diagram is the largest. The red border region where the target X9 is located is the Voronoi partition of X9. In this partition, the direction of the red arrow is the escape strategy of the target X9, which is widely used [13], as shown in figure 1.

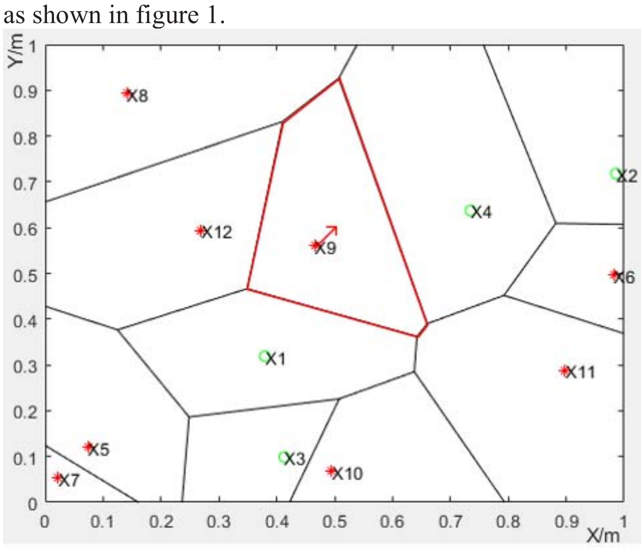

Fig. 1: Escape strategy of all targets Voronoi partition area maximization

The second escape strategy is to maximize the Voronoi diagram partition area of the target itself rather than the total area of the target, which is to ignore the influence of other targets. Figure 2 shows the strategy. The location of the target in figure 1 and figure 2 is the same and the second strategy adopted by the target is significantly different from the first.

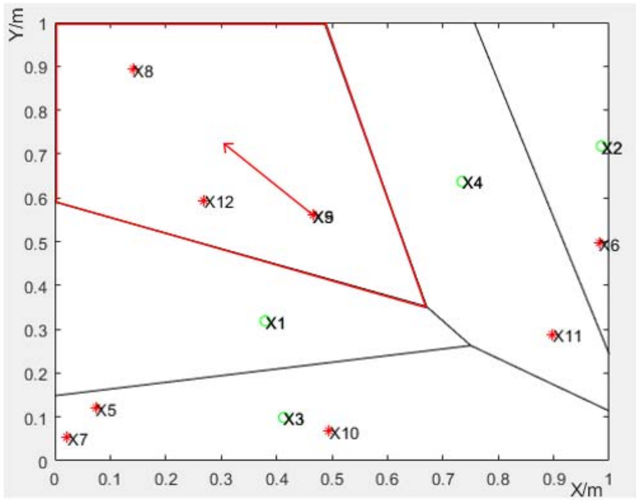

Fig.2: Escape strategy of target self Voronoi partition area maximization

Figures 3 and 4 show the pursuit of eight targets by four pursuers over time. The pursuer pursues the target by using the self-organizing cooperative pursuit algorithm of the improved Voronoi diagram. The green arrow represents the desired target point of the pursuer.

In the figure, X1 and X4 jointly pursue target X5 through cooperative organizations to speed up the pursuit progress, while X2 and X3 pursue independently. As time goes on, the area of the target area decreases, and all targets are successfully captured at the end of the simulation, as shown in figure 4.

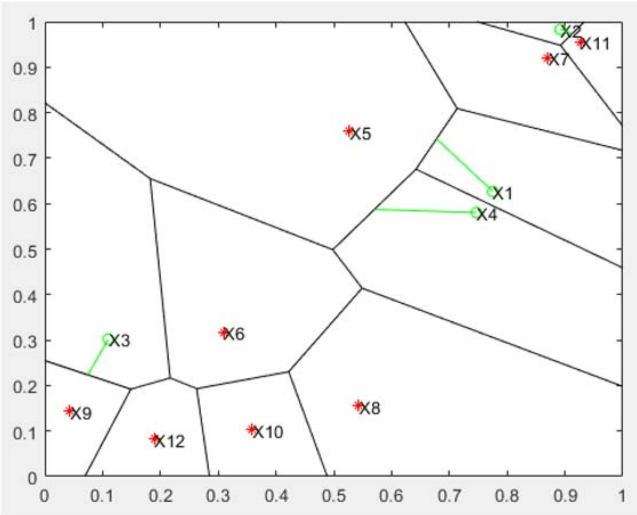

Fig.3: Setup the chase

We carried out 1000 times of simulated reality tests for different algorithm combinations and different escape strategies. The pursuer and the target appear randomly within the convex bounded region. Under two different escape strategies of the target, the performance of the algorithm is analyzed by comparing the pursuer's self-organizing cooperative pursuit algorithm with the traditional algorithm, and comparing the time between the two to complete the pursuit task.

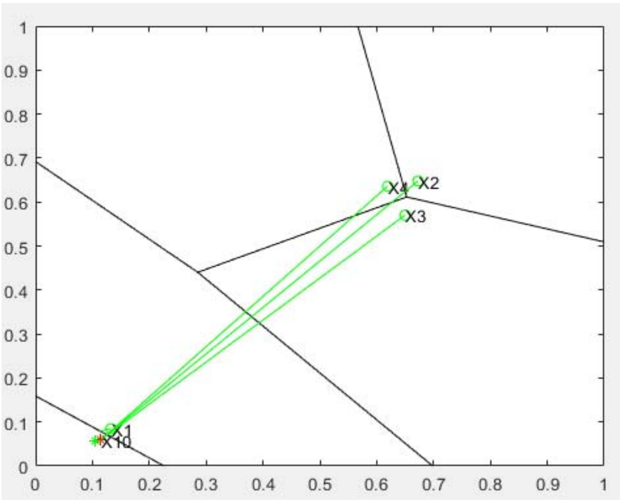

Fig.4: Finished the chase

Figure 5-8 shows the results of 1000 simulation experiments under different escape strategies. The abscissa in the figure is the time taken to complete the task, and the ordinate is the number of chase missions.

4.2 Escape strategy 1: maximum areas of all targets

In 1000 randomized simulations, the target adopted the escape strategy to maximize the partition area of the Voronoi diagram of all targets, and the pursuer used the improved algorithm to compare the average chase time and standard deviation of the traditional algorithm with the improved algorithm.

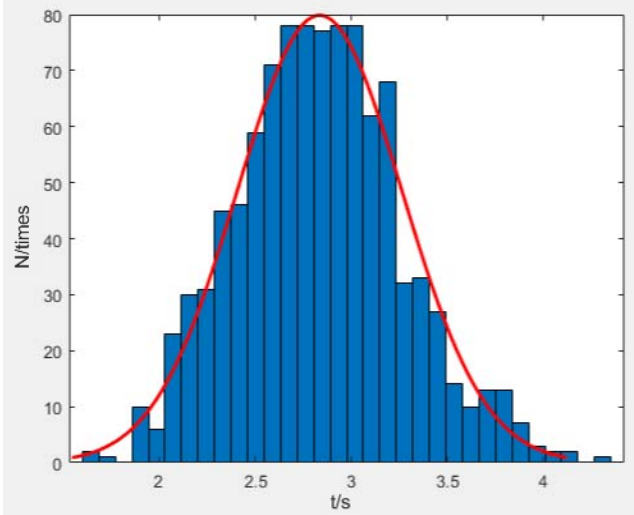

Fig.5: The pursuers adopted self-organizing cooperative algorithm and the targets maximized the partition area of Voronoi graph of all targets

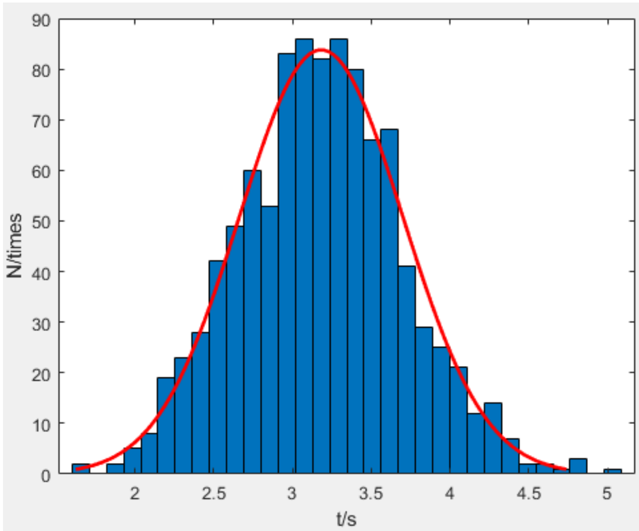

Fig.6: The pursuers adopted traditional algorithm, and the targets maximized the partition area of Voronoi graph of all targets

Table 1: Algorithm performance in escape strategy 1

	Traditional Algorithm	Our Algorithm	Improve rate
Average time	3.18s	2.83s	11%
Standard deviation	0.5189s	0.4294s	17%

In the case of escape strategy 1, our algorithm improved the average capture time and the standard deviation by 11% and 17% compared with the traditional algorithm.

4.3 Escape strategy 2: maximum self-area of one target

The target adopts the escape strategy to maximize the partition area of its Voronoi diagram, and the pursuer used the improved algorithm to compare the average chase time and standard deviation of the traditional algorithm.

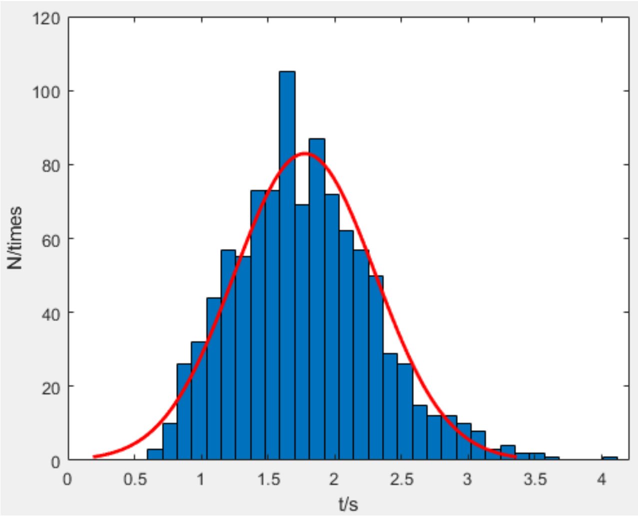

Fig.7: The pursuers adopted the improved algorithm, and the targets maximized the self target Voronoi partition area

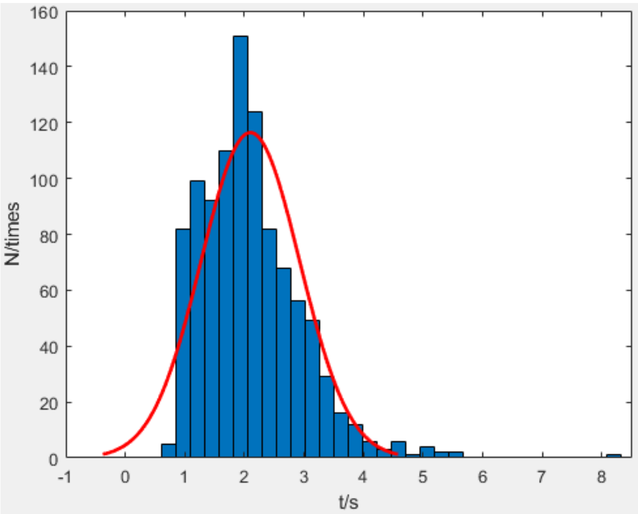

Fig.8: The pursuers adopted the traditional algorithm, and the targets maximized the self Voronoi partition area

Table 2: Algorithm performance in escape strategy 2

	Traditional Algorithm	Our Algorithm	Improve rate
Average time	2.10s	1.77s	16%
Standard deviation	0.8256s	0.5290s	36%

In the case of escape strategy 1, our algorithm improved the average capture time and the standard deviation by 16% and 36% compared with the traditional algorithm.

When the target adopts different escape strategies, it can be seen that our algorithm has a significant increase in the mean value and standard deviation of the pursuit time compared with the traditional algorithm.

5 Conclusion

This paper designs a self - organizing cooperative hunting algorithm for multi - target hunting task. In this method, the shortest distance of points in the common boundary of Voronoi diagram of all pursuers and targets is used to self-organize multiple robots into a set of pursuers, and then the control quantity of respective pursuers is calculated based on the set of pursuers pursuing the targets.

The main advantage of our algorithm over the traditional algorithm is that it embodies the cooperative part and can dynamically adjust the target they pursue according to the distance from evader instead of chasing the same target all the time. The traditional algorithm is that after a target is chased by the pursuers, the next target will be replaced. The traditional algorithm missed the opportunity to capture another evader at a relatively low cost because it ignores the nearby targets during chase another target.

The simulation results show that compared with the traditional algorithm, the self-organizing cooperative method can effectively reduce the mean value and standard difference of the capture time when the target adopts different escape strategies.

References

[1] R. Isaacs, Differential Games. *New York: Wiley*, 1967.

[2] L. A. Petrosjan, Differential Games of Pursuit, ser. *Series on Optimization. Singapore: World Scientific*, 1993, vol. 2.

[3] T. Basar and G. Olsder, Dynamic Noncooperative Game Theory, 2nd ed. *Philadelphia, PA: SIAM*, 1999.

[4] L. C. Evans and P. E. Souganidis, “Differential games and representation formulas for solutions of Hamilton-Jacobi-Isaacs equations,” *Indiana Univ. Math. J.*, vol. 33, no. 5, pp. 773–797, 1984.

[5] I. Mitchell, A. Bayen, and C. Tomlin, “A time-dependent HamiltonJacobi formulation of reachable sets for continuous dynamic games,” *IEEE Transactions on Automatic Control*, vol. 50, no. 7, pp. 947–957, July 2005.

[6] Wang Hao, Ding Lei, Fang Baofu. Pursuit Alliance Generation Algorithm for Multi-robot Pursuit Problem [J]. *Robot*, 2013, 35(2): 142-150.

[7] Sun Bohan, Wang Hao, Fang Baofu. Task Assignment for Emotional Robot Based on Self organizing Algorithm [J]. *Robot*, 2017 (5): 680687.

[8] Zhou Pucheng, Hong Bingrong, Wang Yuehai. Cooperative Pursuit of Multiple Robots in Dynamic Rnvironment [J]. *Robot*, 2005, 27(4): 289-295.

[9] Hollinger G, Kehagias A, Singh S. Probabilistic strategies for pursuit in cluttered environments with multiple robots[C]. *IEEE International Conference on Robotics and Automation.* IEEE, 2007: 3870-3876.

[10] Chung C F, Furukawa T. Coordinated pursuer control using particle filters for autonomous search-and-capture[J]. *Robotics and Autonomous Systems*, 2009, 57(6-7): 700-711.

[11] Malafeyev O, Alferov G, Andreyeva M. Group strategy of robots in game-theoretic model of interception with incomplete information[C]. *International Conference on Mechanics-Seventh Polyakhov's Reading*. IEEE, 2015: 1-3.

[12] Zheng J, Yu H, Liang W. Probabilistic strategies to coordinate multiple robotic pursuers in pursuit-evasion games[C]. *IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2007: 559-564.

[13] Huang H, Zhang W, Ding J. Guaranteed decentralized pursuit-evasion in the plane with multiple pursuers[C]. *IEEE Conference on Decision and Control and European Control Conference.* IEEE, 2011: 4835-4840.

[14] Huang H, Zhou Z, Zhang W. Safe-reachable area cooperative pursuit[J]. *IEEE Transactions on Robotics*, 2012: 10-5.

[15] Zhou Z, Zhang W, Ding J. Cooperative pursuit with Voronoi partitions[J]. *Automatica*, 2016, 72: 64-72.